

# SAP Database Guide: Oracle



**SAP NetWeaver 7.1**  
**February 2008**



## Copyright

© Copyright 2008 SAP AG. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP AG. The information contained herein may be changed without prior notice.

Some software products marketed by SAP AG and its distributors contain proprietary software components of other software vendors.

Microsoft, Windows, Outlook, and PowerPoint are registered trademarks of Microsoft Corporation.

IBM, DB2, DB2 Universal Database, OS/2, Parallel Sysplex, MVS/ESA, AIX, S/390, AS/400, OS/390, OS/400, iSeries, pSeries, xSeries, zSeries, z/OS, AFP, Intelligent Miner, WebSphere, Netfinity, Tivoli, Informix, i5/OS, POWER, POWER5, OpenPower and PowerPC are trademarks or registered trademarks of IBM Corporation.

Adobe, the Adobe logo, Acrobat, PostScript, and Reader are either trademarks or registered trademarks of Adobe Systems Incorporated in the United States and/or other countries. Oracle is a registered trademark of Oracle Corporation.

UNIX, X/Open, OSF/1, and Motif are registered trademarks of the Open Group. Citrix, ICA, Program Neighborhood, MetaFrame, WinFrame, VideoFrame, and MultiWin are trademarks or registered trademarks of Citrix Systems, Inc.

HTML, XML, XHTML and W3C are trademarks or registered trademarks of W3C®, World Wide Web Consortium, Massachusetts Institute of Technology.

Java is a registered trademark of Sun Microsystems, Inc.






JavaScript is a registered trademark of Sun Microsystems, Inc., used under license for technology invented and implemented by Netscape.

MaxDB is a trademark of MySQL AB, Sweden.

SAP, R/3, mySAP, mySAP.com, xApps, xApp, SAP NetWeaver, and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world. All other product and service names mentioned are the trademarks of their respective companies. Data contained in this document serves informational purposes only. National product specifications may vary.

These materials are subject to change without notice. These materials are provided by SAP AG and its affiliated companies ("SAP Group") for informational purposes only, without representation or warranty of any kind, and SAP Group shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP Group products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

## Icons in Body Text

Icon	Meaning
	Caution
	Example
	Note
	Recommendation
	Syntax

Additional icons are used in SAP Library documentation to help you identify different types of information at a glance. For more information, see *Help on Help → General Information Classes and Information Classes for Business Information Warehouse* on the first page of any version of *SAP Library*.

## Typographic Conventions

Type Style	Description
<i>Example text</i>	Words or characters quoted from the screen. These include field names, screen titles, pushbuttons labels, menu names, menu paths, and menu options.  Cross-references to other documentation.
<b>Example text</b>	Emphasized words or phrases in body text, graphic titles, and table titles.
EXAMPLE TEXT	Technical names of system objects. These include report names, program names, transaction codes, table names, and key concepts of a programming language when they are surrounded by body text, for example, SELECT and INCLUDE.
Example text	Output on the screen. This includes file and directory names and their paths, messages, names of variables and parameters, source text, and names of installation, upgrade and database tools.
<b>Example text</b>	Exact user entry. These are words or characters that you enter in the system exactly as they appear in the documentation.
<Example text>	Variable user entry. Angle brackets indicate that you replace these words and characters with appropriate entries to make entries in the system.
EXAMPLE TEXT	Keys on the keyboard, for example, F2 or ENTER.

SAP Database Guide: Oracle.....	20
New Features.....	20
New Features in SAP NetWeaver 7.1.....	20
New Features in SAP NetWeaver 7.0.....	23
New Features in SAP Release 6.40.....	24
New Features in SAP Release 6.30.....	25
New Features in SAP Release 6.20.....	26
New Features in SAP Release 6.10.....	27
Getting Started with Oracle and the SAP System .....	28
Database Security .....	29
Computing Center Management System .....	29
Specification of SAP Tables in the ABAP Dictionary .....	30
Oracle Database Storage Parameters in the ABAP Dictionary .....	30
Support on SAP Service Marketplace .....	32
Database System Configuration.....	33
Setting Up Archiving .....	34
Mirroring the Control File .....	36
Mirroring the Online Redo Log Files.....	37
Organizing Disk Storage.....	38
SAP Naming Conventions for Tablespaces and Data Files .....	39
Database Parameters.....	42
Environment Variables (UNIX).....	42
Environment Variables (Windows) .....	44
Directory Structure (UNIX).....	46
Directory Structure (Windows).....	51
Users and Roles .....	55
Oracle Databases on Raw Devices .....	57
Raw Devices and BR*Tools.....	58
Raw Devices with BRBACKUP and BRRESTORE .....	59
Limitations of the Oracle Database System .....	60
Approach to Oracle DBA.....	61
Instance Management.....	61
Recreate Database .....	65
Space Management .....	67
Managing Tablespaces.....	68
Managing Data Files.....	71
Segment Management.....	73
Reorganization.....	76
Export/Import .....	79

Special Export and Import Functions with BRSPACE .....	80
Reorganization Case Study .....	82
Reorganization with the Redefinition Package .....	85
Database Backup .....	86
Backup Overview .....	87
Why Back Up the Database? .....	88
What Needs Backing Up? .....	90
Database Backup Types .....	93
Online and Offline Backup .....	94
Consistent Online Backup .....	95
Complete Backup .....	96
Incremental Backup .....	97
Backup Cycles .....	98
Backup Approach with Daily Complete Backups .....	99
Backup Approach for Very Large Database with Partial Backups .....	101
Backup Approach with One-Day Retention Period .....	102
Backup Media .....	103
Volume Management .....	104
Volume Initialization .....	105
Volume Label Check .....	107
Volume Expiration Period .....	109
Used Volumes .....	111
Scratch Volume .....	112
Selecting Volumes Manually .....	112
Selecting Volumes with External Tools .....	113
Selecting Volumes Automatically .....	114
Tape Volume Size .....	116
Hardware Compression .....	118
Software Compression .....	119
Backup Methods .....	121
Backup to Multiple Disks .....	121
Backup to a Remote Disk .....	122
Backup to a Remote Tape Device .....	124
Two-Phase Backup .....	125
Structure-Retaining Database Copy .....	128
Parallel Backup .....	129
Unattended Backup .....	131
BRBACKUP and BRARCHIVE Backups in One Run .....	134
Backup Verify .....	136
Grouping Offline Redo Log Files .....	140

Advanced Backup and Recovery .....	141
Backup of Large Oracle Databases .....	141
Backup Devices for Large Databases .....	142
Backup of Large Databases to Tape with BRBACKUP .....	142
External Backup Programs for Large Databases .....	143
Parallel Backup of Large Databases to Disk with BRBACKUP.....	144
Optimization of File Distribution.....	145
Optimization with a Logical Volume Manager .....	146
Partial Backups.....	148
Tablespace Backups .....	149
Backup Example for a Large Database.....	150
Speeding Up the Backup.....	151
Standby Database.....	151
Standby Database Configuration .....	153
Standby Database: BRARCHIVE Backup of Offline Redo Log Files.....	156
Standby Database: BRBACKUP Backup of Database Files.....	157
Standby Database: Restore and Recovery .....	159
Standby Database: Remote Database Connect Requirements.....	160
Split Mirror Backup.....	161
Split Mirror Online Backup.....	165
Split Mirror Offline Backup.....	166
Split Mirror Backup: Software Configuration .....	168
Backup with Automatic Tape Changers.....	172
Mount and Dismount Commands.....	173
Autoloader Backup Example .....	175
Veritas Quick I/O Feature.....	176
External Backup Programs .....	177
Fixing an Online Backup Crash .....	180
Abort of Archive, Backup, or Restore.....	182
Restore and Recovery.....	183
Complete Database Recovery.....	187
Database Point-In-Time Recovery .....	190
Tablespace Point-in-Time Recovery .....	194
Whole Database Reset.....	198
Restore of Individual Backup Files .....	200
Restore and Application of Offline Redo Log Files.....	203
Disaster Recovery .....	204
Database System Check.....	207
Update Statistics .....	207
BR*Tools for Oracle DBA.....	210

Getting Started with BR*Tools.....	213
Configuration of BR*Tools .....	213
Configuring the Scroll Line Count for BR*Tools.....	214
Configuring the UNIX Command at for BR*Tools Batch Functions .....	214
Setting the Option To Log Displayed Information for BRSPACE .....	215
Effects of Autoextend and Resize on BR*Tools .....	216
Starting BR*Tools .....	217
BR*Tools User Interface .....	219
How to Use BR*Tools .....	225
Checking BR*Tools Release Information .....	230
BR*Tools in Action .....	230
Instance Management with BR*Tools .....	230
Starting Up the Database with BR*Tools .....	231
Shutting Down the Database with BR*Tools.....	234
Altering the Database Instance with BR*Tools .....	237
Altering Database Parameters with BR*Tools .....	240
Recreating a Database with BR*Tools.....	244
Showing Instance Status with BR*Tools.....	248
Showing Database Parameters with BR*Tools.....	251
Showing Database Owners with BR*Tools.....	254
Space Management with BR*Tools.....	256
Extending a Tablespace with BR*Tools.....	258
Creating a Tablespace with BR*Tools .....	261
Dropping a Tablespace with BR*Tools .....	264
Altering a Tablespace with BR*Tools.....	267
Altering a Data File with BR*Tools .....	271
Moving a Data File with BR*Tools .....	275
Showing Tablespaces with BR*Tools .....	278
Showing Data Files with BR*Tools .....	282
Showing Redo Log Files with BR*Tools .....	285
Showing Control Files with BR*Tools .....	287
Showing Disk Volumes with BR*Tools.....	290
Segment Management with BR*Tools.....	292
Reorganizing Tables with BR*Tools .....	295
Rebuilding Indexes with BR*Tools .....	299
Exporting Tables with BR*Tools .....	302
Importing Tables with BR*Tools.....	307
Altering Tables with BR*Tools .....	310
Altering Indexes with BR*Tools.....	314
Showing Tables with BR*Tools.....	318

Showing Indexes with BR*Tools .....	321
Showing Table Partitions with BR*Tools.....	324
Showing Index Partitions with BR*Tools.....	328
Showing Segments with BR*Tools .....	331
Showing Segment Extents with BR*Tools .....	334
Showing Free Extents with BR*Tools .....	337
Backup and Database Copy with BR*Tools .....	340
Backing Up the Database with BR*Tools.....	342
Backing Up the Offline Redo Log Files with BR*Tools .....	344
Copying the Database with BR* Tools.....	346
Non-Database Backup with BR*Tools .....	347
Backing Up a Database Disk Backup .....	348
Verifying a Database Backup with BR*Tools.....	350
Verifying an Offline Redo Log Backup with BR*Tools .....	351
Additional Functions for Backup and Database Copy with BR*Tools.....	353
Updating Compression Rates with BR*Tools.....	353
Preparing RMAN Backups with BR*Tools.....	354
Deleting Database Disk Backups with BR*Tools .....	355
Deleting Offline Redo Log Backups on Disk with BR*Tools.....	357
Controlling of BRARCHIVE Run with BR*Tools.....	358
Initializing BRBACKUP Tape Volumes with BR*Tools.....	359
Initializing BRARCHIVE Tape Volumes with BR*Tools.....	360
Restore and Recovery with BR*Tools .....	361
Complete Database Recovery with BR*Tools .....	363
Database Point-In-Time Recovery with BR*Tools .....	364
Tablespace Point-In-Time Recovery with BR*Tools.....	366
Whole Database Reset with BR*Tools .....	367
Restore of Individual Backup Files with BR*Tools.....	369
Restore and Application of Offline Redo Log Files with BR*Tools .....	375
Disaster Recovery with BR*Tools .....	380
Managing Flashback Database with BR*Tools.....	386
Procedures for Restore and Recovery with BR*Tools.....	389
Setting Point In Time and Tablespaces for Recovery .....	389
Checking the Status of Database Files - I.....	391
Selecting Database Backups .....	393
Checking the Status of Database Files - II.....	395
Checking the Status of Tablespaces.....	397
Exporting the Tablespaces Not Being Recovered .....	400
Restoring Control Files.....	403
Restoring Data Files.....	404



Restoring and Applying an Incremental Backup .....	409
Restoring and Applying Offline Redo Log Files .....	410
Performing Flashback Database .....	416
Opening the Database.....	418
Check and Verification with BR*Tools .....	421
Checking the Database System with BR*Tools .....	423
Validating the Database Structure with BR*Tools .....	424
Verifying Database Blocks with BR*Tools .....	425
Database Statistics with BR*Tools .....	426
Updating Database Statistics with BR*Tools .....	427
Collecting Missing Statistics with BR*Tools .....	429
Deleting Harmful Statistics with BR*Tools .....	430
Managing Database Statistics with BR*Tools.....	432
Additional Functions with BR*Tools.....	436
Showing Profiles and Logs with BR*Tools.....	437
Cleaning Up DBA Logs and Tables with BR*Tools .....	438
Adapting Next Extents with BR*Tools.....	439
Methods of Adapting Next Extent Size.....	441
Changing the Password of the Database User with BR*Tools .....	442
Creating or Changing Synonyms for DBA Tables .....	443
BR*Tools in Detail .....	444
BRBACKUP .....	444
Backing up Database Files .....	444
Backing Up Non-Database Files and Directories .....	445
Completion of BRBACKUP Backups .....	446
Hardware Compression for BRBACKUP .....	447
Command Options for BRBACKUP .....	448
-a -archive.....	450
-b -backup.....	450
-bd -backup_delete.....	451
-c -confirm.....	451
-db -delete_backup.....	453
-d -device .....	453
-e -execute.....	454
-f -fillup .....	455
-g -abort .....	455
-h -help.....	456
-i -initialize.....	456
-k -compress .....	457
-l -language.....	457

-m -mode .....	458
-n -number .....	460
-o -output .....	460
-p -profile.....	460
-q -query .....	461
-r -parfile .....	461
-s -saveaset .....	462
-t -type.....	462
-u -user .....	464
-v -volume .....	464
-w -verify .....	465
-V -VERSION.....	466
BRBACKUP Logs.....	466
Names of the BRBACKUP Detail Logs .....	466
BRBACKUP Detail Log.....	468
BRBACKUP Summary Log .....	469
BRARCHIVE .....	470
Hardware Compression for BRARCHIVE .....	471
Command Options for BRARCHIVE .....	471
-a -archive.....	473
-b -backup.....	473
-c -confirm.....	474
-d -device .....	474
-e -execute.....	475
-f -fill .....	475
-g -abort .....	476
-h -help.....	476
-i -initialize.....	476
-k -compress.....	477
-l -language.....	477
-m -modify .....	477
-n -number .....	478
-o -output .....	478
-p -profile.....	478
-q -query .....	479
-r -parfile .....	479
-s -sc -ds -dc -sd -scd -ss -ssd -cs -cbs .....	479
-u -user .....	481
-v -volume .....	481
-w -verify .....	482

-V -VERSION.....	482
BRARCHIVE Logs .....	483
Names of the BRARCHIVE Detail Logs.....	483
BRARCHIVE Detail Log .....	484
BRARCHIVE Summary Log .....	485
BRRESTORE .....	486
Restoring Files .....	487
Completion of BRRESTORE Runs.....	488
Examples of BRRESTORE Runs .....	489
Command Options for BRRESTORE .....	490
-a -archive -a1 -archive1.....	491
-a2 -archive2.....	492
-b -backup -b1 -backup1 .....	493
-b2 -backup2.....	493
-c -confirm.....	493
-d -device .....	494
-e -execute.....	495
-f -fillup .....	495
-g -abort .....	495
-h -help.....	496
-i -interval.....	496
-k -compress.....	496
-l -language.....	497
-m -mode .....	497
-n -number .....	498
-n2 -number2 .....	499
-o -output .....	499
-p -profile.....	500
-q -query .....	500
-r -parfile .....	500
-u -user .....	501
-w -verify .....	501
-V -VERSION.....	502
BRRESTORE Logs.....	502
Names of the BRRESTORE Detail Logs.....	502
BRRESTORE Detail Log.....	503
BRRESTORE Summary Log.....	503
BRRECOVER .....	504
Command Options for BRRECOVER .....	505
-a -tsp -tablespace .....	506

-b -backup.....	507
-c -confirm.....	507
-d -device.....	507
-e -degree.....	509
-g -scn -change.....	510
-h -help.....	510
-i -interval.....	510
-j -ins -instance.....	511
-l -language.....	511
-m -pit -time.....	511
-n -seq -sequence.....	512
-n -seq1 -sequence1.....	512
-o -rpt -point.....	512
-p -profile.....	513
-r -parfile.....	513
-s -scroll.....	513
-t -type.....	513
-u -user.....	514
-V -VERSION.....	514
-w -own -owner.....	514
BRRECOVER Logs.....	514
BRRECOVER Detail Log.....	514
BRRECOVER Summary Log.....	517
BRSPACE.....	517
Command Options for BRSPACE.....	519
-c -confirm.....	520
-f -function.....	520
-f dbstart.....	521
-f dbshut.....	522
-f dbalter.....	523
-f dbparam.....	524
-f dbcreate.....	525
-f dbshow.....	528
-f tsextend.....	530
-f tscreate.....	532
-f tsdrop.....	536
-f tsalter.....	536
-f dfalter.....	537
-f dfmove.....	538
-f tbreorg.....	539

-f idrebuild .....	543
-f tbexport.....	545
-f tbimport.....	548
-f tbalter .....	551
-f idalter .....	552
-f mstats .....	553
-f mfback .....	555
-h -help.....	555
-l -language.....	556
-o -output .....	556
-p -profile.....	557
-q -query .....	557
-s -scroll .....	557
-u -user .....	557
-V -VERSION.....	558
BRSPACE Logs .....	558
BRSPACE Detail Log .....	559
BRSPACE Summary Log.....	561
BRSPACE Structure Change Log .....	562
BRSPACE Parameter Change Log.....	566
BRCONNECT .....	569
Database System Check with BRCONNECT .....	570
BRCONNECT Default Conditions for Database Administration.....	572
BRCONNECT Default Conditions for Database Operations.....	579
Adapt Next Extents with BRCONNECT .....	581
Internal Rules for Determining Next Extent Size .....	582
Update Statistics with BRCONNECT.....	583
-force with Update Statistics .....	585
Deletion of Damaging Statistics .....	586
Verification of Table and Index Structure .....	586
Internal Rules for Update Statistics .....	587
Update Statistics for InfoCube Tables.....	589
Sample Sizes for Update Statistics .....	591
Changing Database User Passwords with BRCONNECT.....	592
Clean Up Old Logs and Trace Files with BRCONNECT .....	593
Additional BRCONNECT Functions.....	594
Command Options for BRCONNECT .....	595
-c -confirm.....	596
-f -function.....	596
-f check .....	596

-f chpass .....	597
-f cleanup .....	598
-f crsyn .....	600
-f dbshut .....	600
-f dbstart .....	601
-f dbstate .....	601
-f next .....	601
-f stats .....	603
-h -help .....	611
-l -language .....	612
-o -output .....	612
-p -profile .....	613
-q -query .....	613
-u -user .....	613
-V -VERSION .....	614
BRCONNECT Logs .....	614
Names of the BRCONNECT Detail Logs .....	614
BRCONNECT Detail Log .....	615
BRCONNECT Summary Log .....	615
BRTOOLS .....	616
Command Options for BRTOOLS .....	616
-c -confirm .....	617
-h -help .....	617
-i -interval .....	618
-l -language .....	618
-p -profile .....	618
-s -scroll .....	619
-u -user .....	619
-w -show .....	619
-V -VERSION .....	620
Profiles, Logs, Messages, and Return Codes for BR*Tools .....	620
Initialization Profile init<DBSID>.sap .....	620
archive_copy_dir .....	621
archive_dupl_del .....	622
archive_function .....	622
archive_stage_dir .....	623
backup_dev_type .....	623
backup_mode .....	626
backup_root_dir .....	627
backup_type .....	628

check_cond .....	629
check_exclude.....	633
check_owner .....	634
cleanup_brarchive_log .....	634
cleanup_brbackup_log .....	634
cleanup_brconnect_log .....	634
cleanup_brrrecover_log.....	634
cleanup_brrrestore_log.....	635
cleanup_brspace_log .....	635
cleanup_check_msg.....	635
cleanup_db_log .....	635
cleanup_disk_archive .....	635
cleanup_disk_backup.....	636
cleanup_exp_dump .....	636
cleanup_ora_trace.....	636
cleanup_owner .....	636
cleanup_xdb_log .....	637
compress .....	637
compress_cmd .....	638
compress_dir .....	638
copy_in_cmd .....	639
copy_out_cmd .....	639
cpio_disk_flags .....	640
cpio_flags .....	640
cpio_in_flags.....	640
db_services .....	641
dd_flags .....	641
dd_in_flags .....	641
disk_copy_cmd.....	642
dismount_cmd .....	642
exec_parallel .....	643
exp_dump_dir.....	644
exp_table .....	644
expir_period.....	644
imp_table .....	645
mount_cmd.....	645
mount_par_file.....	646
new_db_home .....	646
next_exclude .....	647
next_limit_count.....	647

next_max_size.....	647
next_owner .....	648
next_special.....	648
next_table .....	649
orig_db_home.....	649
parallel_instances.....	649
pipe_copy_cmd .....	650
post_shut_cmd .....	650
post_split_cmd.....	651
pre_shut_cmd.....	652
pre_split_cmd .....	652
primary_db.....	653
rebuild_index .....	653
recov_copy_dir .....	654
recov_degree.....	654
recov_interval .....	654
recov_type .....	655
remote_host.....	655
remote_user .....	655
reorg_table .....	656
restore_mode .....	656
resync_cmd .....	658
rewind.....	658
rewind_offline .....	659
rman_channels .....	659
rman_compress.....	660
rman_copies .....	660
rman_diskratio .....	660
rman_filesperset.....	661
rman_maxcorrupt .....	661
rman_maxpiecesize.....	661
rman_maxopenfiles .....	662
rman_parms .....	662
rman_pool.....	663
rman_proxy.....	663
rman_rate .....	663
rman_send.....	664
rman_maxsetsize .....	664
saveset_members .....	664
scroll_lines.....	665



show_period .....	665
space_copy_dir .....	665
split_cmd .....	666
split_options.....	666
split_resync.....	667
stage_copy_cmd .....	667
stage_db_home.....	667
stage_root_dir.....	668
standby_db .....	668
stats_bucket_count.....	669
stats_change_threshold .....	669
stats_dbms_stats.....	669
stats_exclude.....	671
stats_info_cubes.....	671
stats_limit_time .....	672
stats_method .....	672
stats_owner .....	673
stats_parallel_degree .....	674
stats_sample_size .....	674
stats_special .....	674
stats_system_interval .....	676
stats_table .....	676
tape_address .....	678
tape_address_arch.....	678
tape_address_ctl .....	679
tape_address_ctl_arch .....	679
tape_address_rew .....	680
tape_address_rew_arch.....	680
tape_copy_cmd .....	681
tape_pos_cmd .....	683
tape_size .....	683
tape_size_arch .....	684
tape_use_count.....	684
uncompress_cmd .....	684
util_options .....	684
util_par_file .....	685
util_vol_access .....	685
util_vol_nlist .....	686
util_vol_unit.....	686
volume_archive .....	687

volume_backup .....	688
Logs for BR*Tools .....	689
Log Types .....	689
File System Logs .....	690
Database Logs for BRBACKUP, BRARCHIVE, BRSPACE, and BRCONNECT .....	690
Log Supplements.....	691
Messages and Return Codes for BR*Tools .....	692
Common Features of BRBACKUP and BRARCHIVE .....	693
Supported Backup Media.....	694
Effects of the Command Options .....	695
cpio Continuation Tape .....	696
cpio Error.....	697
Canceling a Backup .....	697
Other Tools for Oracle DBA .....	697
Database Recovery with SQLPLUS.....	698
Types of Database Errors.....	698
Error Analysis.....	699
Recovery after User Errors .....	700
Recovery after Statement Errors .....	701
Recovery after Process Errors.....	701
Recovery after an Instance Error.....	701
Recovery after Media Errors.....	702
Recovering from One Control File Missing.....	703
Recovering from All Control Files Missing.....	704
Recovering from Current Online Redo Log Missing.....	707
Recovering from One Inactive Online Redo Log Missing.....	708
Recovering from User Tablespace Missing.....	710
Recovering from SYSTEM Tablespace Missing.....	712
Recovering from Index Tablespace Missing.....	713
Recovering from Errors During the Archiving of Online Redo Logs.....	714
Performing an Incomplete Recovery .....	715
Finishing an Incomplete Recovery .....	717
Automating the Recovery .....	719
Updating the Control File .....	720
Oracle Recovery Manager .....	721
RMAN Backup Strategies .....	723
RMAN Incremental Backups After Structural Changes.....	725
RMAN Restore of Incremental Backups.....	725
RMAN Backup with the SAP Backup Library .....	726

RMAN Backup with an External Backup Library .....	729
RMAN Incremental Backups Without a Backup Library .....	731
RMAN Backup of the Offline Redo Log File .....	734
RMAN Tape Layout .....	734
RMAN Backup Verify .....	735
RMAN Save-Set Grouping.....	736
RMAN-Relevant Profile Parameters .....	737
The SAP Tools with Windows .....	741
SAP Conventions (Windows).....	741
Naming Conventions for Files (Windows).....	742
Executables.....	742
Starting the SAP Utility Programs .....	743
Database Analysis .....	744
Backup Strategy (Windows) .....	744
NTBackup .....	745
BRBACKUP/BRARCHIVE .....	745
Other Backup Programs.....	746
Structure-Retaining Database Copy or Restore on Windows.....	746
Offline Backup with Oracle Fail Safe for Cluster Systems.....	746
Oracle Real Application Cluster .....	747
RAC with BR*Tools .....	749
RAC with BRSPACE .....	750
RAC with BRBACKUP .....	751
RAC with BRARCHIVE .....	752
RAC with BRRESTORE and BRRECOVER .....	754
RAC with the init<DBSID>.sap Profile .....	755

# SAP Database Guide: Oracle

This component lets you administer your Oracle database with the SAP system. Read this documentation to make sure that you administer your database as efficiently as possible, which helps your company get the most from its SAP system.

## Implementation Considerations

For more information if you are new to Oracle database administration with the SAP System, see [Getting Started with Oracle and the SAP System](#).

For more information about installing the Oracle database with an SAP system, see the documentation on SAP Service Marketplace:

► [service.sap.com/instguides](https://service.sap.com/instguides) ◀

## Features

This documentation covers the following main areas:

- [Approach to Oracle DBA](#)
- [BR\\*Tools for Oracle DBA](#)
- [Other Tools for Oracle DBA](#)

For more information about new features in this release, see [New Features in SAP Release 6.40](#).

## Constraints

 Caution

Make sure that you use the SAP tools BRBACKUP, BRARCHIVE, BRRESTORE, BRRECOVER, BRSPACE, and BRCONNECT in accordance with your conditions of use for the SAP system and other SAP products.

## New Features

This section describes new features for Oracle database administration.

## New Features in SAP NetWeaver 7.1

There are the following new features in SAP NetWeaver 7.1:

- Verification of database and archive log files with RMAN

See SAP Note [1016173](#) for details of the changes, which affect the following sections:

- [Backup Verify](#)
- BRARCHIVE [-w|-verify use\\_rmv|first\\_rmv|only\\_rmv](#)
- BRBACKUP [-w|-verify use\\_rmv|only\\_rmv](#)
- BRRESTORE [-w|-verify use\\_rmv](#)
- Support for Oracle Data Pump in BRSPACE

See SAP Note [976435](#) for details of the changes, which affect the following sections:

- BRSPACE [-f tbexport](#)
- BRSPACE [-f tbimport](#)
- [Special Import and Export Functions](#)

- Administration of database statistics with BRSPACE

See SAP Note [1033125](#) for details of the changes, which affect the following sections:

- [Managing Database Statistics with BRTools](#)
- BRSPACE [-f mstats](#)
- RMAN support for split-mirror and standby-database backup, and support for split-mirror backup of a standby database

See SAP Note [968507](#) for details of the changes, which affect the following sections:

- [RMAN-Relevant Profile Parameters](#)
- [Split Mirror Backup](#)
- [Split Mirror Backup: Software Configuration](#)
- [Standby Database](#)
- `init<DBSID>.sap` parameters:
  - [backup\\_type](#)
  - [standby\\_db](#)
  - [disk\\_copy\\_cmd](#)
  - [post\\_shut\\_cmd](#)
  - [post\\_split\\_cmd](#)
  - [pre\\_shut\\_cmd](#)
  - [pre\\_split\\_cmd,](#)
- Support for RMAN save sets with disk backups

See SAP Note [1101530](#) for details of the changes, which affect the following sections:

- [disk\\_copy\\_cmd](#)
- [rman\\_compress](#)
- Automatic handling of Oracle 10g RAC Cluster Services
 

See SAP Note [1033126](#) for details of the changes, which affect [db\\_services](#).
- Improvements to handling of database statistics
 

See SAP Notes [892296](#), [865366](#), and [863811](#) for details of the changes, which affect the following sections:

  - [Update Statistics](#)
  - [Sample sizes for update statistics](#)
  - BRCONNECT [-f stats](#)
  - `init<DBSID>.sap` parameters:
    - [stats\\_bucket\\_count](#)
    - [stats\\_dbms\\_stats](#)
    - [stats\\_sample\\_size](#)
    - [stats\\_table](#)
- Improvements to table reorganization and index rebuild
 

See SAP Notes [1016172](#), and [1080376](#) for details of the changes, which affect the following sections:

  - [Reorganization](#)
  - [Reorganizing Tables with BR\\*Tools](#)
  - [Rebuilding Indexes with BR\\*Tools](#)
  - BRSPACE [-f tbreorg -r|-sortind -m|-mode online|offline](#)
  - BRSPACE [-f idrebuild -m|-mode online|offline](#)
- Various new options and parameters
 

SAP Note [1060696](#) for details of the changes, which affect the following sections:

  - BRCONNECT
    - [-f dbstate](#)
    - [-f dbstart](#)
    - [-f dbshut](#)
    - [-f crsyn](#)
    - [-f chpass](#)
    - [-f check](#)
  - BRRESTORE [-i|-interval](#)

- BRSPACE:
  - [-f tbreorg](#)
  - [-f idrebuild](#)
  - [-q|-query](#)
- `init<DBSID>.sap` parameters:
  - [stats\\_special](#)
  - [check\\_cond](#)
  - [db\\_services](#)
  - [rman\\_compress](#)
  - [man\\_maxcorrupt](#)
- Flashback database
 

See SAP Notes [1125923](#), [966073](#), and [966117](#) for details of the changes, which affect the following sections:

  - [Managing Flashback Database with BR\\*Tools](#)
  - [Performing Flashback Database](#)
  - [Database Point-In-Time Recovery with BR\\*Tools](#)
  - [Whole Database Reset with BR\\*Tools](#)
  - [-f mfbck](#)
- Abort of archive, backup, or restore
 

See SAP Note [1129197](#) for details of the changes, which affect:

  - [Abort of archive, backup, or restore](#)
  - [brarchive -q|-abort](#)
  - [brbackup -q|-abort](#)
  - [brrestore -q|-abort](#)



## New Features in SAP NetWeaver 7.0

There are the following new features in SAP NetWeaver 7.0:

### Note

You can already find these new features in higher patch levels of BR\*Tools 6.40.


- BRSPACE function [recreate database](#)

- Online conversion from `LONG` and `LONG RAW` fields to `CLOB` and `BLOB`, enabling all SAP tables to be subsequently [reorganized online](#). For more information, see *SAP Note 646681*.
- Tablespace renaming (Oracle 10g) in [Altering a Tablespace with BR\\*Tools](#) supported
- Shrink segment feature (Oracle 10g) in [Altering Tables with BR\\*Tools](#) and [Altering Indexes with BR\\*Tools](#) supported
- [Backup verify](#) enhancements
- Rebuild of NOLOGGING indexes after recovery, as described in [Restore and Recovery](#)
- BRCONNECT check condition `CRITICAL_TABLESPACE`, as described in [BRCONNECT Default Conditions for Database Administration](#)
- Parameters and options for BR\*Tools:
  - `init<DBSID>.sap` parameters [pipe\\_copy\\_cmd](#) and [archive\\_dupl\\_del](#)
  - Backup and restore modes `partial`, `non_db`, and `incr_all`, as described in [BRBACKUP -m|-mode](#)
  - Disk mirror split with [BRBACKUP -q split](#)
  - BRCONNECT options `stop|suspend|resume` for [BRCONNECT -f stats -u](#)
  - "Secure copy command" `scp` in `init<DBSID>.sap` parameter [stage\\_copy\\_cmd](#)



## New Features in SAP Release 6.40

There are the following new features in SAP Web Application Server (SAP Web AS) Release 6.40.

- New SAP tool for administration of Oracle databases, [BRSPACE](#), which you can use for:
  - [Instance Management](#)
  - [Space Management](#)
  - [Segment Management](#)
-  Note

We are no longer delivering SAPDBA.
- The following applies to BRSPACE:
  - BRSPACE is part of BR\*Tools and replaces the SAPDBA functions that have not so far been replaced by other BR\*Tools.
  - As of SAP Web Application Server 6.40, SAPDBA is no longer being released.



- You can continue to use SAPDBA 6.20 linked to Oracle 9i with SAP Web AS 6.40. However, we strongly recommend you to only use BR\*Tools instead.
- BR\*Tools 6.40, including BRSPACE can be used for all SAP Releases based on Oracle 9i.
- For more information on BR\*Tools, see the following SAP Notes:
  - [646681](#)
  - [647697](#)
  - [668640](#)
- New functionality for split mirror disk backup with the SPLITINT interface program, implemented using the following new parameters:
  - For BRBACKUP: `online_mirror` and `offline_mirror` in [brbackup -tl-type](#)
  - For the SPLITINT program: [split\\_options](#) and [split\\_resync](#) in the initialization profile `init<DBSID>.sap`



## New Features in SAP Release 6.30

There are the following new features in SAP Web Application Server (SAP Web AS) Release 6.30:

- [New user interface for BR\\*Tools](#)

There is now a GUI and a character interface for all BR\*Tools. You can now perform a wide range of functions from menus:

- [Backup and Database Copy with BR\\*Tools](#)
- [Restore and Recovery with BR\\*Tools](#)
- [Check and Verification with BR\\*Tools](#)
- [Database Statistics with BR\\*Tools](#)
- [Additional Functions with BR\\*Tools](#)

- [BRRECOVER for database recovery](#)

You can use this new tool to recover your database.

For more information, see SAP Note 602497.

- [Restore and Recovery](#)

This explains the concepts behind restore and recovery.

- New command option and initialization profile parameters
  - Command option
    - [BRRESTORE -n2|number2](#)

- Initialization profile parameters:
  - [recov\\_type](#)
  - [recov\\_copy\\_dir](#)
  - [recov\\_interval](#)
  - [recov\\_degree](#)
  - [scroll\\_lines](#)
  - [show\\_period](#)
- [Database Recovery with SQLPLUS](#)

We have updated the documentation for this.



## New Features in SAP Release 6.20

There are the following new features in SAP Web Application Server (SAP Web AS) Release 6.20:

- [BR\\*Tools](#) now supports Oracle 9i.
- RMAN backup without BACKINT:
  - [RMAN Backup with an External Backup Library](#)
  - SAP Note [420698](#)
- Update statistics for partitioned tables with BRCONNECT:
  - [Update Statistics with BRCONNECT](#)
  - SAP Note [424243](#)
- Update statistics for InfoCube tables with BRCONNECT:
  - [Update Statistics for InfoCube Tables](#)
  - SAP Note [428212](#)
- BRCONNECT support for Oracle monitoring in SAP Transaction RZ20:
  - [Monitoring the Oracle Database](#), especially [Database Health Alerts](#)
  - SAP Note [483659](#)
- New command options and initialization profile parameters
  - [Command Options for BRCONNECT](#):
    - [-o|output](#): new option process
    - [-f stats](#): new options `-f nocasc` and `-v index_store|cascade_store`
    - [-f next](#): new option `-f nocasc`

- Initialization profile parameters:
  - [check\\_exclude](#): new values `non_sap` and `all_part`
  - [cleanup\\_check\\_msg](#)
  - [next\\_exclude](#): new value `all_part`
  - [next\\_special](#): new value `all_sel`
  - [stats\\_dbms\\_stats](#)
  - [stats\\_table](#): new value `all_ind`
- SAP Notes [419679](#), [424239](#), [445884](#), and [483639](#).
- SAPDBA support for online reorganization of single tables
 

SAPDBA now supports online reorganization based on the Oracle internal PL/SQL functions.
- SAPDBA support for LOB columns
 

SAPDBA now supports the reorganization of tables with all types of large object (LOB) columns. Large objects are recreated with the same physical characteristics as before the reorganization.



## New Features in SAP Release 6.10

There are the following new features in SAP Release 6.10:

- [BRARCHIVE](#) support for [backup of offline redo log files](#) with Oracle Recovery Manager (RMAN)
 

This enables you to develop a comprehensive strategy for database files and offline redo log files. You can also take advantage of internal block consistency checking by RMAN for offline redo log files.
- [Software compression](#) for backups on remote disk
 

This helps to reduce the network traffic load required for backups. For more information, see [backup\\_dev\\_type](#).
- Support of `util_file_online` logic for [offline backups](#)

This enables you to fully implement [split mirror](#) and snapshot scenarios in BACKINT.
- New user interface for [BRCONNECT](#)

The main new functions are:

  - [Database system check](#)
  - [Adapt next extents](#)
  - [Update statistics](#)
  - [Clean up old logs and trace files](#)

For more information about the new commands, see [Command Options for BRCONNECT](#).

- SAPDBA support for the following new features:
  - Creation and extension of locally managed (that is, “bitmap”) tablespaces and reorganization of locally managed tablespaces.
  - [Veritas Quick I/O](#) to administer files from file systems as if they were raw devices



## Getting Started with Oracle and the SAP System

This section gives you an overview of database administration for Oracle databases running with the SAP System. The aim is to help you get started as quickly as possible by giving you concise information and pointers to further details.

To avoid error situations or bottlenecks in the database, you need to know where to find extra information that goes beyond the scope of this documentation.

### Process



Note

You do not have to follow the sequence below rigidly. It is only a suggestion. However, be sure to consider all the items listed.

1. You read the Oracle documentation thoroughly.
2. You read the Release Notes that appear with each new SAP Release. To do this, choose **Help** **Release Notes** in the SAP system.

See also [New Features](#).

3. You take necessary measures for [database security](#).



Note

For information about operating system security, see the documentation provided by your operating system vendor.

For example, for more information about operating system security on Microsoft Windows, see:

[www.microsoft.com/security](http://www.microsoft.com/security)

4. You learn about how you can [check your database](#) and develop an approach to [database backup](#).
5. You read the sections of the SAP Library on the ABAP Dictionary to learn about the conditions for creating [tables in the ABAP Dictionary](#) and on the database.
6. You read about the [Oracle Database Storage Parameters in the ABAP Dictionary](#).
7. If you have a problem, you use [SAP Service Marketplace](#) for support.

8. If you intend to use raw devices, see [Oracle Databases on Raw Devices](#).
9. You take note of the [Limitations of the Database System](#).

## Result

Now you are ready to [configure the database system](#).



## Database Security

Make sure that you take all relevant security precautions for your Oracle database.

For more information on Oracle database security with your SAP system, see:

- [Oracle Under UNIX](#)
- [Oracle Under Windows](#)



Note

You also need to take security precautions at operating system level. For more information, see:

- SAP documentation on operating system security with the SAP system:
  - [SAP System Security Under UNIX/LINUX](#)
  - [SAP System Security Under Windows](#)
- The documentation provided by your operating system vendor

For example, for more information about operating system security on Microsoft Windows, see:

[www.microsoft.com/security](http://www.microsoft.com/security)



## Computing Center Management System

You can use the Computing Center Management System (CCMS) to administer your Oracle database. For more information, see [CCMS: Oracle](#).

## Features

You can use CCMS to:

- Schedule a range of database administration activities - such as database backup, backup of the offline redo log files - using the [DBA Planning Calendar](#). You can choose from a range of action patterns that include the most commonly needed activities.
- [Display backup logs and status](#)
- [Update statistics for the cost-based optimizer](#)

- [Check the database system](#)
- [Monitor database operations](#)
- [Monitor database alerts](#)

## Specification of SAP Tables in the ABAP Dictionary

As soon as you create a database table for the SAP system, you can influence its storage parameters by maintaining its technical configuration in the [ABAP dictionary](#). This technical configuration is used to optimize the space requirements and access response of individual tables.

For more information, see [Creating Tables](#).

### Process

We recommend you to always maintain at least the parameters [Data Class](#) and [Size Category](#) in the technical configuration for each table:

- `Data class`

The data class logically defines the tablespace where your table is stored. When you select the correct logical table type, your table is automatically assigned to the correct database area when it is created. The F1 help for data class provides information on how to select the proper value for the table.

Data class is assigned to tablespace (for data) using the TAORA table. Data class is assigned to tablespace (for indexes) using the IAORA table.

- `Size category`

The size category specifies the estimated space requirements of the table on the database using the categories 0 to 4. When you create a table an `INITIAL` extent is reserved in the database. If you need more space later, storage space is added corresponding to the selected category (`NEXT` extent). The F4 help on size category displays the number of data records that will fit in the assigned storage area of the database for each of the categories. A maximum of 300 extents ought to be enough for storing table contents, assuming a database block size of 8 KB.

Size category is assigned to extent size using an entry in the table DD09L and an analysis of the TGORA table for tables or IGORA for indexes. These values of the ABAP Dictionary only represent starting values. For further database operations, use the options provided by BRCONNECT for automatically adapting the size of the `NEXT` extent for all tables. For example, see [f next](#).

#### Caution

This section on size category does not apply to locally managed tablespaces.

If the table is defined logically, you have to create it in the database in a second step.

## Oracle Database Storage Parameters in the ABAP

# Dictionary

The [database utility](#) is the interface between the ABAP Dictionary and the relational database underlying the SAP system. You can use it to create, convert, and delete ABAP Dictionary objects, as well as database table, database views, and other SAP objects. You can do this online or in the background.

In a conversion, the definition of a table in the database is adapted to its changed definition in the ABAP Dictionary. You can also implement various analysis options, for example, to display table and index definitions, or check the consistency of objects.

For some SAP objects, particularly transparent tables, you can set user-specific database parameters, such as `INITIAL EXTENT`, `NEXT EXTENT`, `MINIMUM EXTENTS`, `MAXIMUM EXTENTS`, `TABLESPACE`, `FREELIST GROUPS`, `FREELISTS`, `PCT FREE`, `PCT USED`, `INDEX ORGANIZED`, `PARTITION COLUMN LIST`, `COLUMN LIST`. You can also set flags to determine which parameters you want to apply the next time a table is created (deleted and created, converted) and which ones you want to take effect immediately. However, you cannot change all the values immediately. You can only change `MAXEXTENTS`, `NEXT`, `PCTFREE`, and `PCTUSED` immediately (these values are valid if a new storage area is requested for the object).

## Prerequisites

To use the database utility, you need an authorization for authorization object `S_DDIC_OBJ`, such as `S_DDIC_ALL`. Make sure that the authorizations in your system are set so that only the database administrator is authorized to configure the database parameters. Check the settings and change them if necessary.

## Features

Note the following parameters:

- `INDEX ORGANIZED`

This parameter lets you build a table in the same way as an index (that is, using a b\*tree), so saving space.

The advantages of index-organized tables are:

- Less storage, since data only stored in b\*tree instead of in table and index
- Rowid not stored in index entry of b\*tree
- Faster key-based access to table data

The disadvantages of index-organized tables are:

- `UNIQUE` constraints not allowed
- Cannot be stored in a cluster
- Cannot contain `LONG` columns (but `LOB` columns are possible)

- `PARTITION BY`

This allows you to partition tables by using a range expression.

The advantages of partitioned tables are:

- Logical attributes (such as table or indexed columns, constraints) are same in all partitions
- Physical attributes (tablespace, storage parameters) might differ in the partitions
- Data of partitioned objects can be handled in the same way as unpartitioned tables
- Partitions of a table can be separately exported, imported, dropped, set offline, backed up, and so on
- Queries can be performed only on specific partitions

The disadvantages of partitioned tables are:

- Bitmap indexes on partitioned tables can only be local
- Rule-based optimizer is not available for partitions

For more information on the other parameters, see the Oracle documentation.

## Activities

For more information on how to call the database utility and set storage parameters, see:

- [Database Utility](#)
- [Storage Parameters](#)



## Support on SAP Service Marketplace

If problem situations occur in the system, SAP Service Marketplace provides fast, effective help. Here you can address questions to SAP directly and immediately receive an initial response. You receive the same information that SAP itself uses for support work.

SAP support staff use incoming customer messages to write SAP Notes. SAP developers also create Notes to help you fix potential problems or to offer missing information.

## Process

1. You sign on to SAP Service Marketplace with the quick link `message` to enter your problem message:

[service.sap.com/message](https://service.sap.com/message)

The system searches automatically for SAP Notes matching the words in your problem message.

2. You can also use the alias `notes` to reach the SAP Notes area, where you can search manually for SAP Notes to help fix your problem.

[service.sap.com/notes](https://service.sap.com/notes)

3. There are different ways to search for helpful notes, such as:

- Entering the application area, such as BC-DB-ORA-DBA
- Entering the SAP Release



- Specifying the number of the note, if known
- Searching with free text, by entering a meaningful keyword, as in the following examples.

#### Example

```
brbackup, brarchive, brconnect, brrestore, brrecover,
brspace, restore, maxextents, offline redo logs, control
file, stuck, archivelog, tape_size, expir_period,
reorganization, restart, ora-<error number>, BR<error
number>, init.ora, init.sap, reconnect
```

## Database System Configuration

This section tells you how to configure your Oracle database system.

### Prerequisites

During database installation, which is not covered in this documentation, you need to change the initial database user passwords. You can use [BRCONNECT -f chpass](#) to do this at any time.

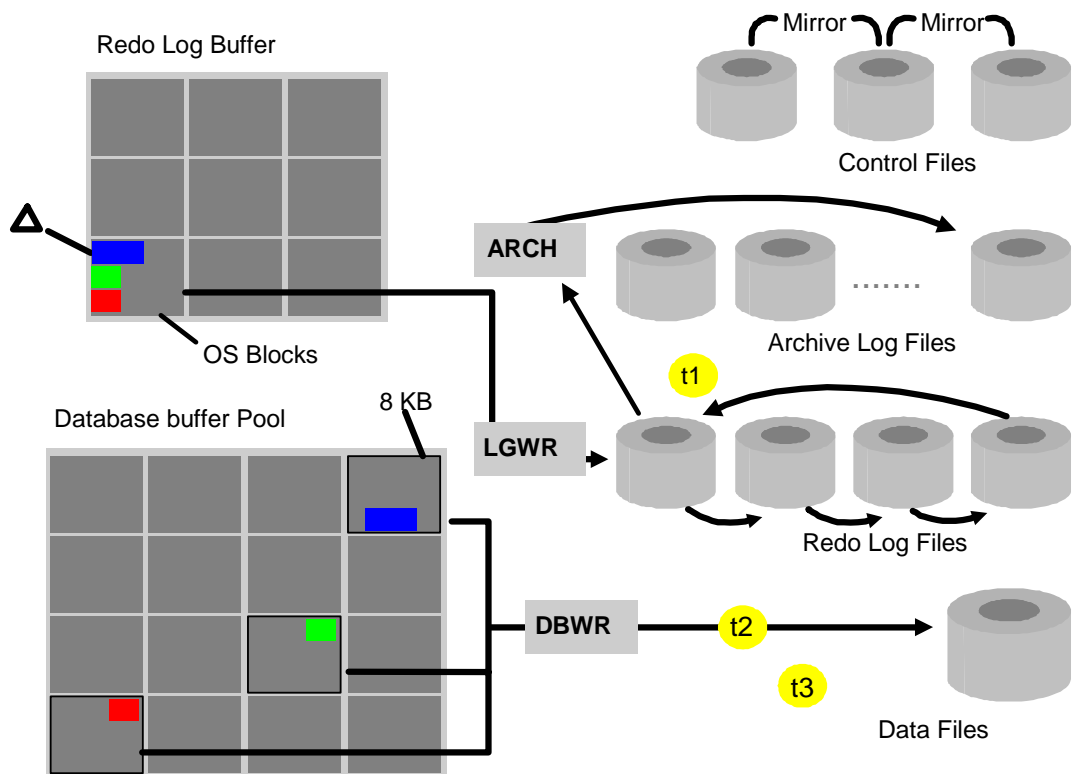
### Process

1. You [set up archiving](#), making sure that the database runs in ARCHIVELOG mode and with automatic archiving enabled.
2. You [mirror the control files](#) on separate disks.
3. You [mirror the online redo log files](#) on separate disks (at operating system level and/or using Oracle resources). With Oracle support, you can set this up when you install the database system.
4. You [organize disk storage](#) by installing online redo log files, offline redo logs files (that is, online redo log files saved to the archive directory), and data files to separate disks.
5. You check the [database parameters](#). You make sure that the block sizes of the database system and the operating system are the same for security and performance reasons. The block size of the operating system is reformatted to 8K during the installation of the SAP system.

The AIX operating system is an exception, because the block size of the database system must be 8 KB and the block size of the operating system must be 4 KB.

6. You follow the [SAP Naming Conventions for Tablespaces](#) and familiarize yourself with the tablespace structure of the SAP System, being sure to monitor the critical tablespaces, as shown in the graphic below.

Structure of the Oracle database



7. You familiarize yourself with environment variables and set them as required:
  - [Environment Variables \(UNIX\)](#)
  - [Environment Variables \(Windows\)](#)
8. You familiarize yourself with the directory structure:
  - [Directory Structure \(UNIX\)](#)
  - [Directory Structure \(Windows\)](#)
9. You familiarize yourself with [users and roles](#).

## Setting Up Archiving

This procedure tells you how to check the archiving parameters and ARCHIVELOG mode for your Oracle database and, if necessary, how to change these.

 Caution

It is important that:

- The database runs in ARCHIVELOG mode
- Automatic archiving is enabled

After correct installation of an SAP system, the Oracle database meets both these criteria. In this case, the online redo log files are automatically archived when full (that is, following a redo log switch). This is important because it allows the online redo log files to be reused for fresh archive data, so that archiving of the log files can continue at all times.

## Prerequisites

The following parameters in the `init<DBSID>.ora` file control the archiving process for the Oracle database:

Parameter	Note
<code>log_archive_start = true</code>	Standard in Oracle 10g
<code>log_archive_dest = &lt;directory&gt;/&lt;file prefix&gt;</code>	Part of the file name
<code>log_archive_format = &lt;Oracle default&gt;</code>	Use the Oracle default

`log_archive_start = true` causes the background archive process ARCH to be started automatically when the database is started. This means that automatic archiving is enabled.

`log_archive_dest` defines the archive directory of the online redo log files for archiving. It is delivered with the specification of the following SAP standard path (this example is for a single instance installation on UNIX):

```
OS> <SAPDATA_HOME>/oraarch/<ORACLE_SID>arch
```

See [SAP Note 316642](#) for information on how to avoid an “archiver stuck” error by changing the archive directory from the previous value `saparch` to `oraarch`.

For more information, see [Environment Variables \(UNIX\)](#) and [Environment Variables \(Windows\)](#).

### Note

The Oracle database names the offline redo log files using the string `<ORACLE_SID>arch` followed by the log sequence number.

You can [back up the offline redo log files with BR\\*Tools](#). The offline redo log files are the copies of the online redo log files saved in the archive directory.




## Procedure

1. Make sure that:

- The archive directory (under UNIX: `oraarch`) exists.
- The directory is not write-protected.
- The directory has enough free space. Otherwise, the archiving process cannot archive any log files and no further actions are possible on the database (this is known as “Archiver Stuck”).

### Note

For more information about the BR\*Tools commands mentioned below, see [Altering the Database Instance with BR\\*Tools](#) and [-f dbalter](#).

2. Check the archiving status of the database by choosing  *Instance Management*  *Show instance status*  in BR\*Tools.




The system displays database instance details.

3. Check that:

- o Archivelog mode is set to ARCHIVELOG
- o Archiver status is set to STARTED

4. If you need to reset ARCHIVELOG mode, do this in one of the following ways:

1. Do one of the following:

- Choose  *Instance Management*  *Alter database instance*  in BRGUI or BRTOOLS and choose the action *Set archivelog mode*.

- Enter the following from the command line:

```
brspace -f dbalter -a archlog
```

- Enter the following commands in the Oracle tool SQLPLUS:

```
SQL> connect / as sysdba
```

```
SQL> startup mount
```

```
SQL> alter database archivelog
```

```
SQL> alter database open
```

```
SQL> archive log list
```

BRSPACE or SQLPLUS reconfigures the database to set ARCHIVELOG mode on.

2. Repeat steps 2 and 3 to check that ARCHIVELOG has been set correctly.

5. If you need to enable automatic archiving (that is, to start the ARCH process), do the following:

0. Enter the following commands in the Oracle tool SQLPLUS:

```
SQL> connect / as sysdba
```

```
SQL> alter system archive log start;
```

```
SQL> archive log list
```

1. Repeat steps 2 and 3 to check that ARCHIVELOG has been set correctly.

## Mirroring the Control File

It is essential to mirror the control file in your Oracle database.

If you only have one copy of the control file and lose this copy due to a disk error or other problems, it is probably impossible to completely recover the database. The result is inevitable data loss.

To prevent this happening, create multiple copies of the control file. When the SAP system is installed, the control file is mirrored to at least two additional disks (often three). You can also mirror the control file yourself.

 Caution

Always make sure that all the control files are on different disks.

## Prerequisites

The default database profile (`init<DBSID>.ora` profile) delivered with the system makes sure that the control file and its mirror copies are stored in directories that are mounted on different disks.

## Procedure

If necessary, change the standard mirroring of the control file by setting the `control_files` parameter.

 Example

Here is a sample entry in profile `init<DBSID>.ora`:

```
control_files = (?/dbs/cntrl<SAPSID>.dbf,  
?/sapdata1/cntrl/ctrl<SAPSID>.dbf, ?/sapdata2/cntrl/ctrl<SAPSID>.dbf)
```

The question mark `?` is the official Oracle placeholder for the home directory of the database system (for example, `/oracle/C11/102_64`).

## Mirroring the Online Redo Log Files

It is essential to mirror the online redo log files in your Oracle database. This procedure describes how you can check the mirroring with `BRSPACE`. Also, the system check in `BRCONNECT` checks mirroring.

If you lose one or more online redo log files, you can no longer recover the database changes recorded in them. This means that you can only recover the database up to the first gap in the online redo log records. For this reason, we strongly recommend that you mirror the online redo log files. We recommend that you make at least one copy.

If an online redo log file of a group is lost, the database remains in operation. Oracle then uses the remaining member (or members) of this group to log the database changes. In such a case, you must recover the original mirroring of the online redo log files as quickly as possible.

For more information see:

- [Database Recovery with SQLPLUS](#)
- [Recovery: One Inactive Redo Log Missing](#)

## Prerequisites

When an SAP System is installed using Oracle resources, an online redo log group normally consists of the original online redo log file and a mirror copy of this file (that is, the group has two members).

## Procedure

1. Do one of the following to check mirroring:
  - o Choose **Space management** **Additional space functions** **Show redo log files** in BRGUI or BRTOOLS.
  - o Enter `brspace -f dbshow -c rfinfo` from the command line.
  - o Call the Oracle program SQLPLUS:

```
SQL> connect / as sysdba

SQL> select * from v$logfile;
```
  - o Run the [system check with BRCONNECT](#). The condition to check mirroring of the online redo log files is `REDOLOG_FILE_MIRROR`. For more information, see [BRCONNECT Default Conditions for Database Administration](#).
2. You can make additional mirror copies. For more information, see the appropriate Oracle documentation. In addition, many systems support hardware-based file mirroring. For more information, see your operating system documentation.

For more information on the security of offline redo log files, see [Backing Up the Offline Redo Log Files with BR\\*Tools](#).



## Organizing Disk Storage

We recommend that you store the files of the database system on different physical disks. In this example, the control file is mirrored twice:

### Example

Disk Number	Directory	Contents of Directory
1	origlogA	Online redo log files from the first and third group (Set A)
2	origlogB	Online redo log files from the second and fourth group (Set B)
3	mirrlogA	Mirrored online redo log files from the first and third group (Set A)
4	mirrlogB	Mirrored online redo log files from the second and fourth group (Set B)
5	sapdata1	Database files, mirror of the control file
6	sapdata2	Database files, mirror of the control file
7	sapdata<n>	Other database files in <code>sapdata3</code> up to <code>sapdata&lt;n&gt;</code> , each on

Disk Number	Directory	Contents of Directory
	>	separate disks when possible

 Note

A disk assignment similar to this is essential for reliable database operation. For performance reasons, we recommend you to distribute the online redo log groups to four disks, as shown above.

## Procedure

1. Make sure that the number and descriptions of the control file and its mirrors agree with the entry in the `init<DBSID>.ora` profile.
2. The database files can be distributed across any number of disks.
3. Make sure that the offline redo log files (that is, online redo log files saved to the archive directory) are not stored on the same disk as the online redo log files.
4. The offline redo log files are not shown in the above example.

## SAP Naming Conventions for Tablespaces and Data Files

In SAP systems with the Oracle database, tablespaces and data files are named according to the conventions described in this section.

- You might need to [create a new tablespace](#), for example, in the following situations:
  - During the repository switch in a SAP system upgrade
  - When moving a table to a separate tablespace (for example, for administrative reasons)
- For more information on extending a tablespace, see [Extending a Tablespace with BR\\*Tools](#).

 Recommendation

We strongly recommend you to create new tablespaces and add data files to extend existing ones in accordance with the SAP naming conventions (see *Structure* below).

## Structure

 Note

There is a new convention for naming tablespaces. This is generally valid for new installations or new tablespaces as of SAP Web Application Server 6.10.

However, the new naming convention is also used if you have Multiple Components in One Database (MCOB) for SAP 4.6C and SAP 4.6D.

## Tablespace Naming Convention as of SAP Web Application Server 6.10

The following table contains an overview of all the tablespaces in an SAP system and how they are used:

Tablespace Name	Use
SYSTEM	Oracle system tablespace
SYSAUX	Oracle auxiliary system tablespace
PSAPTEMP	Temporary objects (system default temporary tablespace)
PSAPTEMP<name>	Additional temporary objects, if required. Example: PSAPTEMP2 .
PSAPUNDO	Undo tablespace (used from Oracle 9i instead of PSAPROLL)
PSAP<SID>	All objects of the SAP component <SID>. Example: PSAPC11.
PSAP<SID>ES<rel> [X] or PSAP<SID>EL<rel> [X]	Exchange tablespace for component <SID> upgrade. Example: PSAPC11ES649.
PSAP<SID>USR	Customer-specific objects for component <SID>. Example: PSAPC11USR.
PSAP<SID><name>	Additional customer-specific objects for component <SID>. Example: PSAPC11DAT.

The association of objects to PSAP<SID>, PSAP<SID>USR, or PSAP<SID><name> is controlled using the TABART in DD09L, TAORA, and IAORA. Exchange tablespaces have no TABART.

## Tablespace Naming Convention Before SAP Web Application Server 6.10

The following syntax is used for naming tablespaces (TSP):

- PSAP<name>D for data tablespaces
- PSAP<name>I for index tablespaces
- PSAP<TSP> if it is not important to differentiate

The following table contains an overview of all the tablespaces in an SAP system and how they are used:

Tablespace name	Use
Oracle Tablespaces	These tablespaces are required for operation of the Oracle DBMS, and contain no SAP data.



<b>Tablespace name</b>	<b>Use</b>
SYSTEM	Oracle system tablespace
SYSAUX	Oracle auxiliary system tablespace
PSAPROLL	Rollback segments
PSAPTEMP	Sort processes
<b>SAP Netweaver tablespaces</b>	
PSAPLOADD/I	Screen and report loads (ABAP)
PSAPSOURCED/I	Screen and report sources (ABAP)
PSAPDDICD/I	ABAP Dictionary
PSAPPROTD/I	Log-like tables (such as spool)
PSAPEL46D/I	Exchange tablespace loads
PSAPES46DD/I	Exchange tablespace sources
<b>Application</b>	
PSAPCLUD/I	Cluster tables
PSAPPOOLD/I	Pooled tables (such as ATAB)
PSAPSTABD/I	Master data, transparent tables
PSAPBTABD/I	Transaction data, transparent tables
PSAPDOCUD/I	Doc., Sapscript, Sapfind
<b>Customers</b>	
PSAPUSER1D/I	Customer tables

## Data Files

When you add a new data file to extend a tablespace, BRSPACE attempts to add a new file to the standard SAP directory in which the most recent data file of the tablespace was stored. The most recent file is the one with the highest relative file number.

The naming conventions are as follows:

- **Directory:** <SAPDATA\_HOME>/sapdata<n>/<tablespace suffix>\_<file number>
- **File:** <tablespace suffix>.data<file number>

<tablespace suffix> is the second half of the tablespace name, for example, UNDO for the tablespace PSAPUNDO.

<n> is the sequentially assigned number of the SAP directory in which the data file will be stored.

<file number> is the sequentially assigned number of the data file in the tablespace. The same number also appears in the subdirectory that is created for each new file in a tablespace.

For example, the first data file for PSAPUNDO is called:

- Directory: <SAPDATA\_HOME>/sapdata<n>/undo\_1
- File: undo.data1

For more information on <SAPDATA\_HOME>, see:

- [Environment Variables \(UNIX\)](#)
- [Environment Variables \(Windows\)](#)



## Database Parameters

The SAP system for the Oracle database comes with a standard initialization profile for the database parameters:

- UNIX  

```
OS> <ORACLE_HOME>/dbs/init<DBSID>.ora
```
- Windows  

```
OS> <ORACLE_HOME>\database\init<DBSID>.ora
```

For example (UNIX):

```
/oracle/C11/dbs/initC11.ora
```

This profile contains the default parameter settings recommended for the SAP system. Copy this standard profile so that you can access the original parameters, if necessary.



## Environment Variables (UNIX)

Environment variables define values used by the Oracle database and BR\*Tools. This section describes the variables used when the operating system is UNIX.

The database uses the environment values for many different purposes. BR\*Tools also use the values.

## Structure

The following variables are required:

- ORACLE\_SID

System ID of the database instance

Example: C11

SAPSID or `sapsid` refers to the SAP System ID.

DBSID or `dbsid` refers to the name of the database instance (database instance system ID).

When a single instance is installed, SAPSID and DBSID are the same.

- ORACLE\_HOME

Home directory of the Oracle software.

Standard: `/oracle/<DBSID>/<database version>`

- SAPDATA\_HOME

Directory of the database files.

Standard: `/oracle/<DBSID>`

 Note

The variables ORACLE\_SID, ORACLE\_HOME and SAPDATA\_HOME must always be set. There is no default.

The following environment variables must only be set if the corresponding paths deviate from the defaults specified here:

- SAPARCH

Directory for the BRARCHIVE logs.

Default value: `$$SAPDATA_HOME/saparch`

- SAPBACKUP

Directory for the BRBACKUP, BRRESTORE, and BRRECOVER logs.

Default value: `$$SAPDATA_HOME/sapbackup`

- SAPCHECK

Directory for the BRCONNECT logs.

Default value: `$$SAPDATA_HOME/sapcheck`

- SAPREORG

Directory for the BRSPACE logs. It is also the standard directory for export dump files, if the parameter `exp_dump_dir` in the profile `init<DBSID>.sap` is not set.

Default value: `$$SAPDATA_HOME/sapreorg`

- SAPTRACE

Directory for Oracle trace files and the alert file.

Default value: `$SAPDATA_HOME/saptrace`

- `SAPDATA1`

Directory of the database data files.

Default value: `$SAPDATA_HOME/sapdata1`

(The same for `SAPDATA<n>`, `n=1, . . . 99`).

The environment variables `SAPDATA<n>` must only be defined if directories are used that differ from the default.

- `TWO_TASK`

Identification of a remote database system.

This environment variable must be left unset.

Other environment variables that you can set for BR\*Tools:

- `BR_LINES`

Definition of the number of lines in list menus.

Recommended height: greater than or equal to 20 lines 20 lines.

For more information, see [Configuring the Scroll Line Count for BR\\*Tools](#).

- `BR_LANG`

Definition of the message language:

- E: English
- D: German

- `BR_TRACE`

Setting the trace function for error analysis. For more information, see SAP Note 29321.

## More Information

[Environment Variables \(Windows\)](#)



## Environment Variables (Windows)

Environment variables define parameter values used by the Oracle database and BR\*Tools. This section describes the variables used when the operating system is Windows.

The database uses the parameter values for many different purposes. BR\*Tools also uses the parameter values.

## Structure

The following variables are required:

- ORACLE\_SID

System ID of the database instance

Example: C11

SAPSID or sapsid refers to the SAP System ID.

DBSID or dbsid refers to the name of the database instance (database instance system ID).

When a single instance is installed, SAPSID and DBSID are the same.

- ORACLE\_HOME

Home directory of the Oracle software.

Standard: <drive>:\orant\<>database version>

Example: D:\orant\ora102

- SAPDATA\_HOME

Directory of the database files.

Standard: <drive>:\oracle\<>DBSID>

Example: E:\oracle\C11



### Note

The variables ORACLE\_SID and SAPDATA\_HOME must always be set. There is no default.

The following environment variables must only be set if the corresponding paths deviate from the defaults specified here:

- SAPARCH

Directory for the BRARCHIVE logs.

Default value: %SAPDATA\_HOME%\saparch

- SAPBACKUP

Directory for the logs from BRBACKUP, BRRESTORE, and BRRECOVER.

Default value: %SAPDATA\_HOME%\sapbackup

- SAPCHECK

Directory for the BRCONNECT logs.

Default value: %SAPDATA\_HOME%\sapcheck

- SAPREORG

Directory for the BRSPACE logs. It is also the standard directory for export dump files, if the parameter `exp_dump_dir` in the profile `init<DBSID>.sap` is not set.

Default value: `%SAPDATA_HOME%\sapreorg`

- SAPTRACE

Directory for Oracle trace files and the alert file.

Default value: `%SAPDATA_HOME%\saptrace`

- SAPDATA1

Directory of the database data files.

Default value: `%SAPDATA_HOME%\sapdata1`

(The same for `SAPDATA<n>`, `n=1, . . . 99`).

#### Note

You can distribute the `SAPDATA` directories across several different drives, without defining the environment variables `SAPDATA<n>`. The environment variables only have to be defined individually if directory names are used that deviate from the default.

For example: `SAPDATA1= F:\data\prod\sapdata1`.

Other environment variables that you can set for BR\*Tools:

- BR\_LINES

Definition of the number of lines in list menus.

Recommended height: greater than or equal to 20 lines.

For more information, see [Configuring the Scroll Line Count for BR\\*Tools](#).

- BR\_LANG

Definition of the message language:

- E: English
- D: German

- BR\_TRACE

Setting the trace function for error analysis. For more information, see SAP Note 29321.

## More Information

[Environment Variables \(UNIX\)](#)



## Directory Structure (UNIX)

This section describes the directory structure for the Oracle database with the UNIX operating system.

The directories contain a range of files such as profiles, log files, scripts, executables, and so on.

The following conventions apply to this section:

Term	Meaning
\$<name>	Environment variable
<DBSID>	\$ORACLE_SID
<TSP>	Tablespace short name For example, BTABD for the tablespace PSAPBTABD.

## Structure

There are the following main structures:

- \$ORACLE\_HOME for Oracle-specific objects with the default directory  
/oracle/<DBSID>/<Oracle version>
- \$SAPDATA\_HOME for SAP-specific objects with the default directory  
/oracle/<DBSID>
- Executables with the directory /usr/sap/<SAPSID>/SYS/exe/run

## Oracle Home Directory

This directory contains the following subdirectories:

- dbs/
  - o spfile<DBSID>.ora: Oracle spfile
  - o init<DBSID>.ora: Profile for Oracle
  - o init<DBSID>.sap: Profile for BR\*Tools
  - o cntrl<DBSID>.dbf: Database control file
- bin/ contains binaries such as oracle, sqlplus, exp, imp  
rwsr-xr-x ora<dbsid> dba oracle

## SAP Home Directory

For the files starting log\_, g means group, and m means member:

- origlogA/
  - o log\_g11m1.dbf

- o log\_g13m1.dbf
- origlogB/
  - o log\_g12m1.dbf
  - o log\_g14m1.dbf

Mirrored redo logs are optional - although we strongly recommend using them - so the specification for `mirrlogA` and `mirrlogB` might vary:

- mirrlogA/
  - o log\_g11m2.dbf
  - o log\_g13m2.dbf
- mirrlogB/
  - o log\_g12m2.dbf
  - o log\_g14m2.dbf
- sapdata1/
  - o cntrl/cntrl<DBSID>.dbf: Database control file
  - o system\_1/system.data1: First SYSTEM tablespace file
  - o btabd\_1/btabd.data1: Example of a SAP data file for tablespace PSAPBTABD
  - o ...
- sapdata2/
  - o cntrl/cntrl<DBSID>.dbf: Database control file
  - o system\_2/system.data2
  - o btabi\_1/btabi.data1 Example of a SAP data file for tablespace PSAPBTABI
  - o ...
- sapdata<n>/
- saparch/
  - o arch<DBSID>.log BRARCHIVE summary log
  - BRARCHIVE detail logs:
    - o <encoded timestamp>.sve: Original saved
    - o <encoded timestamp>.svd: Original saved and deleted
    - o <encoded timestamp>.cpy: Original copied
    - o <encoded timestamp>.cpd: Original copied and deleted
    - o <encoded timestamp>.dsv: Deleted, were saved once



- o <encoded timestamp>.dcp: Deleted, were saved twice
  - o <encoded timestamp>.ssv: Parallel saved on two stations
  - o <encoded timestamp>.ssd: Parallel saved on two stations and deleted
  - o <encoded timestamp>.cps: Copy and save
  - o <encoded timestamp>.cds: Copy, delete and save
  - o <encoded timestamp>.qua: Query which tapes to be used
  - o <encoded timestamp>.cma: Determination of software compression rate
  - o <encoded timestamp>.tia: Tape initialization
  - o <encoded timestamp>.fst: Stop archiving using brarchive -f stop
  - o <encoded timestamp>.vra: Offline redo log file verification with RMAN
  - o <encoded timestamp>.aab: Abort archiving with brarchive -g|-abort
- sapbackup/
    - o back<DBSID>.log: BRBACKUP summary log
    - o <encoded timestamp>.xyz: BRBACKUP detail log, where:
      - x = a (whole, previously all), p (partial) , f (full) , i (incremental)
      - y = n (online) or f (offline)
      - z = t (tape), p (pipe), d (disk), f (util\_file), v, (util\_vol), r (RMAN), s (remote disk, stage),
      - xyz = qub: Query for which tapes are to be used
      - xyz = cmb: Determination of software compression rate
      - xyz = tib: Tape initialization
      - xyz = rmp: RMAN preparation run
      - xyz = dbv: Database verification with DBVERIFY
      - xyz = ddb: Delete of disk backup
      - xyz = bab: Abort backup with brbackup -g|-abort
    - o <encoded timestamp>.xyz: BRRESTORE detail log, where:
      - xyz = rsb : Restore backup files
      - xyz = rsa: Restore archive files
      - xyz = rsf: Restore individual files
      - xyz = qur: Query which tapes to be used
      - xyz = rab: Abort restore with brrestore -g|-abort

- o <encoded timestamp>/: Disk backups
  - Copies of database files
- o <DBSID>/: Copies of profiles and log files
  - spfile<DBSID>.ora
  - init<DBSID>.ora
  - init<DBSID>.sap
  - back<DBSID>.log Summary log
  - <encoded timestamp>.xyz: Detail log
- o [BRRECOVER log files](#)
- sapcheck\
  - o <encoded timestamp>.sta: log of [brconnect -f stats](#)
  - o <encoded timestamp>.chk: log of [brconnect -f check](#)
  - o <encoded timestamp>.nxt: log of [brconnect -f next](#)
  - o <encoded timestamp>.cln: log of [brconnect -f cleanup](#)
- sapreorg/
  - o [BRSPACE logs](#)
  - o BRSPACE writes scripts, parameter files, and restart files as follows to the directory:
    - <encoded datestamp>/ddl.sql: Data Definition Language (DDL) statements for [table reorganization](#)
    - <encoded datestamp>.edd/expdat.dmp: export dump file for [table export](#)
    - <encoded datestamp>/parfile.exp: parameter file for [table export](#)
- saptrace/
  - o background/
    - alert\_<DBSID>.log: Oracle Alert file
    - \*.trc: Oracle trace files
  - o usertrace/
    - \*.trc: User trace files
- oraarch/: Oracle offline redo log files

## DBA Executables in Directory /usr/sap/<SAPSID>/SYS/exe/run

- brarchive

- brbackup
- brconnect
- brrecover
- brrestore
- brspace
- brtools

## More Information

[Directory Structure \(Windows\)](#)



## Directory Structure (Windows)

This section describes the directory structure for the Oracle database with the Windows operating system.

The directories contain a range of files such as profiles, log files, scripts, executables, and so on.

The following conventions apply to this section:

Term	Meaning
%<name>%	Environment variable
<DBSID>	%%ORACLE_DBSID%
<TSP>	<p>Tablespace short name</p> <p>For example, BTABD for the tablespace PSAPBTABD.</p>

## Structure

There are the following main structures:

- %ORACLE\_HOME% for Oracle-specific objects with the default directory  
:\oracle\<DBSID>
- %SAPDATA\_HOME% for SAP-specific objects with the default directory  
:\oracle\<DBSID>
- Executables with the directory x:\usr\sap\<DBSID>\SYS\exe\run

## Oracle Home Directory

This directory contains the following subdirectories:

- database\
  - spfile<DBSID>.ora: Oracle spfile
  - init<DBSID>.ora: Profile for Oracle
  - init<DBSID>.sap.: Profile for BR\*Tools
  - cntrl<DBSID>.dbf: Control file
- bin\
 

Contains binaries such as sqlplus, exp, imp, and so on

## SAP Home Directory

For the files starting log\_, g means group and m means member:

- origlogA\
  - log\_tg101m1.dbf
  - log\_tg103m1.dbf
- origlogB\
  - log\_tg102m1.dbf
  - log\_tg104m1.dbf

Mirrored redo logs are optional – although we strongly recommend using them – so the specification for mirrlogA and mirrlogB might vary:

- mirrlogA\
  - log\_tg101m2.dbf
  - log\_tg103m2.dbf
- mirrlogB\
  - log\_tg102m2.dbf
  - log\_tg104m2.dbf
- sapdata1\
  - cntrl\cntrl<DBSID>.dbf: Control file
  - system\_1\system.data1: SYSTEM tablespace file
  - btabd\_1\btabd.data1: Example of a SAP data file for tablespace PSAPBTABD
  - ...
- sapdata2\
  - cntrl\cntrl<DBSID>.dbf: Control file
  - btabi\_1\btabi.data1: Example of a SAP data file for tablespace PSAPBTABI

- sapdata<n>\
- saparch\
  - arch<DBSID>.log: BRARCHIVE summary log

**BRARCHIVE detail logs:**

- <coded timestamp>.sve: Original saved
  - <coded timestamp>.svd: Original saved and deleted
  - <coded timestamp>.cpy: Original copied
  - <coded timestamp>.cpd: Original copied and deleted
  - <coded timestamp>.dsv: Deleted, were saved once
  - <coded timestamp>.dcp: Deleted, were saved twice
  - <coded timestamp>.ssv: Parallel saved on two stations
  - <coded timestamp>.ssd: Parallel saved on two stations and deleted
  - <coded timestamp>.cps: Copy and save
  - <coded timestamp>.cds: Copy, delete and save
  - <coded timestamp>.qua: Query which tapes to be used
  - <coded timestamp>.cma: Determination of software compression rate
  - <coded timestamp>.tia: Tape initialization
  - <coded timestamp>.fst: Stop archiving using brarchive -f stop
  - <encoded timestamp>.vra: Offline redo log file verification with RMAN
  - <encoded timestamp>.aab: Abort archiving with brarchive -g|-abort
- sapbackup\
    - back<DBSID>.log: BRBACKUP summary log
    - rest<DBSID>.log: BRRESTORE summary log
    - <coded timestamp>.xyz: BRBACKUP detail log, where:
      - x = a (whole, previously all), p (partial) , f (full), , i (incremental)
      - y = n (online) or f (offline)
      - z = t (tape), p (pipe), d (disk), f (util\_file), v, (util\_vol), r (RMAN), s (remote disk, stage)
      - xyz = qub: Query for which tapes are to be used
      - xyz = cmb: Determination of software compression rate
      - xyz = tib: Tape initialization

- xyz = rmp: RMAN preparation run
    - xyz = dbv: Database verification with DBVERIFY
    - xyz = ddb: Delete of disk backup
    - xyz = bab: Abort backup with brbackup -g|-abort
  - <coded timestamp>.xyz: BRRESTORE detail log, where:
    - xyz = rsb: Restore backup files
    - xyz = rsa: Restore archive files
    - xyz = rsf: Restore individual files
    - xyz = qur: Query which tapes to be used
    - xyz = rab: Abort restore with brrestore -g|-abort
  - <coded timestamp>\: Disk backups
    - spfile<DBSID>.ora
        - init<DBSID>.ora
        - init<DBSID>.sap
        - back<DBSID>.log: Summary log
        - <coded timestamp>.xyz: Detail log
      - [BRRECOVER log files](#)
- sapcheck/
  - <encoded timestamp>.sta: log of [brconnect -f stats](#)
  - <encoded timestamp>.chk: log of [brconnect -f check](#)
  - <encoded timestamp>.nxt: log of [brconnect -f next](#)
  - <encoded timestamp>.cln: log of [brconnect -f cleanup](#)
- sapreorg\
  - [BRSPACE logs](#)
  - BRSPACE writes scripts, parameter files, and restart files as follows to the directory:
    - <encoded datestamp>/ddl.sql: Data Definition Language (DDL) statements for [table reorganization](#)
    - <encoded datestamp>.edd/expdat.dmp: export dump file for [table export](#)

- `<encoded datestamp>/parfile.exp`: parameter file for [table export](#)
- `saptrace\`
  - o `background\`
    - `alert_<DBSID>.log`: Oracle Alert file
    - `*.trc`: Oracle trace files
  - o `usertrace\`
    - `*.trc`: User trace files
- `oraarch`: Oracle offline redo log files

### Executables in Directory `<drive>:\usr\sap\<SAPSID>\SYS\exe\run`

- `brarchive.exe`
- `brbackup.exe`
- `brconnect.exe`
- `brrecover.exe`
- `brrestore.exe`
- `brspace.exe`
- `brtools.exe`
- `mkszip.exe`
- `uncompress.exe`
- `cpio.exe`
- `mt.exe`
- `dd.exe`

## More Information

[Directory Structure \(UNIX\)](#)



## Users and Roles

### Operating System Users

In the SAP system the roles of the users `ora<dbsid>` and `<sapsid>adm` on UNIX, or `<sapsid>adm` and `SAPSERVICE<SID>` on Windows, used to be separate. Due to the requirements for RMAN backup, this is no longer true. Both users now belong to the operating system groups `dba` and `oper`, as shown in the tables below.

## Database Roles

- **SYSDBA**  
All authorizations
- **SYSOPER**  
Operator activities, but no read or write authorizations.
- **SAPDBA**

Read and write authorizations to work with BR\*Tools command options, and therefore the DBA functions in the Computer Center Management System (CCMS).

To be able to use the CCMS DBA functions or BR\*Tools command options without restrictions, the OPS\$ user must have both the SYSOPER role and the SAPDBA role.

UNIX			
Operating System Users	Operating System Group	Database Role	Database Users
ora<dbSID>	dba oper	SYSDBA SYSOPER	OPS\$ORA<DBSID>
<sapsid>adm	dba oper	SYSDBA SYSOPER	OPS\$<SAPSID>ADM

Windows			
Operating System Users	Operating System Group	Database Role	Database Users
<sapsid>adm	ORA_<SID>_DBA ORA_<SID>_OPER	SYSDBA SYSOPER	(SYS) OPS\$<DOMAIN>\<SAPSID>ADM
SAPSERVICE<SID>	ORA_<SID>_DBA ORA_<SID>_OPER	SYSDBA SYSOPER	OPS\$<DOMAIN>\SAPSERVICE<SID>

### Note

The OS group on Windows can also be specified globally (without instance name) (ORA\_DB, ORA\_OPER).

## OPS\$ Database User

The Oracle OPS\$ mechanism moves the entire DB security mechanism to the operating system level.

The prerequisite is that a DB user OPS\$<OS\_user> corresponding to the OS user is defined on the database, and identified as externally. It must have been granted the SAPDBA role.



Once you have logged on successfully with the OS user, you can connect to the database with:

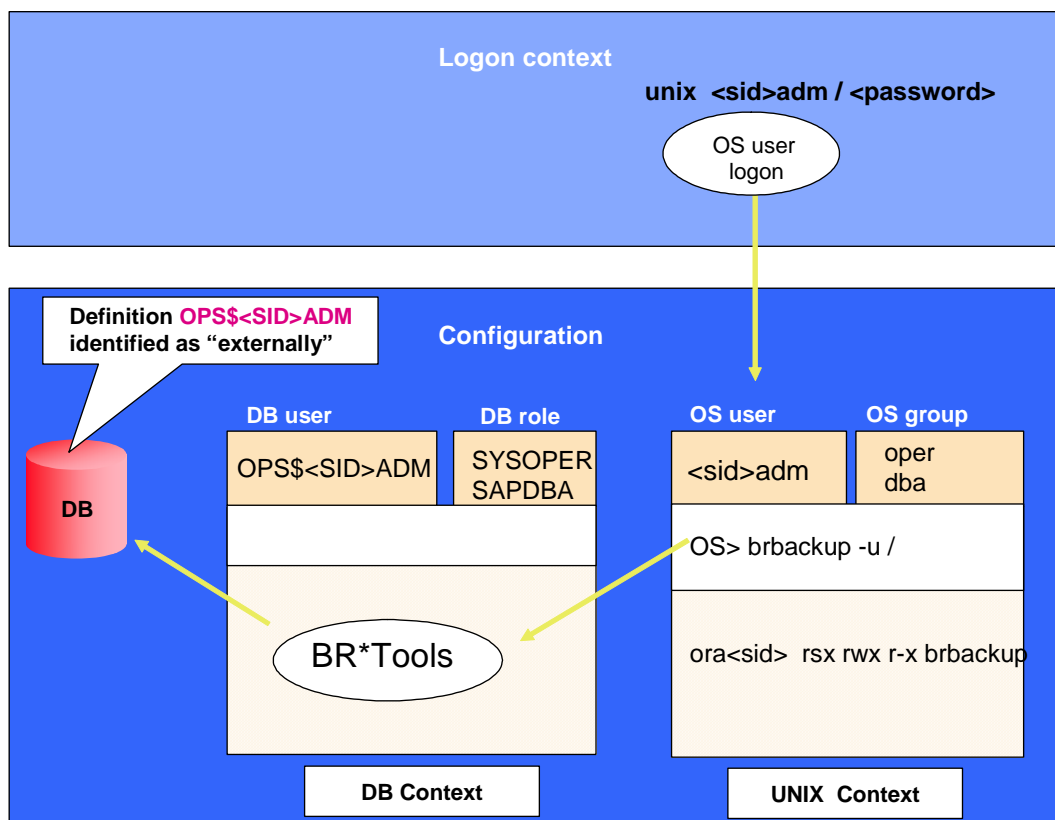
```
SQL> connect /
```

This means you do not have to enter another password. You are then working as OPS\$<OS\_user>. In the same way you can start the program BR\*Tools with:

```
OS> brbackup -u /
```

This OPS\$ mechanism is always used if you call BR\*Tools from the CCMS transaction DB13 in the SAP system.

The OPS\$ Mechanism (UNIX)



## BR\*Tools Database User

The standard DB user used by BR\*Tools is always `SYSTEM`. BR\*Tools connects with the Oracle option `AS SYSOPER` or `AS SYSDBA` for actions such as startup, shutdown, recover, and so on, as well as selecting from V\$ tables when the database is not open.



## Oracle Databases on Raw Devices

You can operate an Oracle database on raw devices. This means that the Oracle database management system avoids the file management of the UNIX system and instead writes data directly to a disk partition. Therefore, the partition is used as a raw device and does not contain a file system. This type of storage improves the speed of data access, but requires its own file management.

The BR\*Tools support raw devices:

- [Raw Devices and BR\\*Tools](#)
- [Raw Devices with BRBACKUP and BRRESTORE.](#)

 Note

Raw devices are only used with the UNIX operating system, not with Windows NT.

## Features

There are the following advantages and disadvantages when using raw devices.

### Advantages

Working with raw devices improves performance for the following reasons:

- Since the usual buffer cache for a file system is not needed, faster data access is possible and less main memory is required.
- You do not have to administer a file system.
- Since no management information has to be stored on the disk, less disk space is required.
- Since the accesses do not have to be synchronized and the management information does not have to be recorded, the load on the CPU is reduced.

### Disadvantages

Working with raw devices makes administration more difficult for the following reasons:

- There is no description of the files residing on the raw devices in the system.
- The configuration of the storage space is inflexible because only one database file is permitted for each raw device (and therefore for each partition). The sizes of the individual partitions must be adjusted to the sizes of the database files. This makes later relocation of the database files to other partitions more difficult.
- It is possible to save raw devices with the `dd` command, but be aware of the disadvantages (for example, no end-of-media handling).



## Raw Devices and BR\*Tools

The following information is important if you intend to use BR\*Tools with raw devices:

- Be sure to observe the SAP naming convention for tablespaces on raw devices:

```
<SAPDATA_HOME>/sapraw/<TSP>_<Number> -> <Raw-Device-Dir>/<Device>
```

This is composed as follows:

- <Raw-Device-Dir> specifies the central directory containing the devices.
- <Device> consists of <DBSID><TSP>\_<Number>.

- o `<Number>` is the sequence number assigned to the raw devices (or files) belonging to the tablespace.

See also [Environment Variables \(UNIX\)](#).

Each tablespace “file” (held on a raw device) visible to Oracle is a symbolic link to a raw device. BR\*Tools checks that the naming convention has been observed.

#### Example

Tablespace PSAPDOCUD

```
/oracle/C11/sapraw/docud_1 -> /dev/rdisk/C11docud_1
```

Compare this with the [SAP naming conventions for tablespaces](#) in the file system.

- The database link structure is recorded in the [structure log](#) `struc<DBSID>.log` for each new file added to the database using BRSPACE.  
  
If the database is recovered using BRRECOVER, BRRECOVER uses this structure log to check whether the link structure is still complete and immediately repairs it if not. If a tablespace is extended or a new tablespace is created or dropped, BRSPACE updates the structure log.
- Each raw device can contain only one tablespace file, because this is an Oracle requirement. BRSPACE can determine the size of this partition. Therefore, in the case of [tablespace extension](#) (adding a file on a raw device), for example, the size of the file is checked automatically if it fits the raw partition.



## Raw Devices with BRBACKUP and BRRESTORE

BRARCHIVE remains unchanged in a raw device configuration because offline redo log files must always reside in a file system.

There are some changes in the BRBACKUP and BRRESTORE programs in a raw device configuration, but these changes have no effect on the functional scope of the programs. The known functionality of BRBACKUP and BRRESTORE for backing up and restoring file systems remains unchanged.

BRBACKUP and BRRESTORE use the `dd` command to access raw devices:

- With `dd`, you can write directly from the raw device to tape or to the raw device from tape (that is, backup to tape or restore from tape) and you can do the same to or from disk.
- For a backup with software compression, `dd` output is sent directly to the `compress`. For a restore with decompression, output of `uncompress` is sent to `dd`.
- For a backup on a remote computer, `dd` output is sent, for example, directly to `rsh` (`/remsh` and so on). For a restore from a remote computer, it is sent from `rsh` (`/remsh` and so on) to the `dd` command.
- You can define options for `dd` command using the `init<DBSID>.sap` parameters [dd flags](#) and [dd\\_in flags](#).

#### Note

Since the `dd` command does not support a `dd` continuation tape (in this case an I/O error is reported), a method which is similar to the `cpio` continuation method is not supported. This means that each individual database file residing on the raw devices must completely fit onto one tape. This restriction does not refer to the BRBACKUP continuation tape management, which means that the BRBACKUP utility can request continuation tapes if they are necessary for backup of the next database files.

## Limitations of the Oracle Database System

The Oracle database has the limitations described in this section. The parameters `MAXEXTENTS`, `DBFILES`, and `MAXDATAFILES` are discussed.

You need to be aware of the limitations described below when you are:

- [Reorganizing Tables with BR\\*Tools](#)
- [Extending a Tablespace with BR\\*Tools](#)

If you use locally managed tablespaces (LMTS) with `autoallocate`, you can avoid the situation where your tables have a large number of small extents.

### Structure

- Maximum number of extents per table or index - `MAXEXTENTS`

For older installations with dictionary managed tablespaces, we recommend setting `MAXEXTENTS` to `UNLIMITED`.

For new installations with LMTS, the `autoallocate` feature optimizes extent growth for tablespaces.

- Maximum number of files per database - `DB_FILES`
  - Soft limit

The SAP software value for `DB_FILES` is 254.

The database system only supports a specific number of data files in the database, depending on the host system, and this is specified by the `DB_FILES` parameter in the `init<DBSID>.ora` profile. If your database approaches this limit, you can reduce the number of data files by reorganizing tablespaces that have more than one file. However, this is not likely to occur with a limit of 254 files for not very large databases.

- Hard limit

The hard limit for `DB_FILES` depends on the operating system but is usually 1022 per tablespace and 65533 per database.

`DB_FILES` can be increased to the value of `MAXDATAFILES`, the value of which was specified when the database was created. `MAXDATAFILES` itself must be less than the permissible maximum number of open files supported by the operating system. The default value for `MAXDATAFILES` is also 254.

 Caution

Do *not* regularly reorganize the database to reduce the number of data files. This maximum possible number of data files is large, so is not normally reached.

## Approach to Oracle DBA

This section helps you to work out how to *approach* database administration (DBA) with the Oracle database.

For more information on how to *perform* Oracle DBA with the tools supplied by SAP, see [Tools for Oracle DBA](#).

### Prerequisites

You have already [started Oracle DBA with the SAP System](#).

### Process

You review the following topics:

- [Instance Management](#)
- [Space Management](#)
- [Segment Management](#)
- [Database Backup](#)
- [Restore and Recovery](#)
- [Database System Check](#)
- [Update Statistics](#)

## Instance Management

This section helps you develop an approach to managing your database instance or – if you have an Oracle Real Application Cluster (RAC) – your instances.

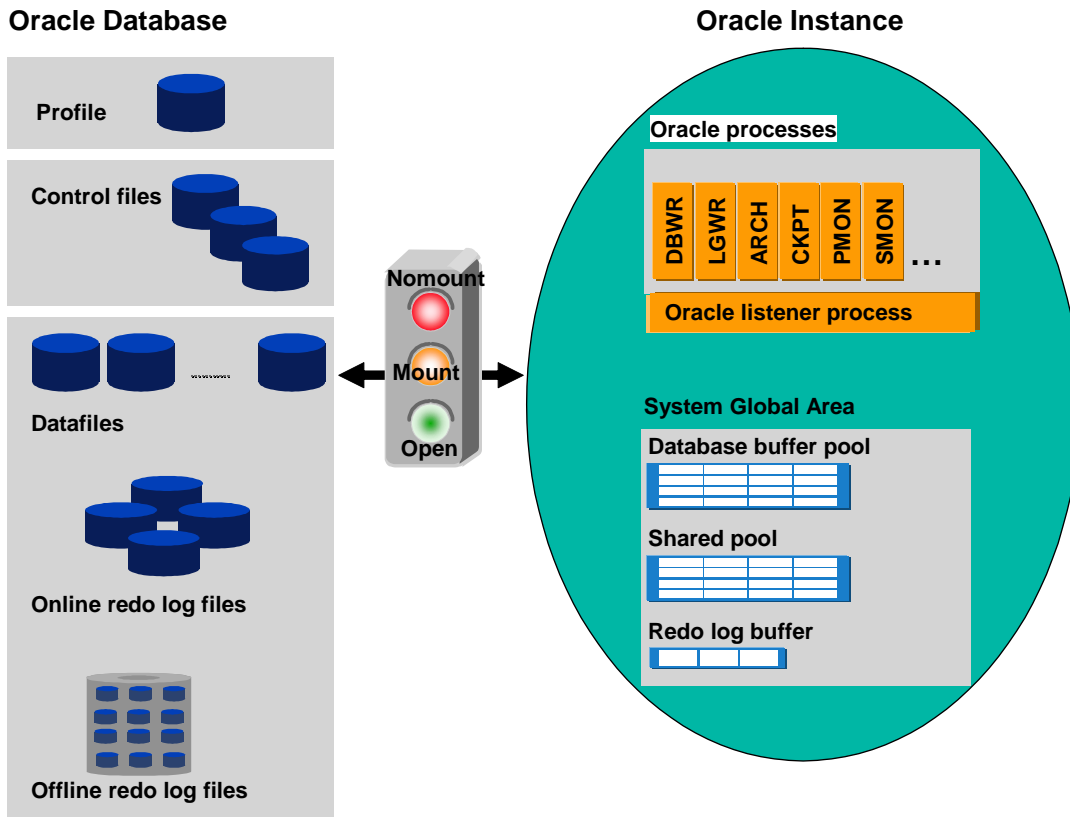
 Note

This section discusses how to approach instance management.

For more information on how to perform instance management, see [Database Instance Management with BR\\*Tools](#).

### Prerequisites

The main components of the database and instance are shown in the following graphic:



When an Oracle database is running, it is associated with an instance. The process of associating the database with an instance is called mounting the database. To make the mounted database accessible to authorized users, you must open it.

## Features

### Starting Up the Database

You can [start up the database](#) as follows:

Type	What Happens	How
No mount	Database instance is built up Operating system resources are allocated using configuration information stored in the profile <code>init&lt;DBSID&gt;.ora</code> or the <code>spfile</code> .	<a href="#">brspace -f dbstart -s nomount</a>
Mount	Database control files are evaluated. Information about the file structure of the database is read Data files and logs are not yet opened.	<a href="#">brspace -f dbstart -s mount</a>
Open	All files in the database system are opened. If required, instance recovery is performed immediately after opening the database. Pending database transactions are ended.	<a href="#">brspace -f dbstart -s open</a>

 Note

If you have an Oracle Real Application Cluster (RAC), you can start up all instances that are currently down with the `all_down` parameter:

[brspace -f dbstart -i all\\_down](#)

## Shutting Down the Database

You can [shut down the database](#) as follows:

Type	What Happens	How
Normal	No new database logon possible.  After all database user have logged off, the database is closed properly: all files are closed, the database is dismounted, and the instance is shut down.  The database is consistent after shutdown.	<a href="#">brspace -f dbshut -m normal</a>
Immediate	Only the current commands are executed.  PMON ends all sessions and performs a rollback of the open transactions.  The database is then closed properly (as for a normal shutdown). The database is consistent after shutdown.  DBWR and ARCH might require up to 1 hour post-processing time.	<a href="#">brspace -f dbshut -m immediate</a>
Transactional	No new connections are allowed and no new transactions can be started.  Oracle waits for all open transactions to finish, then disconnects all users (that is, work processes in the SAP system) and shuts down the database.	<a href="#">brspace -f dbshut -m transactional</a>
Abort	Emergency database shutdown  Users are not logged off and open transactions are not rolled back.  The database is <i>not</i> consistent after shutdown.  An instance recovery is automatically performed at the next database startup.	<a href="#">brspace -f dbshut -m abort</a>

 Note

If you have an Oracle Real Application Cluster (RAC), you can shut down all instances that are currently up with the `all_up` parameter:

[brspace -f dbshut -i all\\_up](#)

## Altering the Database Instance

You can [alter the database instance](#) as follows:

- Switch the current online redo log file

You might want to do this when, for example, you want to apply the data changes immediately to a standby database.

- Force a database checkpoint

You might want to do this when, for example, you want to shorten the database shutdown time after the checkpoint.

For more information, see the Oracle documentation.

- Set archivelog mode

When you set up the database, you normally make sure that you set archivelog mode on.

- Set noarchivelog mode

You can set noarchivelog mode on for short periods to perform essential database administration. The advantages are:

- You save space in the archive directory
- There is a performance gain

### Caution

It is very important that your database normally runs with archivelog mode set and with automatic archiving enabled. This makes sure that the online redo log files, which contain a record of the database transactions, are backed up. This means that you can recover the database in the event of a failure involving data loss.

For more information on Set archivelog mode and Set noarchivelog mode, see [Setting Up Archiving](#).

## Altering Database Parameters

You can [alter database parameters](#) in the following profiles:

- `init<DBSID>.ora` file, which is a normal disk file. It is still used for viewing by many SAP transactions, so must be kept up-to-date with the newer `spfile` (see below).
- `spfile`, which is a new binary server-side parameter file introduced by Oracle, available as part of the standard installation from SAP Web AS 6.40. If your SAP system was upgraded from an older release, you have to create it yourself initially using SQL\*Plus because it was not part of older SAP installations.

### Recommendation

We recommend that you always use BR\*Tools to alter database parameters, because BRSPACE ensures that the two files are synchronized whenever you make a change. Do not change either file manually at operating system level.



You can change parameters with the following scope:

Scope	Where the Change Occurs	When the Change Occurs	Summary
MEMORY	Memory of the currently running instance	Immediately, but does not persist when database is restarted	Immediate but not persistent
SPFILE	<code>spfile</code> and <code>init&lt;DBSID&gt;.ora</code>	At next database startup and persists from then on	Persistent but not immediate
BOTH	<code>spfile</code> and <code>init&lt;DBSID&gt;.ora</code> and memory of the currently running instance	Immediately and persists through subsequent database startups	Immediate and persistent

When you perform parameter changes with BRSPACE, it maintains a history of all changes in the [BRSPACE Parameter Change Log](#).

 Note

The name of the Oracle spfile on single database installations is `spfile<DBSID>.ora` and on RAC installations `spfile.ora`.

## Recreating the Database

For more information, see [Recreate Database](#).

## Recreate Database

You can recreate the database, that is, set up the SYSTEM tablespace. Logically this means that the SYSTEM tablespace is reorganized. The UNDO and TEMP tablespaces are also recreated. However, SAP data is unaffected. Only the metadata of SAP tablespaces are exported and imported.

You can recreate the database to:

- Migrate the SYSTEM tablespace from dictionary to locally managed tablespaces (LMTS)
- Defragment the SYSTEM tablespace
- Solve the extent problem in the SYSTEM tablespace – for more information, see *SAP Note* [651796](#).

 Note

This section describes how to approach recreate database.

For more information on how to perform recreate database, see [Recreating a Database with BR\\*Tools](#).

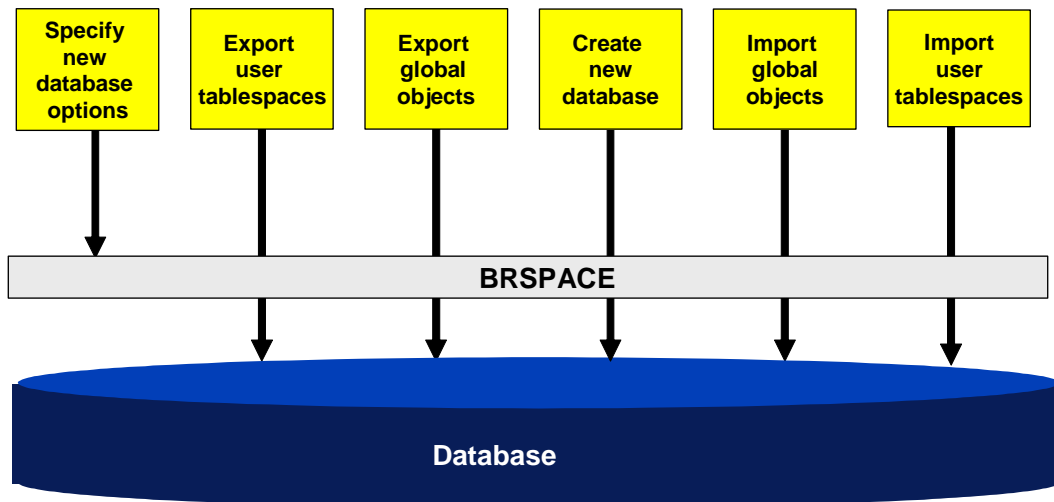
It is possible to use BRSPACE and BRTOOLS for heterogeneous database copies. For more information, see *Enhanced Support for Heterogeneous Database Copies* in *SAP Note* [1003028](#).

## Prerequisites

The prerequisites are described in [Recreating a Database with BR\\*Tools](#).

## Features

The following graphic shows how recreate database works:



## Activities

1. Specify new database options.

You define the following database options:

- Passwords for `SYS` and `SYSTEM` users
- Maximum number of instances
- Maximum number of datafiles
- Maximum number of redo log groups
- Maximum number of redo log members in a group
- Maximum size of redo log history
- Attributes for files of `SYSTEM`, `SYSAUX`, temporary and undo tablespaces
- Attributes for files of original and mirror online redo logs

2. Export user tablespaces phase.

BRSPACE exports the metadata - that is, database object definitions - of all user tablespaces using the transportable feature of the Oracle EXPDP (Data Pump) tool. This logically detaches the user tablespaces from the original database.

3. Export global objects phase.

BRSPACE exports global object definitions - such as users, roles, links, sequences, synonyms, views, procedures, triggers, and so on - using the full export feature of the Oracle EXPDP tool.

4. Create new database phase.

BRSPACE drops the old database and recreates a new one based on the options you specified. Oracle catalog views and stored procedures are also recreated, as described in the Oracle documentation.

5. Import global objects phase.

BRSPACE imports global object definitions previously exported using the full import feature of the Oracle IMPDP (Data Pump) tool.

6. Import user tablespaces phase.

BRSPACE imports the metadata of all user tablespaces exported before using the transportable tablespace feature of the Oracle IMPDP tool. Therefore, the user tablespaces are logically reattached to the new database.

## Space Management

This section helps you develop an approach to managing the space of your Oracle database.

### Note

This section discusses the approach to space management.

For more information on how to perform space management, see [Space Management with BR\\*Tools](#).

For more information on reorganization, see [Segment Management](#)

## Prerequisites

You consider whether to use raw devices or a file system. Raw devices are generally 10 to 20% faster on UNIX systems. However, with Oracle direct I/O or [Veritas Quick I/O](#) the difference is reduced.

We recommend raw devices only for experienced database administrators because the administration is more complex. For example, only one Oracle file can be set up on each raw device.

## Process

1. You monitor the database closely:
  - You regularly run the [database system check](#) so that you can detect space problems before they become serious.
  - If required, you run these reports on a one-off basis:
    - [Showing Tablespaces with BR\\*Tools](#)
    - [Showing Data Files with BR\\*Tools](#)

### Note

You also need to monitor available disk space at the operating system level with, for example:

[Showing Disk Volumes with BR\\*Tools](#)

Make sure that you plan for additional disk space in time to accommodate data growth.

2. You [manage tablespaces](#) to:
  - Extend a tablespace by adding a new file to avoid overflow
  - Create a new tablespace, for example, when switching from a dictionary managed to a locally managed tablespace
  - Rename tablespace, for example, after having reorganized all tables from an old into a new tablespace
  - Drop tablespaces, for example, after an upgrade
3. You [manage data files](#) to:
  - Resize the file, usually to prevent overflow
  - Turn on the Oracle AUTOEXTEND option to avoid data file overflow.
  - Rename data file, for example, to follow SAP naming conventions
  - Move the data file, for example, after you have added new disk storage to your system and want to use it for existing data files.
  - Drop empty data file because it is no longer needed

## Managing Tablespaces

You manage tablespaces in your Oracle database as part of [Space Management](#). You can extend, create, drop, and alter tablespaces.

You especially need to avoid tablespace overflow, which is when a tablespace runs out of freespace in the allocated file or files. This happens when an object requires a new extent but there is either no freespace or insufficient freespace in the tablespace.

### Note

You do not need to start a backup immediately after structural changes to the database such as tablespace extension, tablespace create, or drop. If the database crashes before a backup was performed BRRECOVER can reapply the changes during the recovery procedure based on information stored in the [BRSPACE structure change log](#), `struc<DBSID>.log`.

## Prerequisites

Tablespace overflow can occur in the following situations:

- Operations that greatly extend tables in the tablespace
  - Be sure to plan certain operations (for example, client copy or batch input) carefully, because they might extend tables excessively.
- Poor monitoring of the tablespace

During normal operation, database objects (that is, tables and indexes) grow steadily. Be sure to monitor the database, anticipate growth, and make sure there is always enough disk space available.

## Procedure

1. Extend a tablespace in one of the following ways:

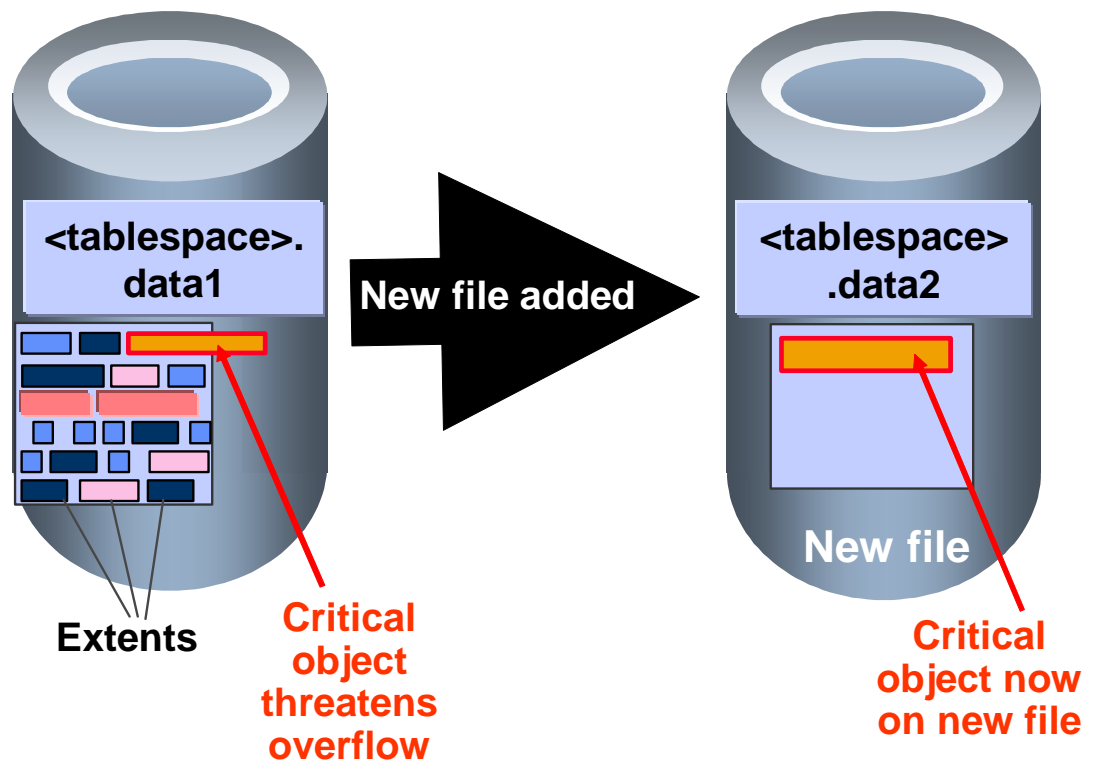
- Add a data file to the tablespace.

Use this method when either of the following conditions applies:

- The existing data files cannot be resized because there is not enough disk space available on the disks where the files are located
- The existing data files have reached their maximum size.

For more information on how to add a data file, see [Extend a tablespace with BR\\*Tools](#).

The following graphic shows the effect of adding a data file:



- Set the AUTOEXTEND option for the data files in the tablespace.
- Resize an existing data file to provide more space.

For more information on setting the AUTOEXTEND option and resizing a data file, see [Managing Data Files](#).

2. [Create a new tablespace](#) for the following reasons:

- To prepare for an upgrade, when you need to create a new tablespace for the SAP software of the new release.
- To prepare for an online reorganization of a tablespace, for example, to switch from a dictionary managed to a locally managed tablespace.
- After you have exported tables from a tablespace and you want to relocate the tablespace to a new disk volume for the import.

Since creating a tablespace is a structural change to the database, BRSPACE:

- Creates a control file backup in the directory `$SAPDATA_HOME/sapreorg/<encoded timestamp>` before and after the procedure.
- Logs the action in the [BRSPACE Structure Change Log](#).

 Note

When you create a new tablespace, you can specify whether the tablespace file:

- Has the AUTOEXTEND option set and, if so, the increment and maximum size.
- Is on a raw device or in the file system. For more information, see *Prerequisites* in [Space Management](#).

1. [Drop a tablespace](#), for example, when the tablespace is no longer required after an upgrade.

BRSPACE only lets you drop an empty tablespace, unless you specify the `-f|-force` option.

Since dropping a tablespace is a structural change to the database, BRSPACE:

- Creates a control file backup in the directory `$SAPDATA_HOME/sapreorg/<encoded timestamp>` before and after the procedure.
- Logs the action in the [BRSPACE Structure Change Log](#).

BRSPACE removes all subdirectories for the data files when it drops the tablespace.

2. [Alter a tablespace](#) for a number of reasons:

- Set a tablespace online or offline:

For systems with Multiple Components in One Database (MCOB), you can set the tablespaces `PSAP<SCHEMA_ID>` belonging to one schema user `SAP<SCHEMA_ID>` offline for maintenance. This does not affect tablespaces from other SAP systems with a different `SAP<SCHEMA_ID>`.

 Caution

The SAP system can only function if all the tablespaces belonging to the schema user are online. Make sure that you find the cause if Oracle has automatically set a tablespace offline, that is, without any intervention from you.

An example of this is when Oracle receives an operating system I/O error when writing to a data file. In this case, Oracle can set the tablespace offline to prevent corrupt blocks.

- Set or reset the backup status

BRBACKUP sets and resets the backup status for a tablespace during and after an online backup.

If BRBACKUP fails during an online backup, tablespaces might be left in backup status. The database system check can report this error when it raises the condition `TABLESPACE_IN_BACKUP`. For more information, see [BRCONNECT Default Conditions for Database Administration](#).

Use BR\*Tools to reset the backup status if you are sure that:

- An online backup has crashed
- There is currently no online backup running

- Coalesce free extents

This combines contiguous extents with free space into a single large extent within a tablespace. It consolidates the storage structure of the tablespace and can improve performance.

Although the database system check automatically coalesces tablespace free extents, you can perform this action on a one-off basis if a large amount of data was deleted. For example, if you have deleted a client or archived data, you can perform this action immediately afterwards.

- Rename tablespace

With this action, available from Oracle 10g, you can rename a tablespace following a reorganization. It enables you to drop the old tablespace and assign its name to the new tablespace.

## Managing Data Files

You manage data files in your Oracle database as part of [Space Management](#). You can resize and move data files and also set the AUTOEXTEND option.

This section discusses the *approach* to managing your data files.

For more information on *how* to manage your data files, see [Altering a Data File with BR\\*Tools](#).

To avoid exceeding the limit for the maximum number of files, you must manage the data files in an Oracle database. To improve your database performance, you can move the data files to raw disks. To simplify database administration, you can move the files to the file system.

### Prerequisites

The following constraints limit the maximum number of data files in an Oracle database:

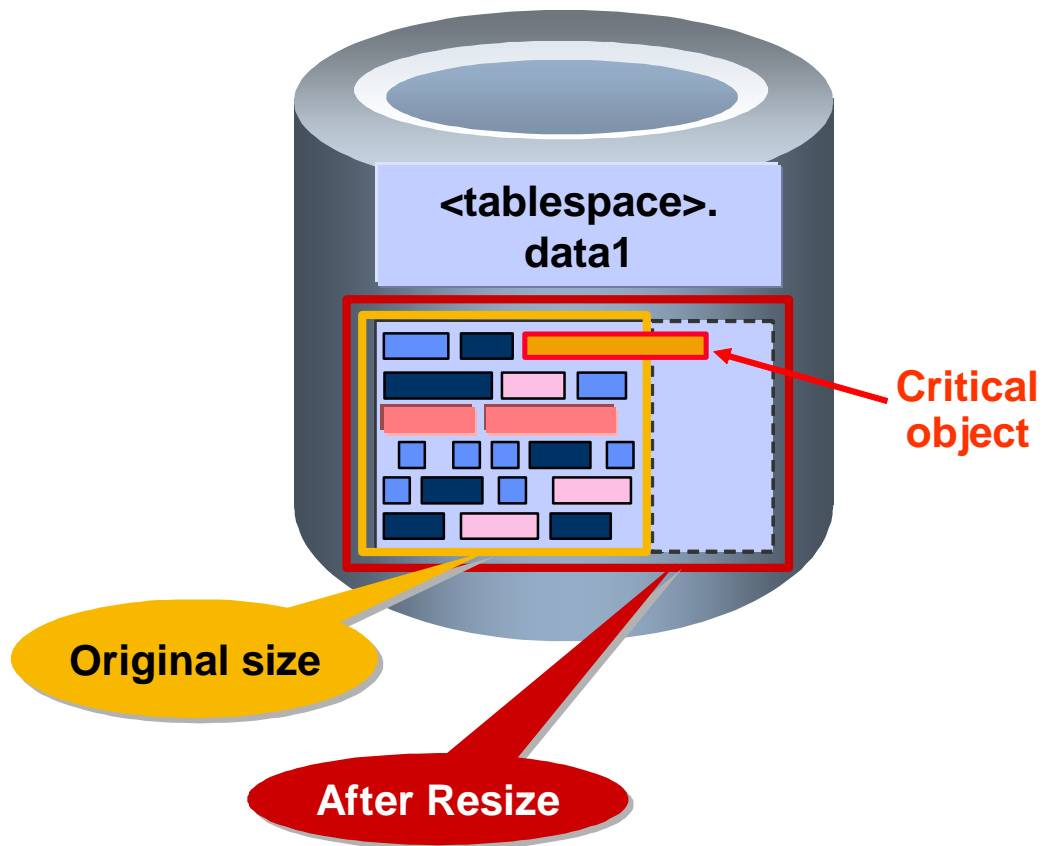
- The `db_files` parameter in the `init<DBSID>.ora` file is usually set to the value of the `maxdatafiles` option of the `create database` command. Currently, the SAP

installation process sets `maxdatafiles` and `db_files` to 254. If you reach this limit, you can no longer add files to the database.

- There is a UNIX kernel limit for the maximum number of open files.
- There is an absolute Oracle maximum of 65533 files in a database and usually 1022 files in a tablespace. However, certain hardware platforms have a limit lower than this absolute maximum.

## Procedure

1. Resize a data file to provide more space for the objects in the file, as shown in the following graphic:



Make sure that you choose a sufficiently large size for the data file, allowing for future growth. Otherwise you might have to repeat the procedure soon.

Use this method when you want to specify a larger size for the data file.

For more information on how to resize a data file, see [Altering a Data File with BR\\*Tools](#).

2. Maintain the `AUTOEXTEND` option on a data file.

With this option, the data file is extended automatically as the data grows. However, the entire disk can still overflow. Therefore, be sure to regularly monitor space on the disk volume.



Use this method when you have enough space on the disk volume and the tablespace is not expected to grow too rapidly.

For more information on how to maintain the autoextend option, see [Altering a Data File with BR\\*Tools](#).

3. You [move data files](#), for example, when you want to relocate the data files to new disk drives for performance or other reasons. To improve your database performance, move the data files to raw disks. To simplify database administration, move the files to the file system.
4. If you reach the limits specified in "Prerequisites" above, then do the following:
  - If you reach the limit for `db_files` or `maxdatafiles`, then do one of the following:
    - Increase the value of the `db_files` parameter, then shut down and restart the database.
    - Reorganize a tablespace consisting of several data files such that the number of files in use is reduced.
    - Recreate the control files specifying larger values for `maxdatafiles`.
    - Recreate the database specifying larger values for `maxdatafiles`
  - If you reach the UNIX kernel limit for the maximum number of open files, you have to change the relevant UNIX kernel parameter, then make sure that the change takes effect (such as shutting down and restarting the database, logging on again at UNIX level, restarting the UNIX system, and so on).
  - In the very unlikely event that you reach the hardware-dependent limit of files in the database (65533 files or less), then you have to perform a reorganization.

## Segment Management

This section helps you develop an approach to managing the segments – that is, the tables and indexes – of your Oracle database.

### Note

This section discusses the approach to segment management.

For more information on how to perform segment management, see [Segment Management with BR\\*Tools](#).

## Prerequisites

With Oracle 9i, you can now perform table reorganization and index rebuild while the database remains *online*. This overcomes the limitations of the old reorganization procedure based on export/import, which is a time-consuming procedure with a risk of data loss. However, you still need to perform export/import for tables containing `LONG` or `LONG RAW` fields. But you can convert `LONG` and `LONG RAW` fields online using [table reorganization in BRSPACE](#) with Oracle 10g.

For more information on how the reorganization works, see [Reorganization with the Redefinition Package](#).

## How the Database Deteriorates

When installed for an SAP System, the Oracle database looks as follows:

- Most of the tables and indexes of a tablespace are stored in only one extent.
- Each tablespace consists of exactly one data file.

This initial database status can change as follows:

- Additional extents

When more space for extra data is required, additional extents are allocated to the tables and indexes of a tablespace. The result might be poorer data access times.

- Additional data files

When a tablespace is full - that is, there is not enough freespace to create a new extent - additional data files must be added (except if the `AUTOEXTEND` option is used).

- Freespace fragmentation

Adding or deleting complete objects causes freespace fragmentation in a tablespace. Free storage space in data files is divided into smaller units. If these are smaller than one requested extent, the space is lost and cannot be used for storing data.

The Oracle system now automatically merges adjacent areas of free space, so this problem is less likely to occur than in the past. Also, this problem should not occur in locally managed tablespaces.

- Internal fragmentation

This occurs if the fill level of the database blocks develops unevenly. The fill level of the individual blocks is initially identical. Inserting and deleting rows causes some blocks to be filled completely, while others remain relatively empty. As a result, space is used inefficiently.

- Block chaining

If a data record does not fit into a database block, block chaining occurs. When the record is accessed, the system must then follow a chain from the first block of the data record to the further blocks. As a result, more time is needed for reading data from the disk.

Since SAP systems usually access table entries using an index, the above changes to the database do not normally significantly increase the time required to access data. However, such changes can increase run times for full-table scans.

## Features

- Reorganization

You can reorganize tables to move them to another tablespace or to recover space in the database and improve performance while the database is online. For more information on the approach to reorganization, see [Reorganization](#) and [Reorganization Case Study](#). You can also use the reorganization function in `BRSPACE` with Oracle 10g to convert `LONG` and `LONG RAW` fields to `CLOB` and `BLOB`.

- Rebuild indexes

You can rebuild fragmented indexes. This improve data access using indexes. For more information on how to rebuild indexes, see [Rebuilding Indexes with BR\\*Tools](#).

- Table export and import

This is the older method of reorganization. You must still use it for tables containing `LONG` or `LONG RAW` fields. For more information, see [Export/Import](#).

- Alter table

- Table monitoring

With table monitoring Oracle automatically collects information on table updates. Although table monitoring has a small performance overhead, the overall effect is to improve performance because table statistics can be more up-to-date.

 Recommendation

For Oracle 9i, we recommend you to turn on table monitoring for all tables. This greatly reduces the run time for update statistics with `BRCONNECT` because it can more quickly identify which tables need update statistics. Therefore, you can easily schedule update statistics to run daily.

For Oracle 10g, this is not required because all tables are monitored by default.

For more information on the approach to update statistics, see [Update Statistics](#).

- Set parallel degree

You can also use alter table to set the degree of parallelism for queries. A higher degree of parallelism improves performance on query statements.

 Caution

Only set the parallel degree if told to by SAP support. Production systems normally run without parallelism.

- Shrink tables – as of Oracle 10g

Shrinks table segments online and in-place. This frees unused space in the table segments.

- For more information on how to alter tables, see [Altering a Table with BR\\*Tools](#).

- Alter index


- Coalesce index

You can coalesce an index to deallocate internal free space. If space allocated to an index has never been used (that is, data has never been written to the data blocks), you can free such space for use by other objects in the tablespace. You can do this online without incurring downtime.

However, remember that the objects for which you have deallocated free space might themselves soon require new extents as they grow with inserts and updates. Therefore, a more permanent solution is to reorganize affected objects and also, if necessary, extend the tablespace.

- Set parallel degree

You can also use alter index to set the degree of parallelism for queries. A higher degree of parallelism improves performance on query statements.

 Caution

Only set the parallel degree if told to by SAP support. Production systems normally run without parallelism.

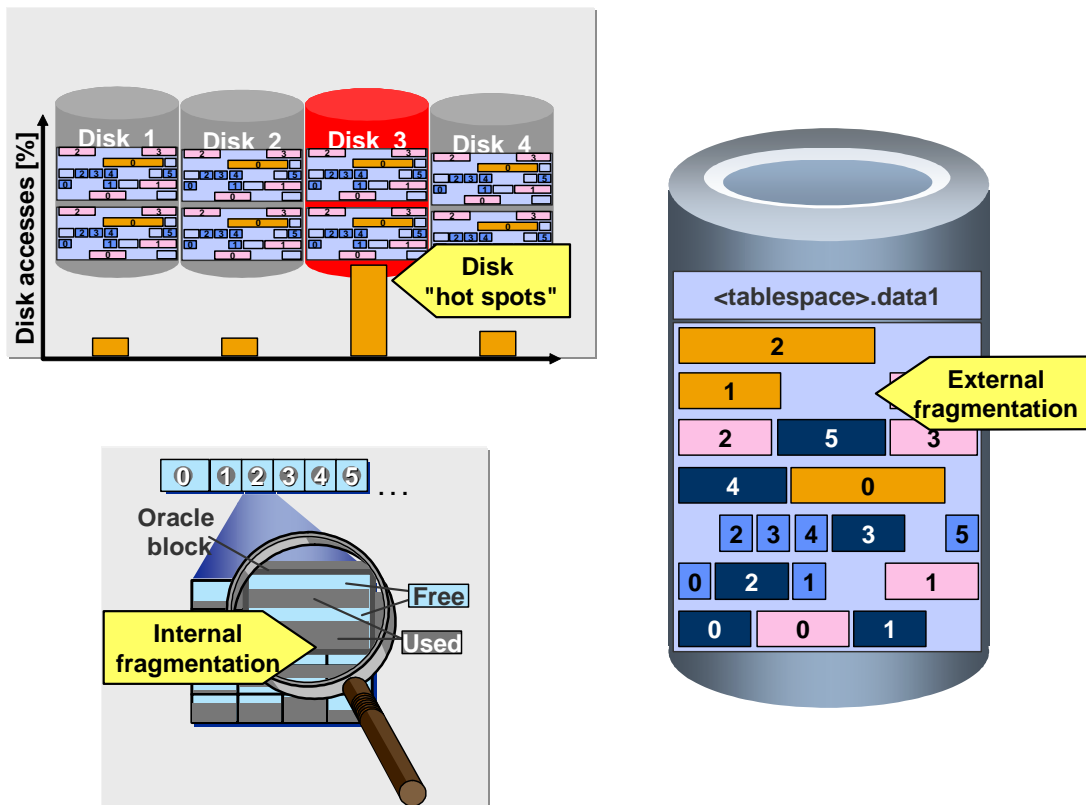
- Shrink indexes — as of Oracle 10g

Shrinks index segments online and in-place. This frees unused space in the index segments.

- For more information on how to alter indexes, see [Altering an Index with BR\\*Tools](#).

## Reorganization

This section helps you develop an approach to reorganization, which improves the structure of the database, and can result in improved performance. BRSPACE performs the reorganization using the new Oracle feature, online table redefinition or offline table move. The following graphic shows some of the reasons for reorganization:



For more information on how reorganization with the Oracle DBMS\_REDEFINITION package works, see [Reorganization with the Redefinition Package](#).

## Prerequisites

### Is Reorganization Really Necessary?

You need to reorganize less often than in the past due to the following:

- With locally managed tablespaces, space allocation inside a tablespace is now more efficient. The parameters `MAXEXTENTS` and `NEXT` no longer exist. Previously, incorrect use of these parameters often caused Oracle to create too many or too large extents, so wasting space.
- Automatic segment space allocation reduces internal fragmentation within Oracle blocks and improves the performance of parallel queries.
- Large disks and RAID systems with large and secure memory buffers reduce I/O hotspots. It is less important to distribute data files manually to different disks during reorganization because the setup itself can improve performance.

However, you might still need to reorganize if the following factors apply:

- You want to transform dictionary managed into locally managed tablespaces.
- You want to move certain large and heavily used tables into separate tablespaces.
- There are fragmented tables or indexes in dictionary managed tablespaces that you are still using. You can identify these by using the database system check. Check the parameters `TOO_MANY_EXTENTS`, `CRITICAL_SEGMENT`, and `PCTINCREASE_NOT_ZERO`. For more information, see [BRCONNECT Default Conditions for Database Administration](#).

To see how you can use reorganization for your system, see [Reorganization Case Study](#).

## Effects of a Reorganization

A reorganization can have the following positive effects on the database:

- The data from one object is merged into a single extent or into fewer extents.
- The data from a tablespace with many small files is merged into one or more larger data files.
- Freespace fragments in an object are merged into larger freespace segments. This process is called “defragmentation”.
- The fill level in the individual blocks is evened out, so reducing internal fragmentation.
- Data chains are resolved in most cases.

## Features

BRSPACE performs the reorganization by default using the new Oracle feature, online table redefinition, with the following advantages:

- Online reorganization improves availability since the SAP system does not need to be stopped for the reorganization. To avoid a performance impact on the SAP system, make sure that you perform the reorganization when the system load is low.



Note

You cannot perform online reorganization for tables with `LONG` or `LONG RAW` fields but you can convert them to `CLOB` or `BLOB` online. After this conversion, you can reorganize all tables online. For more information, see [SAP Note 646681](#).

- Parallel reorganization to improve performance. You can also reorganize without parallelism if you want to minimize the impact on the production database.
- Less risky because Oracle creates a copy of the table and transfers the entire table contents before deleting the original table.
- Consistency is guaranteed because all changes to tables currently being reorganized are preserved.
- Sort of table rows on a specified index (more exactly, on the columns of the index) during the reorganization. This improves later performance for partial sequential access.

For more information, see [-f tbreorg -rj-sortind](#)

- Perform offline reorganization using the `ALTER TABLE MOVE` statement. This can be faster than an online reorganization, especially for small tables. However, since the tables are locked during the move, we recommend you to stop the SAP system for this reorganization.

For more information, see [-f tbreorg -m|-mode](#).

- Easy restart for an aborted reorganization. The restarted reorganization only processes tables that have not yet been reorganized.

#### Note

BRSPACE supports reorganization of partitioned tables and indexes. The reorganization does not change the partitions and their parameters, unless you actually change the Data Definition Language (DDL) statements.

If a partition of a partitioned table or index is in a tablespace that you want to reorganize, BRSPACE reorganizes all other partitions of the object in other tablespaces too, even if you do not specify that you want to reorganize the other tablespaces. In other words, BRSPACE reorganizes all partitions of a partitioned object.

BRSPACE also supports the reorganization of tables with all types of large object (LOB) columns. Large objects are recreated with the same physical characteristics as before the reorganization.

## Activities

You can use table reorganization to:

- Transform data dictionary managed tablespaces into locally managed tablespaces
- Transform tablespaces in an old layout – that is, different tablespaces for data and indexes – into tablespaces in the new layout required for Multiple Components in One Database (MCOD), or vice versa
- Move large tables to a separate tablespace
- Reorganize tables due to internal or external fragmentation

The reorganization case study shows how you can transform data dictionary managed to locally managed tablespaces in the new SAP layout, that is, with a single large tablespace.

For more information on how to perform a table reorganization, see [Reorganizing Tables with BR\\*Tools](#).

#### Note

You can enter `first` for *Create DDL statements* in the BRSPACE menu or the command option `-d|-ddl` so that you can alter the Data Definition Language (DDL) statements for the reorganized tables. BRSPACE pauses and you can change the attributes of the following objects:

- Table
- Storage
- Field
- Index

For more information see [Reorganizing Tables with BR\\*Tools](#) or `-f tbreorg`.

#### Caution

If you change attributes, make sure that they:

- Are syntactically correct
- Do not contain any new fields
- Are compatible with the SAP dictionary

## Export/Import

You can use this function to export and import database objects. BR\*Tools uses Oracle export and import functionality — the Oracle EXP or IMP tools or Oracle data pump – to:

- Export database objects

You can export tables with their data, table and index definitions, and with other database objects such as constraints, grants, views, synonyms and sequences.

- Import database objects

You can import objects that have earlier been exported.

- Export database objects to null device (only with Oracle EXP)

You can use this to validate database objects.

Export and import enables you to back up database objects in addition to other database backups. If you only want to back up particular tables, export is a good method. For example, you can add to the data backup you perform before a reorganization by exporting the objects that are to be reorganized.

This function is only intended for use with objects in a *single* database.

 Caution

Do not use this function for the transport of database objects between databases.

The logical structure of SAP data is so complex that data objects are often distributed across many tables and many tables are linked largely according to the relational database model. Therefore, if you attempt to transport data between systems with this function, you end up with inconsistencies in the SAP system.

Use the SAP correction and transport system to transport objects between SAP systems. Create new SAP databases using the SAP installation procedure.

 Caution

Do *not* use this function for restore.

The data backups from an export are logical backups. This means that you cannot use them as part of an Oracle restore. The exported objects are static and are only consistent with the database if it remains unchanged.

 Note

You can export and import partitioned tables and indexes with BRSPACE. The export and import does not change the partitions and their parameters.

If a partition of a partitioned table or index is in a tablespace that you want to export and import, BRSPACE exports and imports all other partitions of the object in other tablespaces too, even if you do not specify that you want to reorganize the other tablespaces. In other words, BRSPACE exports and imports all partitions of a partitioned object.

## Activities

For more information on how to perform export/import, see:

- [Exporting Tables with BR\\*Tools](#)
- [Importing Tables with BR\\*Tools](#)
- [Special Export and Import Functions with BRSPACE](#)

For more information about how export/import is used in practice, see [Reorganization Case Study](#).



## Special Export and Import Functions with BRSPACE

You can use BRSPACE [-tbexport](#) and [-tbimport](#) to export or import tables and dependent objects, such as indexes, constraints, grants, and so on. With the following special export and import functions you can also export or import other “global” objects such as synonyms, views, procedures, and so on.



You can use the exports and imports listed below with Oracle Data Pump, except the export to NULL device.

## Features

### Note

In the first three points below, it is particularly important to use `-t "*"`  because this makes sure that not only tables and their dependent objects are exported

- Export all objects of the database owner

```
brspace -f tbexport -o <owner>[,<owner>,...] -t "*" 
```

- Export all objects of all SAP database owners

```
brspace -f tbexport -o all -t "*" 
```

- Full database export

```
brspace -f tbexport -o full -t "*" 
```

- Export to NULL device (not for use with Data Pump)

```
UNIX: brspace -f tbexport -u /dev/null
```

```
Windows: brspace -f tbexport -u \nul
```

There is no export dump file created in this case. You can use this type of export to validate the data.

### Caution

Use option `-o` (as described above) if you want to check all SAP tables, as in this example:

```
brspace -c force -f tbexport -o sapr3 -t "*" -u /dev/null
```

Or you can even use the full database export to check the Oracle dictionary:

```
brspace -c force -f tbexport -o full -t "*" -u /dev/null
```

This avoids creating a parameter file with several thousand table names, which can cause the export tool to fail.

- Import all objects of the database owner

From the export dumps created in the first three points above, you can import all tables and their dependent objects either with or without global objects:

- Import of tables of the database owner with their data and dependent objects, but without global objects:

```
brspace -f tbimport -y tables -o <owner>[,<owner>,...] -t "*" 
```

- Import of tables of the database owner with their data and dependent objects together with global objects:

```
brspace -f tbimport -y tables -o <owner>[,<owner>,...] -t
"%"
```

- o Import of the objects of one database owner to another database owner

```
brspace -f tbimport -y tables -o "<old_owner>,->,"
```

```
<new_owner>" -t "*"
```

```
brspace -f tbimport -y tables -o "<old_owner>,->,"
```

```
<new_owner>" -t "%"
```

For more information about import, see BRSPACE [-f tbimport](#).

## Reorganization Case Study

This case study shows how to perform a reorganization with the following aims:

- Convert dictionary managed to locally managed tablespaces
- Convert the tablespace layout to the new SAP standard layout with a single large tablespace, PSAP<SAPSID>.
- Correctly process tables with `LONG` or `LONG RAW` fields, which cannot be reorganized using the online procedure - they require a reorganization using `export/import`. As of Oracle 10g, you can convert `LONG` and `LONG RAW` fields online to `CLOB` and `BLOB` fields. After this conversion, you can convert all tables online.

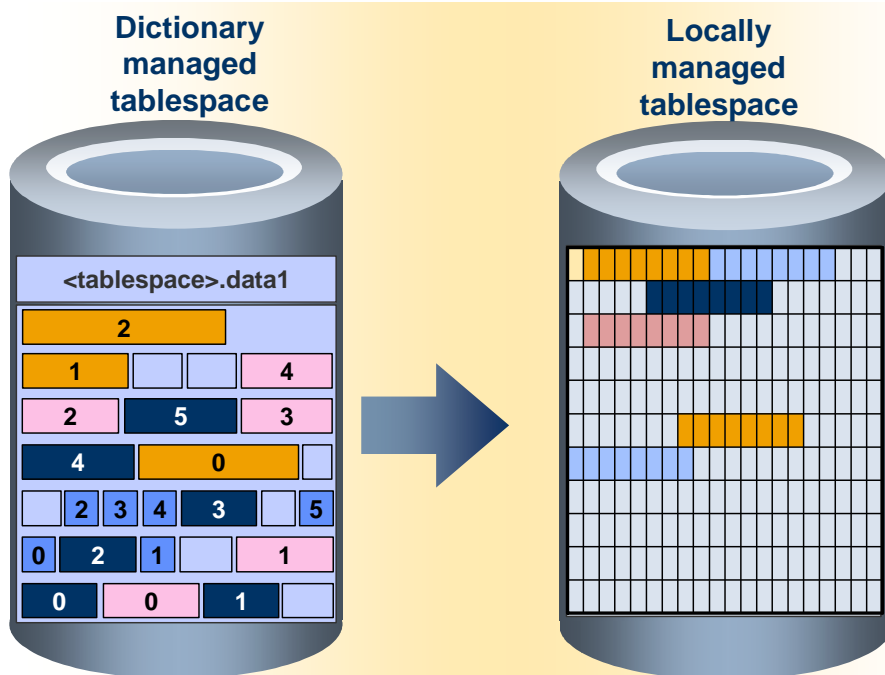
### Recommendation

In this case study we process tables from a group of tablespaces - that is, tablespaces in the [old tablespace layout](#). If you want to process all the tables in your database, we recommend that you do this in groups of tablespaces.

However, if the database is small, you can try processing all the tablespaces in one run.

## Prerequisites

- One of the aims of the case study is to convert dictionary managed to locally managed tablespaces, as shown in the following graphic:



Locally managed tablespaces are now the SAP standard. When you create new tablespaces with BRSPACE, they are by default locally managed.

- When you reorganize a table tablespace, BRSPACE also reorganizes the corresponding index tablespace. For example, if you reorganize PSAPSTABD, then PSAPSTABI is also reorganized.
- You might need to plan downtime for this procedure because tables with `LONG` or `LONG RAW` fields (if selected) require reorganization with the export/import procedure, which cannot be performed online.

## Process

1. You [create a new tablespace](#) called `PSAP<SAPSID>` for the schema owned by `SAP<SAPSID>`, which is locally managed and which stores both data and indexes:
  - Set *Data type in tablespace* in the BRSPACE menu or command option `-d|-data` to `both`. This means the new tablespace will contain both tables and indexes.
  - Set *File autoextend mode* in the BRSPACE menu or command option `-a|-autoextend`.
  - Set the table data class of the new tablespace in the BRSPACE menu or command option `-l|-class` to the list of tablespaces to be reorganized or to `all` if you want to reorganize all tablespaces, except `PSAPUSER*`.
  - Make sure that the tablespace is large enough to hold all the data from your existing tablespaces.

BRSPACE sets up the new tablespace, ready to contain the reorganized tables.

2. You [reorganize the tables](#) in the selected tablespaces using the BRTOOLS menus or the BRSPACE command option `-s|-tablespace`:
  - Set *Tablespace names* to the tablespaces for which you want to reorganize tables. If you want to reorganize all the tables in a component, set *Table owner* in the BRTOOLS menu or BRSPACE command option `-o|-owner` to the name of the SAP owner. Set *Table names* in the BRTOOLS menu or command option `-t|-table` to "\*" to avoid having to make a further selection.
  - Set *New destination tablespace (newts)* in the BRSPACE menu or command option `-n|-newts` to the new tablespace, PSAP<SAPSID>. You do not need to specify a separate index tablespace.
  - To improve the performance of the reorganization, you can set *Parallel threads* or command option `-p|-parallel`.

BRSPACE reorganizes the tables to the new tablespace and deletes the tables from the source tablespaces.

However, BRSPACE cannot reorganize tables with LONG or LONG RAW fields. It displays a warning message and leaves these tables in the source tablespace.

3. You stop the SAP system.
4. You perform an export/import to reorganize the remaining tables with LONG or LONG RAW fields (and associated indexes) into the new tablespace:
  1. You perform a dummy [reorganization](#) to generate Data Definition Language (DDL) statements with which you subsequently create the tables in the new tablespace:
    - Set *Tablespace names* or command option `-s|-tablespace` to the tablespaces containing the remaining tables with LONG or LONG RAW fields (and associated indexes).
    - Set *Create DDL statements (DDL)* in the BRSPACE menu or command option `-d|-ddl` to `only`.
    - Set *New destination tablespace (newts)* in the BRSPACE menu or command option `-n|-newts` to the new tablespace, PSAP<SAPSID>. You do *not* need to specify a separate index tablespace.
  2. You [export the table](#) data from the tablespaces:
    - Set *Tablespace names* or command option `-s|-tablespace` to the tablespaces containing the remaining tables with LONG or LONG RAW fields (and associated indexes).
    - Set *Export dump directory* or command option `-u|-dumpdir` to a directory with enough space to store the data from the exported tables.
  3. You [drop the tablespaces](#) (including contents using command option `-f|-force`) from which you have just exported the table data.
  4. You change to the directory where you stored the DDL statements – that is, \$SAPDATA\_HOME/sapreorg/<coded timestamp> – and enter the following SQLPLUS commands:

```
connect / as sysdba  
  
@ddl.sql
```

This creates the empty tables in the new tablespace.

5. You [import the table](#) data into the new tablespace.
5. You restart the SAP System.

For more information on reorganization see *SAP Note* [646681](#).



## Reorganization with the Redefinition Package

BRSPACE performs online table reorganization using the Oracle package DBMS\_REDEFINITION. For more information on this package, see the Oracle documentation.

### Activities

For each table to be reorganized:

1. BRSPACE calls the package to check whether the table can be redefined online. If a table cannot be processed - for example, if it has LONG or LONG RAW fields - the package returns an error and BRSPACE continues processing, excluding the table from the reorganization.
2. BRSPACE generates Data Definition Language (DDL) statements for the table using the Oracle package DBMS\_METADATA.
3. BRSPACE creates an empty interim table. BRSPACE names the interim table <ORIGINAL\_NAME>#\$.
4. BRSPACE creates a primary key constraint for tables without a primary key but with a unique index, as recommended by Oracle.
5. BRSPACE calls DBMS\_REDEFINITION to start the redefinition process to copy data from the original table to the interim table. This processing occurs in parallel if you set the required option.
6. BRSPACE creates all indexes, constraints, grants, triggers, and comments from the original table on the interim table. Indexes, constraints, and triggers are created with interim names, <ORIGINAL\_NAME>#\$.  
  
When the redefinition is finished, the interim table has all the data and attributes of the original table.

7. BRSPACE calls RDBMS\_REDEFINITION to finish the reorganization. RDBMS\_REDEFINITION briefly locks both the original and interim table before renaming the interim table so that it has the same name as the original table.
8. If all the above steps have completed successfully, BRSPACE drops the original table and renames the indexes, constraints, and triggers back to their original names, frees the space unused by the tables and its indexes, and re-imports their statistics.

# Database Backup

You need to regularly back up your Oracle database so that, in the event of failure, you can restore and recover it.

## Integration

You use the following tools for database backup:

Tool	Use
<a href="#">DBA Planning Calendar</a> – in the DBA Cockpit of the SAP system	Routine backups: <ul style="list-style-type: none"><li>• Backup of database files</li><li>• Backup of offline redo log files</li></ul>
<a href="#">Backup and Database Copy with BR*Tools</a> – uses the BR*Tools character or graphical user interface	<ul style="list-style-type: none"><li>• <a href="#">Backing Up the Database with BR*Tools</a></li><li>• <a href="#">Backing Up the Offline Redo Log Files with BR*Tools</a></li></ul>
<a href="#">BRBACKUP</a>	Backup of database files
<a href="#">BRARCHIVE</a>	Backup of offline redo log files
<a href="#">Oracle Recovery Manager (RMAN)</a> – integrated with BRBACKUP and BRARCHIVE	Backup, restore, and recovery

## Prerequisites

You familiarize yourself with the above tools and make sure that you *regularly* back up the *entire* database. For more information, see [Backup Overview](#).

## Features

- BRBACKUP backs up database files
- BRARCHIVE backs up offline redo log files

For more information, see [Why Back Up the Database?](#) and [What Needs Backing Up?](#)

## Activities

You can perform backups in the following ways:

- DBA Planning Calendar for routine backups
- The menus in BR\*Tools  
  
BRTOOLS calls the tools BRBACKUP, BRARCHIVE, BRRESTORE, or BRRECOVER as necessary to complete the task you have chosen.
- The command line

In this way, you can use the tools BRBACKUP, BRARCHIVE, BRRESTORE, or BRRECOVER, but this requires expert knowledge.

 Recommendation

We recommend you to use the DBA Planning Calendar for routine backup because this enables you to automatically schedule the backup.

We recommend you to use the BR\*Tools menus for restore and recovery because BR\*Tools guides you through the necessary steps.

- BRGUI

## Backup Overview

This section gives you basic information to develop a careful approach to backing up your Oracle database.

### Prerequisites

When designing your approach to backup, archive (that is, backup of offline redo log files), restore, and recovery, consider the following:

- You design an approach based on the needs of your company. You ask yourself questions such as:
  - What level of availability do you require from the database?
  - How long can you afford to shut down the SAP System in the event of data loss? Some backup approaches require a longer restore and recovery time than others.
  - Can you afford to lose data at all? If not, consider [high availability solutions for Oracle](#) such as [Oracle standby databases](#).
- You carefully test your approach before your SAP System goes live, and again after any changes have been made to the approach.
- You document your approach in a plan and make sure that all relevant people know the procedures to follow in the event of problems.

### Process

1. You identify [what needs to be backed up](#).  
Normally, you back up the complete database and the redo log files.
2. You identify the [database backup type](#) that you require.  
Normally, you perform an [online](#) and [complete](#) backup.
3. You work out a [backup cycle](#). We recommend a minimum cycle of 14 days, although 28 days is preferable. For example, with a 28-day cycle, you reuse the backup media after 28 days.

 Recommendation

We recommend that, if possible, you perform at least one offline backup per cycle. It is even better if you perform backups more often (for example, weekly) if possible. Use [incremental backup](#) for larger databases.

4. You schedule regular backups.

#### Recommendation

We recommend that you schedule regular backups with the [DBA Planning Calendar](#), using the [action patterns](#) available there. There are action patterns for different requirements to cover the main aspects of database administration, including backup.

5. You [verify](#) the:
  - Backup tape readability, that is, a check on the contents of the media after the backup
  - Database block consistency, that is, a check on the database itself

If possible, run both types of verify daily, otherwise preferably weekly. At the least, be sure to run a verify once in each backup cycle.

You can back up the database and then verify both the backup media and the database using a single BRBACKUP command, [brbackup -w use\\_dbv|use\\_rmv](#). The `-w use_dbv` option is also available in the [action patterns](#) of the DBA Planning Calendar. Be aware that a verify considerably extends backup run times.

To only verify database block consistency (that is, without a database backup), use the command [brbackup -w only\\_dbv|-w use\\_dbv](#).

To verify table and index structures, use the command [brconnect -f stats -v](#).

To verify the backed up redo log files, use the command [brarchive -w| use\\_rmv](#).

## Result

See the following for examples of backup approaches:

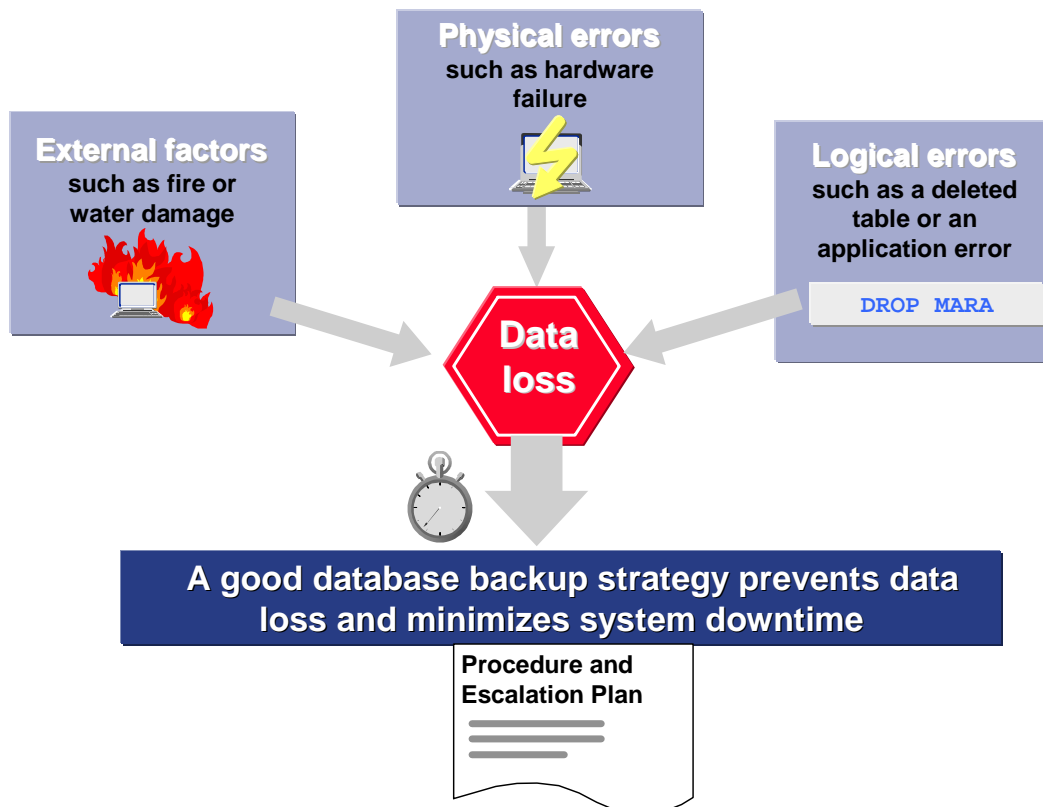
- [Backup Approach with Daily Complete Backups](#)
- [Backup Approach for Very Large Database with Partial Backups](#)
- [Backup Approach with One-Day Retention Period](#)



## Why Back Up the Database?

Without a careful approach to backing up your Oracle database, you run the risk of experiencing excessive system downtime and possibly losing data. The following graphic shows the main ways that data loss occurs:





How you react to data loss depends on how it was caused:

- By external factors or physical errors

You must recover the database up to the point in time when the database crashed. If a full recovery is possible, only the data of uncommitted transactions before the error is lost.

- By logical errors

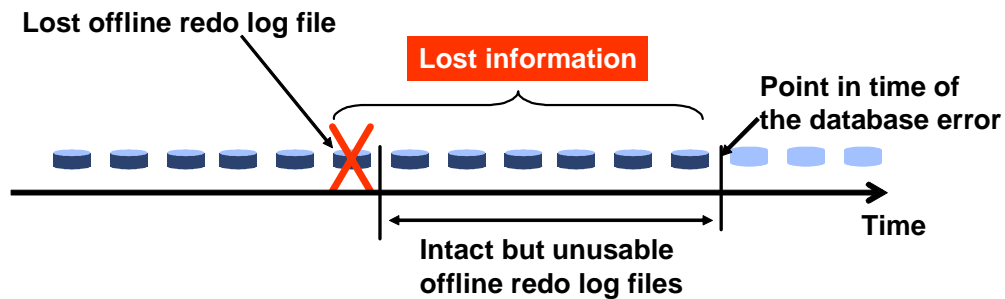
You must recover the database up to a point in time shortly before the error occurred. However, data entered after the error is lost.

To avoid data loss after a logical error, it is sometimes possible to restore the database to a different machine and then export the affected table from that machine to your production database. However, this method is difficult and requires expert knowledge of the application that uses the table.

As well as regularly backing up the database, you also need to archive the redo log files. The following graphic shows how important it is to archive the redo log files:

## Forward recovery

A database backup is restored and you now want to recover data from offline redo log files



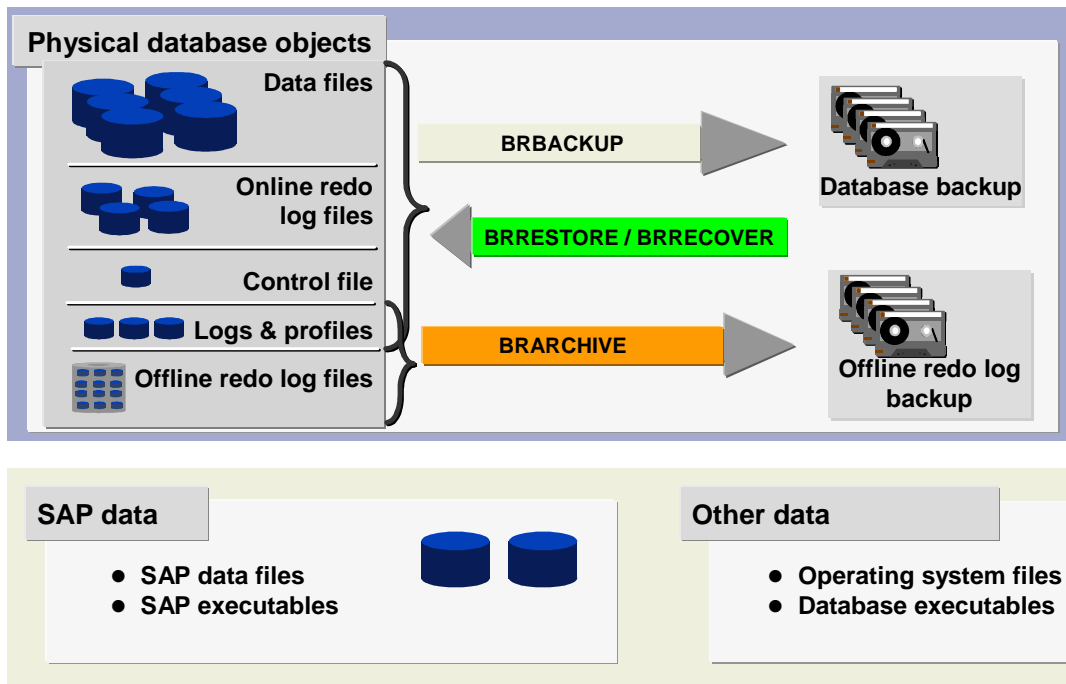
If one offline redo log file is lost, none of the files that follow it can be used

### ↑ Recommendation

We recommend you to keep at least two copies of the offline redo log files on a secure storage medium. For maximum security, store the copies in different locations.

## 💡 What Needs Backing Up?

Apart from deciding whether to perform an [online or offline](#), a [complete](#) or [incremental](#) backup of your Oracle database, you need to decide *what* to back up. The following graphic summarizes the different items that you need to consider in your backup approach:



For more information on the tools, see:

- [BRBACKUP](#)
- [BRARCHIVE](#)
- [BRRESTORE](#)
- [BRRECOVER](#)

The rest of this section discusses what you need to back up from a logical viewpoint.

## Backing Up the Complete Database

- Consider the following when you decide the frequency of complete database backups:
  - The frequency of complete database backups should depend on the degree of activity in your database. High database activity increases the number of redo log files written between complete backups, which increases the time required for any necessary recovery.
  - Performing frequent complete backups reduces the number of redo log files that must exist in order to make a complete recovery. This reduces the data loss if one of these files is lost.

For more information, see [Backup Cycles](#).

- If a redo log file is lost, you can often not completely recover the database after an error, even if you have a complete database backup. Instead, you can only recover up to the gap in the redo log file sequence.
- SAP recommends keeping several generations of complete backups and the corresponding redo log files. This ensures that you can still recover the database, even if the last complete backup is lost.

- Although not necessary - since BRRECOVER can recover the database even after structure changes - you can simplify recovery of the database by backing up at least the changed tablespaces and the control file after every structure change (that is, new, changed, or deleted tablespaces, or new data files). After a reorganization you can back up the affected tablespace. Follow the instructions for the tablespace backup below.
- Use BRBACKUP to back up the database and BRARCHIVE to back up the offline redo log files. See [Common Features of BRBACKUP and BRARCHIVE](#).

## Backing Up a Tablespace

Backing up tablespaces that are changed frequently can reduce the time required for any necessary recovery. When a more recent backup of an intensively used tablespace is available, fewer redo log entries have to be processed in order to recover the tablespace. If you can back up the entire database on a daily basis, tablespace backups are not necessary.

However, tablespace backups are no replacement for frequent backups of the complete database because:

- If you only perform tablespace backups for a long period of time, this increases your dependence on the archived redo log files, and therefore the risk of data loss if one of the redo log files is lost.
- If tablespace backups are used, you decide what has to be backed up. BRBACKUP supports the backup operation itself, but does not help you decide which tablespaces to back up. Therefore, it is possible that you might forget to back up certain tablespaces.

You can use [tablespace backup](#) for large databases.

## Backing Up the Control File

Another type of partial backup is to back up the control file. The control file records the physical file structure of the database. Therefore, you should back up the control file after every structure change.

Mirrored control files protect you against the loss of a single control file. If data files are damaged, an older control file that mirrors the corresponding structure of the database may be necessary for recovery. For this reason, mirroring the control files is by no means a replacement for backing up the control file after every change in the structure of the database.

When BRBACKUP is used to back up the database files, the control file is always saved along with them. The control file is saved before and after the operation for various administration measures with BRSPACE (for example, tablespace extension or reorganization of a table).

## Backing Up a Test System

The data of a test database might not have to be backed up as often, depending on how your test system is used. If you accept the restriction that you will only be able to recover the database from the last offline backup, you can operate the database in *NOARCHIVELOG* mode. If you do not back up the database at all, you will have to reinstall the database in a recovery situation.

## Backing Up Executable Programs and Other SAP Components

In addition to backing up database files and offline redo log files, we recommend you to also back up the following *non-database* files:

- Permanent Files

You can find these files, for example, on UNIX systems in the subdirectories `/usr/sap/<SAPSID>/SYS` and `<ORACLE_HOME>/<ORACLE_VERSION>` (UNIX) or `\\sapmnt\<SAPSID>\SYS` and `<ORACLE_HOME>` (Windows). They include executable programs and profiles of the SAP System and of the Oracle database system. We recommend you to back up the SAP directories after an SAP System upgrade and the Oracle directories after a database upgrade.

- Temporary Files

You can find these files, for example, on UNIX systems in the subdirectory `/usr/sap/<SAPSID>/<INSTANCE>`. The loss of these files is not critical, and does not cause data inconsistency. SAP provides tools that can reset the references to these files in the database, when required.

BRBACKUP can back up non-database files as well as database files in the same run, but we recommend you to back up non-database files in a separate run. For more information, see [Backing Up Non-Database Files and Directories](#).

 Caution

Backing up non-database files using BRBACKUP is not a replacement for backing up the file system at operating system level. For more information, see the documentation for your operating system.

## Database Backup Types

This section describes the different types of backup that you can make of your Oracle database.

 Note

The term “backup” normally refers to a physical backup, which is sometimes called an image backup.

Oracle and SAP support the following types of database backup:

- Physical backup using [BRBACKUP](#)

Physical backups, sometimes called image backups, are complete block-for-block copies of the database, either online or offline. Physical backups are required to recover the database to a consistent and current state. You can perform the following kinds of physical backups:

- [Online and offline backups](#) of the database
- [Consistent online backups](#), that is, while the database is in use
- [Complete backups](#) of the entire database
- [Incremental backups](#) of only data that was changed since the last full backup
- [Tablespace backups](#) to separately back up intensively used tablespaces between complete backups - this is an advanced function to speed up recovery, which we do not normally recommend.

- [Partial backups](#) to back up only parts of the database, not the entire database.
- Logical backup using BRSPACE – also called export

BRSPACE export uses the Oracle export and import functions to let you back up and recover specific objects – that is, a set of database records – in the database. However, logical backups are not a suitable replacement for physical database backups because you can only recover the database to the condition at the time of the backup. You cannot recover database changes made after a logical backup.

SAP tables are usually used by multiple users or applications, which means that it is not a good idea to make user-related backups with Export/Import. For example, you do not want a user to restore the ATAB table in order to retrieve lost entries in a particular SAP pool table stored in ATAB.

Logical backups of SAP objects can be performed using the SAP tool *R3trans*. *R3trans* exports SAP system objects (among others) from the database into operating system files. If a user then inadvertently deletes an object, that object can be imported from the exported backup file. For more information, see the documentation on *Transport Tools*.

#### Note

There are also the following types of non-database backup:

- Backup of executable programs and other components of the SAP system, described in [What Needs Backing Up?](#)
- Operating system backup – for more information, see the documentation for your operating system

## Online and Offline Backup

### Online Backup

You can perform an online backup with the database running - that is, the users can continue to work normally. The management of database changes by the corresponding Oracle background processes is not affected.

Tablespace online backups alone are inconsistent. To make the database consistent, you need to apply redo log entries from the run-time period of the backup. If you use [Oracle Recovery Manager \(RMAN\)](#) for online backup, it takes care of internal block consistency during the backup.

An online backup is made using operating system tools such as `cpio` or `dd` – for example, under the control of [BRBACKUP](#). Since these tools are not part of the database system, Oracle must be informed about the starting point of a backup. In this way, a unique starting point is defined from which the changes of all the data in a tablespace can be recovered in the event of an error. This process works as follows:

1. The starting point of the backup is set using the command `ALTER TABLESPACE <tablespace name> BEGIN BACKUP`. The header of the tablespace files holds information on the checkpoint and redo log files, that is, the system change number (SCN). When the next redo log file switch or checkpoint occurs (normal database operations continue), the header information remains unchanged.

2. Based on this mechanism, all the files of a tablespace are copied with uniform header information specifying when the backup was started.
3. Once the backup of the tablespace is complete, the command `ALTER TABLESPACE <tablespace name> END BACKUP` makes sure that the header information of the files is updated.

RMAN normally takes care of this process internally, so it is not explicitly performed.

#### Note

For more information on what to do if your database crashes during an online backup, see [Fixing an Online Backup Crash](#).

## Offline Backup

After an offline backup of the complete database, you have a backup of the database that is consistent. If you work with the database after the backup, the backup is no longer up-to-date. In this case, you have to recover the database after you restore the backup, using the redo log files.

You must close the database for an offline backup, which means that you have to stop work in the SAP System. However, the SAP System does not have to be shut down for an offline backup. If the `RECONNECT` mechanism is set in the SAP start profiles, the connection to the SAP System is remade after the database is restarted. This means that the information in the buffers of the SAP System is not lost, which implies better performance immediately following the database startup.

You can use [pre\\_shut\\_cmd](#) and [post\\_shut\\_cmd](#) to execute external commands before or after the database is stopped for an offline backup with BRBACKUP.

## Consistent Online Backup

A consistent online backup of your Oracle database is an alternative to an offline backup when you cannot close the database. An online backup has logically consistent data because the offline redo log files created during a backup are backed up with the database files on the same backup volume. You use [BRBACKUP](#) for a consistent online backup.

A consistent online backup differs from an offline backup in that a recovery of the database - that is, applying the redo log files - is always necessary in order to guarantee consistent data.

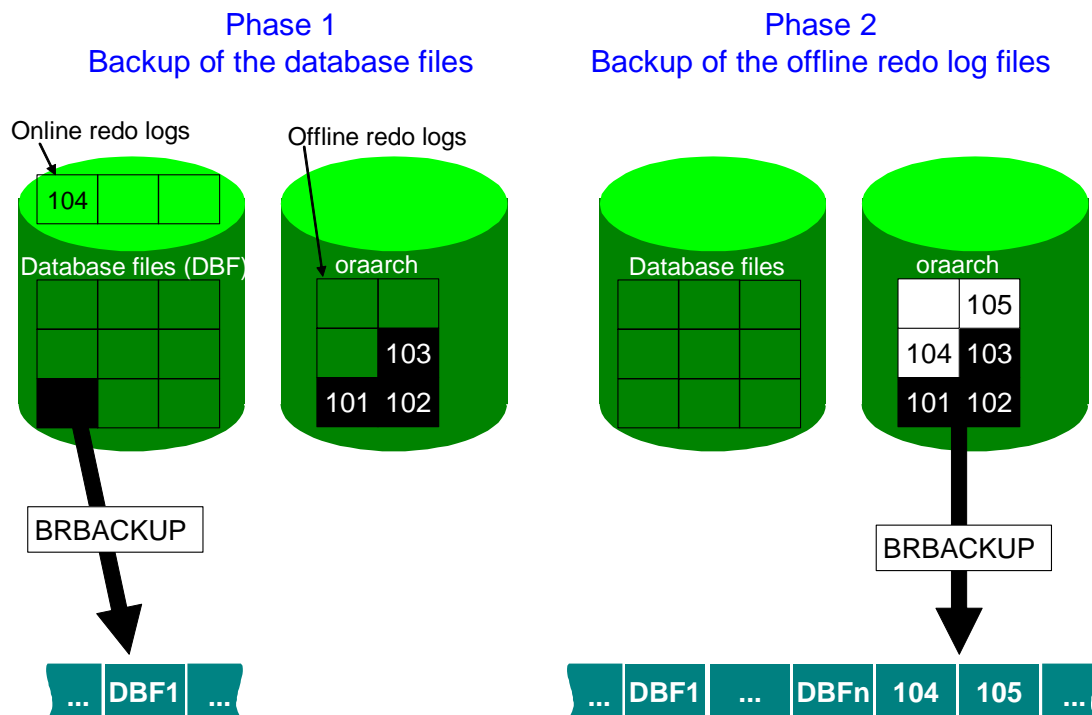
Distinguish between the following, which are completely independent:

- Backup of the offline redo log files using BRBACKUP during a consistent online backup, as described in this section
- Backup of the offline redo log files using BRARCHIVE as a separate operation – for more information, see [-aj-archive](#).

Database recovery is simplified because the offline redo log files of the same backup can be applied. In the same way the database can be reset to an earlier status (*Whole database reset*).

#### Recommendation

We recommend consistent online backups, especially for monthly and yearly backups, when an offline backup is unacceptable. However, consistent online backups cannot replace backups of the offline redo log files with BRARCHIVE.



To perform a consistent online backup, use the BRBACKUP option `-tl-type online_cons` or the relevant `init<DBSID>.sap` profile parameter `backup_type = online_cons`.

To restore the offline redo log files from the BRBACKUP backups, use the BRRESTORE option `-ml-mode archive_logs`. Restore a complete BRBACKUP backup including offline redo log files using the BRRESTORE option `-m full`.

## Complete Backup

This section describes the different types of complete backup for the Oracle database.

SAP backup tools are integrated with the Oracle [Recovery Manager \(RMAN\)](#). You can use RMAN to make [incremental backups](#). However, you cannot start an incremental backup without a preceding full backup.

Complete backup refers to one of the following:

- Whole backup

Backs up all database files, but this backup is not cataloged as a level-0 backup, which means that you cannot use it as a reference backup for an incremental backup with RMAN. The syntax for a whole backup is as follows:

- In profile `init<DBSID>.sap:backup_mode = all`



- With BRBACKUP: `brbackup -m all`
- Full backup
  - Backs up all database files. You can perform a full backup with or without RMAN. If you do not use RMAN, then RMAN is called separately to catalog the backup as level 0. This means that you can use this backup as a reference backup for an incremental backup with RMAN. The syntax for a full backup is:
    - In profile `init<DBSID>.sap:backup_mode = full`
    - With BRBACKUP: `brbackup -m full`

In the context of the Recovery Manager this backup is an incremental level-0 backup.

## More Information

[backup\\_mode](#)

[-m|-mode](#)

[RMAN Backup Strategies](#)



## Incremental Backup

This section describes incremental backups for the Oracle database. [BRBACKUP](#) supports incremental backup with the [Oracle Recovery Manager](#) (RMAN). In an incremental backup, only the changes that have been made since the last full backup are saved.

Incremental backups improve the performance of the database backup, saving both time and backup media. If you do not save as much time as expected, this is because each block has to be checked individually to see if it needs to be backed up, that is, whether it has been changed or not.

Incremental backups are especially useful for regular backups of [large databases](#).

### Caution

An incremental backup cannot be used on its own to recover the database. You must always have the preceding full backup as well.

You can only make incremental backups with the RMAN. For more information, see [RMAN Backup Strategies](#).

To be able to make an incremental backup, you must first make a full backup (level 0). A full backup of the database backs up all Oracle database blocks that have already been used to store data.

You can then make incremental backups. An incremental backup (level 1, cumulative) of the database backs up all Oracle database blocks that have changed since the last full backup (level 0).

## Example

The following describes a weekly backup scenario:

- Sunday: Full backup (level 0) of the database
- Monday to Saturday: Incremental backup (level 1, cumulative) of the database

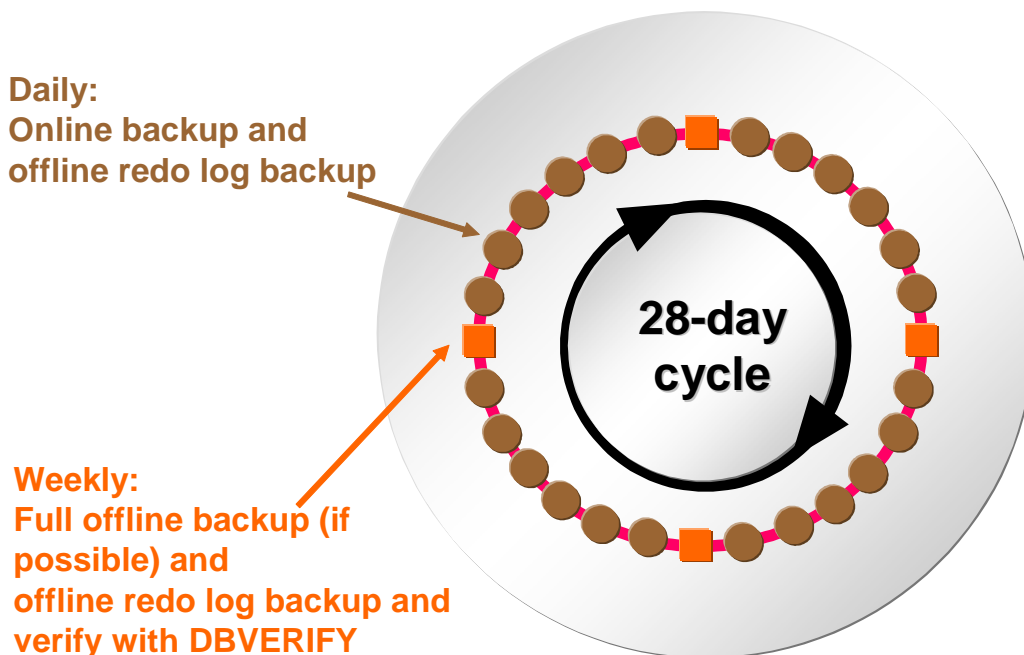
 Note

The SAP incremental backup has the advantage that it comprises a backup (level 0) of the new files that were created as a result of tablespace extensions after the last full database backup. This means that you do not need to make a full backup of the entire database immediately after such extensions.

## Backup Cycles

This section gives you recommendations on how to plan a backup cycle for your Oracle database. We recommend a backup cycle of at least 14 days, preferably 28 days. For more information on the tapes required, see [Backup Media](#).

### Recommended 28-Day Backup Cycle



The guidelines are as follows:

- Perform a full online backup each working day.

- If possible, perform a complete offline backup weekly (for example, at the weekend), or at least once in the cycle. If offline backups are not acceptable due to downtime, complete online backups are also good enough.
- Back up the offline redo log files each working day and after every online and offline backup. Be sure to back up the offline redo log files twice on separate tapes.
- To verify the process, you need to:
  - Verify backups for physical errors at least once in the cycle, preferably once a week
  - Verify the database for logical errors at least once in the cycle, preferably once a week
- Keep the verified full offline backup from each cycle in long-term storage, replacing it with a new initialized tape in the pool.
- Although not necessary you can perform additional backups after changes to the database structure and keep these tapes in long-term storage. You can do this after any of the following:
  - A data file is added
  - A data file is moved to a different location
  - A tablespace is created, dropped, or reorganized

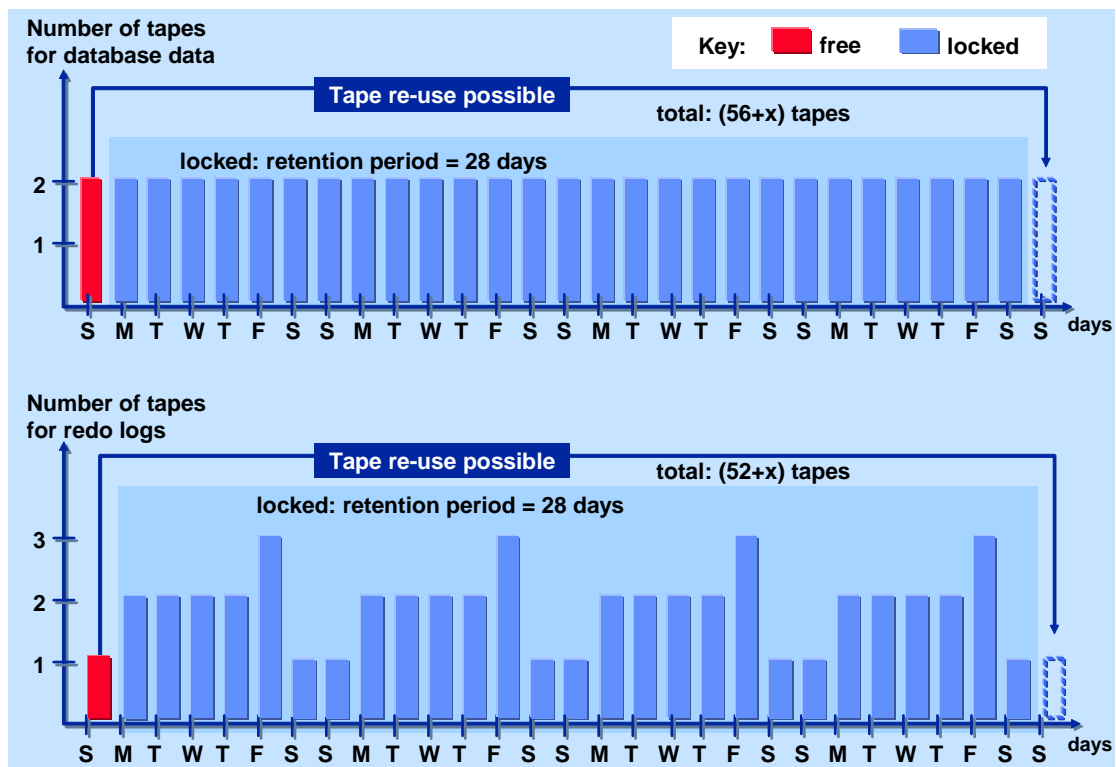


## Backup Approach with Daily Complete Backups

This section describes a sample backup approach for your Oracle database using daily [complete backups](#). This approach is secure and is suitable for small to medium-sized databases.

In this example, the size of the database is less than 200 GB, and daily backups are possible. As the SAP System does not have to be available after 18:00, the backup can be performed offline. Alternatively, the backup can be performed online at a time when the transaction load is low, such as overnight. If DLT drives are used, a full backup of the data (without redo information) fits onto two tapes.

Backup Approach with Daily Complete Backups and 28-Day Retention Period



To be able to deal with a faulty backup, several generations of backups must be available. In this example, the retention period is set to 28 days, so 27 backup generations are always available. The tape pool must also contain several reserve tapes. In this case,  $56 + x$  tapes are required for data backup. The additional  $x$  tapes – approximately 20% of the required number – function as a reserve in case the amount of data to be backed up greatly increases, an extra unplanned backup becomes necessary, or a tape fails.

The redo information generated during the day – buffered on a separate disk that is as large as possible – is also backed up every day using a separate tape pool. As this data is necessary to recover a database after restoring a data backup, the retention period for the tapes must be no less than the retention period for the actual data backup. Particularly in the case of an online backup, you must always back up redo information *directly after* the database backup.

**⚠ Caution**

Without backups of the redo logs, the online backup is worthless.

As the redo data is much more dynamic than the actual data, even more reserve tapes are required. For this example,  $52+x$  tapes are needed, where  $x$  is the number of reserve tapes for redo data. For security reasons, we recommend you to back up redo information *twice*, so the total number of tapes required is  $2 \times (52 + x)$ . The actual number of tapes depends on the hardware implemented and the tape capacity available:

Capacity and Performance of Tapes and Tape Devices		
Tape or Tape Device	Capacity (GB)	Approximate Rate (GB/hour)
IBM 3590/Magstar	20 - 40	10 - 15
DLT 7000	35 - 70	15 - 20

Capacity and Performance of Tapes and Tape Devices		
Tape or Tape Device	Capacity (GB)	Approximate Rate (GB/hour)
DAT (DDS-3)	10 - 20	2 - 4
DST 310/312	50	30 - 50

## More Information

[Backup Approach for Very Large Database with Partial Backups](#)

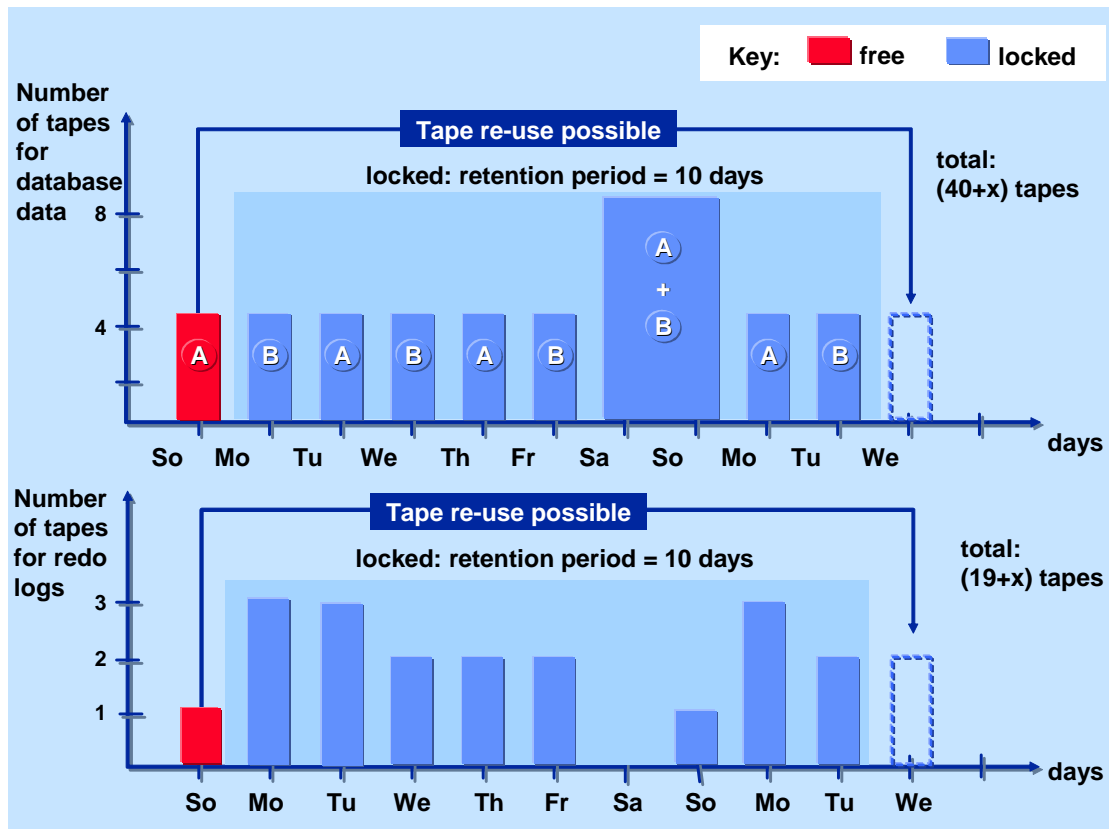
[Backup Approach with One-Day Retention Period](#)

## Backup Approach for Very Large Database with Partial Backups

This section describes a sample backup approach for a very large Oracle database using daily partial backups.

In this example, the database is too large for a complete daily offline or online backup and has to be available 24 hours a day on five working days. Therefore, the backup is spread over two days (part A and part B) and performed online. It runs during the night, as this is the only time when a low transaction load can be expected.

Backup Approach for a Very Large Database with Partial Backups A and B



This strategy is more error-prone than [the first example](#), because the database administrator is responsible for the correct distribution of the data to the partial backups, A and B. The risk of losing data is even higher, because online backups are only consistent in combination with redo information.

The number of tapes required for the data backup is  $40+x$ , where  $x$  is the number of reserve tapes. For security reasons, an additional complete offline backup is performed at the weekend. If this strategy is used with a retention period of seven days, only four generations of backups are available. The redo log files are even more important than in the first example, as the online backups are worthless without them. It is *essential* to back up the redo data twice. Therefore, you need  $2 \times (19 + x)$  tapes for redo information, where  $x$  is the number of reserve tapes.

#### Note

It is possible with [brarchive -cbs](#) to automatically create two copies of backup data using a single set of tapes.

## More Information

[Backup Approach with Daily Complete Backups](#)

[Backup Approach with One-Day Retention Period](#)



## Backup Approach with One-Day Retention Period

This section describes a sample backup approach for your Oracle database using daily full backups but with a retention period of only one day.

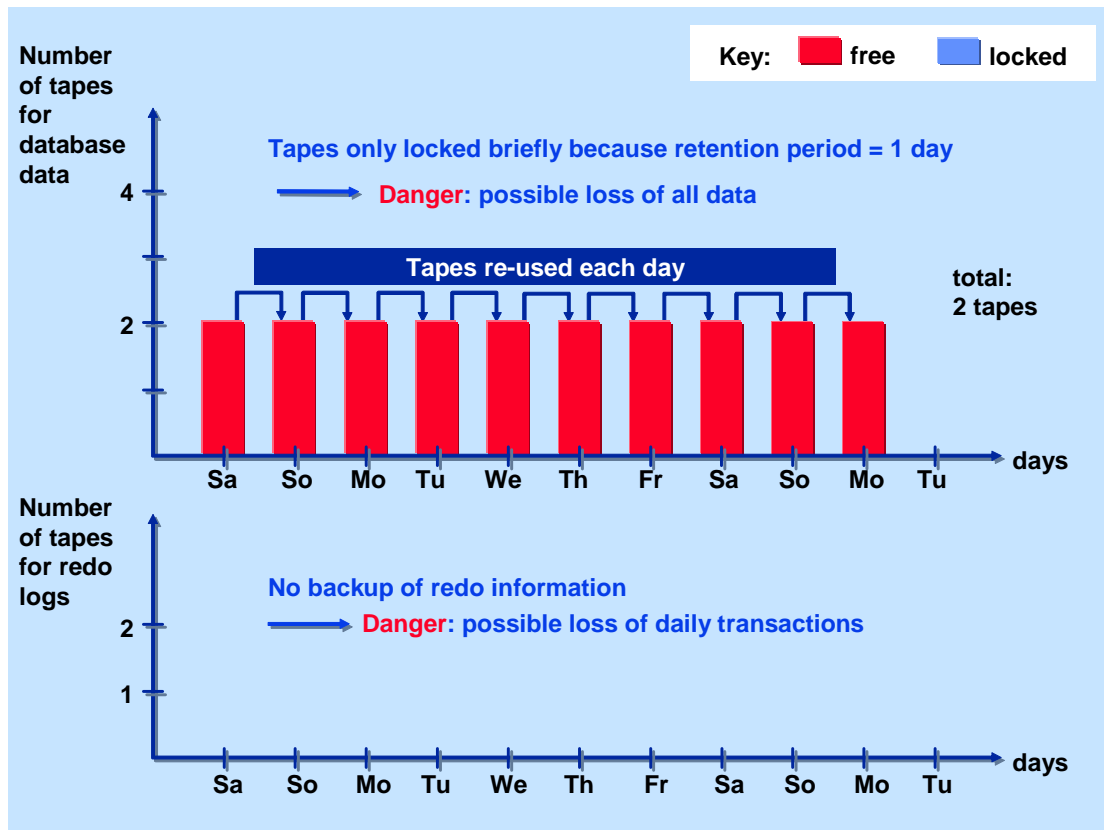
#### Caution

Do *not* follow the example shown in this section. It is included to illustrate a *faulty* backup approach.

In this example, a complete backup is performed offline once a day. However, the retention period is set to 1 day, so the two tapes required are overwritten each day.

In the event of a disk error, the single backup has to be used. If it cannot be read, the database is destroyed. As the redo information has not been saved separately, all transactions executed since the last backup are lost, if a disk failure affects data and redo information.

Faulty Backup Approach



## More Information

[Backup Approach with Daily Complete Backups](#)

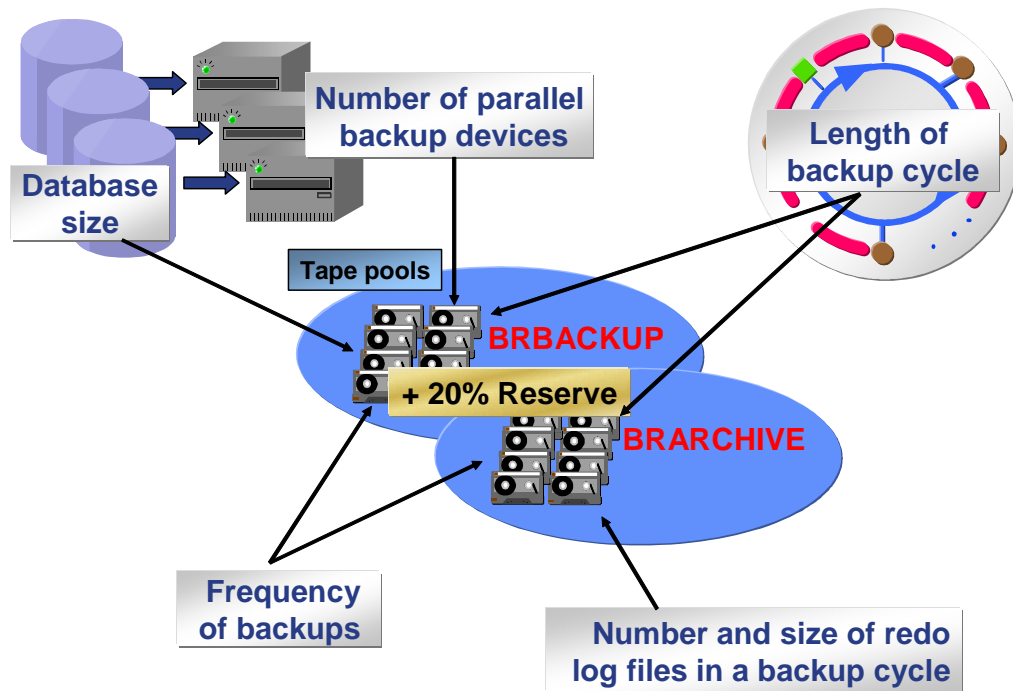
[Backup Approach for Very Large Database with Partial Backups](#)

## Backup Media

You require a pool of tapes for database and offline redo log file backups to back up your Oracle database. Ensure that enough tapes are provided in each tape pool to cover the entire backup cycle. We recommend having 20% more tapes than required to cover database growth and additional backups. Backup tapes can be reused at the end of a backup cycle (that is, normally after 28 days).

Of course, you can also back up the database to disk if you have enough storage space available and later copy it to tape.

The following graphic shows the factors you need to consider when making up a tape pool:



Backup media are normally *locally* connected to the database server. Only back up a production database to a *remote* host if the database is not too large, and the network is stable. You should be able to back up a test database to a remote host without any major problems. You might want to back up the test database to backup devices that are connected to the host on which the production database is running.

For more information on how to manage the tape volumes, see [Volume Management](#).

For more information on data compression, see:

- [Software Compression](#)
- [Hardware Compression](#)

## Integration

If you use an external backup tool, [BRBACKUP](#) calls BACKINT to manage backup media. For more information, see [External Backup Programs](#).

## Volume Management

The tape volumes for the Oracle database are overwritten again at each backup or archive by BRBACKUP or BRARCHIVE. These tools never use the space remaining on the tapes after a backup has finished. New tapes have to be inserted each time you make a backup. For more information on the contents of a volume that has been written by BRBACKUP or BRARCHIVE, see [Used Volumes](#).

BRBACKUP or BRARCHIVE can only use volumes that are correctly [initialized](#). Initialized volumes have an SAP-specific [tape label](#).



The backup volumes must be managed to make sure that they are protected from premature deletion, ensuring that you have access to the required volumes at all times. For more information, see [Volume Expiration Period](#).

## Activities

You can choose volumes as follows:

- [Select volumes manually](#)
- [Select volumes using external tools](#)
- [Select volumes automatically](#)



## Volume Initialization

You need to initialize the tape volumes for Oracle database backup with SAP tools. [BRBACKUP](#) or [BRARCHIVE](#) writes an SAP-specific label – that is, a file with the name `.tape.hdr0` – to the volume concerned. This label file is read when the volume is checked. If the label file does not exist, the check fails and the volume is rejected.

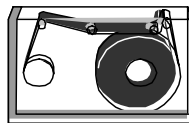
The following graphic summarizes volume initialization:

■ Profile `init<SID>.sap` contains the tape names:

```
...  
volume_backup = (<SID>B01,<SID>B02...  
volume_archive = (<SID>A01,<SID>A02...  
...
```

■ Write the label to the tape that also contains the tape name

`.tape.hdr0`



■ Initialize new tapes, non-SAP tapes, or locked tapes:

`brbackup -i force` or `brarchive -i force`

■ Rename non-locked tapes:

`brbackup -i -v <tape name>` or `brarchive -i -v <tape name>`

## Features

Several devices can be used for initializing volumes. The backup devices defined in the `init<DBSID>.sap` profile in parameters `tape_address` and `tape_address_rew` (or in `tape_address_arch` and `tape_address_rew_arch`) are used serially during a volume

initialization. For more information, see the parameters [tape\\_address](#) and [tape\\_address\\_rew](#). The volumes of all the available backup devices can be changed at the same time.

 Caution

The information in the label is overwritten and the entire tape contents lost if you write to a BRBACKUP or BRARCHIVE volume directly using other tools.

## Activities

 Note

You only have to initialize the following:

- New tapes
- Tapes that have never been used before by BRBACKUP or BRARCHIVE.

Only initialize the volumes once, not repeatedly before every backup. However, to change the name of a volume, you have to reinitialize it. You cannot change volume names during a backup.

1. You initialize all tape volumes to be used by BRBACKUP or BRARCHIVE for the first time. If you want to use automatic tape management, the volume names must be listed in the `init<DBSID>.sap` parameters [volume\\_backup](#) or [volume\\_archive](#).

 Recommendation

We recommend you to write paper labels on the volumes so that you can identify them more easily.

2. You enter the following options during an initialization:

o `-i|-initialize`

Renames volumes that have already been initialized. Only possible for volumes with an expired expiration period. For more information on expiration period, see [Volume Expiration Period](#).

o `-i|-initialize force`

Initializes new volumes or volumes not yet used by BRBACKUP or BRARCHIVE. The expiration period check is not active. Important: This option can also be used to reinitialize locked volumes, which you should never do. If you initialize BRBACKUP or BRARCHIVE tapes with the addition `force`, the `tape_use_count` stored in the tape label is set to 1. Otherwise this value is increased accordingly.

o `-v|-volume`

Enters the names of volumes for initialization.

o `-n|-number`

Specified number of volumes for initialization.

The following examples show how you can use the above options:

### Example

Initialization of volumes with the volume names specified in `volume_backup/volume_archive`:

Use the BRBACKUP or BRARCHIVE option `-i [force]`. Mount a volume and enter the following command:

```
brbackup|brarchive -i [force]
```

The volumes are initialized in sequence with the names specified in the parameters `volume_backup/volume_archive`.

New and non-BRBACKUP or BRARCHIVE volumes must always be initialized with the additional specification `force`.

If you initialize a volume without this additional specification, the expiration period of the volume is checked. Locked volumes or volumes without labels are rejected. Volumes that are not locked are renamed.

### Example

Initialization of 5 volumes with the first 5 volume names specified in `volume_backup/volume_archive`:

```
brbackup|brarchive -i [force] -n 5
```

### Example

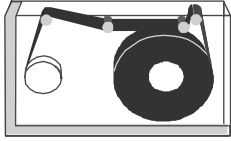

Initialization of volumes with a specific name:

```
brbackup|brarchive -i [force] -v <name1>[,<name2>]...
```

All the volumes for initialization must be mounted in the backup device in the correct order.

## Volume Label Check

You must first [initialize](#) tape volumes before you can use them to back up your Oracle database with BRBACKUP or BRARCHIVE. When BRBACKUP or BRARCHIVE writes to a volume, it first checks the volume label. The following graphic shows the volume label and check:

<b>Volume label contents</b>	<ul style="list-style-type: none"> <li>■ Volume name</li> <li>■ Timestamp of last backup</li> <li>■ Database instance name (ORACLE_SID)</li> <li>■ BRBACKUP or BRARCHIVE action ID (coded timestamp of BRBACKUP or BRARCHIVE log names)</li> <li>■ BRBACKUP or BRARCHIVE function ID (extension of the detail log file name of BRBACKUP or BRARCHIVE)</li> <li>■ Number of backups performed</li> </ul>	
<b>Volume checks by BRBACKUP or BRARCHIVE</b>	<ul style="list-style-type: none"> <li>■ Volume name</li> <li>■ Expiration period</li> <li>■ Use count</li> </ul> <div style="display: flex; align-items: center; gap: 10px;"> <div style="color: red; font-size: 2em;">▶</div> <div style="color: red; font-weight: bold;">Error</div> </div> <div style="display: flex; align-items: center; gap: 10px;"> <div style="color: orange; font-size: 2em;">▶</div> <div style="color: orange; font-weight: bold;">Warning</div> </div>	 <div style="border: 1px solid black; padding: 5px; margin-top: 5px;"> <p style="background-color: #00a0c0; color: white; margin: 0;">Profile <i>init&lt;SID&gt;.sap</i>:</p> <p style="margin: 0;">...  <b>expir_period</b> = 28  <b>tape_use_count</b> = 100          ...</p> </div>

## Activities

You can display the contents of a volume label as follows:

```
brbackup|brarchive -i show -n 1
```

The volume label is always the first file on a volume. This file has a format specific to BRBACKUP or BRARCHIVE and is written to the volume using cpio. The label file is called .tape.hdr0.

BRBACKUP or BRARCHIVE performs the following volume label checks:

1. Before BRBACKUP or BRARCHIVE writes to a volume, it reads the volume label file. If this file does not exist, you have to either initialize the volume or mount another volume.

The following volume label information is checked:

- Volume name
  - An error message (message number BR0216E) is issued if you have mounted a volume with an incorrect name. If a [scratch volume](#) was requested, you can mount any volume.
- Expiration period
  - An error message (message number BR0217E) is issued if the configured expiration period – number of the days specified in the *init<DBSID>.sap* parameter [expir\\_period](#) that must have passed before the volume can be used again – has not ended yet.
- Volume use count

A warning message (message BR0235W) is issued if the volume has been overwritten more frequently than specified in the `init<DBSID>.sap` parameter [tape\\_use\\_count](#).

2. After a backup or archive to a volume is completed, the volume label is checked once more. This is to detect volume, tape device, driver, or hardware errors that would prevent a successful backup, but would not cause an error message to be issued. In extreme cases, several sequential backups might be unusable if these errors are not recognized. The program checks whether the name of the database instance, the action ID, and the function ID match the current values.

## Volume Expiration Period

The volume expiration period for backups of the Oracle database is defined by the `init<DBSID>.sap` parameter [expir\\_period](#). This specifies the period in days during which a volume is locked, that is, cannot be used. When the period expires, you can reuse the volume for a backup.

### Example

`expir_period = 28` means that writing to a volume is possible 28 days after the volume was mounted and used. For example, if you use a volume on Monday 1st July, you cannot use it for another backup until Monday 29th July.

The start time of BRBACKUP or BRARCHIVE determines the first day of the lock for all volumes used for a backup. The time when the volume was first written to does not matter. The expiration period always expires at midnight (that is, 24:00 using the 24-hour clock) of the last day of the lock.

### Recommendation

SAP recommends an expiration period of at least 28 days (the default value is 30 days).

### Recommendation

If you set an expiration period of 0 days, this means that the volume is not locked. The volumes can be overwritten on the same day. Therefore, do not set `expir_period` to 0.

## Structure

The current value of `expir_period` is decisive for whether or not a volume is locked, not the value of the parameter during the backup. This means that the backup volumes are locked for `n` days after the last backup operation, where `n` is the current value of `expir_period`. If the value of `expir_period` is changed, the expiration period for all volumes is automatically changed.

Volumes can be locked physically and logically:

- Physical lock

The volume generation date specified on the volume label is decisive for a physical lock. This generation date is determined when the volume label is written (when a

backup on this volume was started). A volume is locked physically when the system checks the volume label and finds that the expiration period for the volume has not ended yet, that is, the value of the current date is less than the total of the volume generation date stored in the volume label and the value of `expir_period`.

- Logical lock

The internal information in the BRARCHIVE or BRBACKUP logs is decisive for a logical lock. The logs are updated when a database file has been backed up successfully. A volume is locked logically when the automatic volume management system checks the volume and finds that the expiration period stored internally has not ended yet; the value of the current date is less than the total of the volume generation date stored in the BRBACKUP or BRARCHIVE logs and the value of `expir_period`. Under certain circumstances, discrepancies may occur between the physical and logical locks.

You can unlock volumes as follows:

- Unlocking a physically locked volume

During a backup, the volume label was written to the volume but the backup was terminated before the first database file could be written to the volume.

This means that the volume is locked physically but not logically. It is selected from the list by the automatic volume management system – with [volume\\_backup](#) or [volume\\_archive](#) – but is rejected when the physical volume label check takes place. The volume can be reinitialized (with the same name) in order to cancel the physical lock, as follows:

1. Temporarily set the `init<DBSID>.sap` parameter `expir_period` to 0, to circumvent the physical lock.
2. Start BRBACKUP or BRARCHIVE, for example, as follows:  

```
brbackup|brarchive -i -v <volume name>
```
3. Reset the `expir_period` parameter to its previous value.

When performing this operation, do not use the `-i force` option, as this causes the volume use count stored in the volume label to be lost.

- Unlocking a logically locked volume

A volume was reinitialized before the expiration period ended (for example, with the option `-i force`). This means that the volume is no longer locked physically. However, it is not selected by the automatic volume management system because it is still locked logically.

If you still want to use this volume before the logical lock has expired, you can switch off automatic volume management temporarily by mounting the volume on the backup device and starting a backup with the following command:

```
brbackup|brarchive -v SCRATCH
```

For more information, see:

- [brbackup -i](#)
- [brbackup -v](#)
- [brarchive -i](#)

- [brarchive -v](#)

## Used Volumes

This section describes the tape volumes written by BRBACKUP or BRARCHIVE when you back up your Oracle database or back up the offline redo log files.

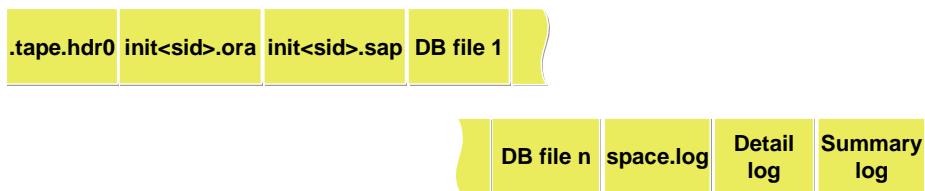
### Structure

After successfully backing up the Oracle database with BRBACKUP and BRARCHIVE, the volumes produced contain the following files:

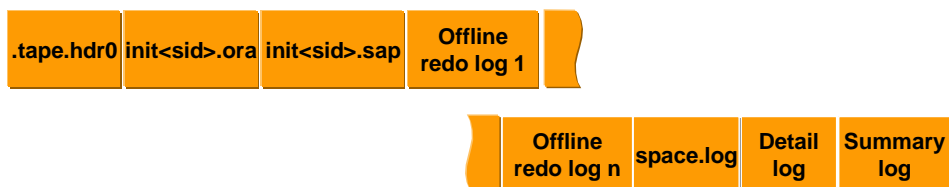
- Label: `tape.hdr0` (position 1)
- `init_ora`: Initialization file `init<DBSID>.ora` (position 2) and `spfile`.
- `init_sap`: Initialization file `init<DBSID>.sap` or the profile file defined under [-p|-profile](#) (position 3).
- Database files (data files, offline redo log files, non-database files)
- `space_log`: BRSPACE summary log `space<DBSID>.log`, database structure log `struc<DBSID>.log`, and parameter change log `param<DBSID>.log`
- `det_log`: Detail log written by BRBACKUP or BRARCHIVE (second-to-last position)
- `sum_log`: Summary log written by BRBACKUP or BRARCHIVE (last position)

Tape Layout for BRBACKUP and BRARCHIVE

#### BRBACKUP:



#### BRARCHIVE:



Information about used volumes is contained in the BRBACKUP and BRARCHIVE logs:

- File system logs
  - Detail log, created for each backup

- Summary log, extended during each backup
- Database logs
  - Table SDBAH with information about backup runs
  - Table SDBAD with information about backup files

## Integration

These log files should only contain information that was written by BRBACKUP or BRARCHIVE. Do *not* change this information manually. The information in database tables SDBAH and SDBAD is evaluated by the Computing Center Management System (CCMS), for example.

For more information, see [Logs for BR\\*Tools](#).

For [automatic volume management](#), BRBACKUP uses the corresponding database log to select available volumes. BRARCHIVE uses the summary file system log. BRARCHIVE cannot rely on the database logs because it also runs when the database has been shut down.



## Scratch Volume

When BRBACKUP or BRARCHIVE requests a scratch volume for backup, this means that you can use any volume for which the expiration period has ended. It does *not* mean that you have to mount a volume with the name `SCRATCH`.

You can also initialize a volume with the name `SCRATCH` explicitly, for example, using `brbackup -i -v SCRATCH`. Such a volume is always accepted, even when a volume with another name is requested. The program still makes sure that the expiration period has expired. In this case the tape mounted is assigned the name of the requested tape. Only use this option in exceptional cases, for example, when an additional tape is requested unexpectedly during a backup operation.

A backup on a tape named `SCRATCH` should never exist. However, this situation might occur when a scratch tape is requested and a tape named `SCRATCH` is mounted.



## Selecting Volumes Manually

You can select the volumes for an Oracle database backup manually. You can do this if the BRBACKUP or BRARCHIVE [automatic volume management](#) is deactivated by using the reserved volume name `SCRATCH`. However, BRBACKUP or BRARCHIVE still checks the expiration period of the volume, and only allows you to use volumes for which the expiration period has ended.

For more information, see [Scratch Volume](#).

## Prerequisites

- Make sure that initialized volumes are available for the backup.



- Determine whether the required expiration period has been configured in profile parameter [expir\\_period](#) and change the value when necessary.

## Procedure

1. You can start the backup with BRBACKUP or BRARCHIVE in one of the following ways:

- Using the profile `init<DBSID>.sap`

1. Enter one of the following parameter values in profile `init<DBSID>.sap`:

```
volume_archive = SCRATCH
```

```
volume_backup = SCRATCH
```

For more information, see [volume\\_backup](#) and [volume\\_archive](#).

2. Start BRBACKUP or BRARCHIVE.

- Using the option `-v SCRATCH`

Leave the profile unchanged (a volume pool can be defined).

Start BRBACKUP or BRARCHIVE with the option `-v SCRATCH`.

## Result

BRBACKUP and BRARCHIVE request the number of scratch volumes needed for the backup (message BR0104I), expired volumes with any name. The volume names in the labels are *not* changed by the backup operation. Any expired BRBACKUP or BRARCHIVE volumes are accepted. See [Volume Expiration Period](#).

### Note

If you use scratch volumes, it might make sense to include the weekdays or days of the month in the volume names. This helps to make the names more meaningful.



## Selecting Volumes with External Tools

You can also use external tools to determine the names of the volumes relevant for the Oracle backup. This might involve an external volume management system or simply a shell script. You can do this if you have deactivated BRBACKUP or BRARCHIVE [automatic volume management](#) by calling them with the option `-v`. However, the expiration period of the volumes is checked anyway.

## Prerequisites

The external tool that you use for volume selection must make sure that only non-locked volumes are suggested for backup. Otherwise, BRBACKUP or BRARCHIVE terminate if they do not find enough free volumes. Make sure that initialized volumes are available for the backup.

## Procedure

1. Determine whether the required expiration period has been configured in profile parameter [expir\\_period](#). Change the value if necessary.
2. Start BRBACKUP or BRARCHIVE with the option `-v <volume list>`. Before starting the backup, BRBACKUP or BRARCHIVE checks whether the mounted volumes agree with those in the volume list and whether the expiration period has expired.

BRBACKUP or BRARCHIVE only use the volumes listed with option `-v` for one backup.

You can select the volume names yourself by defining them in the call with the option `-v|-volume` (using the naming convention, for example, as suggested for automatic volume management).

### Example

```
brbackup -v C11B141,C11B142,C11B143 brarchive -ssd -v  
C11A141,C11A142
```

You can also give the volumes other names. One option would be to include the day of the backup in the volume names. BRBACKUP or BRARCHIVE can use the following sample script to assign the volume names to the day of the month on which the backup was started.

### Example

The name `<DBSID><X><dd><n>` is made up of:

DBSID = ORACLE\_SID (name of the database instance)

X = A for BRARCHIVE or X = B for BRBACKUP

dd = day of month

n = next volume number within one backup

Here is the script:

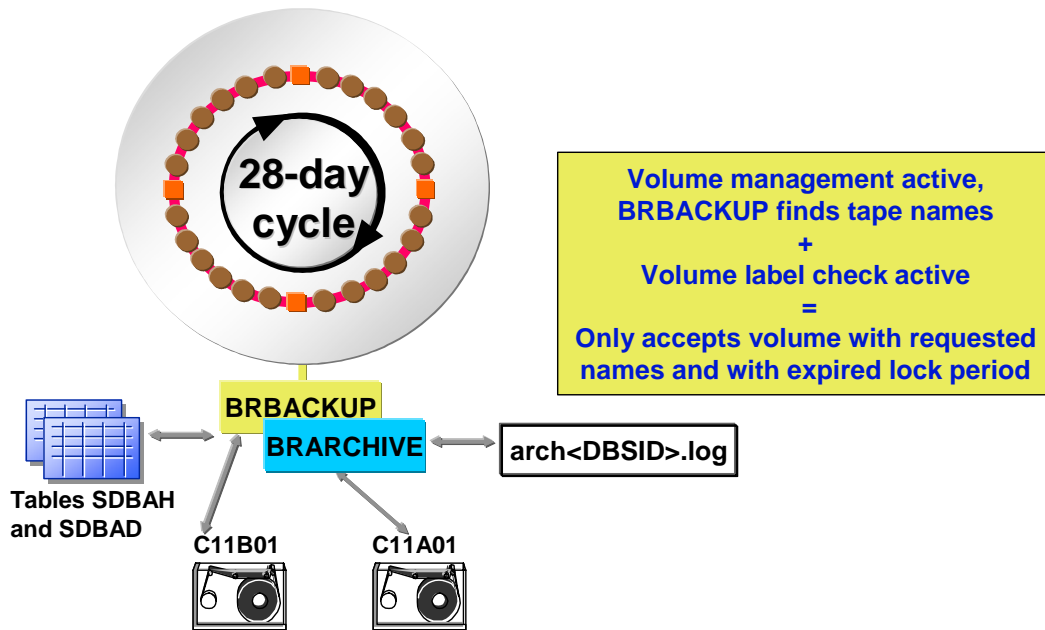
```
day=`date | cut -f 3 -d " "` if [ ${day} -le 9 ]; then  
day=`echo 0${day}`; fi brbackup -v  
C11B${day}1,C11B${day}2,C11B${day}3 -c brarchive -ssd -v  
C11A${day}1,C11A${day}2 -c
```



## Selecting Volumes Automatically

BRBACKUP or BRARCHIVE automatically selects the volumes that are recommended for the next backup. The name of the mounted volume is compared to the name found in the volume pool. In addition, the procedure ensures that current backups are not overwritten.

Automatic Volume Management



## Prerequisites

When you use automatic volume management, do not use weekdays or days of the month in the volume names, because BRBACKUP or BRARCHIVE does not check whether or not a holiday fell in the period since the last backup. SAP recommends that you include the name of the database instance and a sequential number in the tape name, for example <DBSID>A<nn> for BRARCHIVE tapes, <DBSID>B<nn> for BRBACKUP tape volumes.

## Procedure

1. Define a pool of the volumes available for the backup. To do this, enter the corresponding volume names in profile parameter `volume_backup` or `volume_archive`. All the volumes defined there should physically exist. [Initialize](#) new volumes when necessary.
2. Determine whether the required [expiration period](#) has been configured in profile parameter `expir_period`. Change the value when necessary.
3. Start the backup with BRBACKUP or BRARCHIVE without option `-v|-volume`. Only volumes that are not locked are selected from the pool of available volumes.
4. Mount the requested volumes. During the next runs, BRBACKUP and BRARCHIVE attempt to write to all the available volumes in the pool in sequence.
5. You can display the names of the volumes required for the next run as follows:

```
brbackup -q
```

```
brarchive -q
```

6. You can check whether you have mounted the correct volume in the backup device with:

```
brbackup -q check
```

```
brarchive -q check
```

These commands do not actually start the backup. In particular, you can use these options before scheduling a backup with `CRON` or a similar tool.

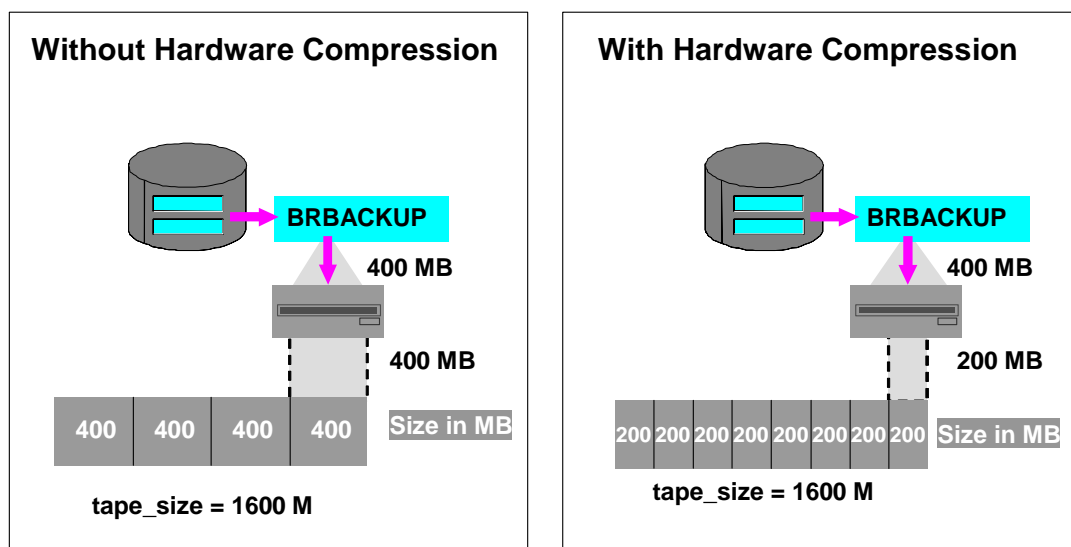
For more information, see [-ql-query](#).

If you want to [select the volumes manually](#) you can deactivate automatic tape management. To do this, set the `init<DBSID>.sap` parameters [volume\\_backup](#) and [volume\\_archive](#) to `SCRATCH` or call `BRBACKUP` or `BRARCHIVE` with the option `-v SCRATCH`. For more information, see [Scratch Volume](#).

## Tape Volume Size

The parameter [tape\\_size](#) always defines the physical tape size (tape length \* write density). The physical tape size is the total volume (in MB or GB) of the data that can actually be written to a volume without compression. Consequently, different amounts of data can be stored on the same tape size, depending on whether you back up with or without compression. Since `BRBACKUP` or `BRARCHIVE` takes this information into account, you do not need to change the physical tape size if you use software compression or hardware compression (for tape devices with hardware compression).

Effect of Hardware Compression on Tape Volume Size



**tape\_size defines the maximum data volume that can be physically written to a tape**


Pay attention to the following:

- If you require several tapes for your database backup, `BRBACKUP` uses the value of parameter `tape_size` to distribute the files for backup to the volumes. During serial backups, the program makes sure that the physical end of tape is not reached. During parallel backups, the program also optimizes the file distribution.
- A single file should not be bigger than the tape size (possibly after compression).

- If parameter `tape_size` is configured too small, the backup program requires more volumes than are actually necessary.
- When parameter `tape_size` is too large, `cpio` reaches the physical end of tape. See [cpio Continuation Tape](#). In this case, you must correct the parameter value.
- When using tape devices with hardware compression, always leave a reserve of around 10% of tape capacity for any errors in the calculation of the compression rate. The compression rate is estimated based on an analogous software compression and is therefore not exactly the same as the actual compression rate of a hardware compression.

 **Caution**

The estimation errors can be larger after a database installation, a tablespace extension, or a reorganization. If the physical end of tape is reached in such a situation, reduce the value of `tape_size` by an additional 10 to 20% of tape capacity.

 **Example**

Typical values for `tape_size` for tape devices *with* hardware compression:

Tape Type	tape_size
60 m DAT tape / 4 mm DDS-1	1200M
90 m DAT tape / 4 mm DDS-1	1800M
120 m DAT tape / 4 mm DDS-2	3800M
112 m Videotape / 8 mm	2000M
112 m Videotape / 8 mm/ high density	4500M
3590 IBM tape	10000M
DLT 2000 – 10/20 GB	10000M
DLT 2000XT – 15/30 GB	15000M
DLT 4000 – 20/40 GB	20000M
DLT 7000 – 35/70 GB	35000M

Make sure that you set the parameters required for using [hardware compression](#).

 **Example**

Typical values for `tape_size` for tape devices *without* hardware compression:

Tape Type	tape_size
-----------	-----------

Tape Type	tape_size
60 m DAT tape / 4 mm DDS-1	1200M
90 m DAT tape/ 4 mm DDS-1	1800M
120 m DAT tape/ 4 mm DDS-2	3800M
112 m Video tape/ 8 mm	2000M
112 m Video tape/ 8 mm / high density	4500M
3490 IBM tape	700M

## Hardware Compression

When backing up your Oracle database to tape, always use hardware compression if your tape devices support it. This reduces backup time because more data can be written to a single volume. Tape units with hardware compression are now industry-standard. The compression method used is normally based on the Lempel-Ziv algorithm. A few operating systems also support hardware compression for disk.

You can also use [software compression](#).

### Features

The amount of data that can actually be written to a tape depends on the compression rate. The average compression rate is between 3 and 5, but this can vary as follows:

- It is lower if the data is mostly already compressed. The compression rate does not improve if the files are compressed again.
- It is higher if new or relatively empty database files are compressed.

BRBACKUP can optimize a backup on tape units with hardware compression if the current compression rates are known before starting the backup. To do this, use [brbackup -k only](#) to approximate compression rates. Repeat this at least once a month to update the compression rates.

After a large data transfer or a reorganization of a tablespace, you must compress the affected tablespaces again. If a database file has no essential changes in two consecutive compression runs, you can consider the compression rate to be constant. You only need check the compression rate again after a longer period (for example, after a year). You can exclude these files from regular compression since the compression rate stays constant.

#### Note

If you are using the BACKINT interface to an external backup tool, the above is not relevant and you do not need to do it.

To reduce compression time for large databases, you can reduce the amount of data by compressing database files individually, or excluding them from compression. You can also run multiple compressions in parallel. You can use parallel compressions to determine the compression rates (that is, without starting a backup).

## Activities

For hardware compression, you set the `init<DBSID>.sap` parameter `compress` to hardware. Be sure to enter the correct address for tape devices with hardware compression in the parameters `tape_address` and `tape_address_rew` (for example, a lower-case `c` can be important).

## Example

- Device type:

```
backup_dev_type =  
tape|pipe|tape_auto|pipe_auto|tape_box|pipe_box
```

- Addresses for the tape device:

```
tape_address = (/dev/rmt/0hnc) tape_address_rew =  
(/dev/rmt/0hc)
```

- Compression parameters:

```
compress = hardware
```

- Tape size:

```
tape_size = 16G
```

## More Information

[backup\\_dev\\_type](#)

[tape\\_address](#)

[tape\\_address\\_rew](#)

[compress](#)

[tape\\_size](#)



## Software Compression

When backing up your Oracle database to tape or disk, you can use software compression. Use software compression only if you have no tape devices with [hardware compression](#). Using hardware and software compression at the same time does *not* improve compression rates.

- Advantages

If you make remote backups over a network, using software compression significantly reduces the network load.

- Disadvantages

- High CPU utilization
- Lengthy backups due to compression process

## Features

### Software Compression and Remote Backup to Parallel Tape Devices

You can use multiple tape devices when you use software compression or make a remote backup to a remote host. This means that unattended backup of large databases is possible even if you do not have a tape device with hardware compression or want to perform remote backups.

If several tapes are required for a backup with software compression or for a remote backup, the existing tape devices are used in parallel, assuming that the number of parallel copy processes was not reduced by changing the [exec\\_parallel](#) parameter. The tape devices must be defined in parameters [tape\\_address](#) and [tape\\_address\\_rew](#).

### Size of the Compression Directory

When software compression is used (`compress = yes`, backup not on disk), BRBACKUP uses the compression rates to determine the space required in the compression directory. The free space must be at least as large as the largest compressed file. The calculated compression rates are stored in a detail log and in the database table SDBAD.

For more information, see [Logs for BR\\*Tools](#).

#### Caution

When BRBACKUP is started for the first time, compression rates are not available. In this case BRBACKUP uses internal default values that are usually smaller than the actual compression rates. For successful compression, make sure that the compression directory has at least as much free space as the largest database file needs before the compression.

If you specify `compress = only` (determine the compression rates), no disk space is required in the compression directory. The sizes are evaluated by reading the compressed files directly (using redirection). As a prerequisite for this, the redirection character ">" must be used in the parameter `compress_cmd` (as set already by default).

## Example

- Device type:

```
backup_dev_type =
tape|pipe|tape_auto|pipe_auto|tape_box|pipe_box
```

- Tape device addresses:

```
tape_address = (/dev/rmt/0mn) tape_address_rew = (/dev/rmt/0m)
```

- Compression parameters:

```
compress = yes
```

- Tape size:

```
tape_size = 18G
```



## More Information

[backup\\_dev\\_type](#)

[tape\\_address](#)

[tape\\_address\\_rew](#)

[compress](#)

[tape\\_size](#)



## Backup Methods

This section describes special methods to perform backup of database files and offline redo log files for the Oracle database:

- [Backup to Multiple Disks](#)
- [Backup to a Remote Disk](#)
- [Backup to a Remote Tape Device](#)
- [Two-Phase Backup](#)
- [Structure-Retaining Database Copy](#)
- [Parallel Backup](#)
- [Unattended Backup](#)
- [BRBACKUP and BRARCHIVE Backups in One Run](#)
- [Grouping Offline Redo Log Files](#)



## Backup to Multiple Disks

You can use [BRBACKUP](#) for Oracle backup to multiple disks if the space available on one disk or logical volume is not sufficient.

To do this, you use the `init<DBSID>.sap` profile parameter `backup_root_dir` to specify the directories on the different disks where you want to save your database files.



### Example

This is an example of the entries required in the `init<DBSID>.sap` initialization profile for parallel backup:

- Device type: `backup_dev_type = disk`
- Backup directories: `backup_root_dir = (/backup/dir1, /backup/dir2)`
- Compression parameters: `compress = no|yes`

BRBACKUP normally uses all the directories specified in `backup_root_dir` in parallel to back up the database files. The number of copy processes corresponds to the number of disks. Since BRBACKUP attempts to optimize the speed of the backup, all the hard disks specified in [backup\\_root\\_dir](#) are written to, except if the number of files you want to back up is smaller than the number of disks. You can change this setting with the `init<DBSID>.sap` parameter [exec\\_parallel](#) or the command option [-e|execute](#).

BRARCHIVE uses the value of parameter `archive_copy_dir` as the destination for offline redo log file copies on disk.

## More Information

[backup\\_dev\\_type](#)

[compress](#)



## Backup to a Remote Disk

You can perform an Oracle backup to a remote disk directly with [BRBACKUP](#) or [BRARCHIVE](#), or as part of an incremental backup using the [Oracle Recovery Manager \(RMAN\)](#).

### Recommendation

We only recommend remote backup when your network is very fast and stable. Therefore, we do not recommend this procedure for production systems in most cases. However, you can use it to back up test systems.

## Prerequisites

- The target directories for the backup defined in the parameter [stage\\_root\\_dir](#) and [archive\\_stage\\_dir](#) must exist on the remote host.
- The name of the remote host and the relevant user must be specified in the initialization profile with the parameters [remote\\_host](#) and [remote\\_user](#).
- If you use FTP as the transmission program, a password is needed for the remote host. Do one of the following for this:
  - Specify the password in the [remote\\_user](#) parameter.

```
remote_user = "<user_name> <password>"
```
  - Let BRBACKUP use the password of the database user, which happens if you do not explicitly specify a password in the `remote_user` parameter. Make sure that the password of the operating system user is the same. However, to achieve this, you must enter the password in the option `-u`. It is *not* possible to use the OPS\$ mechanism.
- No password is needed if you use the RCP command for the remote disk backup. A prerequisite for a successful RCP call is, for example, the following entry in the `.rhosts` file, which is located in the `HOME` directory of the remote user on the remote host:

```
<host_name> <user_name>
```

For more information, see your operating system documentation.

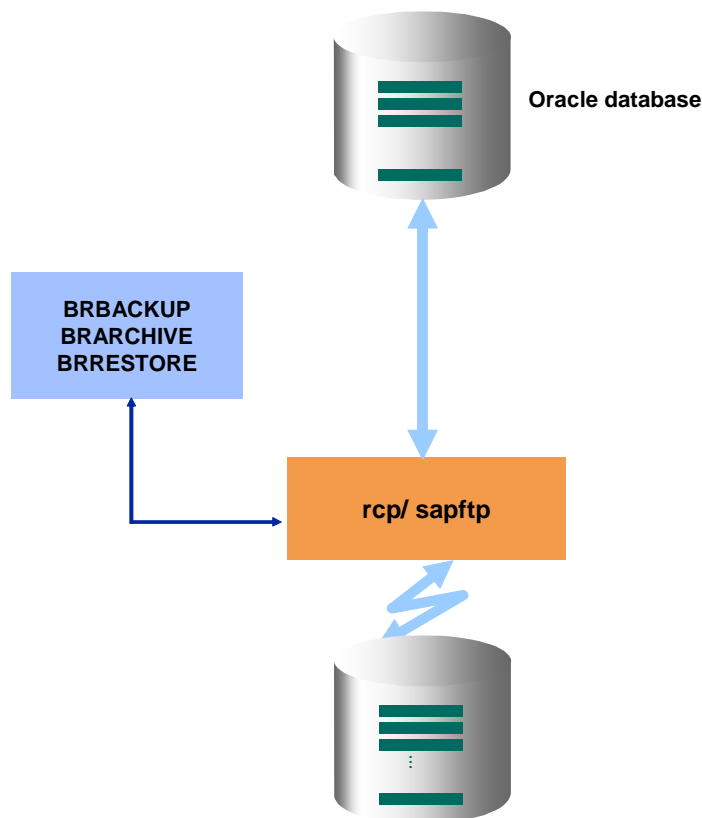
## Features

### Remote Disk Backup with BRBACKUP

A remote disk backup removes the need for a Network File System (NFS) disk mount, if the backup is across the network. The NFS protocol is relatively insecure, which means that the backup must be verified. In contrast, RCP and FTP are relatively secure ways of transferring data over the network, which means that you do not have to verify the backup (but you can still if you want).

You need to meet the prerequisites above and make the following entries in the [initialization profile init<DBSID>.sap](#):

```
backup_dev_type = stage|stage_copy|stage_standby stage_copy_cmd =  
rcp|ftp|scp
```



A remote backup to disk is particularly useful with the [standby database](#) and hierarchical storage management systems.

### Incremental Remote Backup with RMAN

You can use [RMAN](#) to perform an incremental disk backup on a remote host.

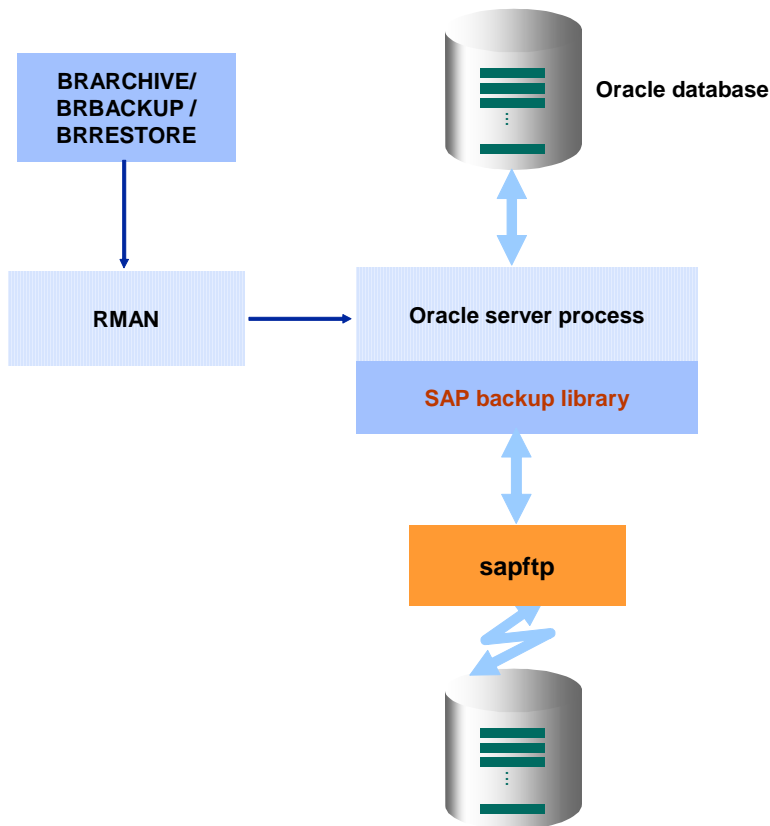
You need to meet the prerequisites above and make the following entries in the [initialization profile init<DBSID>.sap](#):

```
backup_dev_type = stage
```

```
backup_mode = incr
```

You must have the SAP backup library installed on your system.

FTP is always used automatically as the transmission program for incremental backup to remote disk.



## More Information

[RMAN Backup with the SAP Backup Library](#)

[RMAN-Relevant Profile Parameters](#)



## Backup to a Remote Tape Device

With [BRBACKUP](#) and [BRARCHIVE](#) you can back up files of the Oracle database located on a UNIX system to a remote tape device that is connected to a UNIX host in the network. You can use several tape devices on the remote host for backup and these are used in parallel.

The UNIX versions of the local and remote hosts need not be identical. For example, if the database runs on an HP-UX host, you can perform the backup on an AIX host.

Backup to a remote tape device is not supported for remote hosts running a Windows operating system.



### Recommendation

We only recommend remote backup when your network is very fast and stable. Therefore, we do not recommend this procedure for production systems in most cases. However, you can use it to back up test systems.

## Prerequisites

- Make sure that no additional messages (that is, not belonging to command output) are issued on remote login (for example, from or `.cshrc`). Test the command `remsh|rsh <host_name> date`. Only one line with the output of the `date` command should be displayed.
- For a successful remote shell call, check that there is an entry as follows in the `.rhosts` file, which is located in the `HOME` directory of the remote operating system user on the remote host:

```
<local_host_name> <local_user_name>
```

where:

`<local_host_name>` is the host where the database runs

`<local_user_name>` is the operating system user who starts the backup

For more information, see your operating system documentation.

## Activities

The individual database files are transferred to the remote host using a remote shell call, called a “pipe”. You define the remote host with the `init<DBSID>.sap` parameter [remote\\_host](#) and the user with the [remote\\_user](#) parameter.

On the remote host the files are written to tape using the UNIX `dd` command. You define the `dd` command in the profile `init<DBSID>.sap`, as in the following example:

### Example

```
copy_out_cmd = "dd bs=64k of=$"
```

```
copy_in_cmd = "dd bs=64k if=$"
```

The number of parallel copy processes normally corresponds to the number of backup devices available. You can change this setting with the `init<DBSID>.sap` parameter [exec\\_parallel](#) or the command option [-e|execute](#).

## More Information

[Backup to a Remote Disk](#)

## Two-Phase Backup

As an alternative to a direct Oracle backup to tape you can perform a two-phase backup with full support from [BRBACKUP](#) or [BRARCHIVE](#). This backup strategy enables you to easily make a disk backup, as well as having several copies of this and previous backups available. For a recovery, `BRRECOVER` can, in this case, access the disk backup directly. If a restore from tape is required, the files are directly copied to the original directories by [BRRESTORE](#).

### Note

You can also perform the second phase of the two-phase backup with external tools - that is, backup programs, operating system tools, and so on. In this case, you have full responsibility for the complete and correct execution of this phase.

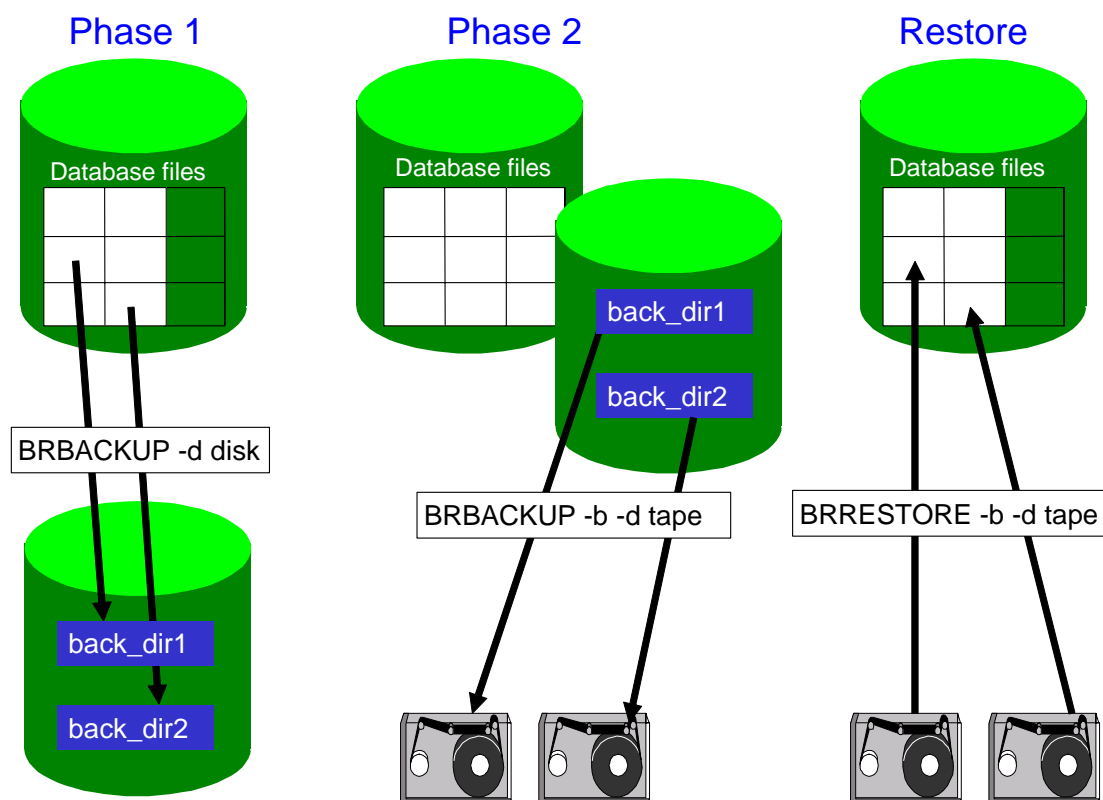
 Note

You cannot use the BACKINT interface to copy backups from raw devices or compressed disk files to tape. If you use raw devices or compressed disk files, you must save them directly to tape with BRBACKUP or operating system tools.

## Features

### Two-Phase Database Backup with BRBACKUP

- Phase 1  
BRBACKUP backup to disk
- Phase 2  
BRBACKUP backup from disk to tape
- Restore phase  
BRRESTORE restore from tape directly to the original directories or restore from the disk to the original directories



To back up to tape (phase 2), start BRBACKUP with the relevant command option. For more information, see [-b|-backup](#).

## Features of Two-Phase Database Backup with BRBACKUP

### Advantages

- The first phase can be much shorter than a direct backup to tape.
- In the case of a recovery the restore phase is shortened, if the backup is directly available on disk.
- When restoring from tape, BBRESTORE can write the backed up files directly to the original directories.
- Volume management and all other automatic actions of BRBACKUP can be fully used in the second phase.

### Disadvantages

- The hardware requirements (that is, disk storage and storage tapes) must be fulfilled. Additional disk storage space is required, compared to a direct tape backup.
- You must start BRBACKUP twice for a two-phase backup.

#### Note

The backup type - that is, [offline or online](#) – and the [extent of the backup](#) – that is, complete or partial - must be identical in the first and second phases. For an offline backup, the database remains open in the second phase.

## Two-Phase Backup of Offline Redo Log Files with BRARCHIVE

Two-phase backup of the offline redo log files using BRARCHIVE runs analog to the BRBACKUP backup. However, you can only make a maximum of two copies to tape. We strongly recommend you to back up the offline redo log files to tape, in addition to a disk backup.

- Phase 1  
BRARCHIVE backup to disk
- Phase 2  
BRARCHIVE backup from disk to tape
- Restore phase  
BBRESTORE restore from tape directly into the original directories or direct applying of offline redo log files in a BRRECOVER recovery from disk (no restore).

To perform the BRARCHIVE backup of the offline redo log files from a disk backup to a tape volume, start BRARCHIVE with the appropriate command option. For more information, see [a|-archive](#).

## Features of Two-Phase Backup of Offline Redo Log Files with BRARCHIVE

### Advantages

- The first phase can be much shorter than direct backup to tape, so that the ORAARCH directory is emptied more quickly.
- In a recovery, the restore phase can be much shorter, if the backup of the offline redo log files is directly available on disk. In this case BRRECOVER

### Disadvantages

- A maximum of two copies of the offline redo log files can be backed up to tape, regardless of the way in which the offline redo log files are backed up: directly to tape or with a disk backup.
- Additional storage space and storage tapes are required.

## Features of Two-Phase Backup of Offline Redo Log Files with BRARCHIVE

### Advantages

- uses the offline redo log files directly from the disk. There is no restore.
- When restoring from tape, BBRESTORE can write the backed-up offline redo log files directly to the ORAARCH directory.
- The volume management of BRARCHIVE can be fully used in the second phase of the backup.

### Disadvantages

- You must start BRARCHIVE twice for a two-phase backup.



## Structure-Retaining Database Copy

With [BRBACKUP](#) you can make a copy of the Oracle database files with exactly the same directory structure. You can use this type of database copy to:

- Generate a test system from a production system
- Set up a [Standby Database](#).
- Have a database backup available that saves you the restore during a recovery. In this case the `sapdata` home directory is renamed as the new `sapdata` home directory of the database copy. The copied files are then the current files and you can apply the offline redo log files directly.

## Prerequisites

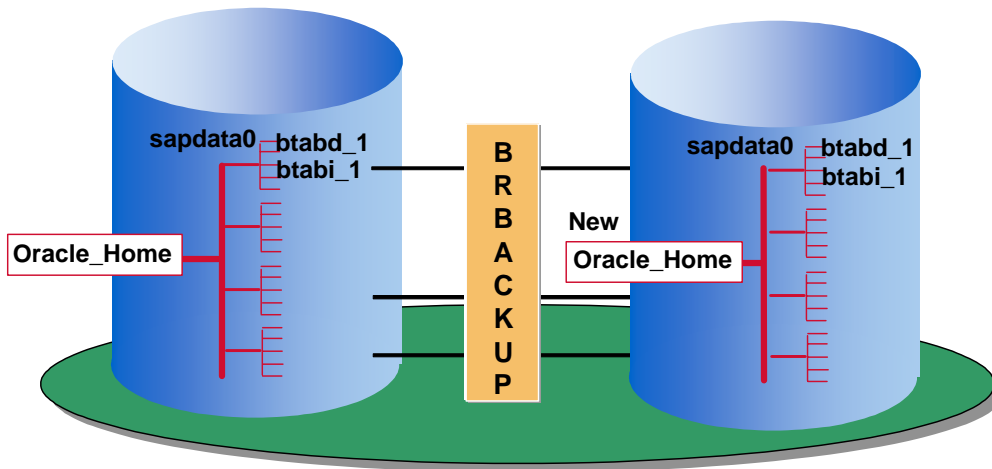
You must create the following directories on the target database:

- `sapdata` directories
- `sapbackup` directory
- `origlogA`, `origlogB`, `mirrlogA`, `mirrlogB` directories of the online redo log files

The corresponding subdirectories are created automatically during copying.



`brbackup -d disk_copy`



#### Example

`/oracle/C11/sapdata2/stabd_1/stabd.data1` is copied to  
`/oracle/C12/sapdata2/stabd_1/stabd.data1`

#### Caution

Since this is a one-to-one copy, software compression is not possible.

## Activities

To copy the database, you have to define the name of the new database home directory (of the database copy) in the `init<DBSID>.sap` profile parameter [new\\_db\\_home](#) (for local disks) or [stage\\_db\\_home](#) (for remote disks). Also set the parameter [backup\\_dev\\_type](#) to `disk_copy|disk_standby|stage_copy|stage_standby` or call up BRBACKUP with the relevant command option, for example, `brbackup -d|-device disk_copy|stage_copy`.

#### Note

Under Windows, the `sapdata` directories can be distributed across several drives. When you make the copy, you can retain this distribution by specifying the appropriate target drives. For more information, see the BRBACKUP parameter [ml-mode](#).

## Parallel Backup

You can use [BRBACKUP](#) and [BRARCHIVE](#) to back up your Oracle database to several backup devices in parallel. The backup devices can be tape or disk drives. Parallel backup

reduces the backup time and allows unattended operation (that is, backup in unattended mode).

Parallel backup is especially useful for [large databases](#). For more information, see [Parallel Backup of Large Databases to Disk with BRBACKUP](#).

## Features

- Parallel backup is possible to local or remote backup devices.
- If the backup devices used support hardware compression, use this by setting the `init<DBSID>.sap` parameter `compress = hardware`. You can also make parallel backups with software compression, using `compress = yes`. For more information, see [compress](#).
- When you perform parallel backups to several backup devices, BRBACKUP attempts to optimize the distribution of the database files among the backup volumes using load balancing, as follows:
  - BRBACKUP attempts to balance the load equally among all the backup devices, which is called time optimization. If this is not possible because of the size of the backup volume, it attempts to divide the data files equally among the individual backup devices, which is called size optimization.
  - Whenever possible, BRBACKUP saves files from one disk on one volume in one backup device, in order to minimize drive head movement during the backup.
- BRARCHIVE only uses the parallel backup option to tape when you start archiving with `brarchive -ss` or `brarchive -ssd`. In this case, the offline redo log files for archiving are saved to both volumes in parallel (or saved and then deleted). For more information, see [-s|-sc|-ds|-dc|-sd|-scd|-ssl|-ssd|-cs|-cds](#).
- The addresses of the tape devices are defined in the following `init<DBSID>.sap` parameters:
  - [tape\\_address](#)
  - [tape\\_address\\_rew](#)
  - [tape\\_address\\_arch](#)
  - [tape\\_address\\_rew\\_arch](#)
- If the `-ss` or `-ssd` option is used, BRARCHIVE only uses the first two tape devices in the list. The addresses of the directories on disk are defined in the `init<DBSID>.sap` parameter [backup\\_root\\_dir](#) or [stage\\_root\\_dir](#) for backup.

### Example

This is an example of the entries required in the [init<DBSID>.sap](#) initialization profile for parallel backup:

- Device type

```
backup_dev_type = tape|disk|pipe|stage
```
- Addresses for tape devices:

```
tape_address = (/dev/rmt/0hnc, /dev/rmt/1hnc)
tape_address_rew = (/dev/rmt/0hc, /dev/rmt/1hc)
```

- Addresses for directories on hard disk:

```
backup_root_dir = (/backup/dir1, /backup/dir2)
```

- Compression parameters:

```
compress = no|software|hardware
```

- Tape size:

```
tape_size = 18G|16G
```

- The number of parallel copy processes normally corresponds to the number of backup devices available. You can change this setting with the `init<DBSID>.sap` parameter [exec\\_parallel](#) or the command option `-e|execute`.

## More Information

[backup\\_dev\\_type](#)

[tape\\_size](#)



## Unattended Backup

With [BRBACKUP](#) and [BRARCHIVE](#), you can back up your Oracle database without monitoring or operator intervention. How you do this depends on your operating system. For more information on backup with Microsoft Windows, see [BRBACKUP/BRARCHIVE](#).

## Features

### Unattended Parallel Backup

You can make unattended backups if you have enough backup devices. This means that you need as many backup devices as volumes are required for the backup. BRBACKUP can then back up to these devices in parallel, without operator intervention to change the volumes. For more information, see [Parallel Backup](#).

### Automatic Tape Changers

If you want to use automatic tape changers, you must define the [rewind\\_offline](#) parameter appropriately and set [backup\\_dev\\_type](#) to `tape_auto` or `pipe_auto`. For more information, see [Backup with Automatic Tape Changers](#).

### Serial Backup

Unattended backup is also possible if fewer parallel copy processes than connected backup devices are possible (for example, due to the impact on performance). To do this, set the parameter [exec\\_parallel](#) to 1. If you need several volumes for a backup, the backup devices are not used in parallel. In this case they are used in accordance with the number of copy processes set.

To make a parallel or serial backup on several tape devices, you must define the addresses of the backup devices in the following `init<DBSID>.sap` parameters:

- [tape\\_address](#)

- [tape\\_address\\_rew](#)
- [backup\\_root\\_dir](#) for a local disk backup
- [stage\\_root\\_dir](#) for a remote disk backup

#### Example

Here is an example of these parameters:

```
tape_address = (dev/rmt/0mn, /dev/rmt/1mn) tape_address_rew =
(dev/rmt/0m, /dev/rmt/1m)
```

## Backups with BRBACKUP or BRARCHIVE in One Run

The complete backup of the database files and the offline redo log files can be executed with a single start of BRBACKUP, using command option [-al-archive](#).

For more information, see [BRBACKUP and BRARCHIVE Backups in One Run](#).

## Backup with CRON

For a successful unattended backup using CRON, you must meet the following requirements:

- The parameters in `init<DBSID>.sap` must be set correctly.
- The Crontab entries must be defined under user `root`.
- BRBACKUP or BRARCHIVE must run under database user `ora<sapsid>` or the operating system user `<sid>adm`. No password is needed due to the OPS\$ mechanism. This means that no password can be seen in a script.
- Enough volumes (for example, tape or disk volumes) must be available, as well as a sufficient number of backup devices (if you want to perform a parallel backup).
- The correct tapes must be mounted in the backup devices. The operator cannot change the tapes during the backup run.

If automatic tape management is active, first determine the tape names by entering the commands `brbackup|brarchive -q`.

Use `brbackup|brarchive -q check` to verify that the required volumes were actually mounted.

For more information, see [brbackup -q](#) and [brarchive -q](#).

If automatic tape management is not active, mount tapes for which the expiration period has passed.

## Example

These are examples of shell scripts to start unattended backups.

#### Example

Online backup of the complete database

Unattended operation

Daily, Monday through Friday

**Backup to start at 22:00**

```
#Min(0-59) Hrs (0-23) Day (1-31) Mon(1-12) WD (0-Sun,...,6-Sat)
00 22 * * 1-5
su - ora<dbsid> -c "brbackup -t online -c force -u"%system/<password>
or under <sapsid>adm:
su - <sapsid>adm -c "brbackup -t online -c force -u /"
```

### Example

**Offline backup of the complete database**

**Unattended operation**

**Daily, Monday through Friday**

**Backup to start at 22:00**

**SAP system shut down**

```
#Min(0-59) Hrs (0-23) Day (1-31) Mon(1-12) WD (0-Sun,...,6-Sat)
00 22 * * 1-5
/backup1.sh
```

**The script backup1.sh in the root directory might look as follows:**

```
su - <sapsid>adm -c "stopsap R3"
su - ora<dbsid> -c "brbackup -t offline -c force -u" <<END
system/<password>
END
su - <sapsid>adm -c startsap
```

### Example

**Offline backup of the complete database**

**Unattended operation**

**Daily, Monday through Friday**

**Backup to start at 22:00**

**No SAP system shut down**

```
#Min(0-59) Hrs (0-23) Day (1-31) Mon(1-12) WD (0-Sun,...,6-Sat)
00 22 * * 1-5
/backup2.sh
```

**The script backup2.sh in the root directory might look as follows:**

```
su - ora<dbsid> -c "brbackup -t offline_force -c force -u" <<END
system/<password>
END
```

### Example

Archiving the offline redo log files

Unattended operation

Daily, Monday through Friday

Backup to start at 08:00

Parallel archiving to two backup devices.

```
#Min(0-59) Hrs (0-23) Day (1-31) Mon(1-12) WD (0-Sun,...,6-Sat)
00 8 * * 1-5
su - ora<dbsid> -c "brarchive -ssd -c force -u"%system/<password>
```

## BRBACKUP and BRARCHIVE Backups in One Run

You can execute the database backup using [BRBACKUP](#) and the backup of the offline redo log files using [BRARCHIVE](#) in a single run. You can use this option to make more effective use of the increasing capacity of storage devices, such as tapes and disks. It also makes an unattended backup easier, since after a backup with BRBACKUP, the BRARCHIVE run is started automatically. You can also perform this procedure in the DBA Planning Calendar of the Computing Center Management System (CCMS)

There are the following options for performing the backup in one run:

- BRBACKUP starts BRARCHIVE, using `brbackup -a`. For tape backups, the tapes are managed by BRBACKUP. For more information, see [-aj-archive](#).
- BRARCHIVE starts BRBACKUP, using `brarchive -b`. For tape backups, the tapes are managed by BRARCHIVE. For more information, see [-bj-backup](#).

We recommend the first option (`brbackup -a`). For tape backups, BRBACKUP uses the tapes defined in [volume backup](#). To execute the backup, BRBACKUP:

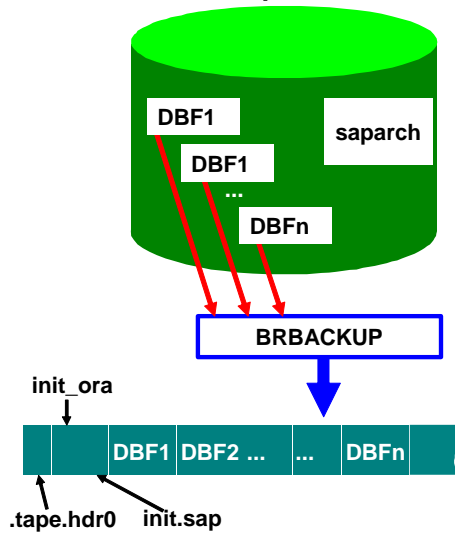
- Checks the volume label
- Backs up the tape header files (`.tape.hdr0`, `init_ora`, `init_sap`)
- Backs up the database files (but does not save logs)
- Calls BRARCHIVE

Then BRARCHIVE:

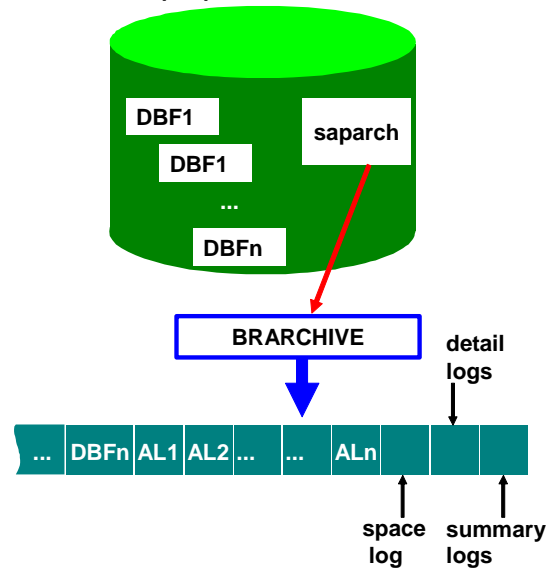
- Backs up the offline redo log files to tape after the backed-up database files (without checking the label and without header files)

- Backs up all logs (that is, for BRBACKUP and BRARCHIVE)

### 1. Backup of the database files, brbackup -a



### 2. Backup of the offline redo log files (AL)



### Overall View



### Example

This is an example of a BRBACKUP command for unattended online backup with two tape devices. The database files are backed up first and then the offline redo log files are backed up to the same tapes.

```
brbackup -m all -t online -c -a -ssd -c
```

For more information, see BRBACKUP command [-a|-archive](#).

If you run the tape administration using [brarchive -b](#), the tapes defined in [volume archive](#) are used. The only change to the backup is the sequence in which the data files and the offline redo log files are written to tape. In this case the logs are backed up by BRBACKUP at the end.

This is an example of a BRARCHIVE command for offline backup with one tape device. The offline redo log files are backed up on tape first and then the database files are copied to the same tape.

```
brarchive -sd -c -b -t offline -c
```

For more information, see BRARCHIVE command [-b|-backup](#).

You can also perform a single combined BRBACKUP and BRARCHIVE run to several tape devices in parallel. For more information, see [Parallel Backup](#).

### Note

Starting BRBACKUP and BRARCHIVE in sequence with BACKINT, or to disk, only means that BRBACKUP and BRARCHIVE run together. It does not automatically mean that the backup is made to the same volume. If you are using BACKINT, you can do this with BACKINT tools in some situations, depending on the BACKINT implementation.

## Backup Verify

When making database backups, you must regularly perform a verify.

### Caution

If you fail to verify backups, you might find that a backup cannot be used in the event of a database restore. Even if a backup is reported as successfully completed, this does not mean that it is always error-free.

There are the following types of verify:

- Backup tape readability

This verify checks the backup media (that is, usually tape). It checks the size of the backup files and whether the data can be read. In some cases, it also compares the backed-up data with the contents of the database files byte-by-byte.

- Database block consistency

This verify checks the database itself block-by-block, validating internal block consistency.

We recommend that, at least once in every [backup cycle](#), you perform *both* types of verify. If possible, perform verify once a week, or even for every backup.

## Integration

- You can perform verify at the command level of the [BRBACKUP](#), [BRARCHIVE](#), and [BRRESTORE](#) tools, as described below.
- You can also perform verify using the action patterns of the [DBA Planning Calendar](#).
- You can now use Oracle Recovery Manager (RMAN) to verify the internal consistency of database and offline redo log files and backups of these files. This is especially useful for offline redo log files because you were unable to verify their internal consistency up to now. For database files, the RMAN functionality offers the same functionality as DBVERIFY.

The relevant command options are:

- [brarchive -w|-verify use \\_rmv|first\\_rmv|only\\_rmv](#)
- [brbackup -w|-verify use \\_rmv|only\\_rmv](#)
- [brrestore -w|-verify use \\_rmv](#)

## Prerequisites

- Verify adds considerably to backup run times.



- Verify is mainly performed on complete backup volumes. A volume is first written and then verified.

## Features

### Verify of Backups with BRBACKUP Without DBVERIFY and RMAN

This type of verify only checks the backup tape readability. The method and extent of a brbackup verify differs according to whether the backup is [online or offline](#):

- Offline backup
 

After the backup the files are copied back to a temporary directory, `compress_dir`, and the contents are compared to the original data in the database byte by byte. This means that backup tape readability and file contents are checked.
- Online backup
 

After the backup the files are copied back to a temporary directory, `compress_dir`, but only the file sizes are compared. The byte-by-byte compare used in an offline backup is not possible with an online backup because the database is constantly changing as updates continue. This means that only backup tape readability and file sizes are checked.

For more information, see [brbackup -w|-verify](#).

### Independent Verify of Backups with BRRESTORE Without DBVERIFY and RMAN

This type of verify with [BRRESTORE](#) only checks the backup tape readability, separately from the backup, at a later time if you want. You can also perform this type of verify on another computer if the backups are available there.

BRRESTORE only checks whether the backup can be read and its size, not the contents. The files are only read, not restored.

For more information, see [brrestore -w|-verify](#)

### Verify of Backups with BRARCHIVE

This type of verify with [BRARCHIVE](#) only checks the backup tape readability. The extent of the check on archived offline redo log files depends on the type of BRARCHIVE backup:

- `brarchive -s|-sc|-ss|-cs`

The backed-up files are restored and compared with the originals byte by byte.
- `brarchive -sd|-scd|-ssd|-cds`

The backed-up files are restored and file sizes are checked. Since the originals were deleted, a check on the contents is not possible.

For more information, see [-brarchive -w|-verify](#)

### Verify of Backups with Oracle DBVERIFY

The Oracle DBVERIFY tool is available for both types of verify, that is, backup tape readability (but without a byte-by-byte comparison) and database block consistency. This means you can recognize errors early (for example, ORA-1578), before they lead to the termination of a program in an application that accesses the blocks.

You can use DBVERIFY as follows:

- Database backup with subsequent restore to a temporary directory (`compress_dir`) and check on the Oracle block consistency:

```
brbackup -w use_dbv
```

- Online check of block consistency on the database files without backup:

```
brbackup -w only_dbv
```

Any number of these verify processes can run in parallel, using parameter `exec_parallel`, option `-e`.

- Temporary restore of a database backup (`compress_dir`) and verify of the database block consistency:

```
brrestore -w use_dbv
```

The restore implicitly checks the readability of the backup.

- You can now also verify backups on local disks – that is, using `backup_dev_type = disk` – with software compression (`compress = yes`).

A verify with DBVERIFY of database backups on tape – that is, using `backup_dev_type = tape | tape_auto | tape_box | pipe | pipe_auto | pipe_box` with software compression – is still not possible. However, this limitation is usually irrelevant since tape backups are normally performed with hardware compression, not software compression.

## Verify of Backups with RMAN

The Oracle Recovery Manager (RMAN) is available for both types of verify, that is, backup tape readability (but without a byte-by-byte comparison) and database block consistency. This means you can recognize errors early (for example, ORA-1578), before they lead to the termination of a program in an application that accesses the blocks. The `RMAN BACKUP VALIDATE` command is used for the verification.

You can use RMAN as follows:

- Database backup with subsequent restore to a temporary directory (`compress_dir`) and check on the Oracle block consistency:

```
brbackup -w use_rmv
```

- Online check of block consistency on the database files without backup:

```
brbackup -w only_rmv
```

Any number of these verify processes can run in parallel, using parameter `exec_parallel`, option `-e`.

- Temporary restore of a database backup (`compress_dir`) and verify of the database block consistency:

```
brrestore -w use_rmv
```

The restore implicitly checks the readability of the backup.

- You can now also verify backups on local disks – that is, using `backup_dev_type = disk` – with software compression (`compress = yes`).

A verify with RMAN of database backups on tape – that is, using `backup_dev_type = tape | tape_auto | tape_box | pipe | pipe_auto | pipe_box` with software compression – is still not possible. However, this limitation is usually irrelevant since tape backups are normally performed with hardware compression, not software compression.

## Verify of BACKINT and RMAN Backups

- Verify of BACKINT Backups

You can now fully verify third-party backups using the BACKINT interface, as specified above in *Verify of Backups with Oracle DBVERIFY* and *Verify of Backups with RMAN*.

The files to be verified are loaded back to the temporary directories (or to a single directory) specified by the `init<DBSID>.sap` parameter `compress_dir`. DBVERIFY or RMAN then checks database block consistency. The verify can be performed in several steps. The required free space must be at least 10% of the backup size (for databases greater than 1 TB, this is at least 100 GB), so that the number of steps required is not too great.

### Caution

Since the verification of database backups in the DBA Planning Calendar always uses `-w use_dbv`, there is a change to the verify of BACKINT backups. The backup is always reloaded and verified. This doubles the runtime of the backup.

If you do not want to fully deactivate the verify, set the `init<DBSID>.sap` parameter `compress_dir` to point to a directory with a small amount of free space (less than 10% of the backup size). BRBACKUP then switches automatically to BACKINT query mode (`only_conf`).

### Note

Of course, you can perform BACKINT backup verification directly using the backup tool (if supported). Some backup tools offer special parameters in the BACKINT parameter file, `init<SID>.utl`, that you can use to activate the verification independent of the option `-w|-verify`. For more information, consult your BACKINT provider.

- Query only whether the backup is known, using the BACKINT query function:

```
brbackup -w only_conf
```

This option does what the option `brbackup -w` used to do, before the functionality of BACKINT verify was extended.

- Verify of RMAN Backups

You can also access an external backup tool using the Oracle Recovery Manager (RMAN) with the parameter `backup_dev_type = rman_util|rman_disk|rman_stage`. This also gives you a complete range of functions for verification:

- The RMAN `VALIDATE` command is used for verifying backups. In this case, data is physically read by the backup medium and checked by RMAN for consistency.

- The verification with DBVERIFY is no longer required because for each backup all saved database files are checked by RMAN for internal Oracle block consistency.

## Grouping Offline Redo Log Files

When you [back up offline redo log files](#) for your Oracle database, you can group the files. If you use an [external backup programs](#) with the BACKINT interface, we recommend you to use this function. In this case, every archiving action activates the interface and often repositions the tape, which usually leads to the creation of a new save set. A reduction in the number of save sets by grouping the offline redo log files speeds up the backup.

By using permanent backup of the offline redo log files with BRARCHIVE [-f|-fill](#), you can prevent a possible overflow of the `saparch` archiving directory. You can also collect a certain number of offline redo log files before BRARCHIVE backs them all up together on tape, by using the command `brarchive -f <number>`, as shown in the following example.

### Example

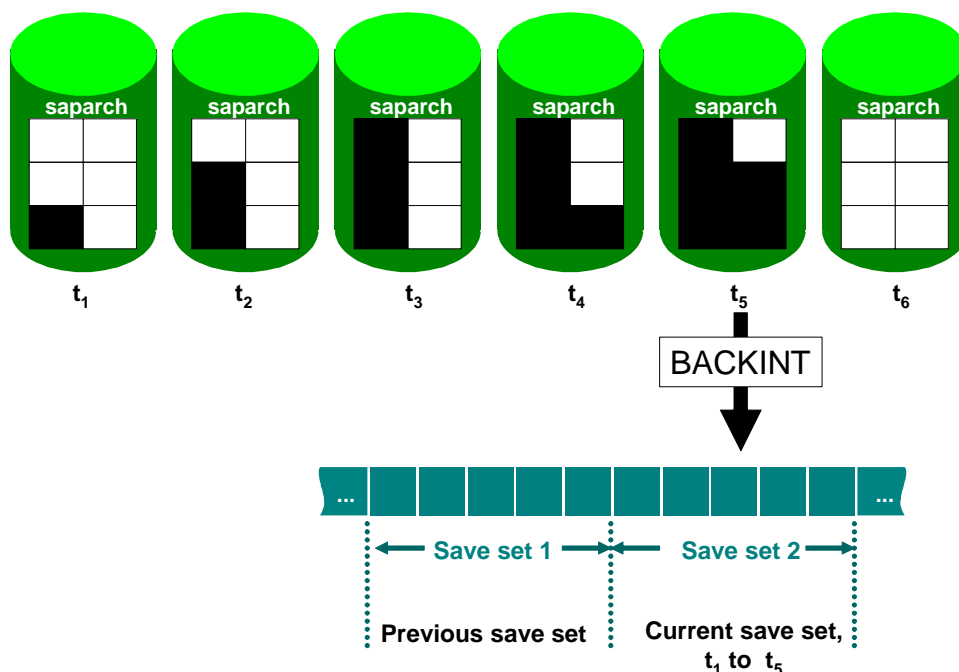
#### Example

This example uses the following command:

```
brarchive -sd -d util_file -f 5
```

After five offline redo log files have been created, BRARCHIVE saves these to tape. It repeats this until either of the following is true:

- BRARCHIVE was stopped with call `brarchive -f stop`
- The maximum number of redo logs specified has been reached





## Advanced Backup and Recovery

This section deals with advanced topics in backup and recovery. Make sure that you have sufficient expertise before using these functions.



## Backup of Large Oracle Databases

This section gives you advice on backing up large Oracle databases. Large databases contain 500 GB - 5 TB or more data. For reasons such as the following, you might not be able to back up the database as often as you want to (that is, daily in most cases):

- Performance problems

There is a heavy load on database server resources, particularly CPU time, system and I/O buses, disk and volume controllers. Therefore, online operations in the SAP System are restricted.

- Lack of time

Although you normally perform backups at times of low system load, such as at night, you might still find that you exceed the 12 hours available.

- Data volume

The amount of database data is too great to back up in the time available.

- Common backup strategy

A common backup strategy for database and non-database files is not a solution since non-database data can better be backed up with operating system tools (such as tar).

## Prerequisites

Your system is normally configured so that there is a single server for the database, where no other large applications run.

It is often *not* recommended to back up large databases across a network because of instability and performance problems.

## Features

Whether you can back up large databases with [BRBACKUP](#) (and `cpio` or `dd` for the copy processes) depends on the following factors:

- Capacity and maximum throughput of the tape devices
- Disk access times
- Maximum throughput of the I/O buses
- Maximum throughput of the system bus
- `cpio` and `dd` performance, determined by internal buffering and blocking. In general, `dd` offers much better performance than `cpio`

BRBACKUP itself places minimal load on the backup process. Any hardware-specific restrictions can only be improved by the hardware vendor.

The hardware configuration for a large database needs careful planning for an optimal backup. This might require multiple tape changes and the management of hundreds of volumes. If possible, use tape jukeboxes or robots supported by the BACKINT interface and external backup programs.

You can reduce the amount of data significantly by using the following features:

- [Incremental backup](#), which we recommend
- [Standby Database](#)
- [Split Mirror Backup](#)



## Backup Devices for Large Databases

This section discusses how to use devices when [backing up a large Oracle database](#).

- Instead of backing up over a network, it is often better to back up to volumes in locally mounted backup devices or directly to hard disks.
- Backup devices:
  - The maximum number of locally connected tape devices supported by [BRBACKUP](#) is 255.
  - Backup devices (for example magneto-optical media) that are addressed with an external backup program can be reached using the BACKINT interface. For more information, see [External Backup Programs for Large Databases](#).
  - BRBACKUP offers only limited support for [automatic tape changers](#) such as jukeboxes. However, you can address such devices using the BACKINT interface to external backup programs.
- Use tape units with a larger capacity and higher throughput rate if possible.
- Factors other than the performance of the backup device also play a large role, such as:
  - Server throughput, in particular, hard disk access times, system bus speed, I/O bus speed.
  - Do not mount too many backup devices on one I/O bus, so as not to overload it. If hard disks and tape units are mounted on the same I/O bus, the load is split between the mounted devices.



## Backup of Large Databases to Tape with BRBACKUP

This section describes how you can use [BRBACKUP](#) to [back up large Oracle databases](#) to tape. BRBACKUP calls `cpio` or `dd` to copy individual database files from disk to tape. As a result, throughput is largely determined by `cpio` or `dd`.

## Prerequisites

Note the following restrictions for backup with BRBACKUP:

- As the database server is working with high load during the backup, we recommend you to reduce other activities on the computer to the absolute minimum during the backup. Virtually all of the computer's resources ought to be available for the backup.

This clearly limits the high availability required of the SAP System. However, there is no simple solution to this conflict.

- Be aware of the features of [hardware compression](#) if you decide to use this.
- No `cpio` continuation mechanism is supported during parallel backups. For more information, see [cpio Continuation Tape](#). This means that you must always completely save individual files to one volume. Always perform tape swapping with BRBACKUP. Therefore, the size of a file to be saved must not exceed the tape capacity. If you work with tape units with hardware compression, the size of the compressed file must not exceed the tape capacity. Make sure that the tape capacity is not set too high, in order to avoid reaching the physical end of the tape.
- Since individual files can only be written in their entirety to a volume, there is a certain amount of wastage when distributing the files to the available tapes. That is, the tape capacity cannot be fully used. When distributing the files, BRBACKUP makes sure that the tape capacity is never exceeded. The size of the tape header files – that is, label, `init<DBSID>.ora` file, and `init<DBSID>.sap` file – and the log files at the end of the tape (central, detailed, and summary log) are not taken into account.

## Features

BRBACKUP offers the following functions for optimal backup of large databases:

- BRBACKUP can back up in [parallel](#) to multiple mounted tape units (up to 255).
- In a parallel backup, BRBACKUP also supports [automatic tape changers](#). Therefore, a completely automatic backup is also possible with tape swapping for one or more tape units during a backup.
- All files to be saved are distributed to the tape volumes inserted in the tape units. BRBACKUP has different optimization targets. For more information, see [Optimization of File Distribution](#).

## More Information

[Optimization with a Logical Volume Manager](#)

[Partial Backups](#)



## External Backup Programs for Large Databases

You might want to use external backup programs if you need to [back up a large Oracle database](#). We provide the BACKINT interface for this purpose. For more information, see [External Backup Programs for Large Databases](#).

## Features

The main advantage of using the BACKINT interface with external backup programs is that you can use other backup media, for example, magneto-optical (MO) media. Alternatively you can transfer volume management to other systems, for example, [automatic tape changers](#), such as jukeboxes and tape robots.

For more information, see the option `util_file_online` with the [backup\\_dev\\_type](#) parameter or the corresponding command option `-d|-device`.

### Note

You can obtain information on the throughput and performance of external backup programs from the vendors providing the external backup.

## Parallel Backup of Large Databases to Disk with BRBACKUP

You can use [BRBACKUP](#) to back up your large Oracle database to [multiple disks](#) in [parallel](#).

## Prerequisites

You must define the directories with the [backup\\_root\\_dir](#) parameter.

## Features

- You can perform the backup with or without [software compression](#).
- You can control the degree of parallelism – that is, the number of parallel copy processes – using the [exec\\_parallel](#) parameter or the BRBACKUP command option `._el-execute`.
- If required, you can make a [two-phase backup](#), that is, first to disk and then to tape.

## Activities

To avoid imposing extra processing load on the database server during the second phase of a two-phase backup, consider the following procedure instead:

1. Unmount the file system from the database server.
2. Mount the file system on a second host.
3. Start the backup from this host.

A requirement for this procedure is that the hard-disk controllers for the backup disks can be physically mounted on both hosts simultaneously. The unmount and mount operations – that is, `umount` and `mount` – are necessary, since the file system cannot be mounted on different computers at the same time due to the buffering mechanism.

Alternatively, you can [back up directly to remote disks](#). This eliminates the need for `umount` and `mount`.

Another approach to reduce processing load on the production database host is to use [split mirror backup](#).





## Optimization of File Distribution

This section describes how [BRBACKUP](#) optimizes file distribution during an Oracle backup. This is especially relevant to [backup of large Oracle databases](#).

### Features

When distributing the files to the tapes, BRBACKUP has different optimization targets:

- All files from a disk saved to the same tape volume

Therefore, try to avoid competing disk accesses, and keep the number of read/write accesses to a minimum.

If a Logical Volume Manager (LVM) is used, this target can only be attained if the logical volumes are not scattered over several physical hard disks. For more information, see [Optimization with a Logical Volume Manager](#).

- Time-based optimization used

Using the backup times for individual files stored in the database, BRBACKUP attempts to minimize the total backup time by keeping the backup time equal on each of the individual tape units or disk volumes. Note the following factors relevant to time-based optimization:

- It helps to avoid the following problem. When using tape units with hardware compression, the backup time cannot be estimated accurately from the file size or the size of the compressed file. Therefore, the optimization is performed from the backup times stored by BRBACKUP in the database.
- The tape capacity in general is not used to its maximum due to time-based optimization. That is, if the optimal backup time is reached for a tape, no further files are saved to this tape, though the tape might still have plenty of space left. Therefore, the total capacity of all available tapes inserted at the same time must significantly exceed the total size of the files to be saved.
- The time-based optimization is cancelled internally if it would cause a tape change, since BRBACKUP always attempts to avoid volume changes. In this case, the entire capacity of the tapes is used. As a result, the backup time can vary for individual volumes.
- All mounted tapes used

To minimize the total backup time, BRBACKUP always attempts to use all mounted tape units, even if the backup would fit onto a smaller number of tapes.

The one exception is when the number of files to be backed up is less than the number of tape devices.

- Files sorted and distributed by size

BRBACKUP sorts the files to be saved according to their size and distributes the largest files to the available tapes first, followed by the smaller ones. As a result, wastage is reduced on individual volumes, because the tape capacity can never be exceeded and the total tape capacity can be better utilized. For more information, see the [tape\\_size](#) parameter.

- Previous backup times used

BRBACKUP is capable of “learning”. It stores the backup times of individual files in the database and uses these in the next backup for time-based optimization (if this is performed).

As a result, the changes to backup time are taken into account for individual files. The backup time can change, for example, if the fill level of the individual database files varies and the files are saved with [hardware compression](#), using a tape unit.

- Parameter `-o|-output` with options `dist, time` used

BRBACKUP or BRARCHIVE use these to:

- Display the file distribution performed by BRBACKUP before starting the actual backup
- Display the backup times of the individual files after the backup is completed

For more information, see [-o|-output](#) (BRBACKUP) and [-o|-output](#) (BRARCHIVE).

See also [Log Supplements](#).



## Optimization with a Logical Volume Manager

This section describes how a Logical Volume Manager (LVM) influences the [optimization of file distribution](#) during an Oracle database backup.

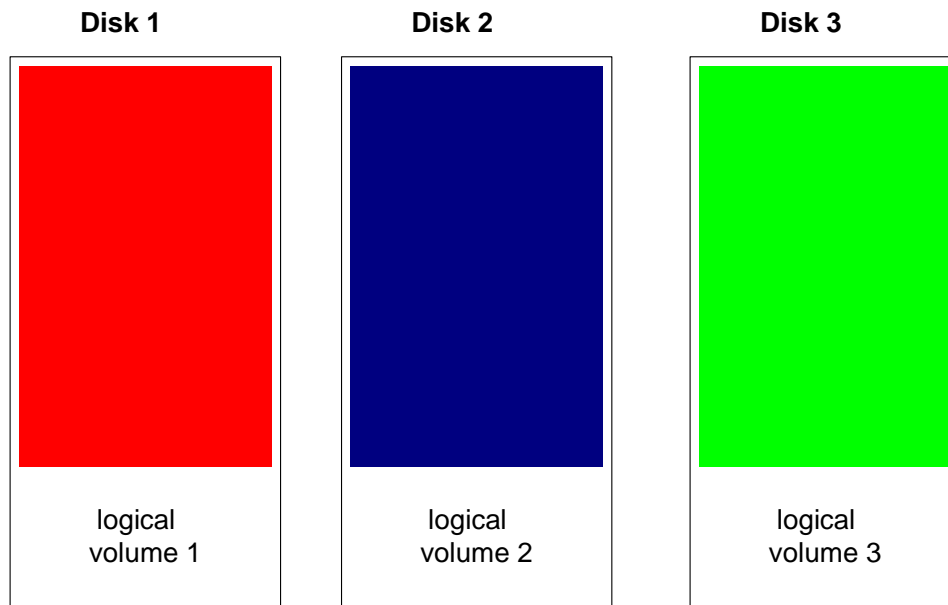
These consideration do not apply to large disk arrays, where the whole database is stored in one array.

If you use an LVM, BRBACKUP can only save all files from a hard disk to tape if the logical volumes are not scattered over several hard disks. Configuration A of the graphic below is better suited for backup with an LVM.

### Features

#### Configuration A

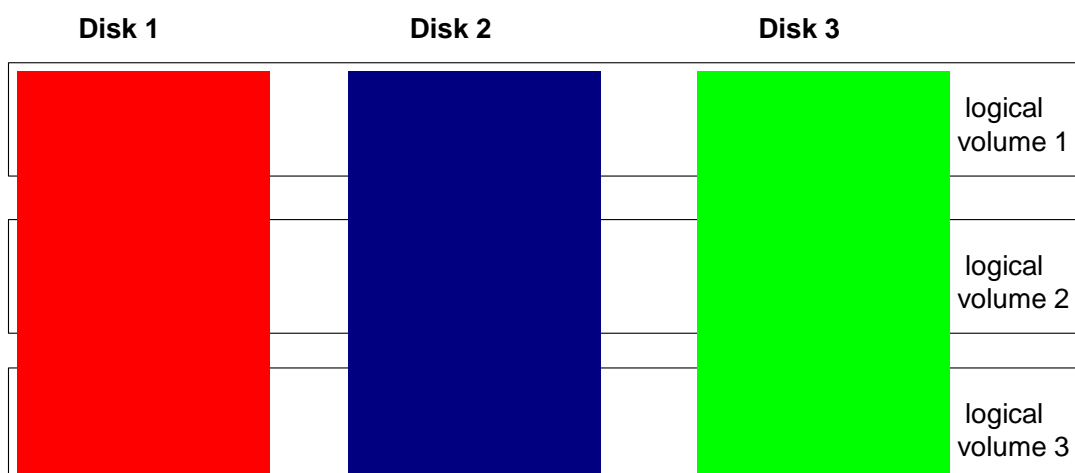
Each logical volume exactly corresponds to one hard disk. For example logical volume 1 = disk 1, logical volume 2 = disk 2, logical volume 3 = disk 3.



### Configuration B

Each logical volume is set up so that it covers areas of all the hard disks:

- Logical volume 1 = area1.disk1 + area1.disk2 + area1.disk3
- Logical volume 2 = area2.disk1 + area2.disk2 + area2.disk3
- Logical volume 3 = area3.disk1 + area3.disk2 + area3.disk3



Configuration A is more efficient than configuration B with respect to database backup. However, configuration B can provide better performance for online operation of the SAP system.

## Activities

When planning the configuration of logical volumes for large databases, you need to find a compromise between the following:

- More effective backup
- Higher SAP System performance

Consider the following factors:

- Advantages of using an LVM  
Easier administration, high flexibility, higher security by using RAID systems
- Disadvantages of using an LVM  
Performance loss through management overhead and possibly, in configuration B, reduced optimization with BRBACKUP
- Although the higher security and availability of your datasets provided by using the LVM has a generally high priority, consider whether you could do without an LVM:
  - You lose the LVM advantages outlined above.
  - However, you can perform a dataset backup much more effectively using BRBACKUP.
- In large databases, the configuration of the database files is not variably selectable. You must consider the effect on physical disk configuration and the influence this has on performance when you make changes to the configuration, such as structure changes due to a tablespace extension or a tablespace reorganization.



## Partial Backups

You can reduce long backup times for your Oracle database by dividing the backup into several partial backups.

### Features

#### Advantages

- The partial backups require less time, so you can perform them daily.
- You can recover the database at any time if the corresponding redo log entries exist. It is also possible to restore the entire database and then recover it.
- You can quickly recover after a media error – that is, a disk crash – as you only need to restore the files of the affected hard disks.

#### Disadvantages

- Restoring the entire database takes as long as the total of all of the partial backups. This situation only normally occurs with logical errors, such as program errors or user errors. However, this is an unlikely event.
- You must ensure that all database files are saved in a partial backup cycle. This is your responsibility as database administrator.

### Recommendation

Back up at [tablespace](#) level (not at database file level, which is also possible), as BRBACKUP can then make sure that all files from a tablespace are backed up.

Splitting extremely large tablespaces into several smaller tablespaces can be useful in this case. For example, you can put extremely large tables into separate tablespaces, and then back them up separately at tablespace level.

Use BRBACKUP command option [-fj-fillup](#) to do this.

## Tablespace Backups

It is important to keep the volume of data to be backed up or restored as small as possible, especially when backing up large Oracle databases. One way to do this is by backing up specific tablespaces.

### Activities

You do *not* have to back up pure index tablespaces during a database backup. Index definitions are stored in the `SYSTEM` tablespace as a matter of course. Therefore, indexes can always be recreated.

### Caution

As it takes longer to recreate indexes than to restore and recover index tablespaces, we do not generally recommend a backup strategy based on the BRBACKUP function `-m all_data` as described below. Only use this function for a good reason.

To back up all tablespaces except index tablespaces, perform one of the following:

- Start BRBACKUP with the command option `-m all_data`.  
For more information, see [-m|-mode](#).
- Set the profile parameter `backup_mode = all_data` in `init<DBSID>sap`.  
For more information, see [backup\\_mode](#).

During the backup, BRBACKUP indicates all pure index tablespaces, so that BRRECOVER and BRRESTORE recognize these. In the restore process, which uses a complete backup (`brrestore -m all`), you can exclude the pure index tablespaces with the command `brrestore -m all_data`. For more information, see [-m|-mode](#).

### Note

After restoring the backed-up data tablespaces, both the index tablespaces and the indexes must be newly created with the definitions in the `SYSTEM` tablespace. BRRECOVER automatically performs this during the recovery.

BRRECOVER can now automatically handle the situation where the old index tablespaces are not restored. After the database is opened, BRRECOVER creates new index tablespaces called `<OLD_TABLESPACE_NAME>I` with the same space attributes as the old tablespaces:

- Oracle 10g: the tablespaces are renamed back to their old names
- Oracle 9i: you can trace them back to their old names by subsequently creating new index tablespaces with the old names and executing an online index rebuild at tablespace level (this is `idrebuild` in `BRSPACE`).

 **Caution**

After new indexes have been created, you must [update optimizer statistics](#) on these indexes. For example, you can use `brconnect -f stats -t <tablespace name>` to do this. For more information, see [-f stats](#). This requires extra time, possibly canceling the expected time reduction from recreating indexes.

We recommend you to test the complete recovery procedure to find out whether you can indeed expect a time reduction of this approach.

## Backup Example for a Large Database

The following example shows how you can perform a large Oracle database backup:

- Size of the database:
 

The size of the database is 400 GB.
- Backup devices
 

The database backup is performed to locally mounted tape units. Four tape units are available in our example.
- Throughput
 

The throughput of the backup is approximately 15 to 20 GB per hour.
- Tape capacity
 

About 70 GB for tape units with [hardware compression](#), assuming the average compression rate for SAP data, about 3 to 4.
- Duration
 

The backup should be possible in less than 10 hours overnight.

If the database can be backed up in parallel to four tape devices, then it can be completed within ten hours, using hardware compression.

 **Note**

Remember that scalability is restricted. This means that backup performance does not rise in proportion to the number of tape devices. Therefore, if you use more tape devices, backup time increases for each device.

## Speeding Up the Backup

This section discusses how you can speed up a backup of your Oracle database. Backups can often take a long time. For example, if you use digital audio tape (DAT) devices as a backup medium, you can back up 8 to 12 GB per hour, using [hardware compression](#). This means that backing up a 50 GB database on one tape device of this type takes approximately five hours.

### Features

#### Parallel Backup

You can improve throughput by performing data backups [in parallel](#). If you have several backup devices, you can make a parallel backup with BRBACKUP by assigning several tape devices to the `init<DBSID>.sap` parameters `tape_address` and `tape_address_rewind` or several disks to parameter `backup_root_dir`. You can then make a backup of, for example, a 100 GB database in parallel on five tape devices in three hours.

For more information, see [tape\\_address](#), [tape\\_address\\_rew](#), and [backup\\_root\\_dir](#).

#### Logical Volumes

If you are using the [logical volume manager \(LVM\)](#), a logical volume is considered to be one disk. Therefore, you should not distribute a logical volume to be used for backup over several physical disks when this is not absolutely essential.

By keeping the logical volume on one physical disk, you minimize read/write head movement on the disk, so speeding up the backup.

### Activities

A good way to reduce backup time is to perform a [two-phase backup](#):

1. You back up to disk, using the [backup\\_dev\\_type](#) parameter.
2. You copy the disk backup to a tape volume, using the [tape\\_copy\\_cmd](#). You can assign the copy operation a lower priority.

## Standby Database

The standby database is supported officially by Oracle as of Version 8. The Oracle documentation contains detailed information on this database configuration. For more information, see [Standby Database Configuration](#).

#### Caution

Only use the standby database if you are an experienced user. If you decide to use the standby database, you are fully responsible for correctly configuring and running the standby database.

## Integration

The SAP tools [BRARCHIVE](#) (for backup of offline redo log files) and [BRBACKUP](#) (for database backup) support the standby database. For more information, see:

- [Standby Database: BRARCHIVE Backup of Offline Redo Log Files](#)
- [Standby Database: BRBACKUP Backup of Database Files](#)

BRARCHIVE and BRBACKUP support the standby database by helping you to:

- Copy the offline redo log files from the production database to the standby database, including a check on the copied files
- Import the offline redo log files into the standby database (recovery), followed by backup of the offline redo log files to tape
- Back up the standby database
- Create and configure a new standby database, and reconstruct the production database after a takeover



### Note

You can use BRBACKUP to perform *split-mirror backups* of your standby database. For more information, see [Split Mirror Backup](#).

You can use BRRECOVER to perform [database point-in-time recovery](#) and [whole database reset](#).

## Oracle Recovery Manager (RMAN) for Standby Database Backup

You can also use the Oracle Recovery Manager (RMAN) to back up your standby database as follows:

- Online backup with `backup_type = online_standby`  
You do *not* need to stop importing offline redo log files during an online backup with RMAN.
- Offline backup with `backup_type = offline_standby`  
You must stop importing offline redo log files during an offline backup with RMAN.

To use the Oracle Recovery Manager (RMAN) for a standby database backup, you need to set one of the following three possibilities in the `init<DBSID>.sap` parameter file:

- `backup_dev_type = rman_util|rman_disk|rman_stage`
- `backup_dev_type = tape|tape_auto|tape_box|pipe|pipe_auto|pipe_box`  
`tape_copy_cmd = rman|rman_dd`
- `backup_dev_type = disk`  
`disk_copy_cmd = rman|rman_set`

You can even use RMAN to perform level-0 full backups and incremental backups of your standby database.



## Features

Before you use the standby database, weigh up the advantages and disadvantages carefully.

### Advantages

- Very low failure rate

All system components are duplicated. The primary and standby instances can run on different hosts. They can also have separate locations depending on the safety requirements.

- Very short downtime

If an error occurs in the primary database system and it is necessary to recover the database, you can perform the recovery very quickly on the standby host. This avoids a time-consuming data-file restore, since these files are already located on the standby host. The only thing you need to do is to import the last entries from the redo log files. Therefore, the standby instance can take over the tasks of the primary instance very quickly.

- Significant decrease of the load on the production host

The database backup requires considerable resources and time for large databases. Since the backup can run on the standby host, the load on the primary instance is reduced significantly. Therefore, the resources on the production host are fully available for production operation, and database operation does not need to be interrupted or restricted for a backup.

### Disadvantages

- High costs

For a standby database scenario, all system components must be available in duplicate. In particular, duplicate hardware resources (CPU, hard disks, and so on) are expensive.

- High system administration expense

You need to set up the standby host. If structural changes are made on the primary database system, make sure these are incorporated on the standby host. When the standby instance has taken over production operation – a “takeover” – you must set up a replacement standby database.

- High requirements for switchover software

For the standby instance to take over production operation, the appropriate switchover software is required. You need to work with the hardware and software suppliers, who are responsible for selecting this software and ensuring that it functions correctly.



## Standby Database Configuration

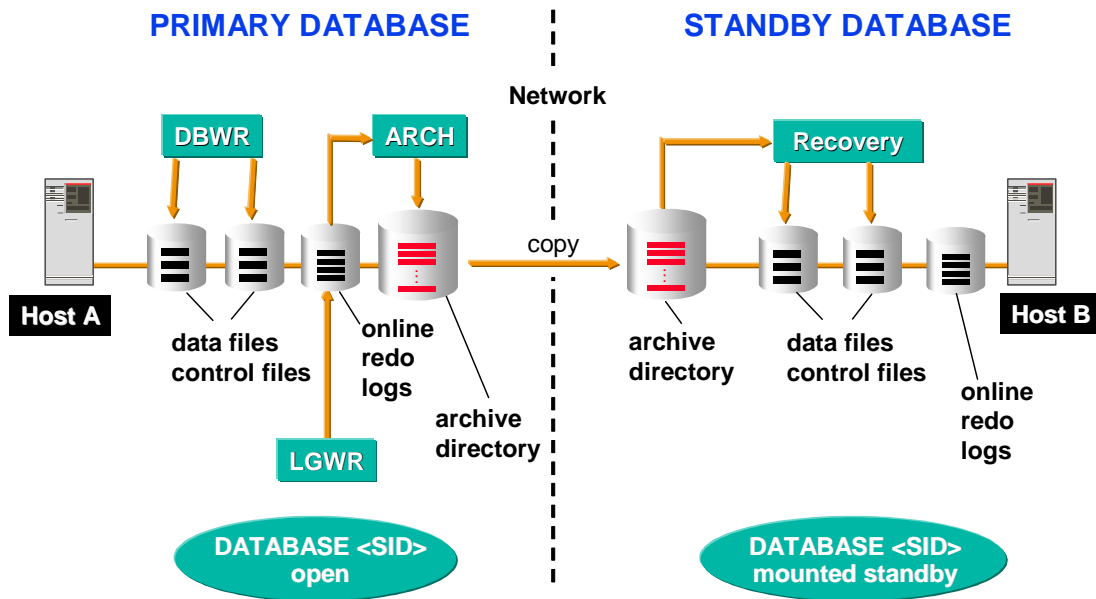
This section describes how to configure the Oracle Standby Database. For more information, including advantages and disadvantages of using the standby database, see [Standby Database](#).

When the primary (that is, production) database is duplicated on a standby database, this is referred to as a standby database configuration. The aim of this configuration is to minimize downtime if the primary database suffers an error, since the standby database can assume the role of the productive database in a very short time.

## Features

The following graphic illustrates the standby database:

Standby Database Configuration



Two identically configured databases operate on two identically configured hosts:

- The primary (that is, productive) Oracle instance is located on the first host. The database is open and fully available for all SQL statements of the SAP System. The primary database system is also the system that directly executes all database requests.
- The standby database is a copy of the primary database and is only intended as a recovery system.

The standby Oracle instance on the second host is in a mounted standby state (not opened) and is recovered constantly. This means that the standby instance incorporates all changes to the data of the primary instance either immediately, or with a chosen delay. To do this, the offline redo log files created in the primary database system are applied (only the redo entries already archived by Oracle can be imported).

If it is necessary to recover the primary database system (for example, after a media error), the standby instance can assume the functions of the primary instance in a very short time. This is a “takeover”, which means that the recovery mode of the standby instance is terminated and the standby database is opened for online operation.

Since all data files are already located on the standby host, costly reloading of the files is avoided. Some redo entries might still need to be applied to the files to enable all transactions to be incorporated in the standby instance. This means that you must first import the missing offline redo log files from the primary instance. You can then try to archive the current online redo log file of the primary instance with the Oracle command `ALTER SYSTEM ARCHIVE LOG CURRENT` and also to import these redo entries in the standby instance.

### Caution

If this command fails, it can be very risky to directly apply the current online redo log file to the standby database. If you try to directly apply the redo entries from the online redo log file, you might crash the standby database.

After the takeover, a standby database needs to be set up again (usually on what was the primary host).

### Caution

Changes to the physical structure of the primary database (creating new files, renaming files, changes to online redo log and control files) are in most cases automatically incorporated in the standby database. In some cases you might need to intervene depending on the type of change (for example, to create a soft link).

If it is not possible to incorporate the changes automatically, the recovery process is stopped, and you need to intervene manually to incorporate the structural change in the standby database. After that, you need to restart the recovery process.

The original names of the primary database ought to be retained. However, BRBACKUP supports renaming all database files to another SAPDATA\_HOME directory of BRBACKUP using the `init<DBSID>.sap` parameter [orig\\_db\\_home](#). If you use this parameter, it is even possible to run the standby database on the same host as the primary database. However, we do not recommend this for high availability.

### Caution

If commands are executed in the primary database with the `UNRECOVERABLE` option, these changes do not appear in the redo log files. It is therefore not possible for the standby instance to receive any information about such changes. In this case, no error messages appear during the recovery process. However, they are recorded in the standby database `ALERT` file. Therefore, be sure to check the `ALERT` file regularly.

### Note

For more information, see the Oracle documentation. The new and changed SQL and SQLPLUS commands are also described there as well as the necessary `init.ora` parameters, which are required for working with a standby database.

## Activities

To create a standby configuration, you make a copy of the production database as follows:

```
backup_dev_type = disk_standby|stage_standby
```

You can then use this as the standby database, as follows:

For more information, see [backup\\_dev\\_type](#).



# Standby Database: BRARCHIVE Backup of Offline Redo Log Files

You can use [BRARCHIVE](#) with the Oracle [Standby Database](#) to back up the offline redo log files.

BRARCHIVE can back up offline redo log files from the primary to the standby instance. This is possible because BRARCHIVE is able to copy offline redo log files to a local or remote hard disk.

## Prerequisites

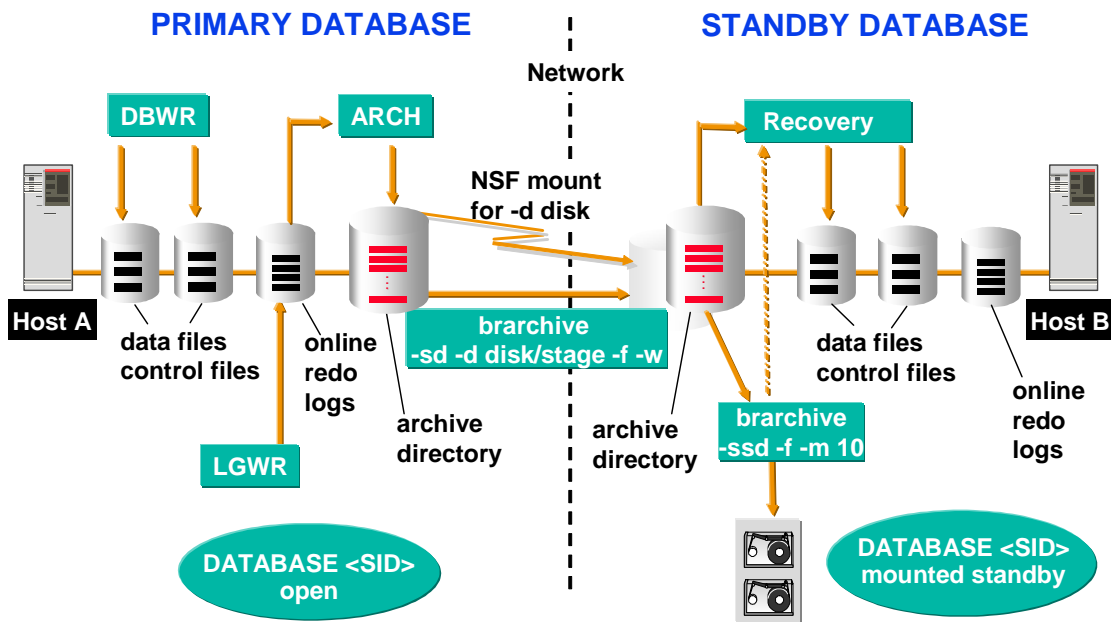
On *UNIX* systems, you can now perform BRBACKUP and BRARCHIVE backups of remote databases with the `OPSS$` user *without* entering the database user and password in the `-u` option. For more information, see [SAP Note 914174](#).

On *Windows* systems, this is possible in the standard Oracle configuration.

## Features

The following graphic illustrates how BRARCHIVE works with the standby database:

BRARCHIVE with Standby Database



- BRARCHIVE process on the primary host

This process copies the offline redo log files to an NFS-mounted or remote directory, which represents the archive directory (usually oraarch) on the standby host. For a mounted directory, the copy process uses an insecure network protocol. In this case, you should use BRARCHIVE with the [-wj-verify](#) option.

- Reliable copy programs

BRARCHIVE can use reliable copy programs such as `rsh`, `ftp`, or `scp` for the copying of the offline redo log files from the primary host onto the standby host, with

the [backup\\_dev\\_type](#) parameter set to stage. Therefore, NFS-mounting and verification are no longer necessary.

- BRARCHIVE process on the standby host

This process waits for the offline redo log files to be copied into the local archive directory. If a redo log file was copied completely, BRARCHIVE applies these redo entries to the standby instance with the [-m|-modify](#) option, backing up the redo log file and deleting it if necessary. Therefore, BRARCHIVE starts the recovery process of the standby database, in which the offline redo log files are processed individually.

- Delaying import of redo entries

You can delay importing the redo entries, using the `<delay>` parameter with the [-m|-modify](#) option. If a logical error occurs in the primary instance (for example, accidental deletion of a table), you can prevent this error from being imported in the standby instance, by stopping the BRARCHIVE run.

- Importing offline redo log files

You import the offline redo log files with the following Oracle command:

```
RECOVER STANDBY DATABASE;
```

When the volume of offline redo log files is high, the import to the standby database occurs parallel to the backup, leading to significantly faster processing. For more information, see [-m|-modify](#).



#### Caution

To import the redo log files, the database user (usually `SYSTEM`) must have `SYSDBA` authorization.



#### Note

BRARCHIVE can now automatically process new database files when importing offline redo log files into the standby database. In the past this caused an error.

However, if you access the database file with a softlink (for example, a raw device file), you must still set up the softlink manually on the standby database. You must do this at the same time as you set up the file on the primary database.



## Standby Database: BRBACKUP Backup of Database Files

You can use [BRBACKUP](#) to back up the data files and control files of the [Standby Database](#).

A major advantage of the standby database is that you do not have to perform backups on the primary (that is, production) database. Instead, you can perform the backup on the standby database using BRBACKUP. This means that the database backup does not add to the load on the primary database host. Since there is no online operation on the standby database, all host resources are available for the database backup.

## Prerequisites

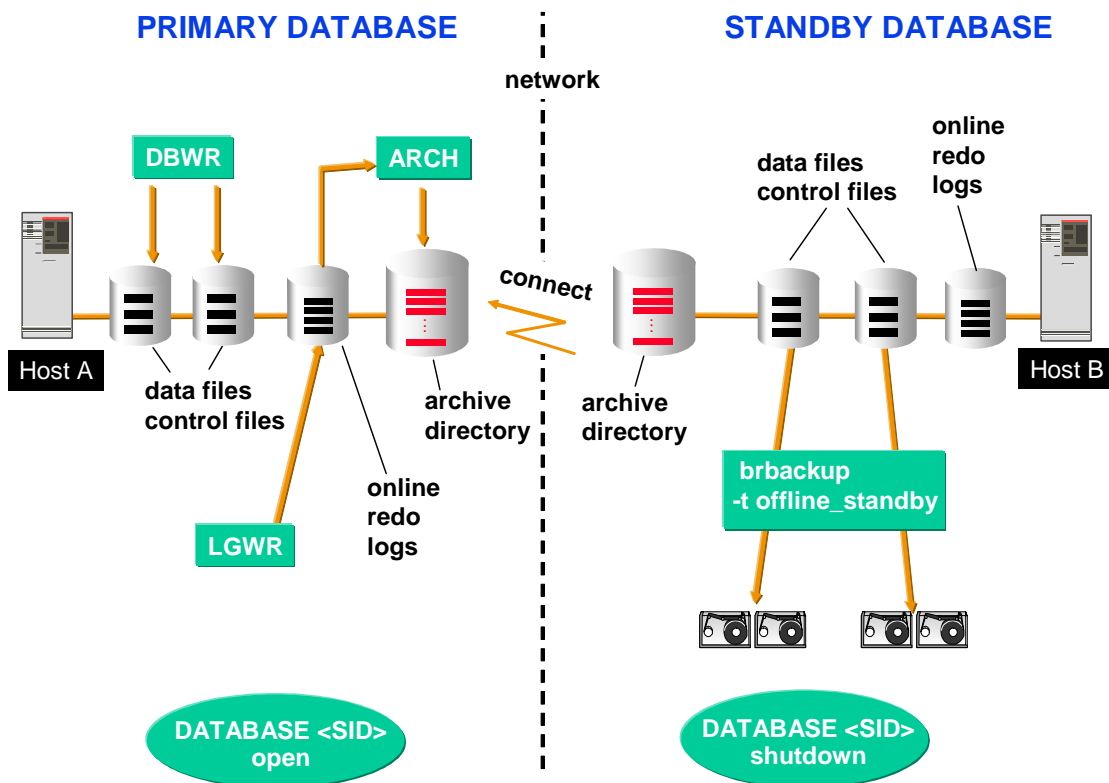
- The standby instance is in the recovery state and cannot be opened. You can only perform an offline backup without RMAN. But with RMAN, the database can still remain in mount standby state. For the BRBACKUP backup of the standby data, you must set the [backup\\_type](#) parameter to `offline_standby` or `online_standby`.
- BRBACKUP supports renaming all database files to another SAPDATA\_HOME directory using the [orig\\_db\\_home](#) parameter. If you use this parameter, you can even run the standby database on the same host as the primary database. However, we do not recommend this for high availability.
- For connection to a remote host, you must meet the [remote database connect requirements](#).

On *UNIX* systems, you can now perform BRBACKUP and BRARCHIVE backups of remote databases with the `OPSS` user *without* entering the database user and password in the `-u` option. For more information, see *SAP Note 914174*.

On *Windows* systems, this is possible in the standard Oracle configuration.

## Features

The following graphic shows how backup occurs in a standby database:



- BRBACKUP logs on to the primary database instance using the instance string from the [primary\\_db](#) parameter. It retrieves the information required on the database structure. This information is written to the backup logs.
- BRBACKUP stops the standby database instance.
- BRBACKUP backs up the standby data.

- After the backup has finished, the original state of the standby database instance is recovered. If the database was in a recovery state, this state is restored, using the Oracle commands `STARTUP NOMOUNT`, `ALTER DATABASE MOUNT STANDBY DATABASE`.

## More Information

[Structure-Retaining Database Copy](#)



## Standby Database: Restore and Recovery

This section describes what you need to do in the event of a failure with the Oracle [Standby Database](#).

### Activities

If the production database fails, you can do one of the following:

- Switch to the standby database, which now becomes the production database

#### Caution

If you switch to the standby database, be sure to perform backups as described below in *Control File Backup and Complete Online Backup After a Switch*.

After you have fixed the problem on the previous production database, you can do one of the following:

- Stay with the swapped configuration

This means that you continue production operation on the previous standby database. In effect, you have swapped the configuration, with the previous production database becoming the standby database.

- Resume the original configuration

This means that you switch back to the previous production database, to resume the configuration from before the failure. For more information, see "Resuming the Original Configuration after a Switch" below.

#### Note

Which option you choose depends on whether the production and standby databases are equivalent in terms of processing power. Often, the standby database is less powerful, in which case it makes sense to switch back to the original production database as soon as possible.

- Fix the problem and continue on the production database. If necessary, restore and recover the database with the redo log files, if available.

## Control File Backup and Complete Online Backup After a Switch

If the production database fails, you can do one of the following after a takeover, you should back up the control file immediately. Otherwise, you cannot recover the standby database at all. For example, you can enter the following command to do this:

```
brbackup -m 0 -t online
```

For more information, see [-m|-mode](#).

We then *strongly recommend* you to perform a complete online backup as soon as possible. For example, you can enter the following command to do this:

```
brbackup -m all -t online
```

This is the only way to make sure that the standby database can be recovered, if necessary.

Be sure to perform this operation *whenever* you switch production from one database to another.

## Resuming the Original Configuration After a Switch

To recreate and configure a production database after a switch, you can use BRBACKUP to perform an offline backup of the current production database (that is, the database that was functioning as a standby before the switch). To do this, set:

```
backup_type = offline_stop
```

For more information, see [backup\\_type](#).

After the backup, you directly switch the database to mount-standby state, so it can resume the role of the standby database. After a restore on the production database, this can resume its role as production database. If you want to set it up in one step (that is, without a restore), you also set one of the following:

```
backup_dev_type = disk_standby|stage_standby
```

```
stage_copy_cmd = rcp|ftp|scp
```

For more information, see [backup\\_dev\\_type](#) or [stage\\_copy\\_cmd](#).



## Standby Database: Remote Database Connect Requirements

Certain conditions are required for the connection to a remote host with the [Oracle Standby Database](#). You must be able to manage the primary Oracle instance from the standby instance, that is, you must be able to start up and shut down this instance from the standby host. You must be able to perform these operations from a local SQLPLUS session.

### Prerequisites

[BRARCHIVE](#) and [BRBACKUP](#), which are started on the standby database server, connect remotely to the primary database. Therefore, make sure that the instance string in the [primary\\_db](#) parameter is defined to Oracle SQL\*Net in the `tsnnames.ora` file.

Test the connection in SQLPLUS with the following command:

```
connect system/<password>@<value_of_primary_db>
```



## Procedure

1. Create an Oracle password file on the primary database:

```
orapwd file=<ORACLE_HOME>/dbs/orapw<DBSID>  
password=<sys_password> entries=10
```

2. Set the `remote_login_passwordfile` parameter in the `init<DBSID>.ora` as follows:

```
remote_login_passwordfile = exclusive
```

If the parameter is entered only after an instance has been started up, you must restart it, so that the parameter becomes effective.

3. Start as user `SYS` and execute the Oracle command:

```
SQL> connect / as sysdba  
  
SQL> grant sysoper to system;
```

This grants the `system` user `SYSOPER` authorization on the primary database instance.

4. If necessary, change the password for the `system` user:

```
SQL> alter user system identified by <password>;
```

### Note

It is not necessary to give the `SYSTEM` user `SYSOPER` authority in the Oracle password file of the production database if the primary database remains open all the time before and during the backup of the standby database.

For more information, see *SAP Note* [131610](#).

## Split Mirror Backup

This section describes how you can use split mirror disk backup with [BRBACKUP](#) to perform an online or offline backup of your Oracle database without downtime. We recommend this especially for online backup of large databases.

### Caution

We provide this information on split mirror backup for advice only. Consult your hardware partner for help in setting up split mirror backup.

It is especially important to make sure that the data on the backup database host is consistent when you perform the backup.

## Integration

- Computing Center Management System (CCMS)

- You cannot use CCMS to schedule split mirror disk backups. This is because BRBACKUP runs on the backup database host if scheduled from the [DBA Planning Calendar](#).
- However, you can use CCMS to monitor backups on the production system, because BRBACKUP sends status information to the production database, which CCMS can then use.
- The BACKINT interface is also supported in the split mirror configuration. This configuration is transparent to BACKINT and there are no extra considerations for BRBACKUP.
- A split mirror disk backup only includes database backups with BRBACKUP. Backups of the offline redo log files using [BRARCHIVE](#) are not included because they do not place significant load on the production host. In a standard setup, you back up the offline redo log files on the database server to local or remote tape devices, on disk or with BACKINT.

If the backup device is connected to the backup server, you can use the remote device – using [backup\\_dev\\_type](#) = pipe|pipe\_auto|pipe\_box – or the BACKINT interface.

- To synchronize BRBACKUP sessions with BRARCHIVE sessions, you can start a BRARCHIVE session immediately after the disks are split and not just at the end of the BRBACKUP run. When the disks have been split, Oracle regards the backup as already finished.
- You can use RMAN for split-mirror backups, where :

[backup\\_type](#) =  
online\_split|online\_mirror|offline\_split|offline\_mirror

For more information about setting the parameters for RMAN with split-mirror backups, see *Profile Parameters and Command Options for init<DBSID>.sap* in [Split Mirror Backup: Software Configuration](#).

You can even do level-0 full backups and incremental backups in this way. As a prerequisite you need to make a sure that the `sapbackup` directory is shared or mounted for the database *and* the backup server.

 **Caution**

To be able to perform an RMAN backup on the backup server, BRBACKUP sets the database on the backup server to mount state. Therefore, the Oracle server software must be fully installed on the backup server.

- You can use split-mirror backups to back up your [standby database](#), but only in *offline* split mode. You need to first stop the import of offline redo log files. This is the new backup type:

[backup\\_type](#) = offstby\_split|offstby\_mirror

Similar to the connection definition to the production database in the `init<DBSID>.sap` parameter [primary\\_db](#), you also need to define a connection to the standby database as follows:

```
standby_db = <conn_name>
```

For more information, see [standby\\_db](#).

## Features

In the split mirror configuration, the disks of the production database host are mirrored, that is, synchronized. BRBACKUP runs on a backup database host, where the backup is performed after the mirror disks have been split and mounted. On the production host, this saves processing power otherwise required for the backup, so that the production SAP System is unaffected by the backup.

BRBACKUP can be used to control the splitting and later synchronization of the disks. If you want to open the database on the backup host and use it as a “reporting server” you can perform the synchronization yourself later.

The actual splitting and later synchronization of disks is executed by a script or program supplied and supported *not* by SAP, but by the manufacturer of the operating system, disk subsystem, or backup software.

If the split disks do not have to be resynchronized immediately after the backup you can use the backup server along with the mirror disks to operate an independent SAP system on the backup server. For this you must install the entire Oracle server software on the backup server.

For more information, see the following:

- To start split mirror backup on the production database server, see *SAP Note* [170607](#).
- To only split the mirror disks without starting a backup, see *SAP Note* [378818](#) and [811637](#).

## Activities

There are two basic scenarios:

- Split Command Scenario  
This uses [split\\_cmd](#) and [resync\\_cmd](#) to split and resynchronize the mirror disks. BRBACKUP controls the splitting and later synchronization of the disks.
- SPLITINT Scenario  
This uses the SPLITINT interface program with [split\\_options](#) and [split\\_resync](#) to split and resynchronize the mirror disks. BRBACKUP calls SPLITINT to actually split and resynchronize the disks. SPLITINT executes the split and resync requests from BRBACKUP using the appropriate disk subsystem calls.

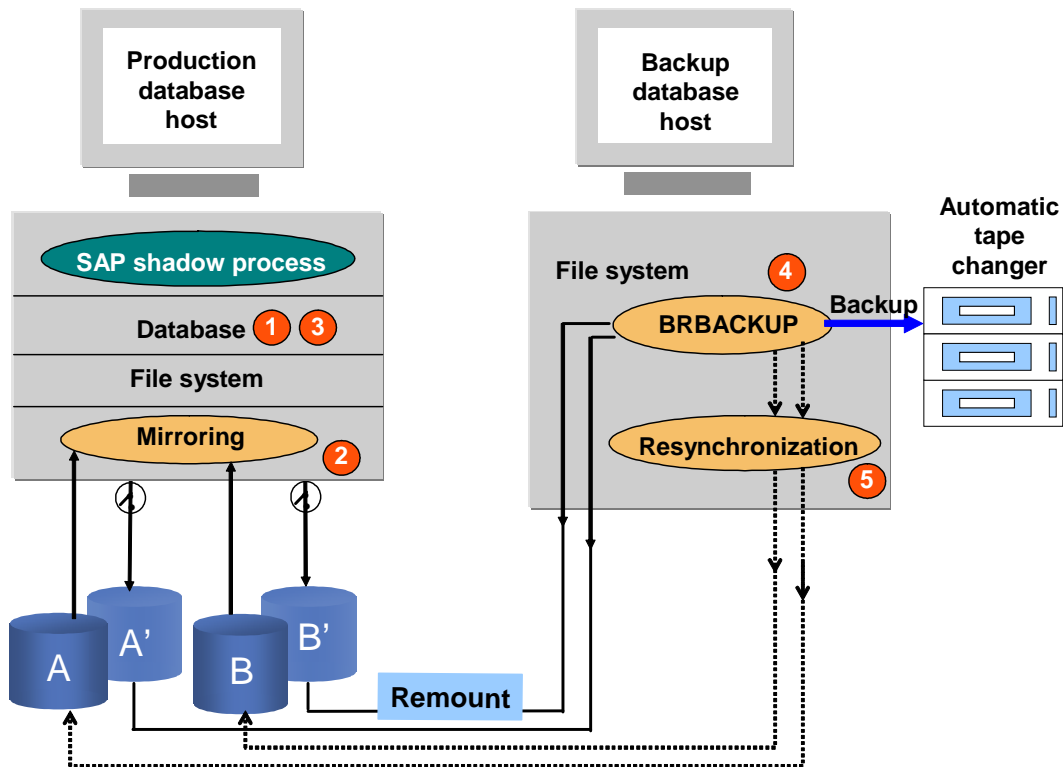
### Note

Database handling with the SPLITINT scenario is considerably faster than with the split command scenario. This is the recommended approach.

For each scenario, there is an online and an offline variant.

The following graphic shows how split mirror backup works:

Split Mirror Backup



	<a href="#">Split Mirror Online Backup</a>	<a href="#">Split Mirror Offline Backup</a>
1	Tablespaces set to status BACKUP	Production database shut down
2	Mirror disks (A' and B' in the graphic) split and connected to backup server using: . Split command scenario: <code>split_cmd</code> SPLITINT scenario: SPLITINT program	
3	Tablespaces reset to normal status	Production database restarted
4	Mirror disks backed up on backup host	
5	Optional: Primary and mirror disks resynchronized using: . Split command scenario: <code>resync_cmd</code> SPLITINT scenario: SPLITINT program	

**↑ Recommendation**

Consider online split mirror disk backup if high availability is an important consideration for your system.

## Split Mirror Online Backup

This section describes how you can use [BRBACKUP](#) to perform a [split mirror](#) online backup with your Oracle database. We especially recommend this for large databases where high availability is an important consideration.

In an online backup the system remains available the whole time. The tablespaces to be backed up are temporarily set to BACKUP status while the mirror disks are split. In comparison to the normal online backup, in which the tablespaces retain this status during the whole backup period, the amount of redo information is significantly reduced.

### Prerequisites

You set the following parameters in the [initialization profile init<DBSID>.sap](#):

- [backup\\_type](#):
  - Split command scenario: `backup_type = online_split`
  - SPLITINT scenario: `backup_type = online_mirror`

Instead of the `backup_type` parameter, you can enter the BRBACKUP command option `-t|-type` as shown below in *Activities*. The BRBACKUP command option takes priority.

- Split command scenario: [split\\_cmd](#) and [resync\\_cmd](#)
- SPLITINT scenario: [split\\_options](#) and [split\\_resync](#)

#### Note

All copies of control files must be located on disks that can be split, for example, in `sapdata` directories.

## Activities

### Split Command Scenario

You start BRBACKUP on the backup host using the following command :

```
brbackup -t online_split
```

For more information, see [-t|-type](#).

BRBACKUP then performs the following steps:

1. Sets the tablespaces to be backed up to backup status using:

```
ALTER TABLESPACE <tablespace name> BEGIN BACKUP;
```
2. Splits the mirror disks using [split\\_cmd](#).
3. Resets the tablespaces to the normal status using:

```
ALTER TABLESPACE <tablespace name> END BACKUP;
```
4. Backs up the mirror disks on the backup host.

5. If you have set [resync\\_cmd](#), synchronizes the mirror disks with the originals.

## SPLITINT Scenario

You start BRBACKUP on the backup host using the following command :

```
brbackup -t online_mirror
```

For more information, see [-t|-type](#).

BRBACKUP and SPLITINT then perform the following steps:

1. BRBACKUP calls SPLITINT without setting the tablespaces to backup status.
2. SPLITINT prepares the split. When complete, SPLITINT sends a message to BRBACKUP.
3. When BRBACKUP receives the message, it sets the tablespaces to backup status.
4. SPLITINT executes the split. When complete, SPLITINT sends a message to BRBACKUP.
5. When BRBACKUP receives the message, it takes the tablespaces out of backup status.
6. SPLITINT completes the split and sends a success message before returning control to BRBACKUP.
7. BRBACKUP starts the actual backup (for example, with BACKINT).

### Note

Resynchronization (the next step) occurs after the actual backup (for example, with BACKINT) has finished. It is optional and only needs to be executed if the setting in `init<DBSID>.sap` is [split\\_resync](#) = yes.

8. SPLITINT performs the resynchronization and sends a success message before returning control to BRBACKUP.
9. BRBACKUP cleans up and terminates.

### Note

For more information, see the documentation *BC-BRI SPLITINT Interface for Oracle Databases*, which you can find here:

► [www.sdn.sap.com/irj/sdn/ora](http://www.sdn.sap.com/irj/sdn/ora) ► Backup and Recovery ◀

## More Information

[Split Mirror Backup: Software Configuration](#)



## Split Mirror Offline Backup

This section describes how you can use [BRBACKUP](#) to perform a [split mirror](#) offline backup with your Oracle database.

In an offline backup the database is shut down to split the mirror disks. Therefore, it is considerably faster than a normal offline backup, in which the database is shut down for the duration of the entire backup.

## Prerequisites

You set the following parameters in the [initialization profile init<DBSID>.sap](#):

- [backup\\_type](#)
  - Split command scenario: `backup_type = offline_split`
  - SPLITINT scenario: `backup_type = offline_mirror`

Instead of `backup_type`, you can enter the BRBACKUP command option `-t|-type` as shown below in *Activities*. The BRBACKUP command option takes priority.

- Split command scenario: [split\\_cmd](#) and [resync\\_cmd](#)
- SPLITINT scenario: [split\\_options](#) and [split\\_resync](#)

### Caution

All copies of control files and online redo log files must be located on disks that can be split, for example, in `sapdata` directories for control files.

On *UNIX* systems, you can now perform offline split-mirror backups of remote databases with the `OPS$` user *without* entering the database user and password in the `-u` option. For more information, see [SAP Note 914174](#).

On *Windows* systems, this is possible in the standard Oracle configuration.

## Activities

### Split Command Scenario

You start BRBACKUP on the backup host using the following command :

```
brbackup -t offline_split
```

For more information, see [-t|-type](#).

BRBACKUP then performs the following steps:

1. Shuts down the database on the productive host, in order to guarantee the consistency of the database for an offline backup.
2. Splits the mirror disks using [split\\_cmd](#).
3. Restarts the database on the production host, so that it is again available for the SAP System.
4. Backs up the mirror disks on the backup host.
5. If you have set [resync\\_cmd](#), synchronizes the mirror disks with the originals.

## SPLITINT Scenario

You start BRBACKUP on the backup host using the following command :

```
brbackup -t offline_mirror
```

For more information, see [-tj-type](#).

BRBACKUP and SPLITINT then perform the following steps:

1. BRBACKUP calls SPLITINT without stopping the database.
2. SPLITINT prepares the split. When complete, SPLITINT sends a message to BRBACKUP.
3. When BRBACKUP receives the message, it stops the database.
4. SPLITINT executes the split. When complete, SPLITINT sends a message to BRBACKUP.
5. When BRBACKUP receives the message, it restarts the database.
6. SPLITINT completes the split and sends a success message before returning control to BRBACKUP.
7. BRBACKUP starts the actual backup (for example, with BACKINT).

### Note

Resynchronization (the next step) occurs after the actual backup (for example, with BACKINT) has finished. It is optional and only needs to be executed if the setting in `init<DBSID>.sap` is [split\\_resync](#) = yes.

8. SPLITINT performs the resynchronization and sends a success message before returning control to BRBACKUP.
9. BRBACKUP cleans up and terminates.

### Note

For more information, see the documentation *BC-BRI SPLITINT Interface for Oracle Databases*, which you can find here:

► [www.sdn.sap.com/irj/sdn/ora](http://www.sdn.sap.com/irj/sdn/ora) ► Backup and Recovery ◀

## More Information

[Split Mirror Backup: Software Configuration](#)

## Split Mirror Backup: Software Configuration

This section describes the software configuration for [split mirror backup](#):

- At least the client software and SQLPLUS components of the Oracle database software must be installed on the backup host.



- The Oracle home directory structure must correspond to the SAP standard installation required by BRBACKUP.

### Default Directories

UNIX	Windows
<code>\$ORACLE_HOME/dbs</code>	<code>%ORACLE_HOME%\DATABASE</code>
<code>\$ORACLE_HOME/bin</code>	<code>%ORACLE_HOME%\BIN</code>
<code>\$SAPDATA_HOME/sapbackup</code>	<code>%SAPBACKUP%</code>
<code>\$SAPDATA_HOME/saparch</code>	<code>%SAPARCH%</code>
<code>\$SAPDATA_HOME/sapreorg</code>	<code>%SAPREORG%</code>
<code>\$SAPDATA_HOME/sapcheck</code>	<code>%SAPCHECK%</code>
<code>\$SAPDATA_HOME/saptrace</code>	<code>%SAPTRACE%</code>

### Note

BRBACKUP must be able to access the control files on the backup server created by the SQL command `alter database backup controlfile to <filename>` on the production server.

BRBACKUP tries to copy the control files to the backup server using the `rcp` or `ftp` command – depending on the value of [stage\\_copy\\_cmd](#) – and the host name for the `HOST_NAME` field of the `V$INSTANCE` view.

Alternatively, you can use Network File Systems (NFS) to mount the `sapbackup` directory from the production server to the backup server.

For more information, see *SAP Note* [156704](#).

- The Oracle and BRBACKUP profiles should be available in `$ORACLE_HOME/dbs` or `%ORACLE_HOME%\DATABASE`.
- Set the [backup\\_type](#) parameter in the `init<DBSID>.sap` profile to `offline_split |online_split|offline_mirror|online_mirror`.
- To establish the connection between the backup server and the production server, you must define the profile parameter [primary\\_db](#) for the SQLNET connection. Before this, you must perform the following steps on the primary database host:

- Create an Oracle password file:

```
orapwd file=<ORACLE_HOME>/dbs/orapw<DBSID> password=<SYS password> entries=10
```

- Set the `remote_login_passwordfile` parameter to `exclusive` in the `init<DBSID>.ora` profile or `sfile`.
- Give the system user `SYSOPER` authorization in the production database. Start SQLPLUS as user `SYS` and execute the following Oracle command:

```
SQL> connect / as sysdba
```

```
SQL> grant sysoper to system;
```

- o If needed, change the password for the `system` user:

```
SQL> alter user system identified by <password>;
```

For more information, see [SAP Note 131610](#)

- The `SYSOPER` authorization and the password file are only required to stop the primary database for the offline split. For an online split, the standard remote connection using the instance string specified in the [primary\\_db](#) parameter is sufficient.
- The primary database should remain open during the entire backup.
- Make sure that the directory `/usr/sap/<SAPSID>/SYS/exe/run` or `\\<Host name>\sapmnt\<SAPSID>\SYS\exe\run` is accessible from the backup host and that it contains at least the programs `BRBACKUP`, `BRCONNECT`, and `BRTTOOLS` (and optionally other BR programs).

#### Note

Normally make sure that the paths of all database files accessed by the production database and backup database are identical, although this is not absolutely essential.

You can mount the database files on the backup host in a different `SAPDATA_HOME` directory by using the `init<DBSID>.sap` parameter [orig\\_db\\_home](#). By using this parameter you can even mount the split-off files on the same host.

- The manufacturer of the external backup tool should install and configure the `BACKINT` interface on the backup host, if used.
- To use the Oracle Recovery Manager (RMAN) for a split mirror backup, you need to set one of the following three possibilities in the [init<DBSID>.sap](#) parameter file:

- o `backup_dev_type = rman_util|rman_disk|rman_stage`
- o `backup_dev_type =  
tape|tape_auto|tape_box|pipe|pipe_auto|pipe_box  
tape_copy_cmd = rman|rman_dd`
- o `backup_dev_type = disk  
disk_copy_cmd = rman|rman_set`

## Profile Parameters and Command Options for `init<DBSID>.sap`

Make sure that you have set the following parameters in the [initialization profile `init<DBSID>.sap`](#):

- [backup\\_type](#)=  
`offline_split|online_split|offline_mirror|online_mirror|offstby_split|offstby_mirror`
- Split command scenario using `offline_split`, `online_split`, or `offstby_split`:
  - o [split\\_cmd](#) = "`<split_cmd> [$]`"

<split\_cmd> is a program or shell script called by BRBACKUP to split the mirror disks.

- o [resync\\_cmd](#) = "<resync\_cmd> [\$]"

<resync\_cmd> is a program or shell script called by BRBACKUP to resynchronize the mirror disks. This parameter must be set so that BRBACKUP performs the resynchronization.

At runtime, BRBACKUP replaces the optional character \$ with the name of the text file that contains the names of all files to be split or resynchronized.

 Note

If [split\\_cmd/resync\\_cmd](#) is completed successfully, an exit code of 0 is returned. Only messages beginning with #INFO are accepted, that is, these are the only messages not interpreted as error messages. If the command is not successful, a return code of > 0 is returned as well as messages describing the cause of the error.

- SPLITINT scenario using [offline\\_mirror](#), [online\\_mirror](#), or [offstby\\_mirror](#):

- o [split\\_options](#) = "<split\_options>"

This defines the additional options used to call SPLITINT. Your SPLITINT vendor can tell you how to set them.

- o [split\\_resync](#) = no|yes

This controls whether the mirror disks need to be resynchronized immediately after the backup has finished.

- [primary\\_db](#) = <inst\_str>

<inst\_str> is an instance string to the production database to connect from the backup server to the production server. The connections are defined in the Oracle configuration file `tnsnames.ora`.

- [standby\\_db](#) = <inst\_str>

<inst\_str> is an instance string to the standby database to connect from the backup server to the standby server. The connections are defined in the Oracle configuration file `tnsnames.ora`. BRBACKUP needs SYSDBA privileges for this connection so as to be able to restart the standby database. For more information, see the Oracle password file above.

- The BRBACKUP call is as follows:

- o Split command scenario:

- Online backup of mirror disks:

```
brbackup -t|-type online_split
```

- Offline backup of mirror disks:

```
brbackup -t|-type offline_split
```

- o SPLITINT scenario:

- Online backup of mirror disks:
 

```
brbackup -t|-type online_mirror
```
- Offline backup of mirror disks:
 

```
brbackup -t|-type offline_mirror
```
- For more information, see [-t|-type](#). This command line option overrides the backup type set in the [backup\\_type](#) parameter in the `init<DBSID>.sap` profile.
- [pre\\_split\\_cmd](#) and [post\\_split\\_cmd](#)

You can use these parameters to execute external commands before or after the BRBACKUP disk split.
- [pre\\_shut\\_cmd](#) and [post\\_shut\\_cmd](#)

You can use these parameters to execute external commands before or after the database is stopped for an offline backup with BRBACKUP.

## Backup with Automatic Tape Changers

In addition to standard backups to local disks or tapes, [BRBACKUP](#) supports several types of Oracle backup with automatic tape changers, such as autochangers, autoloaders, jukeboxes, or tape robots.

### Features

The following table shows the types of backup supported by BRBACKUP:

Backup Type	Parameter	
Local disks	disk	
Remote disks	stage	
Local tape devices with manual tape swapping	tape	
Remote tape devices with manual tape swapping	pipe	
Automatic sequential tape swapping: autochangers or autoloaders	Local tape devices	tape_auto
	Remote tape devices	pipe_auto
Fully automatic tape swapping: jukeboxes or tape robots	Local tape devices	tape_box
	Remote tape devices	pipe_box

The following commands are used to define automatic mounting and dismounting of tapes in the backup device for `tape_box` and `pipe_box`:

- [mount\\_cmd](#)

- [dismount\\_cmd](#)

These commands enable you to administer hundreds of tapes in jukeboxes and autoloaders. For more information, see [Mount and Dismount Commands](#).

## Activities

You can set the parameters from the above table in either of the following ways:

- In the [backup\\_dev\\_type](#) parameter of the [Initialization Profile init<DBSID>.sap](#)
- In the [-dl-device](#) command option of BRBACKUP (this overrides the initialization profile)

Some devices with automatic tape changing require additional time to change the tape. This additional time can be taken into account by adjusting [rewind\\_offline](#). To do this, you can enter a command like the following:

```
rewind_offline = "mt -t $ offline && sleep 60"
```

## More Information

[Autoloader Backup Example](#)



## Mount and Dismount Commands

The mount and dismount commands are relevant if you use [automatic tape changers](#), such as jukeboxes or tape robots, for your Oracle database.

[BRBACKUP](#) or [BRARCHIVE](#) does the following:

1. Before accessing tapes for the first time, they perform the command defined in [mount\\_cmd](#) to automatically mount the tapes.
2. When the backup has finished, BRBACKUP or BRARCHIVE switches the corresponding tape devices to offline mode.
3. Then they call the command defined in [dismount\\_cmd](#) to automatically dismount the tapes.

### Caution

For some tape devices, the tapes might not be switched to offline mode. The dismount command is then sufficient to dismount the tapes. In this case, do not set [rewind\\_offline](#).

The mount and dismount commands used in most implementations require special control drivers, which are defined in the parameters [tape\\_address\\_ctl](#) or [tape\\_address\\_ctl\\_arch](#).

Syntax of the commands mount and dismount commands:

```
mount_cmd = "<mount_cmd> <A> <B> <C> <D>"
```

```
dismount_cmd = "<dismount_cmd> <A> <B> <D>"
```

The command options <A> to <D> mean the following:

- <A> identifies the database to be backed up and has the following structure:

<DBSID>-A for BRARCHIVE and BRRESTORE with option -a|-a1|-a2

<DBSID>-B for BRBACKUP and BRRESTORE without option -b|-b1|-b2

where <DBSID> is the Oracle system ID (that is, the database instance name)

This option can be used to identify a subset of tapes ("sub-pool") for database backup. The additional extension -A and -B enables you to define and administer separate tape sub-pools for BRARCHIVE and BRBACKUP.

- <B> identifies the tape devices, on which the mount or dismount operations are to be performed, using the control driver addresses:

<dev\_addr1>,<dev\_addr2>,...

where <dev\_addr> is the driver address of the tape unit

One or more of the tape devices defined in the parameter [tape address ctl](#) can be addressed.

- <C> defines the tape names to be mounted on the tape devices given in <B>:

<tape name1>,<tape name2>,...|SCRATCH

where <tape name> is the name of the tape to be mounted

The tape names are chosen from the backup volume list [volume backup](#) (from BRBACKUP) or [volume archive](#) (from BRARCHIVE) by the automatic tape administration. The assignment of the tapes on the tape devices can be defined freely by the mount command. The number of tape names can be greater than the number of driver addresses. In this case any tapes from this list can be mounted on the given tape devices.

The reserved tape name SCRATCH means that any unlocked tapes (that is, those for which the expiration period has finished) can be mounted. Which of the available tapes is mounted is decided by the mount command.

- <D> transfers the name of a parameter file to the mount or dismount command, which contains additional configuration parameters for these commands. The name of the file is defined in the [mount par file](#) parameter or by the command option [-r|-parfile](#).



#### Caution

The user must provide the mount or dismount command in the form of a program, shell script or a batch file.

The successful completion of the command is indicated by the exit code 0 and the absence of any output (except for lines starting with #INFO).

## Autoloader Backup Example

If you are using [Backup with Automatic Tape Changers](#), this example shows how to integrate an autoloader, HP 48AL, into the backup strategy for [BRBACKUP](#) or [BRARCHIVE](#).

### Caution

Make sure that you have correctly configured the mount and dismount commands. This is your complete responsibility. Consult your hardware partner if in doubt.

During this, the `mtx` command is used for the mounting and dismounting of the tapes. In this example, we assume that the autoloader has 20 slots. The fifth and sixth characters of the tape name are used for the number of the slot:

Slot 1-10: tapes C11A01, C11A02, ....., C11A10

Slot 11-20: tapes C11B11, C11B12, ....., C11B20

The backup of the database C11 with BRBACKUP and the backup of the offline redo log files each require one tape. Automatic tape administration is switched on.

An extract from the profile `initC11.sap` might look like the following:

```
...
backup_dev_type = tape_box
tape_address = /dev/rmt/0mn
tape_address_rew = /dev/rmt/0m
tape_address_ctl = /dev/scsi/3
mount_cmd = "mount_tape.csh $ $ $"
dismount_cmd = "dismount_tape.csh $ $"
volume_archive = (C11A01, C11A02, ....., C11A10)
volume_backup = (C11B11, C11B12, ....., C11B20)
...
```

**C shell script** `mount_tape.csh`:

```
#!/bin/csh -f
set slot=`echo $3 | cut -b 5-6`
mtx -d $2 -l $slot
echo $slot > $SAPDATA_HOME/sapbackup/.slot
```

**C shell script** `dismount_tape.csh`:

```
#!/bin/csh -f
set slot=`cat $SAPDATA_HOME/sapbackup/.slot`
```

```
mtx -d $2 -u $slot
```

These scripts must be executable and be located in the directory  
/usr/sap/C11/SYS/exe/run.

## Procedure for a BRBACKUP Backup with Mount and Dismount Commands:

After BRBACKUP starts, the automatic tape administration of BRBACKUP chooses a tape (for example, C11B11) from the backup volume list. Internally the mount command is called up to mount the chosen tape in the tape device:

```
mount_tape.csh C11-B /dev/scsi/3 C11B11
```

In this script the following `mtx` command is executed:

```
mtx -d /dev/scsi/3 -l 11
```

This command causes the tape to be mounted from the 11th slot in the tape device. If the command is performed without an error message and with exit code 0, BRBACKUP assumes that the tape has been mounted successfully. After a tape label check, BRBACKUP starts the backup to the tape.

After the backup has finished the tape is again dismounted in the tape device:

```
dismount_tape.csh C11-B /dev/scsi/3
```

In this script the following `mtx` command is executed:

```
mtx -d /dev/scsi/3 -u 11
```

This command dismounts the current tape and returns it to slot 11.

The BRARCHIVE backup procedure is similar to the BRBACKUP backup described here. In this case, a tape C11A01,....,C11A10 is chosen from the slots 1-10.

### Note

For simplification neither the first option (`ORACLE-SID`) nor the last option (`mount_par_file`) of the mount or dismount command has been used in the example described here. You can make use of these options in more demanding scripts or programs, such as to perform parallel backups on several tapes, or to backup several databases and form subpools for these in a tape robot.

## Veritas Quick I/O Feature

BRBACKUP and [BRRESTORE](#) now support the Veritas Quick I/O feature.

### Note

Previously BRBACKUP and BRRESTORE considered the Veritas Quick I/O files to be raw disks. They were saved with `dd` command (if no `BACKINT` was used) with block size equal to Oracle block size. This could cause performance disadvantages. In addition, the files had to be created manually before BRRESTORE was started.



## Features

- The Veritas Quick I/O files are automatically recognized by BRBACKUP and marked in the detail log (see *Q/O* flag in messages BR0118I, BR0119I and BR0120I). This is the basis for processing these files through BRRESTORE.
- In native BRBACKUP backups (that is, without BACKINT), the files are copied to local and remote tape units using `dd`, to local disks using `cp` or `dd` or using `rcp` or `sapftp` onto remote disks, using `backup_dev_type = stage`. Unlike the raw disks, the block size for the `dd` command can be set to any value using parameters `dd_flags` and `dd_in_flags`.
- In BRBACKUP with BACKINT, the "cdev" names are transferred to BACKINT with complete path, as in the following example.

### Example

```
/oracle/DCA/sapdata13/ddici_1/.ddici.data1::cdev:vxfs:
```

- Quick I/O files can be created with relative links, as in the following example:

### Example

```
/oracle/DCA/sapdata13/ddici_1/ddici.data1 ->  
.ddici.data1::cdev:vxfs:
```

We recommend you to usually adhere to the SAP standard naming convention for normal files, as in the examples above.

- BRRESTORE creates the Quick I/O files automatically before data is actually restored. Therefore, you can use any BACKINT program (not only NetBackup by Veritas) for the database backup.

## External Backup Programs

The SAP tools [BRBACKUP](#), [BRARCHIVE](#), and [BRRESTORE](#) provide an interface called BACKINT that can be used to access external backup programs. You can only use this interface if the BACKINT interface program is supported by the supplier of the external backup program.

As of SAP NetWeaver 7.1, you can use the BACKINT interface to:

- Better support snapshot and cloning technology, which is increasingly becoming the industry standard
- Fully implement split-mirror disk technologies in BACKINT, instead of using the SPLITINT interface
- Enable use of BRRECOVER procedures for backup based on snapshot or cloning technologies
- Run backups at the level of disk volume (volume backups) in addition to the currently available backups at file level (file backups)

## Prerequisites

### Disk-Volume Backup

By “disk volume” we normally mean a “logical volume” (file system, mount point, or drive). This is the smallest unit that can be backed up with a snapshot or clone. Depending on the hardware and implementation, the smallest volume unit might also be bigger – for example, a “logical volume group”. The BR\*Tools parameter [util\\_vol\\_unit](#) defines which volume unit is used in a specific configuration.

In the event of a restore, the database files are normally reset to the state of a selected snapshot or clone, which is known as “snap-back” or “clone-back”. However, it is often necessary to access such backups without overwriting the original files – for example, for verification purposes or to copy back into other directories. This type of additional access is controlled by the BR\*Tools parameter [util\\_vol\\_access](#). This access normally uses a separate mount action. However, depending on the implementation, the copying of individual files might be possible.

#### Recommendation

To make the handling of disk volume backups easier, we recommend only complete backups. However, partial backups are still permitted. For partial backups BRBACKUP extends the list of database files to be backed up so that it always includes all database files on the selected volume units.

To create backups at *disk-volume level*, you need to meet the following prerequisites:

- All database files are in sub-directories of `sapdata` directories or in `sapraw` for raw disks.
- All online redo log files are in `origlog` or `mirrlog` directories, or in `sapraw` for raw disks.
- All control files are in `sapdata`, `origlog`, or `mirrlog` directories, or in `sapraw` for raw disks.
- Normally `sapdata`, `origlog`, or `mirrlog` are mount points (UNIX) or are on separate drives (Windows).
- There are no other non-database files in these directories.

The use of the new functionality is only allowed with the corresponding certified backup tools.

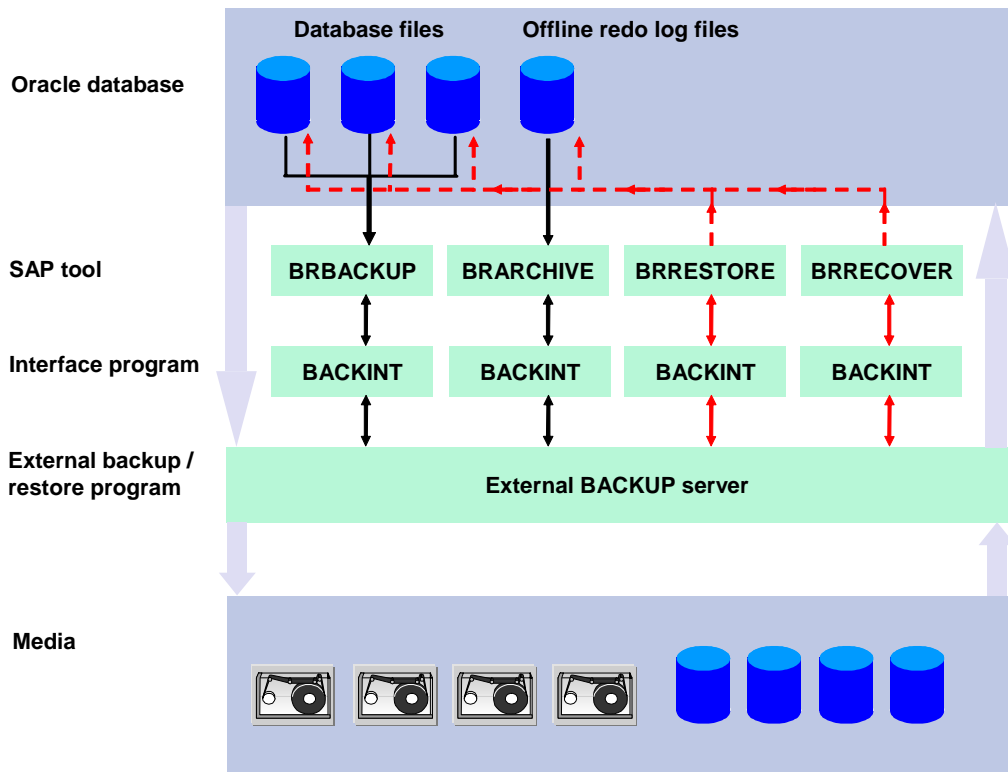
## Features

By using external backup programs you have the following advantages:

- You can use new, manufacturer-specific backup media. For example, the SAP tools do not support direct backup to or restore from optical storage media. However, you can use such media with an external backup program using the BACKINT interface.
- You can set up a consistent backup procedure for file systems and databases.
- Many backup programs are not hardware-specific and can be used in a network, for example, for [backup with an automatic tape changer](#), like tape robots.

## Activities

If you use this interface, tasks are distributed as follows:



The graphic shows that:

- BRBACKUP or BRARCHIVE does the database handling for backups.
- The external backup program manages the backup media.
- BRBACKUP or BRARCHIVE uses BACKINT to pass a backup request to the external backup program. This request contains a list of the files for backup. BRRESTORE and BRRECOVER also use BACKINT to trigger the external program to restore the requested files.

Any parameters that are passed are contained in a parameter file that you define. The external backup program performs all the backup operations.

- BRBACKUP, BRARCHIVE, BRRESTORE, or BRRECOVER evaluates the confirmation messages of the external backup program.

**Note**

Use the following settings in the [initialization profile init<DBSID>.sap](#):

- `backup_dev_type = util_file|util_file_online|util_vol|util_vol_online`
- Parameter file, for example:  
`util_par_file = initC11.utl`

For more information, see [backup\\_dev\\_type](#) and [util\\_par\\_file](#).



## Fixing an Online Backup Crash

This section tells you how to fix a crash during an [online backup with BRBACKUP](#).

During an online backup, BRBACKUP sets tablespaces into backup mode to enable proper recovery if a data file from an online backup needs to be restored. After an online backup BRBACKUP turns off backup mode for all tablespaces.

Therefore, if BRBACKUP crashes, one or more tablespaces remain in backup mode. Also, no further backups can start because BRBACKUP sets a lock at the start of the failed backup.

An online backup crash means that:

### Prerequisites

Distinguish between the following cases, which each require a different procedure:

- BRBACKUP failure

The database and the server remain running but BRBACKUP fails. The following is true:

- The database cannot be shut down to a consistent state. Only shutdown abort is possible, although we strongly recommend you not to perform a shutdown abort.

#### Caution

Do not shut down the database with the command option `shutdown abort`.

The result of this is that the next database startup does not work and you need to perform additional actions such as a media recovery.

- Any other backup that you start with BRBACKUP can fail.

You can check the backup mode of a tablespace with either of the following commands:

- `brconnect -f check:`

BRCONNECT displays the following warning if a tablespace is in backup mode:

```
WARNING, type: TABLESPACE_IN_BACKUP, object: PSAPT00EX.
```

- `brspace -c force -f dbshow -t tslist:`

Check the column *Back* to see if any tablespace is in backup mode.

- Database server or host failure

The database server or the host running the database fails during a backup with BRBACKUP.

A normal database startup is not possible and you see the following error message when starting the database:

...

```

SQL> ORACLE instance started.

Total System Global Area 48307140 bytes
Fixed Size 453572 bytes
Variable Size 41943040 bytes
Database Buffers 5734400 bytes
Redo Buffers 176128 bytes

Database mounted.

ORA-01113: file 5 needs media recovery

ORA-01110: data file 5:
'D:\ORACLE\T00\SAPDATA3\T00_1\T00.DATA1'

...

```

You can solve this in either of the following ways:

- With SQLPLUS to start up the database into mount state from where you can use the `ALTER DATABASE END BACKUP` command. However, you must only use this method if no data file has been restored.

This solution is recommended.

- With recovery using BRRECOVER. However, this can take a long time, depending on the database activity and the size of the tablespace in backup mode.

This solution is not recommended.

## Procedure

### BRBACKUP failure

1. Check if BRBACKUP is running using the `ps` command on UNIX or the task manager on Windows.



Caution

If BRBACKUP is running, do not continue with the procedure.

2. Delete the lock file:
  - UNIX: `$SAPDATA_HOME/sapbackup/.lock.brb`
  - Windows: `$SAPBACKUP%\lock.brb`
3. Start BRTOOLS or BRGUI and choose **Space management** **Alter tablespace** **Reset backup status**.

For more information, see [Altering a Tablespace with BR\\*Tools](#).

4. Choose *continue* twice to get a list of all tablespaces in backup mode.
5. Select all tablespaces and choose *Continue* to turn the backup mode off.

6. Perform a [backup with BRBACKUP](#) to check if everything is OK.

## Database Host or Server Failure

You can solve this problem in the following ways:

- To avoid a long recovery, use SQLPLUS:

1. Start SQLPLUS:

```
sqlplus "/as sysdba"
```

2. Start up the database to mount state:

```
SQL> startup mount
```

3. Execute the command:

```
SQL> alter database end backup;
```

### Caution

Only use `alter database end backup` if no data file has been restored.

Never use this command after having restored a data file because it might leave your database in an inconsistent and unusable state.

4. Open the database for normal user access:

```
SQL> alter database open;
```

- For more information, see [Database Recovery with SQLPLUS](#).
- If no other problem has occurred and the above SQLPLUS method is not possible, perform a step-by recovery with the following BRRECOVER command:

```
brrecover -t complete
```

Continue with the step-by-step recovery as prompted by BRRECOVER.

For more information, see [Complete Database Recovery with BR\\*Tools](#).



## Abort of Archive, Backup, or Restore

You can abort runs of BRARCHIVE, BRARCHIVE, or BRRESTORE. For more information on the command syntax to do this, see [brarchive -g|-abort](#), [brbackup -g|-abort](#), or [brrestore -g|-abort](#).

### Soft Abort with Option -g stop

The system waits until all active copy processes (for example, `cpio`, `dd`, `cp`, `copy`) have finished before stopping the archive, backup, or restore and performing a clean-up. This leaves the database in a clean state. Any files that are being processed when the abort command is issued are completely copied before the abort takes effect. Therefore, the abort takes longer, up to several minutes.

This kind of abort is especially useful for native backups or restores, such as with `cpio`, `dd`, `cp`, `copy`, `compress`, `uncompress`, `mkszip`, and the SAP SBT (RMAN) backup library.

If you are using third party backup tools such as BACKINT (`backup_dev_type = util_file|util_file_online`) or third party SBT backup libraries (`backup_dev_type = rman_util|rman_disk|rman_stage`), a soft abort does not usually work. This is because it is not possible to prematurely terminate an active BACKINT or backup-library call. Since the backup or restore is usually processed in a single call, it is usually almost finished (apart from the writing of profiles and logs) before an abort can take effect.

### Normal Abort with Option -g term

This sends a terminate signal to the running processes of BRARCHIVE, BRBACKUP, or BRRESTORE, which causes the tool to immediately try to stop active copy processes or BACKINT or RMAN calls. However, this is not always possible, especially if the backup or restore is not running in a separate process group, because of the danger of also aborting other processes.

On Windows, if the backup was started from the DBA Planning Calendar, such an abort might also affect SAP processes. On UNIX, if the backup was started with a CRON job, a `Not owner` error might occur. For more information about a possible solution to this problem, see SAP Note [1129197](#). Otherwise you must manually abort the processes using the `kill` command on UNIX or the Task Manager on Windows.

BRARCHIVE, BRBACKUP, or BRRESTORE normally clean up, so that the database is left in a clean state after the abort.

### Hard Abort with Option -g term

This sends a kill signal to the running processes of BRARCHIVE, BRBACKUP, or BRRESTORE, which causes an immediate abort without clean-up. Active copy processes or BACKINT or RMAN calls continue running. If required, you must manually stop these with the `kill` command on UNIX or the Task Manager on Windows. The database remains in the state it was in at the time of the abort. To restart the database after an aborted offline backup, you can use the following BRSPACE command:

```
brspace -c force -f dbstart -s open -f
```

To reset the backup status of the tablespaces after an aborted online backup, you can use the following BRSPACE command:

```
brspace -c force -f tsalter -a endback -t all_ts
```

 Caution

Only use the hard abort if absolutely necessary, such as when a normal abort has failed.

## Restore and Recovery

This section helps you to develop an approach to restore and recover of your Oracle database in the event of failure.

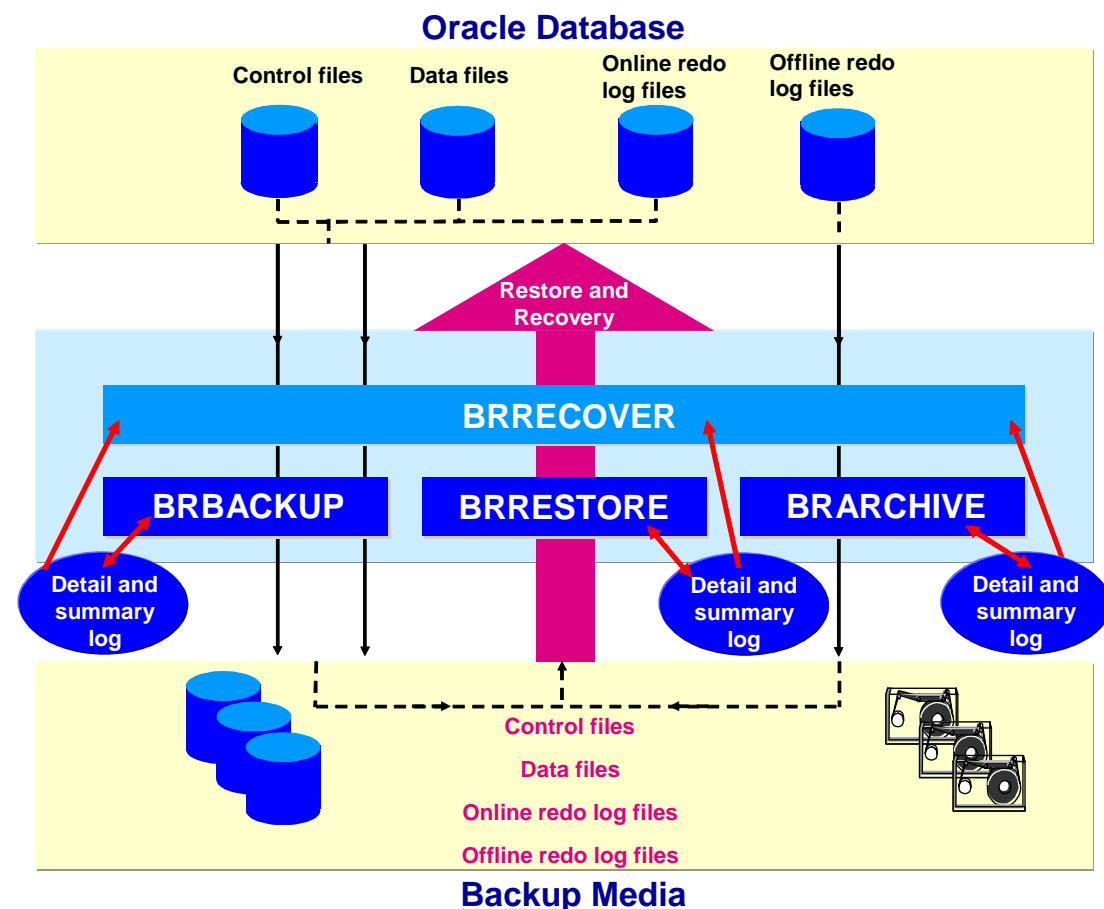
 Caution

Always analyze the problem carefully before attempting to recover your database. If in doubt, seek support from SAP. The business costs of an incorrect or delayed recovery can easily outweigh consultancy fees.

**↑ Recommendation**

We strongly recommend you to practice restore and recovery on a test system as similar as possible to your production system. Repeat this regularly, especially after you have modified the production system.

The following graphic shows an overview of the restore and recovery process:



It is possible to use BRRECOVER and BRRESTORE for homogeneous database copies. For more information, see *Enhanced Support for Homogeneous Database Copies* in [SAP Note 1003028](#).

## Integration

You use the following tools for database restore and recovery:

Tool	Use
<a href="#">Restore and Recovery with BR*Tools</a>	Restore and recovery with the BR*Tools user interface



Tool	Use
<a href="#">BRRESTORE</a>	Restore of database and offline redo log files
<a href="#">BRRECOVER</a>	Recovery of database and offline redo log files
<a href="#">Database Recovery with SQLPLUS</a>	Database recovery when BR*Tools is not sufficient
<a href="#">Oracle Recovery Manager</a> – integrated with BRBACKUP, BRARCHIVE, and BRRESTORE	Restore and recovery

## Prerequisites

Before you start to restore and recover your database, try to locate:

- The backups of the missing or faulty data files, or the required offline redo log files made with [BRBACKUP](#) or [BRARCHIVE](#).
- The [BRBACKUP logs](#) and [BRARCHIVE logs](#), which are very important for the restore because they contain all the information about the backups, such as directories, volumes, and timestamps. BRRECOVER uses the logs to find the backups of the data files and the offline redo log files.

### Caution

If you lose the entire database system, including such items as the BRBACKUP and BRARCHIVE logs and profiles, you need to perform [Disaster Recovery](#).

### Note

BRRECOVER recovery is not based on BRBACKUP or BRARCHIVE logs in database tables, because these tables are not available when you recover the database.

You might also need to meet other prerequisites, depending on the context of the recovery.

## Features

- Restore
 

BRRESTORE restores data files that have been damaged or are missing, using backups of the database files.
- Recovery
 

BRRECOVER recovers transactions lost since the database backup, using backups of the offline redo log files to roll forward the lost transactions.

BRRECOVER supports database recovery after:

  - Media errors, such as a disk crash or accidental deletion of database files

- User errors such as software problems or when a user accidentally drops a table
- Disaster, when the entire database is lost, including backup profiles and logs

After recovery, BRRECOVER automatically rebuilds `NOLOGGING` indexes that were created during or after the backups used. For more information, see [SAP Note 849485](#).

## Activities

You can perform restore, and recovery tasks from either of the following:

- The menus in BR\*Tools

BR\*Tools calls the tools BRBACKUP, BRARCHIVE, BRRESTORE, or BRRECOVER as necessary to complete the task you have chosen.

- The command line

In this way, you can use the tools BRBACKUP, BRARCHIVE, BRRESTORE, or BRRECOVER, but this requires expert knowledge.



### Recommendation

We recommend you to use the DBA Planning Calendar for routine backup because this enables you to automatically schedule the backup.

We recommend you to use the BR\*Tools menus for restore and recovery because BR\*Tools guides you through the necessary steps.

Proceed as follows to restore and recover your database:

1. To analyze the problem, check the:
  - Database alert log
  - Trace files belonging to the background processes in the directory `$ORACLE_HOME/saptrace/background`.

For more information, see [Error Analysis](#).
2. Ask yourself the following questions:
  - What is the status of the database? Is it available or not?
  - What kind of error has occurred? A media or a user error?
  - Which files are corrupt?
  - What type of file is affected? Data files, control files, online redo log files?
  - If a media error has occurred, is software or hardware mirroring available?
  - Do you have a [standby database](#)?
3. If a user error has occurred, the database is still available, and you have enough time, perform a [complete, offline backup](#) before starting to restore and recover the database.

4. If a media error has occurred, you must replace the affected equipment and recreate the file system as it existed before the error.
5. Decide what kind of restore and recovery you want to perform:

Scenario	Aim	Solution
Media error such as a disk crash	You want to recover to the time of failure	<a href="#">Complete Database Recovery</a>
User or software error	You want to recover to a selected point in time	<a href="#">Database Point-In-Time Recovery</a>
User or software error within one component of a multi-component database	You want to recover the affected component to a selected point in time.	<a href="#">Tablespace Point-in-Time Recovery</a>
Either of the following applies: <ul style="list-style-type: none"> <li>○ Error in which all copies of redo log files are lost, but the database file backup is available</li> <li>○ You performed a complete or incremental offline or consistent online backup immediately before the error, such as during a software upgrade.</li> </ul>	You want to reset the database to the state it was in at the most recent offline or consistent online backup.	<a href="#">Whole Database Reset</a>
Loss of entire database system, including backup profiles and logs	You want to recover as much as possible	<a href="#">Disaster Recovery</a>

 **Caution**

Only if you are very experienced, you might want to consider the following in an exceptional situation:

- [Restore of individual backup files](#)
- [Restore and application of offline redo log files](#)

Be sure to consult SAP Support first if you are unsure about using these functions.

## Complete Database Recovery

This section tells you about restoring damaged or lost data files after a failure in your Oracle database, and then recovering the database to the time of failure. You normally do this after a media error, such as a disk crash. With this function you can:

1. Restore lost data files by using appropriate backups
2. Recover the restored data file to the time of failure using the redo log files

This function consists of a number of phases that are executed either manually or automatically by BRRECOVER in a predetermined sequence.

 Note

This section discusses how to approach complete database recovery.

For more information on how to perform a complete database recovery, see [Complete Database Recovery with BR\\*Tools](#).

## Integration

This function performs a *complete* recovery to the time of failure. If you want to perform a *point-in-time* recovery – that is, a recovery to some time other than the time of failure – see:

- [Database Point-In-Time Recovery](#)
- [Tablespace Point-In-Time Recovery](#)

## Prerequisites

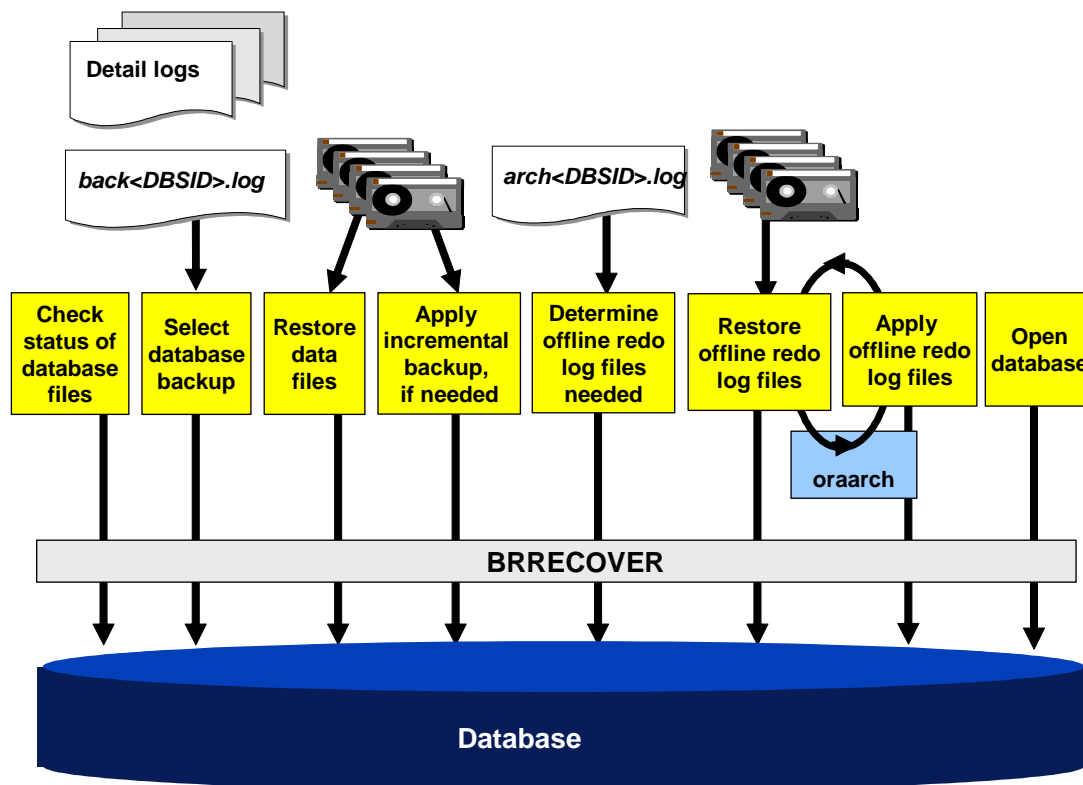
To finish complete recovery, you must have the files shown in the following table, either as originals or backups. The table shows what you must do if the files are *not* available:

Missing File Type	Solution if Unavailable
init<DBSID>.dba init<DBSID>.ora spfile<DBSID>.ora	Restore from backup as described in <a href="#">Disaster Recovery</a> .
<a href="#">BRBACKUP Logs</a> and <a href="#">BRARCHIVE Logs</a>	Restore from backup as described in <a href="#">Disaster Recovery</a> .
Control files	Copy one of the control file mirrors if available. Otherwise, use one of the following: <ul style="list-style-type: none"> <li>• <a href="#">Database Point-In-Time Recovery</a></li> <li>• <a href="#">Whole Database Reset</a></li> </ul>
Online redo log files	Copy one of the redo log file mirrors if available.  If the non-current redo log group is involved, use ALTER DATABASE to drop and recreate the group.  If the current group is involved, use one of the following: <ul style="list-style-type: none"> <li>• <a href="#">Database Point-In-Time Recovery</a></li> <li>• <a href="#">Whole Database Reset</a></li> </ul>
Database files	These files are automatically restored and recovered during complete database recovery.
Offline redo log files	Perform <a href="#">database point-in-time recovery</a> to the most recent available redo log file. If none is available, perform a <a href="#">whole</a>

Missing File Type	Solution if Unavailable
	<a href="#">database reset</a> from an offline backup or an online consistent backup.

## Features

The following graphic shows how a complete recovery works:



## Activities

### 1. Check the Status of Database Files phase

BRRECOVER checks the status of all files in the database (that is, the control files, online redo log files, and data files). BRRECOVER does the following:

1. To update the `v$` views, BRRECOVER stops the database instance if started and sets the database to `mount` status. BRRECOVER logs any recorded errors in data files to the detail log created in the `sapbackup` directory with the `crv` suffix (for complete recovery.)
2. BRRECOVER refers to entries in Oracle's `v$` views, such as `v$datafile`, `v$recover_file` to determine the status of database files.
3. BRRECOVER does not create missing `sapdata` directories automatically, so they must exist beforehand. However, BRRESTORE automatically creates missing subdirectories under `sapdata` directories during the Restore Data Files phase.

### 2. Select Database Backup phase

BRRECOVER determines the eligible backups using the entries in the BRBACKUP summary log file `back<DBSID>.log` (return code 0 or 1). The associated detail logs show whether the required data files were in the backup. The data files can be compiled from various backups. To minimize the subsequent recovery time, BRRECOVER always suggests the most recent backup.

BRRECOVER also roughly checks the availability of offline redo log files.

You can also select an incremental backup to be restored before applying offline redo log files. In this case, BRRECOVER automatically selects the corresponding full backup to restore missing files.

### 3. Restore Data Files phase

BRRECOVER calls BRRESTORE to restore the data files to their original location.

### 4. Apply Incremental Backup phase

If you selected an incremental backup during the Select Database Backups phase, BRRECOVER calls BRRESTORE to restore and apply the selected incremental backup.

### 5. Determine Offline Redo Log Files Needed phase

BRRECOVER determines the offline redo log files required for a complete recovery. The BRARCHIVE summary log file `arch<DBSID>.log` lists the backups of the offline redo log files. BRRECOVER takes existing online redo log files and offline redo log files in `saparch` or `oraarch` into consideration.

### 6. Restore Offline Redo Log Files phase

BRRECOVER calls BRRESTORE to restore the offline redo log files that have been found back to the `saparch` or `oraarch` directory.

### 7. Apply Offline Redo Log Files phase

BRRECOVER calls SQLPLUS to apply offline redo log files to the database.

Offline redo log files are applied to the database in groups of at most 100 files. If you have more than 100 files to apply, the restore and apply phases are repeated as necessary.



#### Note

The restore and apply phases can be executed in parallel to minimize total recovery time.

### 8. Open Database phase

During this phase BRRECOVER:

1. Opens the database
2. Checks the status of database files and tablespaces



## Database Point-In-Time Recovery

You can use this function to fully restore and then recover your Oracle database to a specified point in time (PIT). You normally use this function when there has been a logical error – that is, a user or software error – and you want to recover the database to the point immediately before the error. In this way, you minimize lost data.

This section discusses how to approach database point-in-time recovery.

For more information on how to perform a database point-in-time recovery, see [Database Point-In-Time Recovery with BR\\*Tools](#).

You can also now use point-in-time recovery for a [standby database](#).

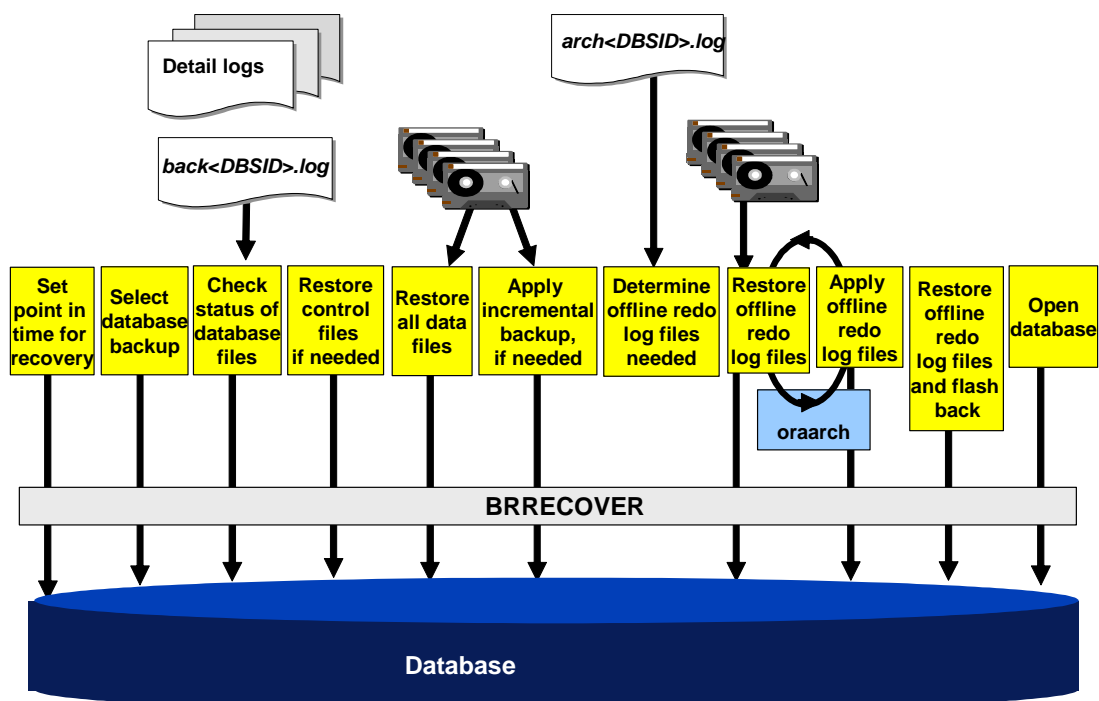
## Prerequisites

- We recommend you to:
  - Perform a [full offline or online backup](#). If the database is running, use SAP tools, otherwise use operating system tools.
  - Back up all offline redo log files using BRARCHIVE. For more information, see [-a|-archive](#).
- You must have the following data available if you are not using [flashback database](#):
  - The [BRBACKUP logs](#) and the [BRARCHIVE Logs](#)
  - The data file backups and an incremental backup if required
  - All offline redo log files between the data backup and the chosen PIT

Any types of database files – data, online redo, control – might be unavailable on disk.

## Features

The following graphic shows how database point-in-time recovery works:



## Activities

### 1. Set Point In Time for Recovery phase

You enter the recovery end-point in BRRECOVER by choosing one of the following:

- Point in time
- Redo log sequence number
- System change number

### 2. Select Database Backup phase

BRRECOVER determines the eligible backups using the entries in the BRBACKUP summary log file `back<DBSID>.log` (return code 0 or 1). The associated detail logs show whether the required data files were in the backup. The data files can be compiled from various backups. To minimize the subsequent recovery time, BRRECOVER always suggests the most recent backup.

BRRECOVER also roughly checks the availability of offline redo log files.

You can also select an incremental backup to be restored before applying offline redo log files. In this case, BRRECOVER automatically selects the corresponding full backup to restore all data files.

You can also select flashback database if activated and if the recovery endpoint is covered by the flashback data.

### 3. Check Status of Database Files phase

BRRECOVER checks the status of database files to determine which will be overwritten.

### 4. Restore Control Files phase

BRRECOVER calls BRRESTORE to restore control files if needed, that is, if they are unavailable or unsuitable for the selected backups.

### 5. Restore Data Files phase

BRRECOVER calls BRRESTORE to restore all the data files to their original location.

### 6. Apply Incremental Backup phase

If you selected an incremental backup during the Select Database Backups phase, BRRECOVER calls BRRESTORE to restore and apply the selected incremental backup.

### 7. Determine Offline Redo Log Files Needed phase

BRRECOVER determines the offline redo log files required for a recovery. The BRARCHIVE summary log file `arch<DBSID>.log` lists the backups of the offline redo log files. BRRECOVER takes existing online redo log files and offline redo log files in `saparch` or `oraarch` into consideration.

### 8. Restore Offline Redo Log Files phase

BRRECOVER calls BRRESTORE to restore the offline redo log files that have been found back to the `saparch` or `oraarch` directory.

### 9. Apply Offline Redo Log Files phase



BRRECOVER calls SQLPLUS to apply offline redo log files to the database.

Offline redo log files are applied to the database in groups of at most 100 files. If you have more than 100 files to apply, the restore and apply phases are repeated as necessary.

 Note

The restore and apply phases can be executed in parallel to minimize total recovery time.

## 10. Restore Offline Redo Log Files and Flashback phase

When database flashback is active, BRRECOVER checks whether the given recovery timestamp or system change number (SCN) is available in the flashback data. If so, you can use flashback to perform the database point-in-time recovery.

BRRECOVER calls BRRESTORE to restore the offline redo log files needed for the flashback to the `saparch` or `oraarch` directory (if required), then performs flashback database.

 Caution

Before you use database flashback in your production environment, make sure you have tested it thoroughly.

Database flashback does *not* replace regular backups.

For more information, see [Managing Flashback Database with BR\\*Tools](#).

## 11. Open Database phase

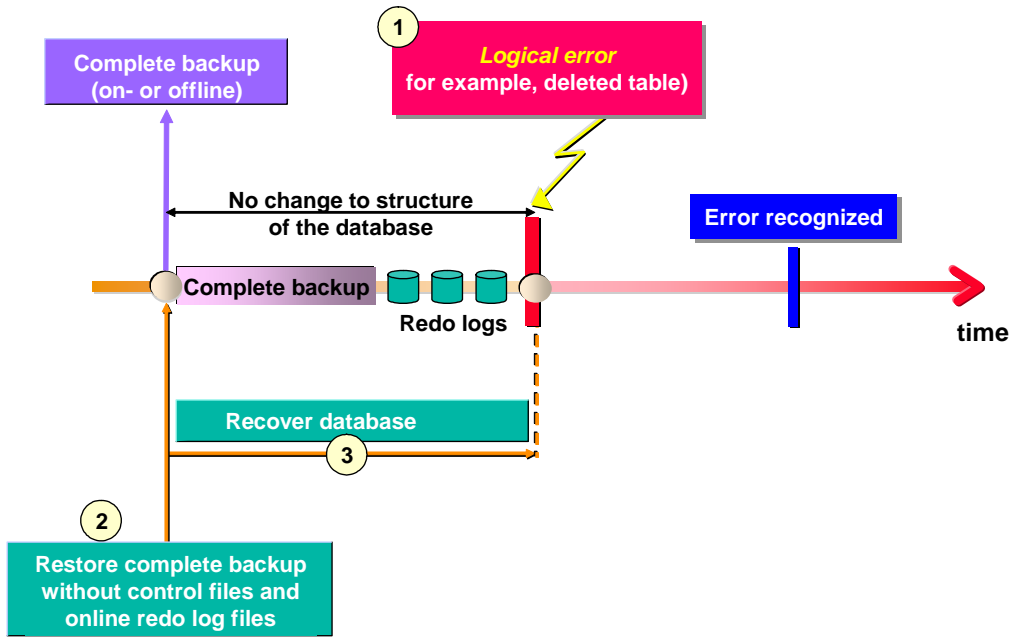
During this phase BRRECOVER:

1. Opens the database with the option `RESETLOGS`
2. Creates missing temporary files
3. Checks the status of database files and tablespaces
4. Deletes unnecessary files that are no longer used by the database

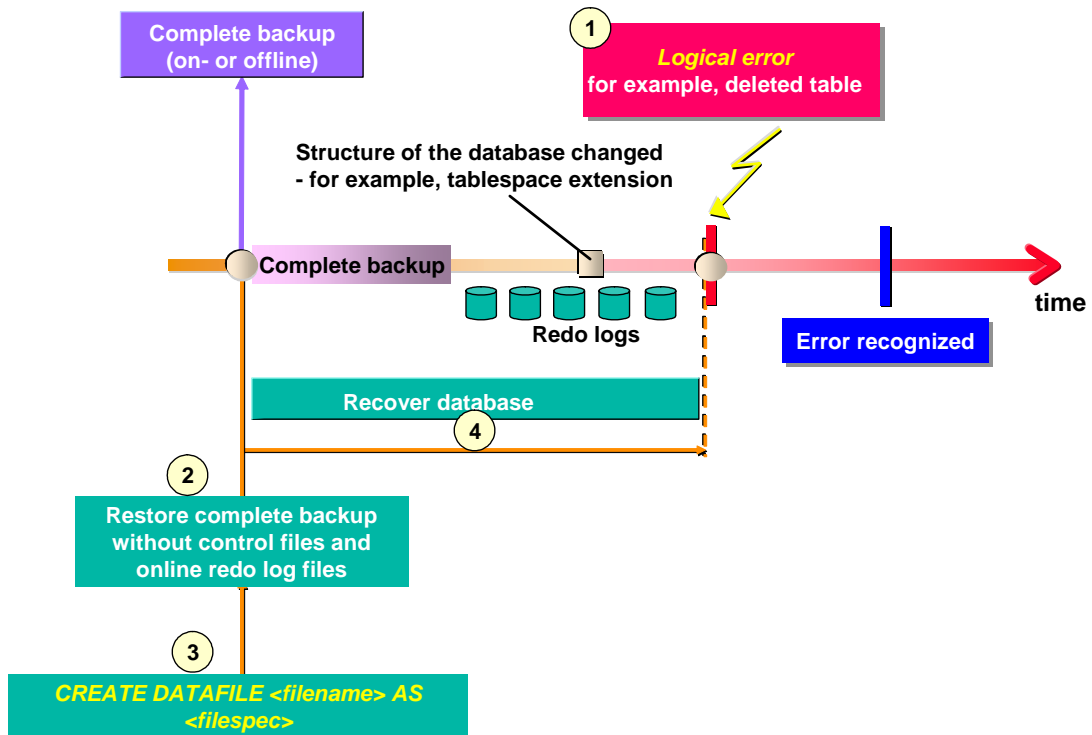
## Example

Here are two typical scenarios in which you can use database point-in-time recovery:

Logical Error



Logical Error with Preceding Structure Change



## Tablespace Point-in-Time Recovery

You can use this function to fully restore and then recover a group of Oracle tablespaces to a specified point in time (PIT). You normally use this function when there has been a logical

error - that is, a user or software error - and you want to recover the tablespace to the point immediately before the error. In this way, you minimize lost data.

Tablespace PIT recovery is especially useful for Multiple Components in One Database (MCOB). It lets you restore the tablespaces for a single component - for example, if a component upgrade has failed - without affecting the other components in the same database.

This section discusses how to approach tablespace point-in-time recovery.

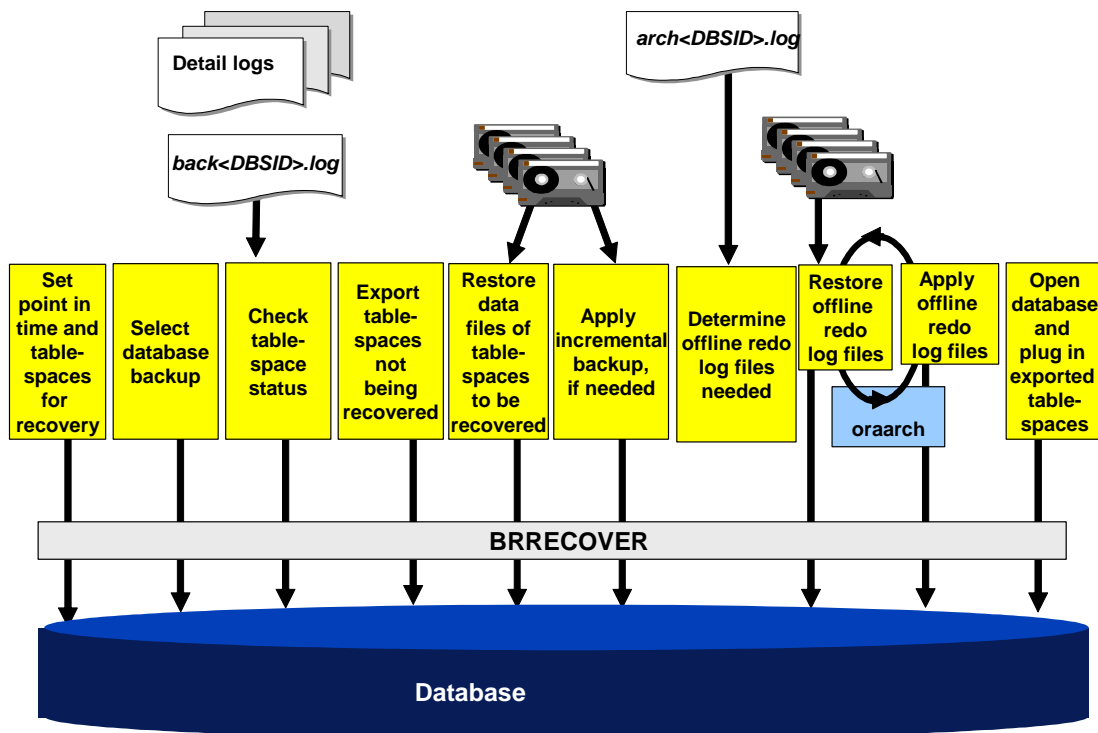
For more information on how to perform a tablespace point-in-time recovery, see [Tablespace Point-In-Time Recovery with BR\\*Tools](#).

## Prerequisites

- We recommend you to:
  - Perform a [full offline or online backup](#). If the database is running, use SAP tools, otherwise use operating system tools.
  - Back up all offline redo log files using BRARCHIVE. For more information, see [-a|-archive](#).
- You must have the following data available:
  - The [BRBACKUP logs](#) and the [BRARCHIVE logs](#)
  - The data file backups and an incremental backup if required
  - All offline redo log files between the data backup and the chosen PIT
- You must be able to open the database before starting the recovery procedure.

## Features

The following graphic shows how tablespace point-in-time recovery works:



## Activities

### 1. Set Point In Time and Tablespaces for Recovery phase

You enter the recovery end-point in BRRECOVER by choosing one of the following:

- Point in time
- Redo log sequence number
- System change number

You specify the tablespaces that you want to recover or a SAP owner for a multi-component database. BRRECOVER automatically finds tablespaces containing segments belonging to this owner.

### 2. Select Database Backup phase

BRRECOVER determines the eligible backups using the entries in the BRBACKUP summary log file `back<DBSID>.log` (return code 0 or 1). The associated detail logs show whether the required data files were in the backup. The data files can be compiled from various backups. To minimize the subsequent recovery time, BRRECOVER always suggests the most recent backup.

BRRECOVER also roughly checks the availability of offline redo log files.

You can also select an incremental backup to be restored before applying offline redo log files. In this case, BRRECOVER automatically selects the corresponding full backup to restore all data files.

### 3. Check Tablespace Status phase

BRRECOVER checks if the:

- Tablespace group to be recovered is self-contained

- Tablespace group to be exported is self-contained

This means that database objects within this group must not have references to objects outside the group.

#### 4. Export Tablespace not Being Recovered phase

BRRECOVER sets the tablespaces not being recovered to `READ ONLY` and exports the tablespaces' meta data using the Oracle EXPDP (Data Pump) tool.

#### 5. Restore Data Files of Tablespaces to Be Recovered phase

BRRECOVER calls BRRESTORE to restore data files of the tablespaces to be recovered, including the `SYSTEM` and rollback tablespaces, placing them in their original location.

#### 6. Apply Incremental Backup phase

Before applying an incremental backup, BRRECOVER sets the data files of exported tablespaces to `OFFLINE`. Therefore, these tablespace are not recovered.

If you selected an incremental backup during the Select Database Backups phase, BRRECOVER calls BRRESTORE to restore and apply the selected incremental backup.

#### 7. Determine Offline Redo Log Files Needed phase

BRRECOVER determines the offline redo log files required for a recovery. The BRARCHIVE summary log file `arch<DBSID>.log` lists the backups of the offline redo log files. BRRECOVER takes existing online redo log files and offline redo log files in `saparch` or `oraarch` into consideration.

#### 8. Restore Offline Redo Log Files phase

BRRECOVER calls BRRESTORE to restore the offline redo log files that have been found back to the `saparch` or `oraarch` directory.

#### 9. Apply Offline Redo Log Files phase

Before applying the offline redo log files, BRRECOVER sets the data files of the exported tablespaces to `OFFLINE`, if not yet done.

BRRECOVER calls SQLPLUS to apply offline redo log files to the database.

Offline redo log files are applied to the database in groups of at most 100 files. If you have more than 100 files to apply, the restore and apply phases are repeated as necessary.



#### Note

The restore and apply phases can be executed in parallel to minimize total recovery time.

#### 10. Open Database and Plug In Exported Tablespaces phase

During this phase BRRECOVER:

1. Opens the database with the option `RESETLOGS`
2. Creates missing temporary files

3. Temporarily drops exported tablespaces from the database.
4. Imports meta data of exported tablespaces using the Oracle IMPDP (Data Pump) tool.
5. Sets imported tablespaces to `READ/WRITE`.
6. Checks the status of database files and tablespaces
7. Deletes unnecessary files that are no longer used by the database

## Whole Database Reset

This section tells you about resetting your Oracle database after a failure. You normally need to do this if either of the following applies:

- An error occurred in which all copies of the redo log files are lost, but the database file backup is available.
- You performed a complete offline backup or a consistent online backup immediately before the error, such as during a software upgrade.

With this function you can reset the database to a previous consistent state, at the time of either a *complete offline* or a *consistent online backup*. If you reset from an online backup, the consistent end point of the backup is used.

This section discusses how to approach whole database reset.

For more information on how to perform a whole database reset, see [Whole Database Reset with BR\\*Tools](#).

You can now also use whole database reset for a [standby database](#).

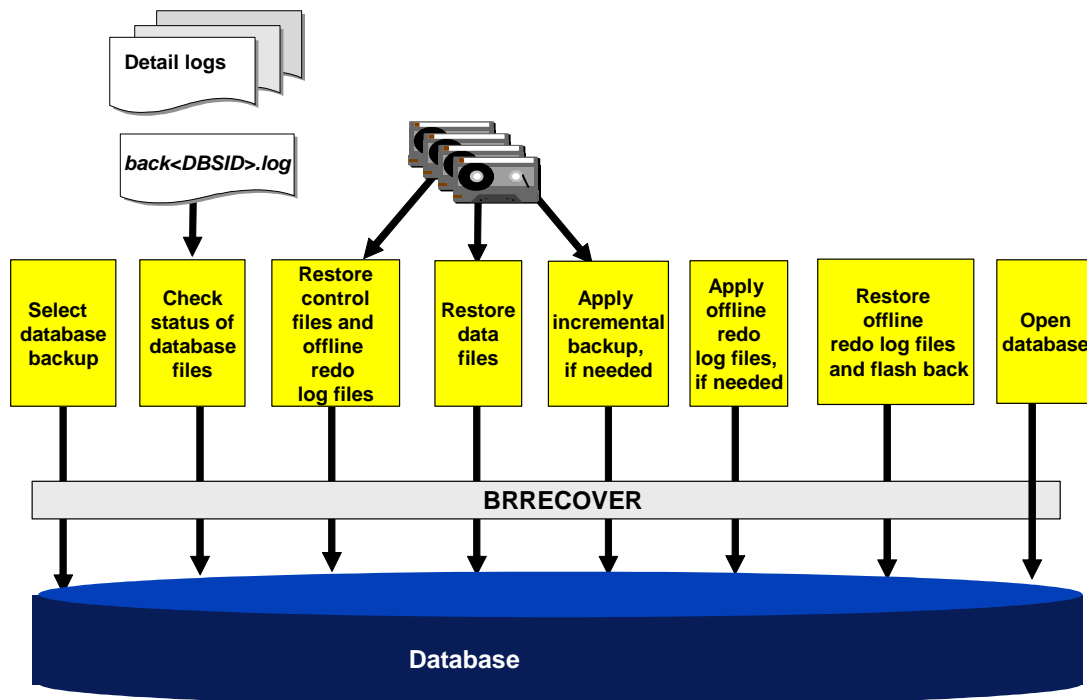
### Prerequisites

- We recommend you to perform a full [offline backup](#). If the database is running, use SAP tools, otherwise use operating system tools.
- You have the following data available if restore points are not used:
  - The [BRBACKUP logs](#)
  - The data file backups and, if selected, an incremental backup
  - If restoring from a consistent online backup, the accompanying redo log files that were saved in the same BRBACKUP run

Any types of database files - data, online redo log, control - might be unavailable on disk.

### Features

The following graphic shows how a whole database reset works:



## Activities

### 1. Select Database Backup phase

BRRECOVER determines the eligible backups using the entries in the BRBACKUP summary log file `back<DBSID>.log` (return code 0 or 1). The associated detail logs show whether the required data files were in the backup. The selected backup must be a complete offline or a consistent online backup. To minimize the subsequent recovery time, BRRECOVER always suggests the most recent backup.

You can also select an offline or consistent online incremental backup to be restored. In this case, BRRECOVER automatically selects the corresponding full backup to restore all data files.

You can also select a restore point if defined and still usable.

### 2. Check Status of Database Files phase

BRRECOVER checks the status of database files to determine which files will be overwritten.

### 3. Restore Control Files and Offline Redo Log Files phase

BRRECOVER calls BRRESTORE to restore control files. At the same time, offline redo log files are restored if a consistent online backup was selected.

### 4. Restore Data Files phase

BRRECOVER calls BRRESTORE to restore all the data files to their original location.

### 5. Apply Incremental Backup phase

If you selected an incremental backup during the Select Database Backups phase, BRRECOVER calls BRRESTORE to restore and apply the selected incremental backup.

## 6. Apply Offline Redo Log Files phase

If a consistent online backup was selected, BRRECOVER calls SQLPLUS to apply the restored offline redo log files to the database.

## 7. Restore Offline Redo Log Files and Flashback phase

When restore points are defined, you can use flashback for the database reset. It is always possible to use a guaranteed restore point. However, it is only possible to use a normal restore when the timestamp or system change number (SCN) is contained in the flashback data.

In this case, BRRECOVER calls BRRESTORE to restore offline redo log files needed for the flashback to the `saparch` or `oraarch` directory (if required), then resets the database to the chosen restore point using flashback.

### Caution

Before you use database flashback and restore points in your production environment, make sure you have tested it thoroughly.

Database flashback does *not* replace regular backups.

For more information, see [Managing Flashback Database with BR\\*Tools](#).

## 8. Open Database phase

During this phase BRRECOVER:

1. Opens the database
2. Creates missing temporary files
3. Checks the status of database files and tablespaces
4. Deletes unnecessary files which are no longer used by the database

## Restore of Individual Backup Files

You can restore individual backups files to your Oracle database. This is intended for *experts*, so make sure you fully understand how it works before using it.

This section discusses how to approach the restore of individual backup files.

For more information on how to perform a restore of individual backup files, see [Restore of Individual Backup Files with BR\\*Tools](#).

### Prerequisites

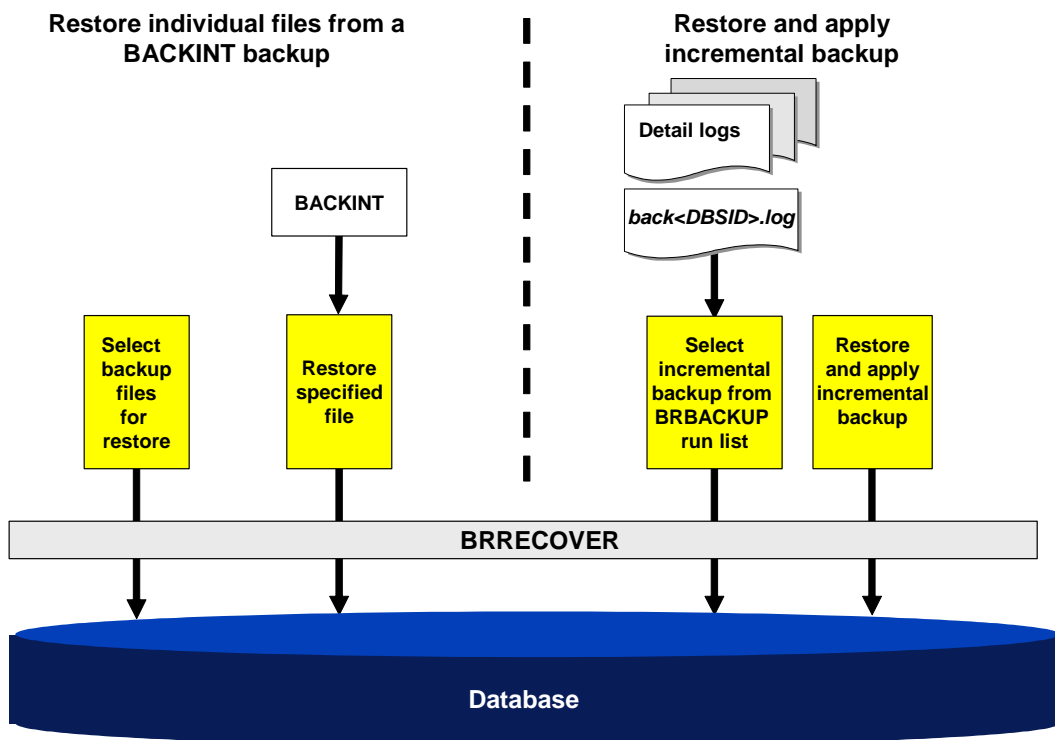
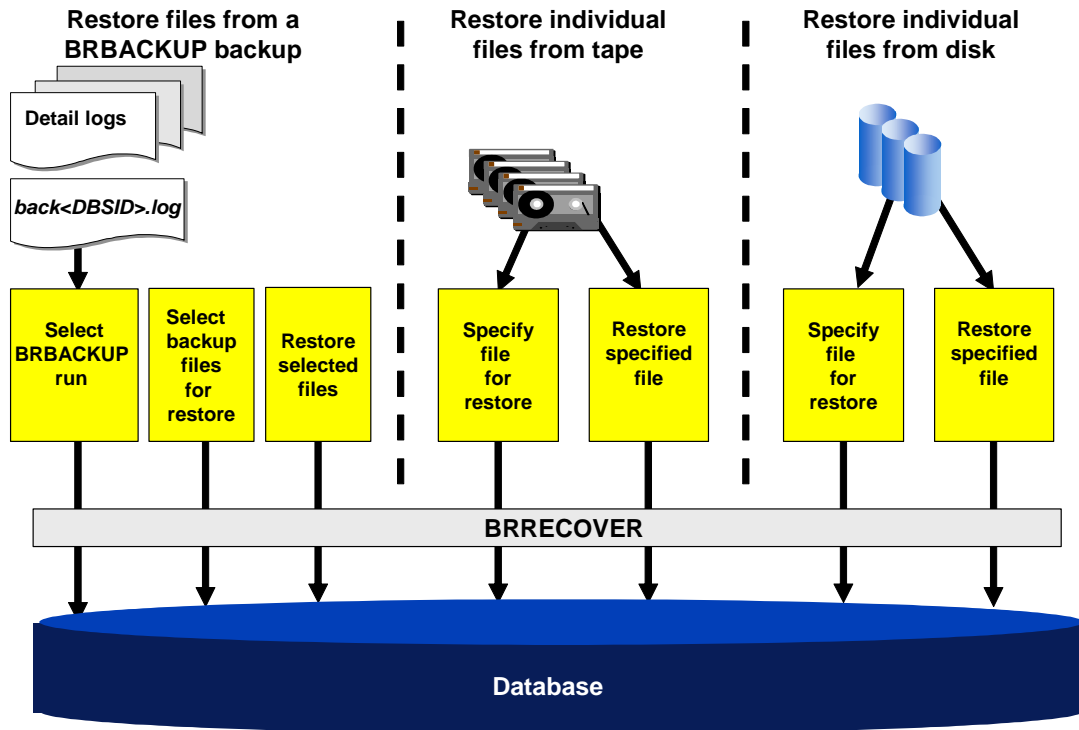
- We recommend you to perform a full [offline or online backup](#). If the database is running, use SAP tools, otherwise use operating system tools.
- You must have the following data available:
  - The [BRBACKUP logs](#) when restoring from a BRBACKUP backup or an incremental backup



- The data file backups and an incremental backup if required

## Features

The following graphic shows how restore of individual backup files works:



## Activities

You can perform the following functions to restore individual backup files:

- Restore files from a BRBACKUP backup
  1. Select BRBACKUP run phase

You select a BRBACKUP run from the backup list. You can select a BRBACKUP run that finished with an error.
  2. Select backup files for restore phase

You select files to restore from the list of files backed up in the selected run.
  3. Restore selected files phase

BRRECOVER calls BRRESTORE to restore the selected files.
- Restore individual files from tape
  1. Specify file for restore phase
    - You can specify a file on a local or a remote tape device.
    - You can specify not only database files but also non-database files and directories or offline redo log files.
    - You must specify the file position on tape and the restore destination.
  2. Restore the file phase

BRRECOVER calls BRRESTORE to restore the specified file.
- Restore individual files from disk
  1. Specify file for restore phase
    - You can specify a file on a local or a remote disk.
    - You can specify not only database files but also non-database files (but not directories) or offline redo log files.
    - You must specify the backup file name and the restore destination.
  2. Restore the file phase

BRRECOVER calls BRRESTORE to restore the specified file.
- Restore individual files from BACKINT backup
  1. Specify file for restore phase
    - You can specify not only database files but also non-database files and directories (if supported by BACKINT) or offline redo log files.
    - You must specify the backup file name, the BACKINT backup ID and optionally the restore destination.
  2. Restore the file phase

To restore the file, BRRECOVER calls BRRESTORE, which then calls BACKINT.

- Restore and apply incremental backup
  1. Select incremental backup phase

You select an incremental BRBACKUP run from the backup list. You can select a BRBACKUP run that finished with an error.
  2. Restore and apply incremental backup phase

BRRECOVER calls BRRESTORE to restore and apply the selected incremental backup.



## Restore and Application of Offline Redo Log Files

You can restore and apply offline redo log files - called archive log files in the BR\*Tools menus - to your Oracle database. This is intended for *experts*, so make sure you fully understand how it works before using it.

This section discusses how to approach restore and application of redo log files.

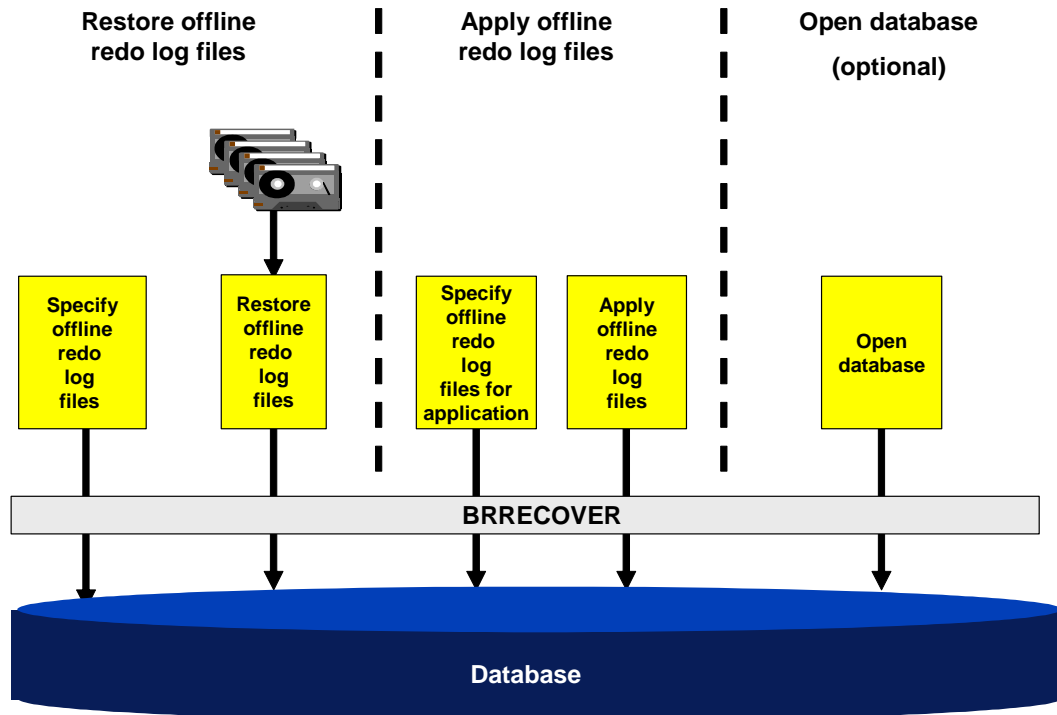
For more information on how to perform a restore and application of redo log files, see [Restore and Application of Offline Redo Log Files with BR\\*Tools](#).

### Prerequisites

- We recommend you to:
  - Perform a [full offline or online backup](#). If the database is running, use SAP tools, otherwise use operating system tools.
  - Back up all offline redo log files using BRARCHIVE. For more information, see [-a|-archive](#).
- You must have the [BRARCHIVE logs](#) available.

### Features

The following graphic shows how restore and application of individual backup files works:



## Activities

You can perform the following functions to restore and apply offline redo log files:

- Restore offline redo log files
  1. Specify offline redo log files for restore phase
 

You specify the offline redo log files that you want to restore.
  2. Restore offline redo log files phase
 

BRRECOVER calls BRRESTORE to restore the specified offline redo log files.
- Apply offline redo log files
  1. Specify offline redo log files for application phase
 

You specify which offline redo log files you want to apply.
  2. Apply offline redo log files phase
 

BRRECOVER calls SQLPLUS to apply the offline redo log files.
- Open database (optional)
 

If required, you can open the database after applying the offline redo log files.



If you lose your entire Oracle database system (possibly including hardware), and have not taken any special security precautions - such as setting up a [Standby Database](#) - then you have to recover the system as much as possible, step by step. This section describes how to begin reinstalling the system in the event of such a disaster, and how to keep data loss to a minimum by using [BRRECOVER](#).

 Note

This section discusses how to approach disaster recovery.

For more information about how to perform disaster recovery, see [Disaster Recovery with BR\\*Tools](#).

## Integration

You can only restore profiles and log files using this function. This is a preparation step for subsequent database recovery with one of the following:

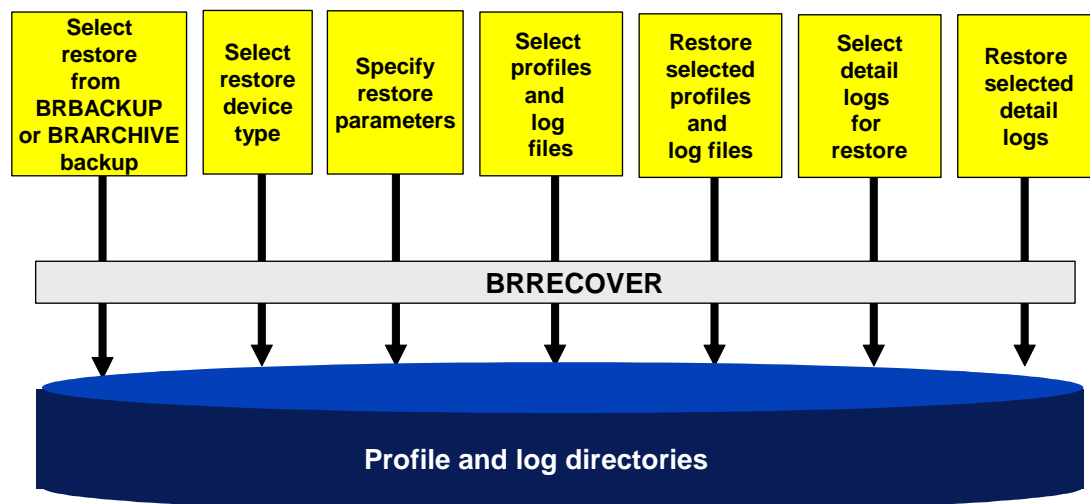
- [Database point-in-time recovery](#)
- [Whole database reset](#)

## Prerequisites

- SAP and Oracle software is correctly installed
- The file systems – that is, `sapdata<x>` directories – exist and are configured as before the disaster.

## Features

The following graphic shows how disaster recovery works:



## Activities

1. Select restore from BRBACKUP or BRARCHIVE backup.

You select the BRBACKUP or BRARCHIVE backup that contains the required profiles and logs.

2. Select restore device type phase

You select the device to be used from the following possibilities:

- Local tape
- Remote tape
- Local disk
- Remote disk
- Backup utility

3. Specify restore parameters phase

Depending on your choice of restore device type, you specify appropriate restore parameters.

4. Select profiles and log files phase

You select the profiles and log files that you want to restore from the following:

- [Backup profile](#)
- [Oracle profile](#)
- BACKINT / mount profile – for more information, see [util\\_par file](#)
- Detail log from one of the following:
  - [BRARCHIVE Detail Log](#)
  - [BRBACKUP Detail Log](#)
- Summary log from one of the following
  - [BRARCHIVE Summary Log](#)
  - [BRBACKUP Summary Log](#)
  - [BRSPACE Summary Log](#)
  - [BRSPACE Structure Change Log](#)
  - [BRSPACE Parameter Change Log](#)

5. Restore selected profiles and log files phase

BRRECOVER directly restores the selected profiles and log files without calling BRRESTORE.

6. Select detail logs for restore phase

You select the detail logs that you want to restore. You can only select logs on the selected restore device type, such as a disk or external backup tool (BACKINT). For tape device type, the details logs are determined by the mounted tape volume.

7. Restore selected detail logs phase

BRRECOVER directly restores the selected detail logs from the disk, tape, or external backup tool without calling BRRESTORE.

## Database System Check


This section helps you develop an approach to checking the database system. By running regular checks of the database system, you can identify potential problems and take the necessary action. You can schedule regular checks in the DBA Cockpit, using the DBA Planning Calendar. You can also run checks directly with BRCONNECT.

### Prerequisites

For more information on how BRCONNECT performs database system check, see [Database System Check with BRCONNECT](#). This is the same functionality as used in the DBA Planning Calendar.

### Process

1. You choose a method to run the database system check:
  - Regularly, preferably daily, by [setting it up in the DBA Planning Calendar](#) – recommended for normal use

 Recommendation

We recommend you to select an action pattern when you set up the DBA Planning Calendar. Action patterns let you schedule the database system check daily.

  - One-off by using the BRCONNECT command [-f check](#) – recommended for non-standard checks
2. You check the results using CCMS and take any necessary action:
  - You normally use the [alert monitor](#) to check the results of the system check.
  - For a detailed technical view, see [Displaying Alert Messages from Database System Check](#).
3. If required, you configure database system check using CCMS. This includes activating or deactivating conditions and changing the threshold and severity levels (that is, error, warning, or exception). For more information, see [Configuring Database System Check \(Oracle\)](#).

## Update Statistics

By running update statistics regularly, you make sure that the database statistics are up-to-date, so improving database performance. You can schedule the checks in the Computing Center Management System (CCMS) of the SAP System, using the DBA Planning Calendar.

The Oracle cost-based optimizer (CBO) uses the statistics to optimize access paths when retrieving data for queries. If the statistics are out-of-date, the CBO might generate inappropriate access paths (such as using the wrong index), resulting in poor performance.

From Release 4.0, the CBO is a standard part of the SAP System. If statistics are available for a table, the database system uses the cost-based optimizer. Otherwise, it uses the rule-

based optimizer. Since from Oracle 10g the rule-based optimizer is no longer supported, in this case auto-sampling is performed.

Update statistics supports partitioned tables. For more information, see *SAP Note* [424243](#).

## Prerequisites

For more information about how BRCONNECT performs update statistics, see [Update Statistics with BRCONNECT](#).

You can improve the performance of update statistics as follows:

- Parallel processing

You can implement this with BRCONNECT or DBMS\_STATS:

- BRCONNECT

You can implement BRCONNECT parallel processing as follows:

- Command call `brconnect -p <number of threads>`
- Parameter `stats_parallel_degree` in the `init<DBSID>.sap` file

### Example

Here are some examples:

```
brconnect -c -u / -f stats -p 4
```

```
stats_parallel_degree = 4
```

Each of these sets the number of threads – that is, the degree of parallelism – to 4. The default is 1, which means that update statistics runs in non-parallel, that is, serial mode. In either case, each table is processed by a single thread. For more information, see *SAP Note* [403713](#).

- DBMS\_STATS Package (standard from Oracle 10g)

There is a parallel processing option in this package that considerably reduces run times for very large tables (that is, several hundred GB). Each table can be processed by several threads.

For more information, see *SAP Note* [408532](#).

### Note

BRCONNECT calls DBMS\_STATS to create the statistics for individual partitions with the granularity `PARTITION`. This optimizes and significantly accelerates processing for large partitioned tables.

To increase the accuracy of statistics for partitioned tables, the sample size is doubled (`<standard sample size> * 2`) and DBMS\_STATS recreates the global statistics for the entire table if half the threshold is exceeded (`<change_threshold> / 2`).



You can use both the above types of parallel processing in the same BRCONNECT run.

- Oracle table monitoring

With this feature, the Oracle database system constantly updates information concerning record counts for database tables, entering the results in the `DBA_TAB_MODIFICATIONS` table. It only takes BRCONNECT a short time to read this results table and determine whether update statistics is required for a given database table.

The activation of the table monitoring attribute is now automated with the `monit` option as follows:

```
brconnect -u / -c -f stats -t all -f monit
```

The advantages of this are:

- For tables with the monitoring attribute activated by BRCONNECT, new statistics are collected at the same time. This gives Oracle a good basis for collecting information about changes to the table.
- Tables that are dynamically created by the SAP system, such as in SAP BW, are automatically (that is, without any manual intervention) processed by BRCONNECT on the next processing run.

For more information, see [brconnect -f stats](#) and *SAP Notes* [408527](#) and [628590](#).

 Caution

As of Oracle 10g, table monitoring is active by default.

- BRCONNECT support for system and Oracle dictionary statistics

You can use BRCONNECT to call the `DBMS_STATS` package to collect system statistics, as follows:

```
brconnect -u system /<password> -c -f stats -t SYSTEM_STATS -i  
<minutes>
```

Or you can use BRCONNECT to collect Oracle dictionary statistics as follows:

```
brconnect -u / -c -f stats -t ORADICT_STATS
```

For more information, see [brconnect -f stats](#).

You can also determine the interval in minutes for collecting system statistics using the `init<SID>.sap` parameter [stats system interval](#).

## Process

You choose one of the following approaches to update statistics:

- DBA Planning Calendar in the Computing Center Management System (CCMS)

For more information, see [Update Statistics for the Cost-Based Optimizer in CCMS \(Oracle\)](#). The DBA Planning Calendar uses the BRCONNECT commands.

 Recommendation

We recommend you to use this approach in production operation because you can easily schedule update statistics to run automatically at regular intervals. We recommend you to update statistics at least weekly or even daily for Oracle 10g.

- [BRCONNECT](#)
- [Manage database statistics with BRSPACE](#)

## BR\*Tools for Oracle DBA

SAP provides the tools BRBACKUP, BRARCHIVE, BRRESTORE, BRRECOVER, BRSPACE, BRCONNECT, and BRTOOLS to manage and protect the data in your Oracle database.

This section describes how to *perform* Oracle DBA with BR\*Tools. For more information on how to work out an *approach* to Oracle DBA, see [Approach to Oracle DBA](#)

For more information on other tools, including the Oracle tool SQL\*Plus, see [Other Tools for Oracle DBA](#).

### Integration

The BR\*Tools are:

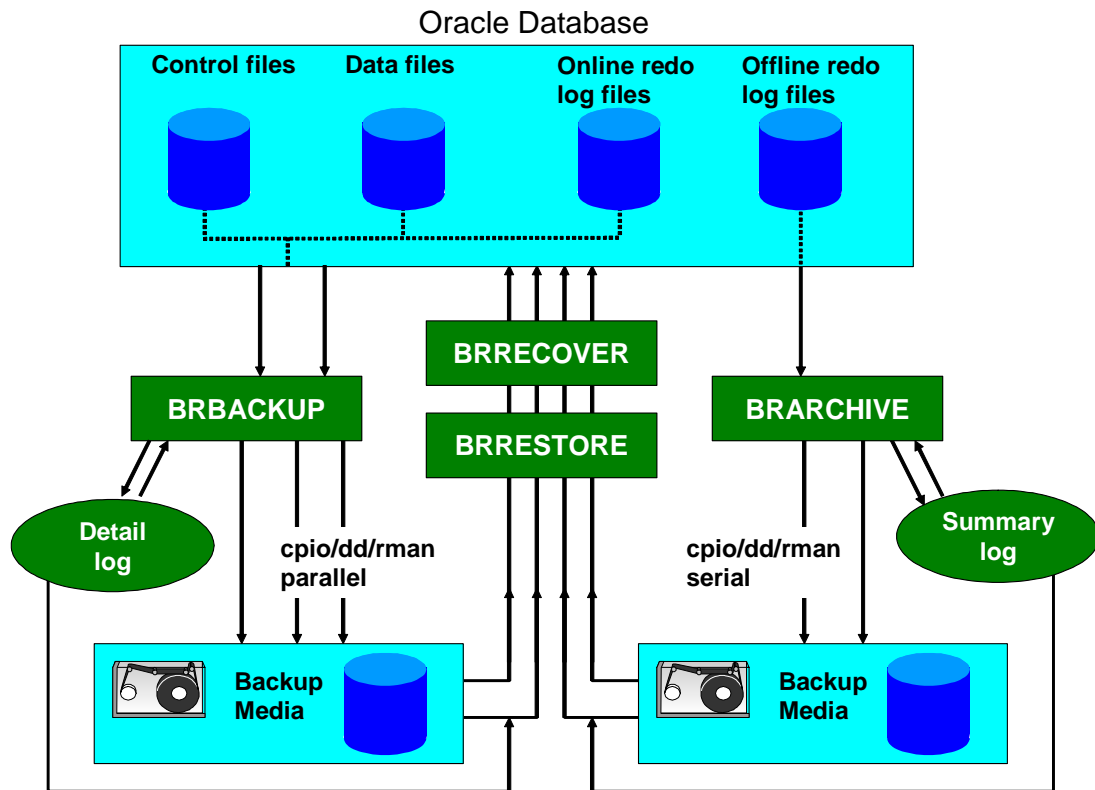
- Integrated with [BRTOOLS](#), which is the character-based interface that calls BRBACKUP, BRARCHIVE, BRRESTORE, BRRECOVER, BRSPACE, and BRCONNECT
- Available for the operating systems UNIX and Microsoft Windows

#### Note

Distinguish between the following:

- BR\*Tools is the program package containing BRBACKUP, BRARCHIVE, BRRESTORE, BRRECOVER, BRSPACE, BRCONNECT, and BRTOOLS.
- BRTOOLS is the program that displays the menus from which the other BR programs are called.

As an example, the following graphic shows the integration of the backup, restore, and recovery tools:



## Prerequisites

The BR\*Tools are installed automatically on the database server in the directory `/usr/sap/<SAPSID>/SYS/exe/run`.

We classify BR\*Tools as functional, help, batch, and interactive tools:

### Tool Types for BR\*Tools

Type	Tool	Description
Functional	BRBACKUP, BRARCHIVE, BRRESTORE, BRRECOVER, BRSPACE, and BRCONNECT	Perform actions directly on database files or objects
Help	BRTOOLS and BRCONNECT	<ul style="list-style-type: none"> <li>BRTOOLS is the menu program that can call all other functional tools interactively</li> <li>BRTOOLS and BRCONNECT are also called internally: <ul style="list-style-type: none"> <li>BRTOOLS is called by BRBACKUP, BRARCHIVE, and BRRESTORE for backup, restore, and verification.</li> <li>BRCONNECT is called by BRBACKUP to monitor the database during a backup</li> </ul> </li> </ul>

## Tool Types for BR\*Tools

Type	Tool	Description
Batch	BRBACKUP, BRARCHIVE, BRRESTORE, BRCONNECT	Only run on their own in batch, without menus. You can call them interactively with BRTOOLS.
Interactive	BRSPACE, BRRECOVER, BRTOOLS	Run interactively with their own menus. BRSPACE and BRRECOVER can also run in batch mode.

### Note

There is a menu in BRCONNECT to change the database password in GUI mode.

## Features

### Toolset for BR\*Tools

Tool	Purpose
<a href="#">BRBACKUP</a>	Backs up data files, control files, and online redo log files of the database
<a href="#">BRARCHIVE</a>	Backs up offline redo log files
<a href="#">BRRESTORE</a>	Restores data files, control files, and redo log files
<a href="#">BRRECOVER</a>	Recovers database files and restores profiles and log files
<a href="#">BRSPACE</a>	Manages the database instance, space, and segments (that is, tables and indexes)
<a href="#">BRCONNECT</a>	<ul style="list-style-type: none"><li>• Performs database administration tasks such as statistics update, check database system, adapt next extents, clean up logs and DBA tables</li><li>• Functions as a help tool to monitor the database during a backup</li><li>• Displays the menus from which the other BR programs are called</li></ul>
<a href="#">BRTOOLS</a>	<ul style="list-style-type: none"><li>• Functions as an internal help tool started by BRBACKUP, BRARCHIVE, and BRRESTORE</li></ul>
BRGUI	Functions as a Java-based GUI, working as the front-end display program for BR*Tools

The following features apply to all the BR\*Tools:

- [Menu-driven interface](#), including GUI, in English or German
- Context-sensitive supporting information
- Detail and summary logs

- Available with operating systems UNIX and Microsoft Windows
- Command-line options for experts

## Activities

For more information, see [Getting Started with BR\\*Tools](#).



## Getting Started with BR\*Tools

To get started with BR\*Tools, you need to configure it and learn how to use it.

### Prerequisites

If you are new to Oracle database administration with the SAP system, see [Getting Started with Oracle and the SAP System](#).

### Process

1. You [configure BR\\*Tools](#).
2. You [start BR\\*Tools](#).
3. You learn about the [BR\\*Tools user interface](#).
4. You learn how to [use BR\\*Tools](#).
5. If necessary, you check [BR\\*Tools release information](#).



## Configuration of BR\*Tools

To [get started with BR\\*Tools](#), you need to configure it.

### Prerequisites

- You have made any required settings in the initialization profile. The default [Initialization Profile init<DBSID>.dba](#) is:
  - UNIX: <ORACLE\_HOME>/dbs/init<DBSID>.sap
  - Windows: %<ORACLE\_HOME>%\database\init<DBSID>.sap

Changes to profile parameters become active when you start BR\*Tools.
- You have set the environment variables when you configured the database:
  - [Environment Variables \(UNIX\)](#)
  - [Environment Variables \(Windows\)](#)
- You are familiar with the directory structure:
  - [Directory Structure \(UNIX\)](#)
  - [Directory Structure \(Windows\)](#)

## Process

1. You [configure the scroll line count for BR\\*Tools](#).
2. You [configure the UNIX command at for BR\\*Tools batch functions](#).
3. You [set the option to log displayed information for BRSPACE](#).
4. You check the [effects of autoextend and resize on BR\\*Tools](#).



## Configuring the Scroll Line Count for BR\*Tools

The BR\*Tools character interface uses 20 lines for scrolling in lists. You can lengthen or shorten the display. For example, you can choose to use more than 20 lines for the scroll line count if your command windows have more than 20 lines.



### Note

This section is not relevant for the graphical user interface, BRGUI, where you can scroll freely.

## Prerequisites

The environment variable `BR_LINES` and the parameter [scroll\\_lines](#) in the initialization parameter file, `init<DBSID>.sap`, control the BR\*Tools list display. For more information on environment variables, see:

- [Environment Variables \(UNIX\)](#)
- [Environment Variables \(Windows\)](#)

## Procedure

To select a different number for the scroll line count, set the following environment variable (operating-system specific) *before* you start BR\*Tools:

```
BR_LINES <Number of lines for scrolling>
```



## Configuring the UNIX Command at for BR\*Tools Batch Functions

This section tells you how to configure the UNIX command `/usr/bin/at`, which you can use to schedule BR\*Tools functions in batch mode.

## Prerequisites

The `at` command has the following authorizations:

```
r-sr-xr-x root root at
```

To use `at`, you must make an entry in the file `/usr/lib/cron/at.allow`. Add `ora<dbsid>` to the list of authorized users.

The running `at` process has root authorization, and analyzes the jobs created by BR\*Tools in file `/usr/spool/cron/atjobs`.

## Procedure

1. To list all `at` jobs, enter the command `at -l`.

These jobs are transparent files that you can display using UNIX commands such as `vi`, `view`, `more`, and so on.

2. Make sure the proper entries have been made, and then test your configuration.

For example, reorganize a small table or tablespace such as `PSAP<SCHEMA_ID>USR`. This helps you avoid processes crashing due to incorrect configuration of the `at` command.



## Setting the Option To Log Displayed Information for BRSPACE

You can have BRSPACE log the information displayed in a show function for BRSPACE.

### Prerequisites

- Decide whether you need to set this option:
  - Do not set this option if you want to show the database information and the results do not have to be available at a later time. The displayed data is not recorded in the BRSPACE log files to avoid unnecessary logging.
  - Set the option if you want to record the displayed data in the log files.
- This option is global for BRSPACE. That is, it is valid for each class of displayed information in a BRSPACE run.

### Procedure

Choose one of the following to set the log option:

- BRTOOLS or BRGUI:
  1. Choose **▸** *<Management function>* **▸** *<Show function>* **⏪**.
  2. Enter `yes` in *Create log file*.

#### Example

Choose **▸** *Instance management* **▸** *Show instance status* **⏪**.

Choose *Continue*.

Enter `yes` in *Create log file*.

- Command line:

Use option `-l|-log` of BRSPACE function `dbshow`.

#### Example

Enter `brspace -f dbshow -c dbstate -l` at the command line.

## Effects of Autoextend and Resize on BR\*Tools

*Autoextend* and *Resize* are two Oracle options for influencing the size of the data files of the database system:

- *Autoextend* extends the data files automatically by a specified amount
- *Resize* lets you increase the size of data files manually (up to the maximum file system size), or reduce their size (down to the last used block ID of the data file).

These options influence the BR\*Tools functions below, and have been adjusted accordingly.

### Restore and Recovery

When performing a recovery (that is, importing the offline redo log files), the Oracle database system automatically takes into account the autoextends made to the data files during or after a database backup.

### Database System Check

Parameters for checking freespace in tablespaces that take into account the *Autoextend* option are the following:

- `TABLESPACE_FULL` for absolute freespace
- `CRITICAL_SEGMENTS` for critical segments

For more information, see [BRCONNECT Default Conditions for Database Administration](#).

### Reorganization

- The preventative freespace check includes the *Autoextend* option
- Before you reorganize a table, you can set the *Autoextend* parameters `MAXSIZE` and `INCREMENT_BY` when you alter the data files. The parameter `MAXSIZE` takes into account the memory of the file system or the raw device.

For more information, see [Reorganizing Tables with BR\\*Tools](#).

### Tablespace Management

- You can create the data files of the tablespaces with the parameters `AUTOEXTEND ON (OFF)`, `MAXSIZE` and `INCREMENT_BY`. You can also change these parameters. The parameter `MAXSIZE` takes into account the memory of the file system or the raw device.

For more information, see [Extending a Tablespace with BR\\*Tools](#) and [Creating a Tablespace with BR\\*Tools](#).



- You can increase or reduce the size of the data files of the tablespaces with the *Resize* action. For more information, see [Altering a Data File with BR\\*Tools](#).

## Starting BR\*Tools

You can use this procedure to start BR\*Tools for Oracle. You can only use BR\*Tools to manage a database system that is running on the same host system.

### Prerequisites

- You have configured:
  - [The database system](#)
  - [BR\\*Tools](#)
- Logon user

You can log on as the operating system user who owns the data files of the database system. The standard Oracle user created during the installation of the SAP system is `ora<dbSID>` (UNIX) or `<DBSID>ADM` (Windows).

You can also use some BR\*Tools programs if you are logged on as the SAP System user `<sapsid>adm`. This requires the authorization for the BR\*Tools program to be set accordingly (for example, under UNIX: `rwsr_xr_x oracl1 dba brbackup`).

The advantage of this procedure is that the administrator who works with BR\*Tools does not have the authorizations of user `ora<dbSID>` (who can delete database files directly, for example, and perform other critical operations for the database).

BR\*Tools establishes the connection to the database with a special database user who has authorization to create and delete tablespaces, to create data files, and so on (DBA privileges). This default user is `system`.

### Procedure

1. Log on to the host where your database system is running.

#### Note

If you start BR\*Tools with a script, as a background job or from the command line, you must *not* use special characters (such as `$` or `#`) in the user name and password.

2. If you want BR\*Tools to log on as user `system` to the Oracle database, call it as follows from the command line:

```
OS> <brtool> ...
```

For example, `brbackup -t online ...`

If the default password is not used, you have to use option `-u` `<user>/<password>`.

You can call BR\*Tools with the following command options:

- OS> `<brtool> -p [<path>/]<profile name>`

You can also specify an initialization profile that is different to the standard profile.

If you do not specify option `-p`, BR\*Tools uses the values set in the default profile `init<DBSID>.sap`, which must be available. For more information, see [Configuration of BR\\*Tools](#).

- o OS> `<brtool> -u <user>/<password>`

Specifies a DBA user different to the default user (`system/<default password>`).

`<user name>`: Database user that you defined

`<password>`: Password of your database user.

If possible, avoid starting BR\*Tools with the command option `-u` and the immediate entry of user name and password. In this case, the command line with the DBA user and its password can be displayed in the UNIX process list (for example, by using the `ps` command).

Enter the password interactively. When you use `<brtool> -u <user>`, the system prompts you to enter a valid password. In this case, the password does not appear on the screen as it is entered and is not displayed in the process list. Depending on the operating system the password length is limited to a certain number of places (for example, 8 characters for HP-UX, 32 characters for AIX).

The following examples illustrate different procedures for using passwords:

#### Example

```
OS> <brtool> -u system
```

BR\*Tools prompts you for the password. The password is not visible on the screen.

```
OS> <brtool> -u
```

BR\*Tools prompts you for the user and password. The password appears on the screen.

```
OS> <brtool> -u < <file name>
```

The user and password is written to the file `<file name>`. Access to this file can be restricted with operating system privileges.

```
OS> <brtool> -u /...
```

Call for an `OPSS` user (also applies to background processing). To make sure that the password is not visible in the process list, you can create an `OPSS` user (see Oracle documentation and information in SAP Service Marketplace) in the database and assign the SAPDBA role to the user.

You can get an overview of all the command options by entering the following command:

```
OS> <brtool> -h[elp]
```

For information about other command options that are not mentioned in this section, see the section on command options for the relevant one of the BR\*Tools. For example, for BRSPACE command options, see [Command options for BRSPACE](#).

## BR\*Tools User Interface

The user interface to BR\*Tools provides you with menus to perform a wide range of database administration functions for your Oracle database. The menus are controlled by BRTOOLS, which in turn calls one of the functional BR programs.

You can use BR\*Tools with a:

- Character-based interface, as in the following example, which shows the main menu:

```
BR0280I Time stamp 2003-03-06 11.30.57

BR0656I Choice menu 1 - please make a selection

-----
-

BR*Tools main menu

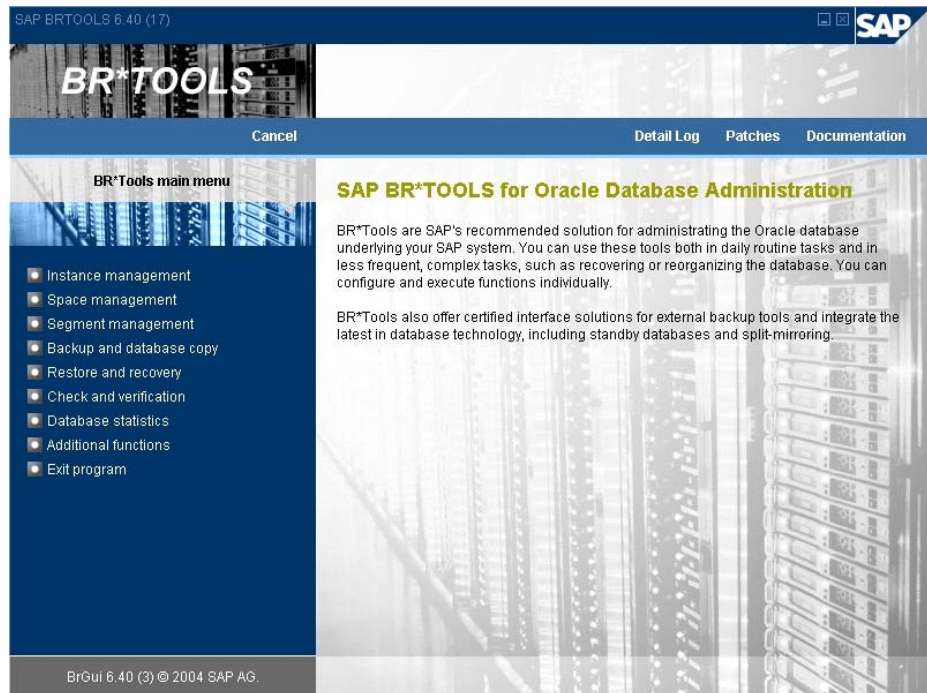
1 = Instance management
2 - Space management
3 - Segment management
4 - Backup and database copy
5 - Restore and recovery
6 - Check and verification
7 - Database statistics
8 - Additional functions
9 - Exit program

Standard keys: c - cont, b - back, s - stop, r - refr, h - help

-----
-

BR0662I Enter your choice:
```

- GUI, as in the following example, which shows the main menu:



#### Note

BRGUI is the graphical interface for BR\*Tools. It displays output and gathers user input for BR\*Tools.

## Features

You see the following types of menus when using BRTOOLS:

- Control

This type of menu leads you from step to step in a pre-defined sequence. You can repeat steps in the correct sequence. Here is an example:

```
BR0655I Control menu 101 - please decide how to proceed
```

```
-----
Complete database recovery main menu
```

```
1 = Check the status of database files
```

```
2 * Select database backup
```

```
3 * Restore data files
```

```
4 * Restore and apply incremental backup
```

```
5 * Restore and apply archive log files
```

```
6 * Open database and post-processing
```

```
7 - Exit program
8 - Reset program status
Standard keys: c - cont, b - back, s - stop, r - refr, h - help
-----
```

It is normally possible for you to repeat steps from a control menu but this is not always a sensible thing to do. Only repeat steps when you understand what the effects will be.

- **Choice**

This type of menu lets you make an independent choice from the menu in any sequence. You can repeat the choice as often as required. Here is an example:

```
BR0656I Choice menu 120 - please decide how to proceed
-----

Restore of individual backup files main menu
1 = Restore files from BRBACKUP backup
2 - Restore individual files from tape
3 - Restore individual files from disk
4 - Restore individual files from backup utility
5 - Restore and apply incremental backup
6 - Exit program
7 - Reset program status
Standard keys: c - cont, b - back, s - stop, r - refr, h - help
-----
```

- **Input**

This type of menu lets you enter required parameters or options. Here is an example:

```
BR0657I Input menu 123 - please check/enter input values
-----

BRRESTORE main options for restore from BRBACKUP backup
1 - BRRESTORE profile (profile) ..... [initGC2.sap]
2 - BRBACKUP backup run (backup) ..... [bdjwgvvh.fnf]
3 - Fill-up previous restores (fillup) . [no]
4 - Restore device type (device) ..... [util_file]
5 - BACKINT/Mount profile (parfile) .... [initGC2.utl]
6 # Database user/password (user) ..... [system/*****]
7 ~ Restore destination (rest_dest) .... []
```

8 - Files for restore (mode) ..... [2,11-12]  
Standard keys: c - cont, b - back, s - stop, r - refr, h - help  
-----

- **List**

This type of menu displays a list of items from which you select one entry. Here is an example:

```
BR0658I List menu 121 - please select one entry
-----

BRBACKUP database backups for restore

Pos. Log Start Type Files Device Rc
1 = bdjwhckx.ffd 2003-01-29 17.30.51 offline 110/0 disk 0
2 - bdjwhadu.fft 2003-01-29 17.05.14 offline 112/0 tape 1
3 - bdjwgyrq.fff 2003-01-29 16.48.42 offline 112/0 util_onl 0
4 - bdjgwgtj.fnt 2003-01-29 16.26.55 onl_cons 115/0 tape 0
5 - bdjwgvvh.fnf 2003-01-29 16.16.29 onl_cons 115/0 util_onl 0

BR0280I Time stamp 2003-01-29 19.05.29
```

There are a number of sub-types for the list menu:

- **List menu with optional single selection**

```
BR0658I List menu 365 - you can select one entry
-----
-----

List of BRSPACE export runs

Pos. Run Date Tables Dumps Size[KB]
1 - sdlmvogy.tbe 2003-09-11 20.28.36 1 1 2
2 - sdlmqobv.tbe 2003-09-10 20.01.43 2 1 14
3 - sdlmqimk.tbe 2003-09-10 18.58.42 2 1 5
4 - sdlmqgin.tbe 2003-09-10 18.34.29 2 1 18
5 - sdlmqfmw.tbe 2003-09-10 18.25.06 2 1 12
...

Press <Rtn> to scroll, <n> to select, 'c' to continue,
's' to stop scrolling..
```

- **List menu with multiple selection**

```
BR0659I List menu 312 + please select one or more entries
```

-----  
-----  
List of tablespaces for alter

Pos. Tablespace Files/AuExt. Type Status SegMan. ExtMan.  
Backup

1 - DRSYS 1/1 DATA ONLINE AUTO LOCAL NO  
2 - EXAMPLE 1/0 DATA ONLINE AUTO LOCAL NO  
3 - INDX 1/1 DATA ONLINE AUTO LOCAL NO  
4 - PSAP1111D 1/1 DATA ONLINE AUTO LOCAL NO  
5 - PSAP1111I 1/1 DATA ONLINE AUTO LOCAL NO

...

Standard keys: c - cont, b - back, s - stop, r - refr, h  
- help

-----  
-----  
o List menu with optional multiple selection

BR0659I List menu 259 + you can select one or more  
entries

-----  
-----  
List of database tablespaces

Pos. Tablespace Type Status ExtMan. SegMan. Backup  
Files/AuExt.

Total[KB] Used[%] Free[KB] ExtSize[KB] FreeExt.  
Largest[KB]

1 - DRSYS DATA ONLINE LOCAL AUTO NO 1/1  
10240 47.50 5376 1038336 2 1038336+:5312:64  
:0:0

2 - EXAMPLE DATA ONLINE LOCAL AUTO NO 1/0  
123520 8.29 113280 0 5 93056:16128:22  
40:1728:128

3 - INDX DATA ONLINE LOCAL AUTO NO 1/1  
5120 1.25 5056 1043456 1 1043456+:5056:0:  
0:0

...

Standard keys: c - cont, b - back, s - stop, r - refr, h  
- help

-----  
-----  
o **List display without selection**

BR0660I List display 352 - no selection possible

-----  
-----  
List of tables for reorganization

Pos. Owner Table Part. Rows Space[KB] Data[KB:%]

1 - SAPR3 DBABARL NO 429 64 42:65

2 - SAPR3 DBABD NO 1068 320 253:79

3 - SAPR3 DBABL NO 763 256 189:74

4 - SAPR3 DBADFL NO 0 64 0:0

5 - SAPR3 DBAERR NO 0 64 0:0

...

Standard keys: c - cont, b - back, s - stop, r - refr, h  
- help

-----  
-----  
• **Display in form of input menu, but no input possible**

BR0692I Display menu 260 # no input possible

-----  
-----  
Information about tablespace DRSYS

1 - Tablespace type (type) ..... DATA

2 - Tablespace status (status) ..... ONLINE

3 - Extent management (extent) ..... LOCAL

4 - Segment space management (space) ..... AUTO

5 - Backup status (backup) ..... NO

6 - Number of files in tablespace (files) .... 1

7 - Number of autoextensible files (autoext) . 1

8 - Total tablespace size in KB (total) ..... 10240

9 - Used space in tablespace in % (used) ..... 47.50

10 - Free space in tablespace in KB (free) .... 5376

11 - Maximal extension size in KB (extsize) ... 1038336

12 - Number of free extents (freext) ..... 2



```
13 - Largest free extents (largest) .....  
1038336+:5312:64:0:0
```

Standard keys: c - cont, b - back, s - stop, r - refr, h - help

-----

## How to Use BR\*Tools

You can start [BR\\*Tools](#) in the following ways:

- Interactively from the BRGUI graphical interface or the BRTOOLS character interface
- Directly from the command line using the relevant BR\*Tools options

For BRSPACE, you can also use “quick mode” from the BRGUI or BRTOOLS interface or the command line. For more information, see *Quick Mode for BRSPACE* below.

### Note

There is also an unattended or “batch” mode, but this is only relevant when you have completed your input and started execution to perform some action on the database.

With the option `-c|-confirm`, you can specify that the BR\*Tool runs without operator input in unattended mode, only stopping when absolutely necessary, that is in menus and “yes / no” choices (message BR0676I). All other confirmation prompts and so on are skipped.

With the option `-c|-confirm force`, you can specify that BR\*Tool runs in fully unattended mode, so that an error is generated if input is required and has not already been given.

Whichever start method you use, at the end a functional tool – BRBACKUP, BRARCHIVE, BRRESTORE, BRRECOVER, BRSPACE, or BRCONNECT – directly performs some action on the database.

## Process

### Interactive Start

You start BRGUI or BRTOOLS from the command line and then choose the menu options required:

- [Database Instance Management with BR\\*Tools](#)
- [Space Management with BR\\*Tools](#)
- [Segment Management with BR\\*Tools](#)
- [Backup and Database Copy with BR\\*Tools](#)
- [Restore and Recovery with BR\\*Tools](#)
- [Check and Verification with BR\\*Tools](#)
- [Database Statistics with BR\\*Tools](#)
- [Additional Functions with BR\\*Tools](#)

When you are ready to execute the functional tool, BRGUI or BRTOOLS normally shows you the command that is to be executed. Then BRGUI or BRTOOLS normally calls the correct functional tool to perform the function that you have specified in the menus.

You can enter manually enter the command. For more information, see *Command Line Mode* below. However, you might get errors if you change the command.

#### Note

BRSPACE and BRRECOVER differ in that they also have an interactive mode separate from BRGUI or BRTOOLS. Therefore, BRTOOLS passes control to BRSPACE or BRRECOVER , which themselves gather further input using menus, before starting to perform an action on the database.

In addition, there is a “quick mode” for BRSPACE. If you specify an object name in BRGUI or BRTOOLS, you enter BRSPACE in quick mode. For more information, see below.

There are the following reasons for this:

- To enable full logging of BRSPACE actions – as soon as BRGUI or BRTOOLS calls BRSPACE, a log is created immediately, even though you are still in interactive mode.
- To enable unattended mode without operator input, since you can specify all necessary input before the function is executed.

## Command Line Start

You enter the command options required directly from the command line:

- [Command Options for BRBACKUP](#)
- [Command Options for BRARCHIVE](#)
- [Command Options for BRRESTORE](#)
- [Command Options for BRRECOVER](#)
- [Command options for BRSPACE](#)
- [Command Options for BRCONNECT](#)

If the command options are correct and complete, the SAP tool is executed immediately to perform a function on the database.

## Calling BRSPACE and BRRECOVER from the Command Line

BRSPACE and BRRECOVER have an interactive component to collect option input. If you do not specify a function, the tool is called with the default function when you start it from the command line:

- *Complete Database Recovery* for BRRECOVER
- *Show Database Instance Information* for BRSPACE

Otherwise, the appropriate menus are shown for the function you select.

You can force the tool to run in batch mode without any interactive component by entering the option `-c force`. For more information, see:

- [-cl-confirm](#) for BRRECOVER
- [-cl-confirm](#) for BRSPACE



Be careful with `-c force` because it forces default selection of all unspecified options, which might lead to unexpected results.

If you call BRSPACE from the command line with a function name (for example, extend tablespace) and an object name (for example, tablespace name `SYSTEM`), you go straight into quick mode, as described below.

## Quick Mode for BRSPACE

You can use quick mode to skip the higher-level menus in BRSPACE, including the menu to select the object of your chosen function.

You use quick mode from:

- BRGUI or BRTOOLS menus with the title *BRSPACE options for <function>*
- The command line at operating system level

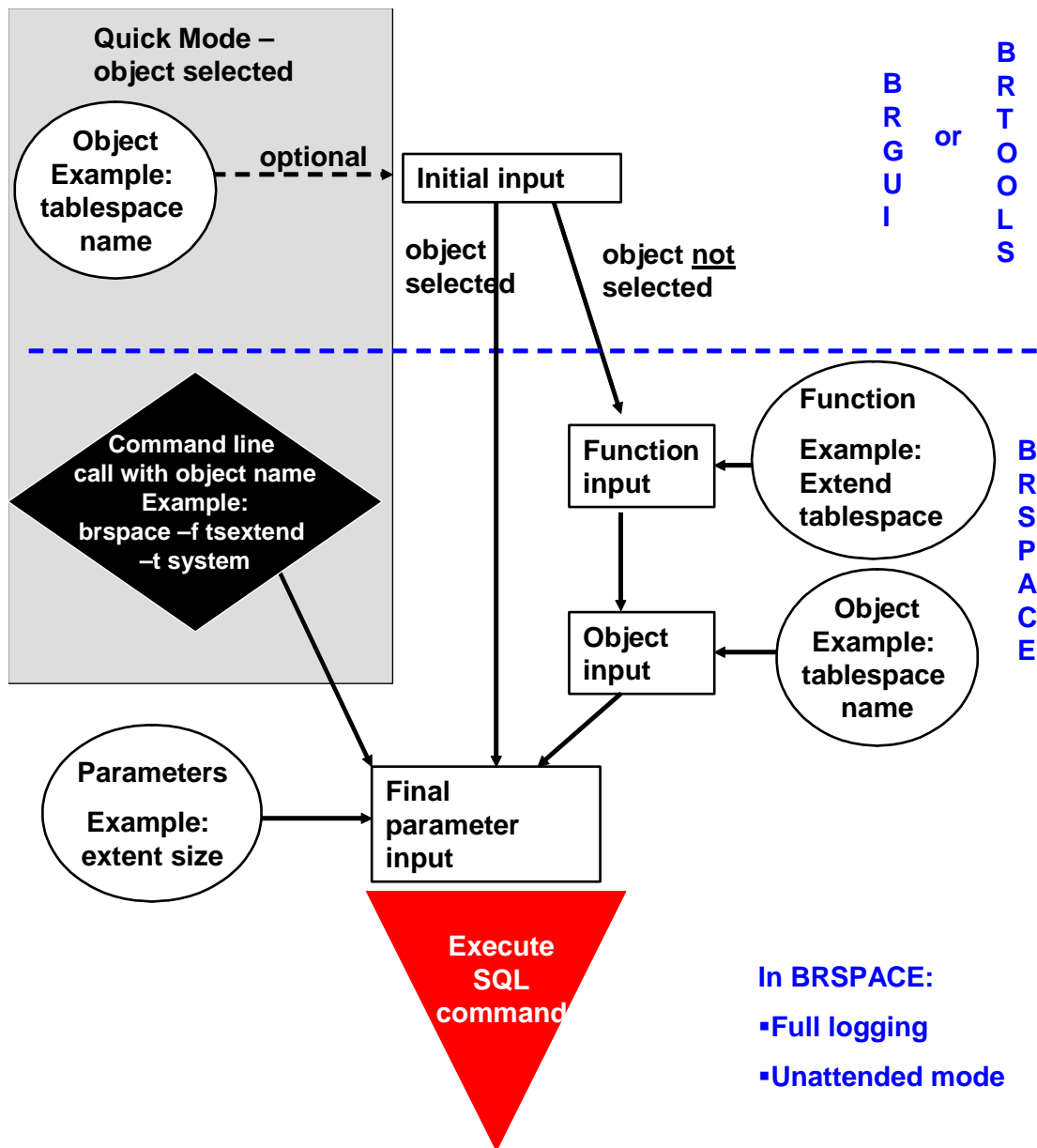
For quick mode, you must specify *at least the function, the object name, and/or the action*. The object – for example, a tablespace name if you want to extend a tablespace – varies according to the function that you choose. You can also enter additional options in quick mode.



The alternative to quick mode is main menu mode, in which you only use the menus in BRSPACE to enter the options required to perform a function. Unless you know the object name, we recommend you to use main menu mode.

In quick mode, BRSPACE skips the menus in which you normally specify the function, the action, and/or the object name. They jump directly to the final input menu to let you enter the remaining required parameters and execute the function immediately.

The following graphic illustrates in outline how quick mode works:



## Exceptions to quick mode

There are the following exceptions to quick mode:

- Function `dbshow` to show database information.

If you do not enter the `class` of the information to be displayed, you see a menu displaying the different categories of information `class`:

*Show database instance information*

*Show database space information*

*Show database object information*

Choose the category and then the `class` that you require.

BRSPACE displays the requested information.

- Alter functions:
  - `dbalter` to alter the database instance
  - `dbparam` to alter database parameters
  - `tsalter` to alter tablespaces
  - `dfalter` to alter data files
  - `tbalter` to alter tables
  - `idalter` to alter indexes

With these functions, you should also enter an action to specify what kind of alter function you require. If you do not enter an action in quick mode, BRSPACE prompts you to select an action before you can continue.

- Multiple Objects

If you select multiple objects in quick mode, BRSPACE displays a list of objects for confirmation. If the list is not what you require, you can go back and make a new selection.

#### Example

- The following example shows quick mode from the command line:

```
brspace -f tsextend -t psapprd
```

BRSPACE starts by displaying the input menu for the function *Extend tablespace*. The field for the object name is filled with your entry.

You can now enter all required options and execute the function immediately.

- The following example shows a command line entry for an alter function where the object (here, the database parameter) is specified, but not the action:

```
brspace -f dbalter -p audit_trail
```

BRSPACE starts by displaying the function main menu for the function that you entered, which is *Alter database parameter* in this example.

You must first select the action before BRSPACE can display the input menu to let you execute the function.

#### Note

When you have finished executing a function and you choose *back* after starting BRSPACE in quick mode, the quick mode is deactivated and you can choose a new object to execute the function again. If you choose *reset program status*, BRSPACE restores your original object selection.

The exception to this is for the functions in segment management, where it makes no sense to display all tables or indexes, because the list would be too long.



## Checking BR\*Tools Release Information

You can check the BR\*Tools release information.

When you contact SAP support to register problems concerning BR\*Tools, it helps to give as much information as possible about the version you are using.

### Procedure

To display release information with the command option, enter the following at the command line:

```
OS> <brtool> -V all
```

For example, `brspace -V all`

For more information, see the relevant command option for the tool that you are using. For example, for more information on how to display release information for BRSPACE, see [-V|-VERSION](#).

### Result

BR\*Tools displays the following release information:

- *BR\*Tools release*
- *BR\*Tools patch level*
- *BR\*Tools patch date*



## BR\*Tools in Action

This section describes how you use BR\*Tools to perform database administration tasks with your Oracle database.



## Instance Management with BR\*Tools

You can manage your Oracle database instance with [BR\\*Tools](#).



Note

This section describes how you perform instance management with BR\*Tools.

For more information on the approach to instance management, see [Instance Management](#).

### Integration

- BRGUI or BRTOOLS calls the SAP tool BRSPACE. You can also manage the database instance by calling BRSPACE from the command line. However, unless you

choose batch mode with the option `-c force`, BRSPACE displays menus to help you enter the required options.

#### Recommendation

We recommend you to normally use BRGUI or BRTOOLS rather than BRSPACE directly. This is because the menus in BRGUI or BRTOOLS simplify entry of the correct options.

- If required, set parameters to control the screen display for BRSPACE in the [Initialization Profile init<DBSID>.sap](#). These are [scroll\\_lines](#) and [show\\_period](#).

## Features

You can perform the following functions for database instance management with BR\*Tools:

- Start up the database
- Shut down the database
- Alter the database instance
- Alter the database parameters
- Recreate the database
- Show the database instance status
- Show database parameters
- Show database owners
- Show flashback status

## Activities

1. You choose *Instance Management* in the BRGUI or BRTOOLS menus or directly from the command line. You can use quick mode if you have specified the instance name(s). For more information on quick mode, see [How to Use BR\\*Tools](#).
2. If required, you change the default values for the parameters in the initialization profile `init<DBSID>.sap` and restart BRTOOLS.
3. If required, you choose **► Instance Management ► Reset program status ◀** to set the defaults used to the values set after you started BRGUI or BRTOOLS.
4. You start database instance management.
5. You check the results of the database instance management in the [BRSPACE logs](#).



## Starting Up the Database with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to start up the database.

#### Note

This section describes how you start the database.

Starting up the database is part of [Instance Management with BR\\*Tools](#).

For more information on the approach to database instance management, see [Instance Management](#).

## Prerequisites

Some steps in this procedure only apply if your database is an Oracle Real Application Cluster (RAC). These are marked "RAC only".

## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:

- BRGUI or BRTOOLS:

1. Choose **► Instance Management ► Start up database ◀**.

BRGUI or BRTOOLS displays the menu *BRSPACE options for database instance startup*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Database instance (instance)</i>	<a href="#">-f dbstart -i -instance</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f dbstart</a> that is to be executed using the current settings.

3. Choose *Continue*.

BRGUI or BRTOOLS prompts you to start BRSPACE.

4. Choose *Continue* to start BRSPACE.

- Command line

You need to enter at least the following command:



```
brspace -f dbstart
```

You can enter more parameters, including the instance name, if required. For more information, see [BRSPACE -f dbstart](#).

 **Note**

Whichever way you start the procedure - with BRGUI or BRTOOLS, or from the command line - you can take advantage of quick mode if you know the object name, in this case the instance name. For more information, see [How to Use BR\\*Tools](#).

RAC only: to start all database instances that are currently down, enter `all_down` in *Database instance (instance)* or `-i all_down` on the command line.

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#).
3. If you have already entered the instance name, continue with step 5 (quick mode).  
Otherwise, BRSPACE displays the *Database instance startup main menu*.
4. Choose *Start up database*, the default selection.

RAC only: BRSPACE displays the *List of database instances for startup*.

Tab 2

List Entry	Meaning
<i>Pos</i>	List sequence number
<i>Name</i>	Instance name
<i>Number / Thread</i>	Instance number and thread
<i>Status</i>	Instance status
<i>Start time</i>	Time the instance was started
<i>RedoSeq</i>	Log redo sequence number
<i>SapConn</i>	Number of SAP connections

5. RAC only: select a database instance or multiple database instances.

 **Example**

These examples apply only to input in character mode.

To select the first three entries in the list, enter 1-3.

To select the first and third entries, enter 1,3.

To select the first three entries and the fifth, enter 1-3,5.

To select all entries, enter 0.

BRSPACE displays the menu *Options for starting up database instance*.

 Note

In normal (that is, non-RAC systems), BRSPACE automatically selects the name of the single instance for you.

6. Set the required options:

Menu Entry	Meaning
Database startup to-state (state)	<a href="#">-f dbstart -s -state</a>
Database open mode (mode)	<a href="#">-f dbstart -m -mode</a>
Force instant restart (force)	<a href="#">-f dbstart -f -force</a>
SQLPLUS command (command)	The SQL command that is to be executed using the current settings. For more information, see your Oracle SQL documentation.

7. To start processing with the selected options, choose *Continue*.

## Result

Check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.dbr` displays the details.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Shutting Down the Database with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to shut down the database.

 Note

This section describes how you shut down the database.

Shutting down the database is part of [Instance Management with BR\\*Tools](#).

For more information on the approach to database instance management, see [Instance Management](#).

## Prerequisites

Some steps in this procedure only apply if your database is an Oracle Real Application Cluster (RAC). These are marked "RAC only".

## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:

- BRGUI or BRTOOLS:

1. Choose  Instance Management  Shut down database .

BRGUI or BRTOOLS displays the menu *BRSPACE options for database instance shutdown*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Database instance (instance)</i>	<a href="#">-f dbshut -i -instance</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f dbstart</a> that is to be executed using the current settings.

3. Choose *Continue*.

BRGUI or BRTOOLS prompts you to start BRSPACE.

4. Choose *Continue* to start BRSPACE.

- Command line

You need to enter at least the following command:

```
brspace -f dbshut
```

You can enter more parameters, including the instance name, if required. For more information, see [BRSPACE -f dbshut](#).

 Note

Whichever way you start the procedure - with BRGUI or BRTOOLS, or from the command line - you can use quick mode if you know the object name, in this case the instance name. For more information, see [How to Use BR\\*Tools](#).

RAC only: to stop all database instances that are currently up and running, enter `all_up` in *Database instance (instance)* or `-i all_up` on the command line.

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#).
3. If you have already entered the instance name, continue with step 5 (quick mode).  
Otherwise, BRSPACE displays the *Database instance shutdown main menu*.
4. Choose *Shut down database*, the default selection.

RAC only: BRSPACE displays the *List of database instances for shutdown*.

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Name</i>	Instance name
<i>Number/Thread</i>	Instance number and thread
<i>Status</i>	Instance status
<i>Start time</i>	Time the instance was started
<i>RedoSeq</i>	Log redo sequence number
<i>SapConn</i>	Number of SAP connections

5. RAC only: select a database instance or multiple database instances.

#### Example

These examples apply only to input in character mode.

To select the first three entries in the list, enter 1-3.

To select the first and third entries, enter 1,3.

To select the first three entries and the fifth, enter 1-3,5.

To select all entries, enter 0.

BRSPACE displays the menu *Options for shutting down database instance*.

#### Note

In normal (that is, non-RAC systems), BRSPACE automatically selects the name of the single instance for you.

6. Set the required options:

Menu Entry	Meaning
Database close mode (mode)	<a href="#">-f dbshut -m -mode</a>
Force instant shutdown (force)	<a href="#">-f dbshut -f -force</a>
SQLPLUS command (command)	The SQL command that is to be executed using the current settings. For more information, see your Oracle SQL documentation.

7. To start processing with the selected options, choose *Continue*.

## Result

Check the results in the [BRSPACE logs](#).

- The [summary log](#) space<DBSID>.log displays the return code.
- The [detail log](#) s<encoded timestamp>.dbs displays the details.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Altering the Database Instance with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to alter the database instance as follows:

- Switch the redo log file
- Force a database checkpoint
- Set archivelog mode
- Set noarchivelog mode

### Note

This section describes how you alter the database instance.

Altering the database instance is part of [Instance Management with BR\\*Tools](#).

For more information on the approach to database instance management, see [Instance Management](#).

## Prerequisites

Some steps in this procedure only apply if your database is an Oracle Real Application Cluster (RAC). These are marked “RAC only”.

## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:

- BRGUI or BRTOOLS:

1. Choose **Instance Management** **Alter database**

BRGUI or BRTOOLS displays the menu *BRSPACE options for alter database instance*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Alter database action (action)</i>	<a href="#">-f dbalter -a -action</a>
<i>Database instance (instance)</i>	<a href="#">-f dbalter -i -instance</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f dbalter</a> command that is to be executed using the current settings.

3. Choose *Continue*.

BRGUI or BRTOOLS prompts you to start BRSPACE.

4. Choose *Continue* to start BRSPACE.

- Command line:

Enter at least the following command:

```
brspace -f dbalter
```

You can enter more parameters, including the instance name and action, if required. For more information, see [BRSPACE -f dbalter](#).

 Note

Whichever way you start the procedure – with BRGUI or BRTOOLS, or from the command line – you can use quick mode if you know the object name, in this case the instance name. For more information, see [How to Use BR\\*Tools](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#).
3. BRSPACE displays the *Alter database instance main menu*.
4. You can use quick mode as follows:
  - If you have already entered the action, continue with step 4.
  - If you have already entered the instance name and action, continue with step 5.
5. Choose or confirm the required action:
  - *Switch redo log file*
  - *Force database checkpoint*
  - *Set archivelog mode*
  - *Set noarchivelog mode*

If you already chose an action in step 1, this action is set to the default.

RAC only: BRSPACE displays the *List of database instances for alter*, unless you have already entered the instance name.

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Name</i>	Instance name
<i>Number / Thread</i>	Instance number and thread
<i>Status</i>	Instance status
<i>Start time</i>	Time the instance was started
<i>RedoSeq</i>	Log redo sequence number
<i>SapConn</i>	Number of SAP connections

6. RAC only: select a database instance or multiple database instances.  
BRSPACE displays the menu *Options for alter of database instance*.

 Note

In normal (that is, non-RAC) systems, BRSPACE automatically selects the name of the single instance for you.

7. Set the required options:

Menu Entry	Meaning
<i>Current archivelog mode (mode)</i> – display only	<a href="#">-f dbalter -m -mode</a>
<i>Alter database action (action)</i> – display only	<a href="#">-f dbalter -a -action</a>
<i>Force instance shutdown (force)</i>	<a href="#">-f dbalter -f -force</a>
<i>SQLPLUS command (command)</i>	The SQL command that is to be executed using the current settings. For more information, see your Oracle SQL documentation.

8. To start processing with the selected options, choose *Continue*.

## Result

Check the results in the [BRSPACE logs](#).

- The [summary log](#) space<DBSID>.log displays the return code.
- The [detail log](#) s<encoded timestamp>.dba displays the details.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Altering Database Parameters with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to alter the database parameters as follows:

- Alter database parameter
- Reset database parameter
- Create `init<DBSID>.ora` profile from `spfile`

### Note

This section describes how you alter database parameters.

Altering database parameters is part of [Instance Management with BR\\*Tools](#).

For more information on the approach to database instance management, see [Instance Management](#).



## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:

- BRGUI or BRTOOLS:

1. Choose **Instance Management** **Alter database parameters**.

BRGUI or BRTOOLS displays the menu *BRSPACE options for alter database parameter*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Alter parameter action (action)</i>	<a href="#">-f dbparam -a -action</a>
<i>Database parameter (parameter)</i>	<a href="#">-f dbparam -p -parameter</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f dbparam</a> command that is to be executed using the current settings.

3. Choose *Continue*.

BRGUI or BRTOOLS prompts you to start BRSPACE.

4. Choose *Continue* to start BRSPACE.

- Command line:

Enter at least the following command:

```
brspace -f dbparam
```

You can enter more parameters, including the parameter name and action, if required. For more information, see [BRSPACE -f dbparam](#).

 Note

Whichever way you start the procedure - with BRGUI or BRTOOLS, or from the command line - you can use quick mode if you know the object name, in this case the parameter name. For more information, see [How to Use BR\\*Tools](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#).
3. BRSPACE displays the *Alter database parameter main menu*.
4. You can use quick mode as follows:
  - If you have already entered the action, continue with step 4.
  - If you have already entered the parameter name and action, continue with step 7.
5. Choose or confirm the required action:
  - *Change parameter value*
  - *Reset parameter value*
  - *Create init.ora from spfile*
6. If you chose *Create init.ora from spfile*, respond to the BRSPACE prompt asking you whether you want to start the function. You have now finished the procedure. See "Results" below.
7. If you have already entered the parameter name, continue with step 7 (quick mode).

BRSPACE displays the database parameter list:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Parameter</i>	Parameter name
<i>Modif.</i>	Modification attributes: <i>no</i> – unmodifiable <i>yes</i> – modifiable for the lifetime of an instance <i>spfile</i> – modifiable in spfile <i>both</i> – modifiable in spfile and for the lifetime of an instance <i>defer</i> – in subsequent sessions, modifiable in spfile and for the lifetime of an instance
<i>Spfile</i>	Whether parameter is specified in spfile: <i>Yes</i> – parameter is specified

List Entry	Meaning
	<i>no</i> – parameter is not specified <i>inst</i> – parameter is specified with an instance-specific value
<i>Inst.</i>	Database instance the parameter is set for: * means all instances
<i>Deflt.</i>	Whether parameter has default value, <i>yes</i> / <i>no</i>
<i>Value</i>	Current parameter value

8. Select a database parameter.

BRSPACE displays the menu, *Options for alter of database parameter*.

9. Set the required options:

Menu Entry	Meaning
<i>Parameter description (desc)</i> – display only	Parameter description from V\$PARAMETER
<i>Parameter type (type)</i> – display only	Type of parameter: Boolean, string, integer, file
<i>Current parameter value (parval)</i> – display only	Current parameter value
<i>Value in spfile (spval)</i> – display only	Current value in spfile
<i>New parameter value (value)</i>	<a href="#">-f dbparam -v -value</a> This entry is locked if reset parameter value was chosen
<i>Scope for new value (scope)</i>	<a href="#">-f dbparam -s -scope</a>
<i>Database instance (instance)</i>	<a href="#">-f dbparam -i -instance</a>
<i>Comment on update (comment)</i>	<a href="#">-f dbparam -c -comment</a>
<i>SQLPLUS command</i>	The SQL command that is to be executed using the current settings. For more information, see your Oracle SQL

Menu Entry	Meaning
(command)	documentation.

10. To start processing with the selected options, choose *Continue*.

## Result

Check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.dbp` displays the details.
- The [parameter change log](#) `param<DBSID>.log` logs all parameter changes.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Recreating a Database with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to recreate a database.



Note

This section describes how you recreate a database with BR\*Tools.

For more information on the approach to recreating a database, see [Recreate Database](#).

## Prerequisites

- The Oracle database must be Version 9.2 or higher.
- All user tablespaces (that is, tablespaces other than the `SYSTEM` tablespace) must be locally managed.
- The automatic UNDO management must be active with the Oracle parameter `undo_management = auto`.
- The database must contain a default temporary tablespace.
- Perform a [complete database backup](#) before you start the database recreate, offline if possible. If BRSPACE fails, you cannot restart the process and you must therefore reset the entire database (for example, with the BRRECOVER “reset” function)

For more information, see *SAP Note* [748434](#).

## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:
  - BRGUI or BRTOOLS
    1. Choose **Instance Management Recreate database**.

BRGUI or BRTOOLS displays the menu *Recreate database main menu*, where you specify the options with which you call BRSPACE.

BRTOOLS now calls the BRSPACE function *recreate database*.

2. Choose *Specify new database options*.

BRSPACE displays the menu *Main options for recreate of database*.

3. Set the required options:

<b>Menu Entry</b>	<b>Equivalent BRSPACE Command Option</b>
<i>Password for SYS user (pass_sys)</i>	<a href="#">-f dbcreate -pass_sys</a>
<i>Password for SYSTEM user (pass_syst)</i>	<a href="#">-f dbcreate -pass_syst</a>
<i>Max. number of instances (max_inst)</i>	<a href="#">-f dbcreate -max_inst</a>
<i>Max. number of data files (max_data)</i>	<a href="#">-f dbcreate -max_data</a>
<i>Max. number of log groups (max_log)</i>	<a href="#">-f dbcreate -max_log</a>
<i>Max. number of log members (max_memb)</i>	<a href="#">-f dbcreate -max_memb</a>
<i>Max. size of log history (max_hist)</i>	<a href="#">-f dbcreate -max_hist</a>

4. Choose *Continue*.

BRSPACE displays the menu *SYSTEM tablespace options for recreate of database*.

5. Set the required options:

<b>Menu Entry</b>	<b>Equivalent BRSPACE Command Option</b>
<i>SYSTEM tablespace file (syst_file)</i>	<a href="#">-f dbcreate -syst_file</a>
<i>Raw disk / link target (syst_rawlink)</i>	<a href="#">-f dbcreate -syst_rawlink</a>
<i>Size of the file in MB (syst_size)</i>	<a href="#">-f dbcreate -syst_size</a>
<i>File autoextend mode (syst_autoext)</i>	<a href="#">-f dbcreate -syst_autoext</a>
<i>Maximum file size in MB (syst_maxsize)</i>	<a href="#">-f dbcreate -syst_maxsize</a>

Menu Entry	Equivalent BRSPACE Command Option
File increment size in MB (syst_incrsize)	<a href="#">-f dbcreate -syst_incrsize</a>

6. Choose *Continue*.

BRSPACE displays the menu *SYSAUX tablespace options for recreate of database*.

7. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
SYSAUX tablespace file (aux_file)	<a href="#">-f dbcreate -aux_file</a>
Raw disk / link target (aux_rawlink)	<a href="#">-f dbcreate -aux_rawlink</a>
Size of the file in MB (aux_size)	<a href="#">-f dbcreate -aux_size</a>
File autoextend mode (aux_autoext)	<a href="#">-f dbcreate -aux_autoext</a>
Maximum file size in MB (aux_maxsize)	<a href="#">-f dbcreate -aux_maxsize</a>
File increment size in MB (aux_incrsize)	<a href="#">-f dbcreate -aux_incrsize</a>

8. Choose *Continue*.

BRSPACE displays the menu *Temp and undo tablespace options for recreate of database*.

9. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
Temp tablespace name (temp_tsp)	<a href="#">-f dbcreate -temp_tsp</a>
Temp tablespace file (temp_file)	<a href="#">-f dbcreate -temp_file</a>
Raw disk / link target (temp_rawlink)	<a href="#">-f dbcreate -temp_rawlink</a>
Size of the file in MB (temp_size)	<a href="#">-f dbcreate -temp_size</a>
Undo tablespace name (undo_tsp)	<a href="#">-f dbcreate -undo_tsp</a>

Menu Entry	Equivalent BRSPACE Command Option
<i>Undo tablespace file (undo_file)</i>	<a href="#">-f dbcreate -undo_file</a>
<i>Raw disk / link target (undo_rawlink)</i>	<a href="#">-f dbcreate -undo_rawlink</a>
<i>Size of the file in MB (undo_size)</i>	<a href="#">-f dbcreate -undo_size</a>

10. Choose *Continue*.

BRSPACE displays the menu *Online redo log options for log group*.

11. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>Redo log file size in MB (log_size)</i>	<a href="#">-f dbcreate -log_size</a>
<i>Original redo log file (orig_file)</i>	<a href="#">-f dbcreate -orig_file</a>
<i>Raw disk / link target (orig_rawlink)</i>	<a href="#">-f dbcreate -orig_rawlink</a>
<i>Mirror redo log file (mirr_file)</i>	<a href="#">-f dbcreate -mirr_file</a>
<i>Raw disk / link target (mirr_rawlink)</i>	<a href="#">-f dbcreate -mirr_rawlink</a>

12. The above menu appears again for subsequent log groups.

13. Choose *Continue*.

BRSPACE returns to the menu *Recreate database main menu*.

14. Choose *Export user tablespaces*.

BRSPACE exports the user tablespaces and returns to the menu *Recreate database main menu*.

15. Choose *Export global objects*.

BRSPACE exports the global objects and returns to the menu *Recreate database main menu*.

16. Choose *Create new database*.

BRSPACE creates a new empty database and returns to the menu *Recreate database main menu*.

17. Choose *Import global objects*.

BRSPACE imports the global objects and returns to the menu *Recreate database main menu*.

18. Choose *Import user tablespaces*.

BRSPACE imports user tablespaces and returns to the menu *Recreate database main menu*.

- Command line:

Enter at least the following command:

```
brspace -f dbcreate
```

You can enter more parameters if required. For more information, see [BRSPACE -f dbcreate](#).

## Result

The new database instance is now recreated and opened.

Check the results in the [BRSPACE logs](#):

- The [summary log](#) space<DBSID>.log displays the return code.
- The [detail log](#) s<encoded timestamp>.dbc displays the details.



## Showing Instance Status with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to show information about the database instance status.

## Prerequisites

Some steps in this procedure only apply if your database is an Oracle Real Application Cluster (RAC). These are marked “RAC only”.

## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:
  - BRGUI or BRTOOLS:

1. Choose **Instance Management** **Show instance status**.

BRGUI or BRTOOLS displays the menu *BRSPACE main options for showing database information*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-pl-profile</a>
<i>Database user/password (user)</i>	<a href="#">-ul-user</a>
<i>Database instance (instance)</i>	<a href="#">-f dbshow -n -instance</a>



Menu Entry	Equivalent BRSPACE Command Option
Create log file (log)	<a href="#">-f dbshow -l -log</a>
Confirmation mode (confirm)	<a href="#">-c -confirm</a>
Scrolling line count (scroll)	<a href="#">-s -scroll</a>
Message language (language)	<a href="#">-l -language</a>
BRSPACE command line (command)	This shows you the <a href="#">BRSPACE -f dbshow -c dbstate</a> command that is to be executed using the current settings.

3. Choose *Continue*.

BRGUI or BRTOOLS prompts you to start BRSPACE.

4. Choose *Continue* to start BRSPACE.

- o Command line:

Enter at least the following command:

```
brspace -f dbshow -c dbstate
```

You can enter more parameters if required. For more information, see [BRSPACE -f dbshow](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#) if you set the option *Create log file* (parameter `-l|-log`).

3.  Note

4. If you started BRSPACE from the command line without the information class name (`-c|-class`) – that is, with `brspace -f dbshow` – BRSPACE displays the *Show database information main menu*.

5. Choose  *Show database instance information*  *Show instance status* .

- 6.


7. RAC only: if you have already entered the name of the database instance, continue with step 4.

RAC only: BRSPACE displays the *List of database instances*:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Name</i>	Instance name
<i>Number / Thread</i>	Instance number and thread

List Entry	Meaning
<i>Status</i>	Instance status
<i>Start time</i>	Time the instance was started
<i>RedoSeq</i>	Log redo sequence number
<i>SapConn</i>	Number of SAP connections

8. RAC only: to see more information, select one or more database instances.

 Example

These examples apply only to input in character mode.

To select the first three entries in the list, enter 1-3.

To select the first and third entries, enter 1, 3.

To select the first three entries and the fifth, enter 1-3, 5.

To select all entries, enter 0.

BRSPACE displays *Information about the status of database instance*:

List Entry	Meaning
<i>Instance number (number)</i>	Instance number
<i>Instance thread (thread)</i>	Instance thread number
<i>Instance status (status)</i>	Instance status
<i>Instance start time (start)</i>	Time the instance was started
<i>Oracle version (version)</i>	Oracle database version
<i>Database creation time (create)</i>	Date and time of database creation
<i>Last resetlogs time (resetlogs)</i>	Date and time that logs were reset
<i>Archivelog mode (archmode)</i>	Archivelog mode
<i>Archiver status (archiver)</i>	Status of the archiver
<i>Current redo log sequence (redoseq)</i>	Current redo log sequence number
<i>Current redo log SCN (redoscn)</i>	Current redo log system change number
<i>Flashback status (flashback)</i>	Flashback status
<i>Number of SAP connections</i>	Number of SAP connections

List Entry	Meaning
(sapcon)	

- RAC only: if you specified multiple instances, choose *Continue* to scroll through.

## Result

If you set the option *Create log file* (parameter `-l|-log`), check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.dbw` displays the details.

For more information about how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Showing Database Parameters with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to show information about database parameters.

### Procedure

- Start the procedure using BRGUI or BRTOOLS, or from the command line:
  - BRGUI or BRTOOLS:
    - Choose **Instance Management** **Show database parameters**.

BRGUI or BRTOOLS displays the menu *BRSPACE main options for showing database information*, where you specify the options with which you call BRSPACE.

- Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Database parameter (parameter)</i>	<a href="#">-f dbshow -p -parameter</a>
<i>Create log file (log)</i>	<a href="#">-f dbshow -l -log</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>

Menu Entry	Equivalent BRSPACE Command Option
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f dbshow -c dbparam</a> command that is to be executed using the current settings.

3. Choose *Continue* to start BRSPACE.

- o Command line:

Enter at least the following command:

```
brspace -f dbshow -c dbparam
```

You can enter more parameters if required. For more information, see [BRSPACE -f dbshow](#).

- BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#) if you set the option *Create log file* (parameter `-l|-log`).

-  Note

- If you started BRSPACE from the command line without the information class name (`-c|-class`) - that is, with `brspace -f dbshow -` BRSPACE displays the *Show database information main menu*.

- Choose  *Show database instance information*  *Show database parameters* .

- 

- If you have already entered the parameter name, continue with step 4.

BRSPACE displays the *List of database parameters*:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Parameter</i>	Parameter name
<i>Modif.</i>	Modification attributes: <ul style="list-style-type: none"> <li>o <i>no</i>: unmodifiable</li> <li>o <i>yes</i>: modifiable for the lifetime of an instance</li> <li>o <i>sfile</i>: modifiable in sfile</li> <li>o <i>both</i>: modifiable in sfile and for the lifetime of an instance</li> </ul>
<i>Sfile</i>	Whether parameter is specified in sfile: <ul style="list-style-type: none"> <li>o <i>yes</i>: parameter is specified</li> </ul>

List Entry	Meaning
	<ul style="list-style-type: none"> <li>○ <i>no</i>: parameter is not specified</li> <li>○ <i>inst.</i>: parameter is specified with an instance-specific value</li> </ul>
<i>Inst.</i>	Database instance the parameter is set for: “*” means all instances
<i>Deflt.</i>	Whether parameter has default value, <i>yes</i> / <i>no</i>
<i>Value</i>	Current parameter value

8. To see more information, select one or more database parameters.

 Example

These examples apply only to input in character mode.

To select the first three entries in the list, enter 1-3.

To select the first and third entries, enter 1, 3.

To select the first three entries and the fifth, enter 1-3, 5.

To select all entries, enter 0.

BRSPACE displays *Information about database parameter*:

List Entry	Meaning
<i>Parameter description (desc)</i>	Parameter description from V\$PARAMETER
<i>Parameter type (type)</i>	Type of parameter: boolean, string, integer, file
<i>Modifiable attribut (modif)</i>	See <i>Modif.</i> in table above
<i>Defined in spfile (spfile)</i>	Whether parameter is specified in spfile: <ul style="list-style-type: none"> <li>○ <i>yes</i>: parameter is specified</li> <li>○ <i>no</i>: parameter is not specified</li> <li>○ <i>inst.</i>: parameter is specified with an instance-specific value</li> </ul>
<i>Database instance (instance)</i>	See <i>Inst.</i> in table above
<i>Default value (default)</i>	Whether parameter has default value, <i>yes</i> / <i>no</i>
<i>Parameter value (value)</i>	Current parameter value

List Entry	Meaning
<i>Value in spfile (spfval)</i>	Parameter value stored in spfile
<i>Comment on update (comment)</i>	Your comment on the parameter update

- If you specified multiple parameters, choose *Continue* to scroll through.

## Result

If you set the option *Create log file* (parameter `-l|-log`), check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.dbw` displays the details.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Showing Database Owners with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to show information about database owners.

### Procedure

- Start the procedure using BRGUI or BRTOOLS, or from the command line:
  - BRGUI or BRTOOLS:

- Choose **Instance Management** **Show database owners**.

BRGUI or BRTOOLS displays the menu *BRSPACE main options for showing database information*, where you specify the options with which you call BRSPACE.

- Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Database owner (owner)</i>	<a href="#">-f dbshow -o -owner</a>
<i>Create log file (log)</i>	<a href="#">-f dbshow -l -log</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>

Menu Entry	Equivalent BRSPACE Command Option
Scrolling line count (scroll)	<a href="#">-s -scroll</a>
Message language (language)	<a href="#">-l -language</a>
BRSPACE command line (command)	This shows you the <a href="#">BRSPACE -f dbshow -c dbowner</a> command that is to be executed using the current settings.

3. Choose *Continue*

BRGUI or BRTOOLS prompts you to start BRSPACE.





4. Choose *Continue* to start BRSPACE.

- o Command line:

Enter at least the following command:

```
brspace -f dbshow -c dbowner
```

You can enter more parameters if required. For more information, see [BRSPACE -f dbowner](#).


2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#) if you set the option *Create log file* (parameter `-l|-log`).
3.  Note
4. If you started BRSPACE from the command line without the information class name (`-c|-class`) – that is, with `brspace -f dbshow` – BRSPACE displays the *Show database information main menu*.
5. Choose  *Show database instance information*  *Show database owners* .
- 6.
7. If you have already entered the owner name, continue with step 4.

BRSPACE displays the *List of database owners*:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Owner</i>	Owner name
<i>Id.</i>	Owner ID
<i>Created</i>	Date that the owner was created
<i>Deft-Tsp.</i>	Default tablespace for owner
<i>Temp-</i>	Temporary tablespace for

List Entry	Meaning
<i>Tsp.</i>	owner
<i>Status</i>	Status of owner

8. To see more information, select one or more database owners.

 Example

These examples apply only to input in character mode.

To select the first three entries in the list, enter 1-3.

To select the first and third entries, enter 1, 3.

To select the first three entries and the fifth, enter 1-3, 5.

To select all entries, enter 0.

BRSPACE displays *Information about database parameter*:

List Entry	Meaning
<i>Owner Id (owner_id)</i>	Database owner ID
<i>Created date (created)</i>	Date that the owner was created
<i>Default tablespace (deft_tsp)</i>	Default tablespace for owner
<i>Temporary tablespace (temp_tsp)</i>	Temporary tablespace for owner
<i>Owner status (status)</i>	Status of owner

9. If you specified multiple owners, choose *Continue* to scroll through.

## Result

If you set the option *Create log file* (parameter `-l|-log`), check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.dbw` displays the details.

For more information on how to view the logs with BR\*Tools, see [Showing Profiles and Logs with BR\\*Tools](#).



## Space Management with BR\*Tools

You can manage the space on your Oracle database with [BR\\*Tools](#)



## Note

This section describes how you manage space with BR\*Tools.

For more information on the approach to manage space, see [Space Management](#).

## Integration

- BRGUI or BRTOOLS calls the SAP tool BRSPACE. You can also perform space management by calling BRSPACE from the command line. However, unless you choose batch mode with the option `-c force`, BRSPACE displays menus to help you enter the required options.

### Recommendation

We recommend you to normally use BRGUI or BRTOOLS rather than BRSPACE directly. This is because the menus in BRGUI or BRTOOLS simplify entry of the correct options.




- If required, set parameters to control the screen display for BRSPACE in the [Initialization Profile init<DBSID>.sap](#). These are [scroll\\_lines](#) and [show\\_period](#).

## Features

You can perform the following functions for space management with BR\*Tools:

- Extend tablespace
- Create tablespace
- Drop tablespace
- Alter tablespace
- Alter data file
- Move data file

## Activities

1. You choose *Space management* in the BRGUI or BRTOOLS menus or directly from the command line. You can use quick mode if you know which objects you require. For more information on quick mode, see [BR\\*Tools User Interface](#)
2. If required, you change the default values for the parameters in the initialization profile `init<DBSID>.sap` and restart BRGUI or BRTOOLS.
3. If required, you choose  *Space Management*  *Reset program status*  to set the defaults used to the values set after BRGUI or BRTOOLS initialization.
4. You start space management.
5. You check the results of space management in the [BRSPACE logs](#).



## Extending a Tablespace with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to extend a tablespace by adding a data file.

### Note

This section describes how you extend a tablespace with BR\*Tools.

For more information on the approach to extending a tablespace, see [Managing Tablespaces](#).

## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:

o BRGUI or BRTOOLS:

1. Choose **Space Management** **Extend tablespace**

BRGUI or BRTOOLS displays the menu *BRSPACE options for tablespace extension*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Tablespace name (tablespace)</i>	<a href="#">-f tsextend -t -tablespace</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f tsextend</a> command that is to be executed using the current settings.

3. Choose *Continue*.

BRGUI or BRTOOLS prompts you to start BRSPACE.

4. Choose *Continue* to start BRSPACE.

o Command line

You need to enter at least the following command:

```
brspace -f tsextend
```

You can enter more parameters, including the tablespace name, if required. For more information, see [BRSPACE -f tsextend](#).

 Note

Whichever way you start the procedure - with BRGUI or BRTOOLS, or from the command line - you can take advantage of quick mode if you know the object name, in this case the tablespace name. For more information, see [How to Use BR\\*Tools](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#).
3. If you have already entered the tablespace name, continue with step 5 (quick mode).

BRSPACE displays the *Tablespace extension main menu*.

4. Choose *Extend tablespace*.

BRSPACE displays the list of tablespaces:

List Entry	Meaning
<i>Pos</i>	List sequence number
<i>Tablespace</i>	Tablespace name
<i>Files/AuExt</i>	Number of data files / number of data files with autoextend
<i>Total [KB]</i>	Total size of all data files of the tablespace
<i>Used[%]</i>	Percentage of space in tablespace that is used
<i>Free [KB]</i>	Amount of space in tablespace that is free
<i>MaxSize [KB]</i>	Maximum size that the tablespace can be autoextended

5. Select a tablespace.

BRSPACE displays the menu *Options for extension of tablespace*.

6. Set the required options:

Menu Entry	Meaning
<i>Last added file name (lastfile)</i> – display only	The name of the last file that was added to the tablespace
<i>Last added file size in MB (lastsize)</i>	The size of the last file that was added to the tablespace

Menu Entry	Meaning
– display only	
New file to be added (file)	<a href="#">-f tsextend -f -file</a>
Raw disk / link target (rawlink)	<a href="#">-f tsextend -r -rawlink</a>
Size of the new file in MB (size)	<a href="#">-f tsextend -s -size</a>
File autoextend mode (autoextend)	<a href="#">-f tsextend -a -autoextend</a>
Maximum file size in MB (maxsize)	<a href="#">-f tsextend -m -maxsize</a>
File increment size in MB (incsize)	<a href="#">-f tsextend -i -incsize</a>
SQL command (command)	The SQL command that is to be executed using the current settings. For more information, see your Oracle SQL documentation.

- If required, repeat the previous step when prompted by BRSPACE with message BR10911.

You can add up to five files to the selected tablespace by repeating the previous step.

 Note

For the *file* option, you can enter not only full path names, but also the `sapdata` path name, `sapdata<N>`, or just `<N>` to specify in which `sapdata` directory the file is to be located. BRSPACE automatically extends the incomplete file name to the full path name according to the [SAP naming convention](#).

- To start processing with the selected options, choose *Continue*.

## Result

Check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.tse` displays the details.
- The [structure change log](#) `struc<DBSID>.log` logs all structure changes.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Creating a Tablespace with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to create a new tablespace.

### Note

This section describes how you create a tablespace with BR\*Tools.

For more information on the approach to creating a tablespace, see [Managing Tablespaces](#).

## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:

○ BRGUI or BRTOOLS:

1. Choose **Space Management** **Create tablespace**.

BRGUI or BRTOOLS displays the menu *BRSPACE options for tablespace extension*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Tablespace name (tablespace)</i>	<a href="#">-f tscreate -t -tablespace</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f tscreate</a> command that is to be executed using the current settings.

3. Choose *Continue*.

BRGUI or BRTOOLS prompts you to start BRSPACE.


4. Choose *Continue* to start BRSPACE.

○ Command line


You need to enter at least the following command:

```
brspace -f tscreate
```

You can enter more parameters, including the tablespace name, if required. For more information, see [BRSPACE -f tscreate](#).

2.  Note
3. Whichever way you start the procedure – with BRGUI or BRTOOLS, or from the command line – you can take advantage of quick mode if you know the object name, in this case the tablespace name. For more information, see [How to Use BR\\*Tools](#).
- 4.
5. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#).
6. If you have already entered the tablespace name, continue with step 4 (quick mode).  
BRSPACE displays the *Tablespace creation main menu*.
7. Choose *Create tablespace*.  
BRSPACE displays the menu *Main options for creation of tablespace*.
8. Set the required main options:

Menu Entry	Meaning
<i>Tablespace name (tablespace)</i>	<a href="#">-f tscreate -t -tablespace</a>
<i>Tablespace contents (contents)</i>	<a href="#">-f tscreate -c -contents</a>
<i>Segment space management (space)</i>	<a href="#">-f tscreate -s -space</a>
<i>SAP owner of tablespace (owner)</i>	<a href="#">-f tscreate -o -owner</a>
<i>Table data class / tabart (class)</i>	<a href="#">-f tscreate -l -class</a>
<i>Data type in tablespace (data)</i>	<a href="#">-f tscreate -d -data</a>
<i>Joined index/table tablespace (join)</i>	<a href="#">-f tscreate -j -join</a>
<i>Uniform size in MB (uniform)</i>	<a href="#">-f tscreate -u -uniform</a>

9.  Note
10. For more information on naming the new tablespace, see [SAP Naming Conventions for Tablespaces and Data Files](#).
- 11.
12. Choose *Continue*.  
BRSPACE displays the menu *Space options for creation of tablespace*.
13. Set the required options:

Menu Entry	Meaning
Tablespace file name (file)	Unless you specified it at the command line, BRSPACE automatically enters the file name according to the SAP naming convention.  For more information, see <a href="#">-f tscreate -f -file</a> .
Raw disk / link target (rawlink)	<a href="#">-f tscreate -r -rawlink</a>
File size in MB (size)	<a href="#">-f tscreate -s -size</a>
File autoextend mode (autoextend)	<a href="#">-f tscreate -a -autoextend</a>
Maximum file size in MB (maxsize)	<a href="#">-f tscreate -m -maxsize</a>
File increment size in MB (incsize)	<a href="#">-f tscreate -i -incsize</a>
SQL command (command)	The SQL command that is to be executed using the current settings. For more information, see your Oracle SQL documentation.

14. If required, repeat the previous step when prompted by BRSPACE with message BR10911.

You can add up to five files to the selected tablespace by repeating the previous step.

 Note

For the *file* option, you can enter not only full path names, but also the `sapdata` path name, `sapdata<N>`, or just `<N>` to specify in which `sapdata` directory the file is to be located. BRSPACE automatically extends the incomplete file name to the full path name according to the [SAP naming convention](#).

If *Data type in tablespace* is *table*, BRSPACE displays the menu *Space options for creation of index tablespace*.

15. If *Data type in tablespace* is *table*, set the required options for the joined index tablespace:

Menu Entry	Meaning
Index tablespace file name (xfile)	<a href="#">-f tscreate -xf -xfile</a>
Raw disk / link target (xrawlink)	<a href="#">-f tscreate -xr -xrawlink</a>
File size in MB (xsize)	<a href="#">-f tscreate -xs -xsize</a>
File autoextend mode (xautoextend)	<a href="#">-f tscreate -xa -xautoextend</a>

Menu Entry	Meaning
Maximum file size in MB ( <i>xmaxsize</i> )	<a href="#">-f tscreate -xm -xmaxsize</a>
File increment size in MB ( <i>xincrsize</i> )	<a href="#">-f tscreate -xj -xincrsize</a>
SQL command ( <i>command</i> )	The SQL command that is to be executed using the current settings. For more information, see your Oracle SQL documentation.

16. If required, repeat the previous step when prompted by BRSPACE with message BR1091I.

You can add up to five files to the selected tablespace by repeating the previous



For the *xfile* option, you can enter not only full path names, but also the `sapdata` path name, `sapdata<N>`, or just `<N>` to specify in which `sapdata` directory the file is to be located. BRSPACE automatically extends the incomplete file name to the full path name according to the [SAP naming convention](#).

17. To start processing with the selected options, choose *Continue*.



BRSPACE creates the table tablespace in step 6 and index tablespaces in this step.

## Result

Check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.tsc` displays the details.
- The [structure change log](#) `struc<DBSID>.log` logs all structure changes.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Dropping a Tablespace with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to drop a tablespace.



This section describes how you drop a tablespace with BR\*Tools.

For more information on the approach to dropping a tablespace, see [Managing Tablespaces](#).



## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:

- BRGUI or BRTOOLS:

1. Choose **Space Management** **Drop tablespace**.

BRGUI or BRTOOLS displays the menu *BRSPACE options for drop tablespace*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Tablespace name (tablespace)</i>	<a href="#">-f tsdrop -t -tablespace</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f tsdrop</a> command that is to be executed using the current settings.

3. Choose *Continue*.

BRGUI or BRTOOLS prompts you to start BRSPACE.

4. Choose *Continue* to start BRSPACE.

- Command line

You need to enter at least the following command:

```
brspace -f tsdrop
```

You can enter more parameters, including the tablespace name, if required. For more information, see [BRSPACE -f tsdrop](#).

 **Note**

Whichever way you start the procedure - with BRGUI or BRTOOLS, or from the command line - you can take advantage of quick mode if you know the

object name, in this case the tablespace name. For more information, see [How to Use BR\\*Tools](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#).
3. If you have already entered the tablespace name, continue with step 5 (quick mode).  
BRSPACE displays the *Tablespace drop main menu*.

4. Choose *Drop tablespace*.

BRSPACE displays the list of tablespaces:

List Entry	Meaning
<i>Pos</i>	List sequence number
<i>Tablespace</i>	Tablespace name
<i>Files/AuExt</i>	Number of data files / number of data files with autoextend
<i>Total [KB]</i>	Total size of all data files of the tablespace
<i>Used[%]</i>	Percentage of space in tablespace that is used
<i>Free [KB]</i>	Amount of space in tablespace that is free
<i>MaxSize [KB]</i>	Maximal size that the tablespace can be autoextended

5. Select a tablespace.

BRSPACE displays the menu *Options for dropping of tablespace*.

6. Set the required options:

Menu Entry	Meaning
<i>Number of files in tablespace (files)</i> – display only	Number of files in the tablespace
<i>Total tablespace size in MB (size)</i> – display only	Total tablespace size
<i>Force tablespace drop (force)</i>	<a href="#">-f tsdrop -f]-force</a>
<i>SQL command (command)</i>	The SQL command that is to be executed using the current settings. For more information, see your Oracle SQL documentation.

7. To start processing with the selected options, choose *Continue*.

 Note

The selected tablespace is only deleted immediately if it is empty (that is, it does not contain any segments).

If the table is not empty, you have to use the option `force` to immediately delete the tablespace.

 Caution

Do *not* delete objects that are still used by the SAP system.

## Result

Check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.tsd` displays the details.
- The [structure change log](#) `struc<DBSID>.log` logs all structure changes.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Altering a Tablespace with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to alter a tablespace as follows:

- Set tablespace online
- Set tablespace offline
- Set backup status
- Reset backup status
- Coalesce free extents
- Rename tablespace (Oracle 10g or higher)

 Note

This section describes how you alter a tablespace with BR\*Tools.

For more information on the approach to altering a tablespace, see [Managing Tablespaces](#).

## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:
  - BRGUI or BRTOOLS:

1. Choose  *Space Management Alter tablespace*. 

BRGUI or BRTOOLS displays the menu *BRSPACE options for alter tablespace*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Alter tablespace action (action)</i>	<a href="#">-f tsalter -a -action</a>
<i>Tablespace names (tablespace)</i>	<a href="#">-f tsalter -t -tablespace</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f tsalter</a> command that is to be executed using the current settings.

3.  Note

4. If required, in *Tablespace names* you can enter the names of multiple tablespaces, but you cannot use wildcards. But this entry and alter action entry are optional.

5. For example, PSAPRAWI,PSAPRAWD is a valid entry, but PSAPRAW\* is not a valid entry.

6.

7. Choose *Continue*.

BRGUI or BRTOOLS prompts you to start BRSPACE.

8. Choose *Continue* to start BRSPACE.

- o Command line:

Enter at least the following command:

```
brspace -f tsalter
```

You can enter more parameters, including the tablespace name and action, if required. For more information, see [BRSPACE -f tsalter](#).

 Note

Whichever way you start the procedure - with BRGUI or BRTOOLS, or from the command line - you can use quick mode if you know the object name, in this case the tablespace name. For more information, see [How to Use BR\\*Tools](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#).
3. BRSPACE displays the *Alter tablespace main menu*.
4. You can use quick mode as follows:
  - If you have already entered the action, continue with step 4.
  - If you have already entered the tablespace name and action, continue with step 6.

 Note

If you have entered multiple tablespaces, BRSPACE displays as confirmation a *List of tablespaces for alter*. You cannot make a selection from this list. If required, go back and make a new selection.

Continue with step 6 (quick mode).

5. Choose or confirm the required action:
  - Set tablespace online
  - Set tablespace offline
  - Set backup status
  - Reset backup status
  - Coalesce free extents
  - Rename tablespace
6. If you have already entered the tablespace name, continue with step 6 (quick mode).

BRSPACE displays the tablespace list:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Tablespace</i>	Tablespace name
<i>Files/AuExt</i>	Number of data files / number of data files with autoextend set
<i>Type</i>	Tablespace type: data, temp or undo

List Entry	Meaning
<i>Status</i>	Tablespace status – online or offline
<i>SegMan.</i>	Segment space management – auto or manual
<i>ExtMan.</i>	Tablespace extent management – local or dictionary
<i>Backup</i>	Tablespace backup status

 Note

BRSPACE only displays the tablespaces that can be processed by your chosen action.

For example, if you choose *Set tablespace online*, only the tablespaces that are currently offline are displayed.

7. Select a tablespace or multiple tablespaces.

 Example

These examples apply only to input in character mode.

To select the first three tablespaces in the list, enter 1-3.

To select the first and third tablespaces, enter 1, 3.

To select the first three tablespaces and the fifth, enter 1-3, 5.

To select all entries, enter 0.

BRSPACE displays the menu, *Options for alter of tablespace*.

8. Set the required options:

Menu Entry	Meaning
<i>Current tablespace status (status)</i> – display only	Current status of the tablespace
<i>Current backup status (backup)</i> – display only	Current backup status of the tablespace
<i>Alter tablespace action (action)</i> – display only	The action that you selected above
<i>Set offline mode (mode)</i>	<a href="#">-f tsalter -m -mode</a>

Menu Entry	Meaning
<i>New tablespace name (name)</i>	<a href="#">-f tsalter -n -name</a>
<i>Force alter tablespace (force)</i>	<a href="#">-f tsalter -f -force</a>
<i>SQLPLUS command line (command)</i>	The SQL command that is to be executed using the current settings. For more information, see your Oracle SQL documentation.

- To start processing with the selected options, choose *Continue*.

## Result

Check the results in the [BRSPACE logs](#).

- The [summary log](#) space<DBSID>.log displays the return code.
- The [detail log](#) s<encoded timestamp>.tsa displays the details.
- The [structure change log](#) struc<DBSID>.log records all structure changes (rename tablespace)

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Altering a Data File with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to alter a data file as follows:

- Set data file online
- Set data file offline
- Switch on and maintain autoextend
- Switch off autoextend
- Resize data file
- Rename data file
- Drop empty data file

### Note

This section describes how you alter a data file with BR\*Tools.

For more information about the approach to altering a data file, see [Managing Data Files](#)

## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:

○ BRGUI or BRTOOLS:

1. Choose **Space Management** **Alter data file**.

BRGUI or BRTOOLS displays the menu *BRSPACE options for alter data file*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Alter data file action (action)</i>	<a href="#">-f dfalter -a -action</a>
<i>Tablespace names (tablespace)</i>	<a href="#">-f dfalter -t -tablespace</a>
<i>Data file names (file)</i>	<a href="#">-f dfalter -f -file</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f dfalter</a> command that is to be executed using the current settings.

3. Choose *Continue*.

BRGUI or BRTOOLS prompts you to start BRSPACE.

4. Choose *Continue* to start BRSPACE.

### Note

If required, in *Data file names* you can enter the names of multiple data files, but you cannot use wildcards. But this entry and alter action entry are optional.

For example, `/oracle/GC2/sapdata3/users_1/users.data1`,  
`/oracle/GC2/sapdata4/xdb/xdb.data1` is a valid entry, but  
`/oracle/GC2/sapdata*` is not a valid entry.



You can also enter Oracle file IDs instead of path names.

- Command line:

Enter at least the following command:

```
brspace -f dfalter
```

You can enter more parameters, including the data file name and action, if required. For more information, see [BRSPACE -f dfalter](#).

2.  Note

3. Whichever way you start the procedure – with BRGUI or BRTOOLS, or from the command line – you can use quick mode if you know the object name, in this case the data file name. For more information, see [How to Use BR\\*Tools](#).

4.

5. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#).

6. BRSPACE displays the *Alter data file main menu*.

7. You can use quick mode as follows:

- If you have already entered the action, continue with step 4.
- If you have already entered the data file name and action, continue with step 6.

 Note

If you have entered multiple data files, BRSPACE displays as confirmation a *List of data files for alter*. You cannot make a selection from this list. If required, go back and make a new selection.

Continue with step 6 (quick mode).

8. Choose or confirm the required action:

- Set data file online
- Set data file offline
- Maintain autoextend
- Switch off autoextend
- Resize data file
- Rename data file
- Drop empty data file

9. If you have already entered the data file name, continue with step 6 (quick mode).

BRSPACE displays the data file list:

List Entry	Meaning
------------	---------

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Tablespace</i>	Tablespace name for the data file
<i>Status</i>	Data file status - online, offline, system or recover
<i>Type</i>	Data file type - regular or raw disk
<i>Size</i>	Data file size
<i>AuExt.</i>	Autoextend setting
<i>File</i>	File name

 Note

BRSPACE only displays the data files that can be processed by your chosen action.

For example, if you choose *Set data file online*, only the data files that are currently offline are displayed.

10. Select a data file or multiple data files.

 Example

These examples apply only to input in character mode.

To select the first three data files in the list, enter 1-3.

To select the first and third data files, enter 1, 3.

To select the first three data files and the fifth, enter 1-3, 5.

To select all entries, enter 0.

BRSPACE displays the menu *Options for alter of data file*.

11. Set the required options:

Menu Entry	Meaning
<i>Current data file status (status)</i> - display only	Current status of the data file
<i>Current datafile size in MB (currsize)</i> - display only	Current size of the data file

Menu Entry	Meaning
Alter data file action (action)  - display only	The action that you selected above
Force offline mode (force)	<a href="#">-f dfalter -f -force</a>
Maximal file size in MB (maxsize)	<a href="#">-f dfalter -m -maxsize</a>
File increment size in MB (incsize)	<a href="#">-f dfalter -i -incsize</a>
New data file size in MB (size)	<a href="#">-f dfalter -s -size</a>
New data file name (name)	<a href="#">-f dfalter -n -name</a>
SQL command (command)	The SQL command that is to be executed using the current settings. For more information, see your Oracle SQL documentation.

12. To start processing with the selected options, choose *Continue*.

## Result

Check the results in the [BRSPACE logs](#).

- The [summary log](#) space<DBSID>.log displays the return code.
- The [detail log](#) s<encoded timestamp>.dfa displays the details.
- The [structure change log](#) struc<DBSID>.log logs all structure changes.

For more information about how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Moving a Data File with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to move a data file.



Note

This section describes how you move a data file with BR\*Tools.

For more information on the approach to moving a data file, see [Managing Data Files](#)

## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:

- BRGUI or BRTOOLS:

1. Choose **Space Management** **Move data file**.

BRGUI or BRTOOLS displays the menu *BRSPACE options for move data file*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Tablespace names (tablespace)</i>	<a href="#">-f dfmove -t -tablespace</a>
<i>Data file names (file)</i>	<a href="#">-f dfmove -f -file</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f dfmove</a> command that is to be executed using the current settings.

3.  Note

4. If required, in *Data file names* you can enter the names of multiple data files, but you cannot use wildcards. But this entry is optional.
5. For example, `/oracle/GC2/sapdata3/users_1/users.data1`, `/oracle/GC2/sapdata4/xdb/xdb.data1` is a valid entry, but `/oracle/GC2/sapdata*` is not a valid entry.
6. You can also enter Oracle file IDs instead of path names.
- 7.
8. Choose *Continue*.  
BRGUI or BRTOOLS prompts you to start BRSPACE.
9. Choose *Continue* to start BRSPACE.

- Command line:

Enter at least the following command:

```
brspace -f dfmove
```

You can enter more parameters, including the data file, if required. For more information, see [BRSPACE -f dfmove](#).

 Note

Whichever way you start the procedure - with BRGUI or BRTOOLS, or from the command line - you can use quick mode if you know the object name, in this case the data file name. For more information, see [How to Use BR\\*Tools](#).

- BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#).
- BRSPACE displays the *Move data file main menu*.
- If you have already entered the data file name, continue with step 5 (quick mode).

 Note

If you have entered multiple data files, BRSPACE displays as confirmation a *List of data files for moving*. You cannot make a selection from this list. If required, go back and make a new selection.

Continue with step 5 (quick mode).

- Choose *Move data file*.

BRSPACE displays the data file list:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Tablespace</i>	Tablespace name for the data file
<i>Status</i>	Data file status - online, offline, system, or recover
<i>Type</i>	Data file type - regular or raw disk
<i>Size</i>	Data file size
<i>File</i>	File name

- Select a data file or multiple data files.

 Example

These examples apply only to input in character mode.

To select the first three data files in the list, enter 1-3.

To select the first and third data files, enter 1, 3.

To select the first three data files and the fifth, enter 1-3, 5.

To select all entries, enter 0.

BRSPACE displays the menu, *Options for moving data file*.

7. Set the required options:

Menu Entry	Meaning
<i>Current datafile size in MB (currsize)</i> – display only	Current data file size
<i>Current link target (currlink)</i> – display only	Destination of link to a raw disk or to a non-sapdata directory, if any
<i>Move destination (destination)</i>	<a href="#">-f dfmove -d -destination</a>
<i>Raw disk / link target (rawlink)</i>	<a href="#">-f dfmove -r -rawlink</a>
<i>Parallel copy processes (parallel)</i>	<a href="#">-f dfmove -p -paralle</a>
<i>Force instance shutdown (force)</i>	<a href="#">-f dfmove -f -force</a>
<i>SQL command (command)</i>	The SQL command that is to be executed using the current settings. For more information, see your Oracle SQL documentation.

8. To start processing with the selected options, choose *Continue*.

## Result

Check the results in the [BRSPACE logs](#).

- The [summary log](#) space<DBSID>.log displays the return code.
- The [detail log](#) s<encoded timestamp>.dfm displays the details.
- The [structure change log](#) struc<DBSID>.log logs all structure changes.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Showing Tablespaces with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to show information about tablespaces.

## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:

- o BRGUI or BRTOOLS:

1. Choose **► Space management ► Additional space functions ► Show tablespaces ◀**.

BRGUI or BRTOOLS displays the menu *BRSPACE main options for showing database information*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Database tablespace (tablespace)</i>	<a href="#">-f dbshow -t -tablespace</a>
<i>Create log file (log)</i>	<a href="#">-f dbshow -l -log</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f dbshow -c tsinfo</a> command that is to be executed using the current settings.

3. Choose *Continue*

BRGUI or BRTOOLS prompts you to start BRSPACE.

4. Choose *Continue* to start BRSPACE.





- o Command line:

Enter at least the following command:

```
brspace -f dbshow -c tsinfo
```

You can enter more parameters if required. For more information, see [BRSPACE -f dbshow](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#) if you set the option *Create log file* (parameter `-l|-log`).

3.  Note
4. If you started BRSPACE from the command line without the information class name (-c|-class) – that is, with `brspace -f dbshow` – BRSPACE displays the *Show database information main menu*.
5. Choose  *Show database space information*  *Show tablespaces* .
- 6.
7. If you have already entered the tablespace name, continue with step 4.

BRSPACE displays the *List of database tablespaces*:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Tablespace</i>	Tablespace name
<i>Type</i>	Tablespace type: data, temp or undo
<i>Status</i>	Tablespace status: online or offline
<i>ExtMan.</i>	Tablespace extent management: local or dictionary
<i>SegMan.</i>	Segment space management: auto or manual
<i>Backup</i>	Tablespace backup status
<i>Files/AuExt.</i>	Number of data files / number of data files with autoextend set
<i>Total[KB]</i>	Total tablespace size
<i>Used[%]</i>	Used space in tablespace
<i>Free[KB]</i>	Free space in tablespace
<i>MaxSize[KB]</i> <i>]</i>	Maximum size the tablespace can be extended to
<i>FreeExt.</i>	Number of free extents in tablespace
<i>Largest[KB]</i>	Size of largest free extents in tablespace

8. To see more information, select one or more tablespaces.

#### Example

These examples apply to input in character mode.

To select the first three entries in the list, enter 1-3.

To select the first and third entries, enter 1, 3.

To select the first three entries and the fifth, enter 1-3, 5.

To select all entries, enter 0.



BRSPACE displays *Information about tablespace*:

List Entry	Meaning
<i>Tablespace type (type)</i>	Tablespace type: data, temp or undo
<i>Tablespace status (status)</i>	Tablespace status: online or offline
<i>Extent management (extent)</i>	Tablespace extent management - local or dictionary
<i>Segment space management (space)</i>	Segment space management - auto or manual
<i>Backup status (backup)</i>	Tablespace backup status
<i>Number of files in tablespace (files)</i>	Number of data files in tablespace
<i>Number of autoextensible files (autoext)</i>	Number of data files with autoextend set
<i>Total tablespace size in KB (total)</i>	Total tablespace size
<i>Used space in tablespace in % (used)</i>	Used space in tablespace
<i>Free space in tablespace in KB (free)</i>	Free size in tablespace
<i>Maximum tablespace size in KB (maxsize)</i>	Maximum size of the tablespace
<i>Allocated disk space in KB (space)</i>	Disk space allocated to the tablespace
<i>Maximum extension size in KB (extsize)</i>	Maximum size the tablespace can be extended to
<i>Number of free extents (freext)</i>	Number of free extents in tablespace
<i>Largest free extents (largest)</i>	Size of largest extents in tablespace in KB

9. If you specified multiple tablespaces, choose *Continue* to scroll through.

## Result

If you set the option *Create log file* (parameter `-l|-log`), check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.dbw` displays the details.

For more information about how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Showing Data Files with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to show information about data files.

### Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:
  - o BRGUI or BRTOOLS:

1. Choose **► Space management Additional space functions ► Show data files ◀**.

BRGUI or BRTOOLS displays the menu *BRSPACE main options for showing database information*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Database tablespace (tablespace)</i>	<a href="#">-f dbshow -s -tablespace</a>
<i>Database file (file)</i>	<a href="#">-f dbshow -f -file</a>
<i>Create log file (log)</i>	<a href="#">-f dbshow -l -log</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f dbshow -c dfinfo</a> command that is to be executed using the current settings.

3. Choose *Continue*

BRGUI or BRTOOLS prompts you to start BRSPACE.





4. Choose *Continue* to start BRSPACE.

- Command line:

Enter at least the following command:

```
brspace -f dbshow -c dfinfo
```

You can enter more parameters if required. For more information, see [BRSPACE -f dbshow](#).

- BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#) if you set the option *Create log file* (parameter `-l|-log`).
-  Note
- If you started BRSPACE from the command line without the information class name (`-c|-class`) – that is, with `brspace -f dbshow` – BRSPACE displays the *Show database information main menu*.
- Choose  *Show database space information*  *Show data files* .
- 
- If you have already entered the data file name, continue with step 4.

BRSPACE displays the *List of database files*:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Tablespace</i>	Tablespace name for the data file
<i>Status</i>	Data file status: online, offline, system or recover
<i>Type</i>	Data file type: raw device or file system
<i>File</i>	Data file name
<i>Id.</i>	Data file ID
<i>Size[KB]</i>	Data file size
<i>Device</i>	Disk device ID
<i>Back.</i>	Backup status
<i>AuExt.</i>	Whether data file has autoextend set
<i>MaxSize[KB]</i> ]	Maximum size of the data file
<i>IncrSize[KB]</i>	Data file increment size

- To see more information, select one or more data files.

 Example

These examples apply only to input in character mode.

To select the first three entries in the list, enter 1-3.

To select the first and third entries, enter 1, 3.

To select the first three entries and the fifth, enter 1-3, 5.

To select all entries, enter 0.

BRSPACE displays *Information about data file*:

List Entry	Meaning
<i>Data file Id (Id)</i>	Data file ID
<i>Data file status (status)</i>	Data file status: online, offline, system or recover
<i>Tablespace name (tablespace)</i>	Tablespace name for the data file
<i>Backup status (backup)</i>	Backup status
<i>Data file type (type)</i>	Data file type: raw device or file system
<i>Disk device Id (device)</i>	Disk device ID
<i>Data file size in KB (size)</i>	Data file size
<i>File autoextend mode (autoextend)</i>	Whether data file has autoextend set
<i>Maximum file size in KB (maxsize)</i>	Maximum size of the data file
<i>File increment size in KB (incrsz)</i>	Data file increment size
<i>Allocated disk space in KB (space)</i>	Disk space allocated to the data file
<i>Data file creation time (ctime)</i>	Creation time stamp of the data file
<i>Data file creation SCN (crscn)</i>	Creation system change number (SCN) of the data file

- If you specified multiple data files, choose *Continue* to scroll through.

## Result

If you set the option *Create log file* (parameter `-1|-log`), check the results in the [BRSPACE logs](#).

- The [summary log](#) space<DBSID>.log displays the return code.
- The [detail log](#) s<encoded timestamp>.dbw displays the details.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#)



## Showing Redo Log Files with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to show information about online redo log files.

### Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:

- BRGUI or BRTOOLS:

1. Choose **Space management** **Additional space functions** **Show redolog files**:

BRGUI or BRTOOLS displays the menu *BRSPACE main options for showing database information*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Database file (file)</i>	<a href="#">-f dbshow -f -file</a>
<i>Create log file (log)</i>	<a href="#">-f dbshow -l -log</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f dbshow -c rfinfo</a> command that is to be executed using the current settings.

3. Choose *Continue*

BRGUI or BRTOOLS prompts you to start BRSPACE.





4. Choose *Continue* to start BRSPACE.

- o Command line:

Enter at least the following command:

```
brspace -f dbshow -c rfinfo
```

You can enter more parameters if required. For more information, see [BRSPACE -f dbshow](#).

- BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#) if you set the option *Create log file* (parameter `-l|-log`).
-  Note
- If you started BRSPACE from the command line without the information class name (`-c|-class`) – that is, with `brspace -f dbshow` – BRSPACE displays the *Show database information main menu*.
- Choose  *Show database space information*  *Show redolog files* .
- 
- If you have already entered the redo log file name, continue with step 4.

BRSPACE displays the *List of database online redolog files*:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Status</i>	Redo log file status
<i>Group</i>	Redo log group number
<i>Thrd</i>	Redo log thread number
<i>Size[KB]</i>	Redo log file size
<i>Device</i>	Disk device ID for the redo log file
<i>Type</i>	Redo log file type: raw device or file system
<i>File</i>	Redo log file name

- To see more information, select one or more redo log files.

 Example

These examples apply only to input in character mode.

To select the first three entries in the list, enter 1-3.

To select the first and third entries, enter 1, 3.

To select the first three entries and the fifth, enter 1-3, 5.

To select all entries, enter 0.

BRSPACE displays *Information about online redo log file*:

List Entry	Meaning
<i>Redolog group number (group)</i>	Redo log group number
<i>Redolog thread number (thread)</i>	Redo log thread number
<i>Redolog file status (status)</i>	Redo log file status
<i>Redolog file type (type)</i>	Redo log file type: raw device or file system
<i>Disk device Id (device)</i>	Disk device ID for the redo log file
<i>Redolog file size in KB (size)</i>	Redo log file size

9. If you specified multiple redo log files, choose *Continue* to scroll through.

## Result

If you set the option *Create log file* (parameter `-l | -log`), check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.dbw` displays the details.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#)



## Showing Control Files with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to show information about control files.

## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:
  - BRGUI or BRTOOLS:

1. Choose **Space management** **Additional space functions** **Show control files**.

BRGUI or BRTOOLS displays the menu *BRSPACE main options for showing database information*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile</i>	<code>-pl-profile</code>

Menu Entry	Equivalent BRSPACE Command Option
(profile)	
Database user/password (user)	<a href="#">-u -user</a>
Database file (file)	<a href="#">-f dbshow -f -file</a>
Create log file (log)	<a href="#">-f dbshow -l -log</a>
Confirmation mode (confirm)	<a href="#">-c -confirm</a>
Scrolling line count (scroll)	<a href="#">-s -scroll</a>
Message language (language)	<a href="#">-l -language</a>
BRSPACE command line (command)	This shows you the <a href="#">BRSPACE -f dbshow -c cfinfo</a> command that is to be executed using the current settings.

3. Choose *Continue*

BRGUI or BRTOOLS prompts you to start BRSPACE.

4. Choose *Continue* to start BRSPACE.

o Command line:

Enter at least the following command:

```
brspace -f dbshow -c cfinfo
```

You can enter more parameters if required. For more information, see [BRSPACE -f dbshow](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#) if you set the option *Create log file* (parameter `-l|-log`).

3.  Note

4. If you started BRSPACE from the command line without the information class name (`-c|-class`) – that is, with `brspace -f dbshow` – BRSPACE displays the *Show database information main menu*.

5. Choose  *Show database space information*  *Show control files* .

6.

7. If you have already entered the control file name, continue with step 4.

BRSPACE displays the *List of database control files*:

List Entry	Meaning



List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Size [KB]</i>	Control file size
<i>Device</i>	Disk device ID for the control file
<i>Type</i>	Control file type: raw device or file system
<i>File</i>	Control file name

8. To see more information, select one or more control files.

 Example

These examples apply only to input in character mode.

To select the first three entries in the list, enter 1-3.

To select the first and third entries, enter 1, 3.

To select the first three entries and the fifth, enter 1-3, 5.

To select all entries, enter 0.

BRSPACE displays *Information about control file*:

List Entry	Meaning
<i>Control file type (type)</i>	Control file type: raw device or file system
<i>Device device Id (device)</i>	Disk device ID for the control file
<i>Control file size in KB (size)</i>	Control file size

9. If you specified multiple control files, choose *Continue* to scroll through.

## Result

If you set the option *Create log file* (parameter `-l | -log`), check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.dbw` displays the details.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#)



## Showing Disk Volumes with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to show information about database disk volumes.

BRSPACE searches in the `SAPDATA_HOME` directory (on UNIX) or on all local drives (on Windows) for all available `sapdata<N>` directories, regardless of whether they already contain database files. It displays these in the *List of database disk volumes*, as described below. The column *Dbfs/Lnks* indicates how many database files or softlinks already exist in these directories.

### Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:

o BRGUI or BRTOOLS:

1. Choose **► Space management ► Additional space functions ► Show disk volumes ◀**

BRGUI or BRTOOLS displays the menu BRSPACE main options for showing database information, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-pl-profile</a>
<i>Database user/password (user)</i>	<a href="#">-ul-user</a>
<i>Create log file (log)</i>	<a href="#">-f dbshow -ll-log</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-cl-confirm</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-sl-scroll</a>
<i>Message language (language)</i>	<a href="#">-ll-language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f dbshow -c dvinfo</a> command that is to be executed using the current settings.

3. Choose *Continue*.





BRGUI or BRTOOLS prompts you to start BRSPACE.

o Command line:

Enter at least the following command:

```
brspace -f dbshow -c dvinfo
```

You can enter more parameters if required. For more information, see [BRSPACE -f dbshow](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#) if you set the option *Create log file* (parameter `-l|-log`).
3.  Note
4. If you started BRSPACE from the command line without the information class name (`-c|-class`) – that is, with `brspace -f dbshow` – BRSPACE displays the *Show database information main menu*.
5. Choose  *Show database space information*  *Show disk volumes* .
- 6.
7. BRSPACE displays the *List of database disk volumes*:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Device</i>	Disk device ID for the disk volume
<i>Total [KB]</i>	Total space on volume
<i>Used [%]</i>	Used space on volume
<i>Free[KB]</i>	Free space on volume
<i>Dbfs/Lnks</i>	Number of database files or softlinks on volume
<i>DirectoryRawDisk</i>	Name of directory or raw disk where volume is located

8. To see more information, select one or more disk volumes.

 Example

These examples apply only to input in character mode.

To select the first three entries in the list, enter `1-3`.

To select the first and third entries, enter `1, 3`.

To select the first three entries and the fifth, enter `1-3, 5`.

To select all entries, enter `0`.

BRSPACE displays *Information about disk volume*:

List Entry	Meaning
<i>Device Id (device)</i>	Disk device ID for the disk volume
<i>Total space in KB (total)</i>	Total space on volume

List Entry	Meaning
<i>Used space in % (used)</i>	Used space on volume
<i>Free space in KB (free)</i>	Free space on volume
<i>All free space in KB (afree)</i>	All free space on volume including space reserved for superuser
<i>Database file/link count (count)</i>	Number of database files or links on volume
<i>Max. still needed space in KB (needed)</i>	Additional space needed by autoextensible files to achieve maximum file size
<i>Max. missing space in KB (missing)</i>	Missing space if all autoextensible files had maximum file size

9. If you specified multiple disk volumes, choose *Continue* to scroll through.

## Result

If you set the option *Create log file* (parameter `-l | -log`), check the results in the [BRSPACE logs](#):

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail logs](#) `<encoded timestamp>.dbw` displays the details.

For more information about how to view the logs with BR\*Tools, see [Showing Profiles and Logs with BR\\*Tools](#).



## Segment Management with BR\*Tools

You can manage the segments – that is, tables and indexes – on your Oracle database with [BR\\*Tools](#).



Note

This section describes how you manage segments with BR\*Tools.

For more information on the approach to manage segments, see [Segment Management](#).

## Integration

- BRGUI or BRTOOLS calls the SAP tool BRSPACE. You can also manage segments by calling BRSPACE from the command line. However, unless you choose batch mode with the option `-c force`, BRSPACE displays menus to help you enter the required options.



Recommendation

We recommend you to normally use BRGUI or BRTOOLS rather than BRSPACE. This is because the menus in BRGUI or BRTOOLS simplify entry of the correct options.

- If required, set the following parameters for BRSPACE in the [Initialization Profile init<DBSID>.sap](#):
  - [scroll\\_lines](#) and [show\\_period](#). to control the screen display
  - [reorg\\_table](#) to specify a list of tables for reorganization
  - [rebuild\\_index](#) to specify a list of indexes for rebuild
  - [exp\\_table](#) to specify a list of tables for export
  - [imp\\_table](#) to specify a list of tables for import

## Features

You can perform the following functions for segment management with BR\*Tools:

- Reorganize tables
- Rebuild indexes
- Export tables
- Import tables
- Alter tables
- Alter indexes

## Activities

1. You choose *Segment management* in the BRGUI or BRTOOLS menus or directly from the command line. You can use quick mode if you know which objects you require. For more information on quick mode, see [BR\\*Tools User Interface](#)
2. If required, you change the default values for the parameters in the initialization profile `init<DBSID>.sap` and restart BRGUI or BRTOOLS.
3. If required, you choose **Segment Management** **Reset program status** to set the defaults used to the values set after BRGUI or BRTOOLS initialization.
4. You start segment management.

You check the results of segment management in the [BRSPACE logs](#).

## Selecting Objects

You can select the objects – that is, the tables or indexes – for segment management functions in one or more of the following ways:

- By specifying tables or indexes directly by name
- By specifying a tablespace name – all objects in the specified tablespace are selected

- By specifying the owner name – all objects with the specified owner are selected. This is useful for systems with Multiple Components in One Database (MCOB).

## Selecting Multiple Objects

When specifying tables or indexes directly by name, you can use wildcards or a parameter in the [initialization profile init<DBSID>.sap](#) to select *multiple* tables or indexes.

- Wildcards

When you use BRGUI or BRTOOLS, or call BRSPACE from the command line, you can use one of the following types of wildcard:

- Final selection wildcard, using the character “\*”

### Example

If you enter `DBA*`, BRSPACE selects all tables or indexes beginning with the characters “DBA”.

BRSPACE displays a list of tables or indexes meeting your selection criterion, but this is only for confirmation. You cannot make a further selection.

- Pre-selection wildcard, using the character “%”.

### Example

If you enter `DBA%`, BRSPACE preselects all tables or indexes beginning with the characters “DBA”.

BRSPACE displays a list of tables or indexes meeting your selection criterion. You can now make a further selection from this list, if required.

### Note

Note the following features when using wildcards:

- You can put the wildcard anywhere in the object name.  
For example, `DBA*`, `DB*A`, or `*DBA` are all valid..
- You can only use one wildcard.  
For example, `DB*A*B` is not valid.
- Wildcards are *not* valid for tablespaces or owners. That is, you can only use wildcards to directly specify groups of tables or indexes.
- If you use wildcards on the final object to select multiple objects in quick mode, BRSPACE displays a list of objects for confirmation.  
  
The final object is the one specified in the function name – that is, table for the functions reorganize, export, import, or alter tables, or index for the function alter or rebuild indexes.
- You can make a final selection with wildcards only on the final object for the function.

For example, for the function “Alter index” you can only make a final selection with a wildcard by entering “\*” or “<prefix>\*” in the index name field. If you enter “\*” or “<prefix>\*” in the table name field for “Alter index”, this is treated as a pre-selection wildcard.

- You can use the following parameters from the [initialization profile init<DBSID>.sap](#) to specify a long list of object names:
  - [reorg\\_table](#) specifies a list of tables to reorganize
  - [rebuild\\_index](#) specifies a list of indexes to rebuild
  - [exp\\_table](#) specifies a list of tables to export
  - [imp\\_table](#) specifies a list of tables to import



## Reorganizing Tables with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to reorganize tables online.

### Note

This section describes how you reorganize a table with BR\*Tools.

For more information on the approach to table reorganization, see:

- [Reorganization](#)
- [Reorganization Case Study](#)

You can perform the following types of reorganization:

- Reorganize tables online
- Check tables for reorganization
- Clean up tables after aborted reorganization
- Convert `LONG` and `LONG RAW` fields to `CLOB` or `BLOB` online (Oracle 10g or higher)
- Stop reorganization (command-line mode only)
- Suspend reorganization (command-line mode only)
- Resume reorganization (command-line mode only)

For more information, see [-f tbreorg](#).

## Prerequisites

You cannot perform online reorganization for tables with `LONG` or `LONG RAW` fields but you can convert them to `CLOB` or `BLOB` online. After this conversion, you can reorganize all tables online. For more information, see *SAP Note* [646681](#).

## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:

- BRGUI or BRTOOLS:

1. Choose  *Segment Management*  *Reorganize tables* .

BRGUI or BRTOOLS displays the menu *BRSPACE options for reorganization of tables*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Reorganization action (action)</i>	<a href="#">-f tbreorg -a -action</a>
<i>Tablespace names (tablespace)</i>	<a href="#">-f tbreorg -s -tablespace</a>
<i>Table owner (owner)</i>	<a href="#">-f tbreorg -o -owner</a>
<i>Table names (table)</i>	<a href="#">-f tbreorg -t -table</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Extended output (output)</i>	<a href="#">-o -output</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f tbreorg</a> command that is to be executed using the current settings.

3.  Note

4. If required, in *Table names* you can enter the names of multiple tables. You can use wildcards. For more information, see *Selecting Objects* in [Segment Management with BR\\*Tools](#).

5. You can also specify a list of tables to be processed with [reorg\\_table](#) in the initialization profile `init<DBSID>.sap`.



6. In *Tablespace names* and *Table owner*, you can specify multiple objects but you cannot use wildcards. BRSPACE processes all the tables in the specified tablespace name(s) or all tables belonging to the specified table owner(s). But these entries and reorganization action entry are optional.

7.

8. Choose *Continue*.

BRGUI or BRTOOLS prompts you to start BRSPACE.

9. Choose *Continue* to start BRSPACE.

- o Command line:

Enter at least the following command:

```
brspace -f tbreorg
```

You can enter more parameters, including the table names, if required. For more information, see [BRSPACE -f tbreorg](#).

2.  Note

3. Whichever way you start the procedure – with BRGUI or BRTOOLS, or from the command line – you can use quick mode if you know the final object names, in this case the table names. For more information, see [How to Use BR\\*Tools](#).

4. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#).

5.

6. If you have already entered the table names, continue with step 5 (quick mode).

 Note

If you have entered multiple tables, BRSPACE displays as confirmation a *List of tables for reorganization*. If you have not already made a final selection, you can make a selection from this list.

Continue with step 5 (quick mode).

BRSPACE displays the *Table reorganization main menu*.


7. Choose *Reorganize tables*.

BRSPACE displays the table list:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Owner</i>	Table owner
<i>Table</i>	Table name
<i>Pt.</i>	Partitioned

List Entry	Meaning
<i>DgPk.</i>	Parallel degree / primary key: “ ” means no “+” means yes / enabled “-” means yes / disabled
<i>Rows</i>	Number of rows
<i>Space [KB]</i>	Space occupied by the table
<i>Data [KB:%]</i>	Amount of data in the table: percentage of occupied space

8. Select a table or multiple tables.

 Example

These examples only apply to input in character mode.

To select the first three tables in the list, enter 1-3.

To select the first and third tables, enter 1, 3.

To select the first three tables and the fifth, enter 1-3, 5.

To select all tables, enter 0.

BRSPACE displays the menu, *Options for reorganization of tables.*

9. Set the required options:

Menu Entry	Meaning
<i>New destination tablespace (newts)</i>	<a href="#">-f tbreorg -n -newts</a>
<i>Separate index tablespace (indts)</i>	<a href="#">-f tbreorg -i -indts</a>
<i>Parallel threads (parallel)</i>	<a href="#">-f tbreorg -p -parallel</a>
<i>Table/index parallel degree (degree)</i>	<a href="#">-f tbreorg -e -degree</a>
<i>Create DDL statements (ddl)</i>	<a href="#">-f tbreorg -d -ddl</a>
<i>Category of initial extent size (initial)</i>	<a href="#">-f tbreorg -l -initial</a>
<i>Sort by fields of index (sortind)</i>	<a href="#">-f tbreorg -r -sortind</a>
<i>Table reorganization mode (mode)</i>	<a href="#">-f tbreorg -m -mode</a>

10.  Note

11. There is no SQL command line here because the online reorganization is performed in several steps by an Oracle package, which contains multiple procedures.
- 12.
13. To start processing with the selected options, choose *Continue*.

## Result

Check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.tbr` displays the details.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).

For online reorganization, the file `ddl.sql` is created in the subdirectory `<encoded timestamp>` of the directory if the option `-d|-ddl` is set to a value other than `no`. The file contains the Data Definition Language (DDL) statements used for the creation of interim tables during the reorganization. For more information, see [Reorganization](#). If the `-d|-ddl` option is set to `only`, then only DDL statements are created and the actual reorganization is not performed.



## Rebuilding Indexes with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to rebuild indexes.

You can use the following types of rebuild:

- Rebuild indexes online
- Stop rebuild (command-line mode only)
- Suspend rebuild (command-line mode only)
- Resume rebuild (command-line mode only)

For more information, see [-f idrebuild](#).

## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:
  - BRGUI or BRTOOLS:

Choose **► Segment Management ► Rebuild indexes ◀**.

BRGUI or BRTOOLS displays the menu *BRSPACE options for rebuild of indexes*, where you specify the options with which you call BRSPACE.

Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile</i>	<a href="#">-p -profile</a>

Menu Entry	Equivalent BRSPACE Command Option
(profile)	
Database user/password (user)	<a href="#">-u -user</a>
Rebuild action (action)	<a href="#">-f idrebuild -a -action</a>
Tablespace names (tablespace)	<a href="#">-f idrebuild -s -tablespace</a>
Index owner (owner)	<a href="#">-f idrebuild -o -owner</a>
Table names (table)	<a href="#">-f idrebuild -t -table</a>
Index names (index)	<a href="#">-f idrebuild -i -index</a>
Confirmation mode (confirm)	<a href="#">-c -confirm</a>
Extended output (output)	<a href="#">-o -output</a>
Scrolling line count (scroll)	<a href="#">-s -scroll</a>
Message language (language)	<a href="#">-l -language</a>
BRSPACE command line (command)	This shows you the <a href="#">BRSPACE -f idrebuild</a> command that is to be executed using the current settings.

 Note

If required, you can enter the names of multiple objects in *Index names* or *Table names*. You can also use wildcards. For more information, see [Selecting Objects](#) in [Segment Management with BR\\*Tools](#).

However, you can only finally select indexes by entering a wildcard with "\*" or "[<prefix>]\*[<suffix>]" in *Index names*. If you enter "\*" or "[<prefix>]\*[<suffix>]" in *Table names*, it is treated as a preselection, not a final selection.

You can also specify a list of indexes to be processed with [rebuild\\_index](#) in the initialization profile `init<DBSID>.sap`.

In *Tablespace names* and *Index owner*, you can specify multiple objects but you cannot use wildcards.

BRSPACE processes all the indexes in the specified tablespace name(s), or all indexes belonging to the specified index owner(s) or tables. But these entries and rebuild action entry are optional.

Choose *Continue*.

BRGUI or BRTOOLS prompts you to start BRSPACE.

Choose *Continue* to start BRSPACE.

- Command line:

Enter at least the following command:

```
brspace -f idrebuild
```

You can enter more parameters, including the indexes names, if required.  
For more information, see [BRSPACE -f idrebuild](#).

 Note

Whichever way you start the procedure – with BRGUI or BRTOOLS, or from the command line – you can use quick mode if you know the final object names, in this case the index names. For more information, see [How to Use BR\\*Tools](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#).
3. If you have already entered the index names, continue with step 5 (quick mode).

 Note

If you have entered multiple indexes, BRSPACE displays as confirmation a *List of indexes for rebuild*. If you have not already made a final selection, you can make a selection from this list.

Continue with step 5 (quick mode).

BRSPACE displays the *Rebuild indexes main menu*.

4. Choose *Rebuild indexes*.

BRSPACE displays the index list:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Owner</i>	Index owner
<i>Table</i>	Table name
<i>Index</i>	Index name
<i>Pt.</i>	Partitioned
<i>Dg.</i>	Parallel degree
<i>Tablespace</i>	Tablespace name

5. Select an index or multiple indexes.

## Example

These examples only apply to input in character mode.

To select the first three indexes in the list, enter 1-3.

To select the first and third indexes, enter 1, 3.

To select the first three indexes and the fifth, enter 1-3, 5.

To select all indexes, enter 0.

BRSPACE displays the menu *Options for rebuild of indexes*.

6. Set the required options:

Menu Entry	Meaning
<i>New destination tablespace (newts)</i>	<a href="#">-f idrebuild -n -newts</a>
<i>Parallel threads (parallel)</i>	<a href="#">-f idrebuild -i -indts</a>
<i>Index parallel degree (degree)</i>	<a href="#">-f idrebuild -e -degree</a>
<i>Category of initial extent size (initial)</i>	<a href="#">-f idrebuild -l -initial</a>
<i>Index rebuild mode (mode)</i>	<a href="#">-f idrebuild -m -mode</a>
<i>SQL command (command)</i>	This shows you the SQL command that is to be executed using the current settings.

7. To start processing with the selected options, choose *Continue*.

## Result

Check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.idr` displays the details.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Exporting Tables with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to export tables to an operating system file.

### Note

This section describes how you export tables with BR\*Tools.

For more information on the approach to table export, see [Export/Import](#).

 Note

Do not use this procedure for the transport of database objects between databases.

Do not use this procedure for restore.

## Procedure

Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:

o BRGUI or BRTOOLS:


1. Choose  *Segment Management*  *Export tables* .

BRGUI or BRTOOLS displays the menu *BRSPACE options for export tables*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Export utility (utility)</i>	<a href="#">-f tbexport -l -utility</a>
<i>Tablespace names (tablespace)</i>	<a href="#">-f tbexport -s -tablespaces</a>
<i>Table owner (owner)</i>	<a href="#">-f tbexport -o -owner</a>
<i>Table names (table)</i>	<a href="#">-f tbexport -t -tables</a>
<i>Export dump directory</i>	<a href="#">-f tbexport -u -dumpdir</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Extended output (output)</i>	<a href="#">-o -output</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command</i>	This shows you the <a href="#">-BRSPACE -f -tbexport</a>

Menu Entry	Equivalent BRSPACE Command Option
<i>line (command)</i>	command that is to be executed using the current settings.

3.  Note
4. If required, in *Table names* you can enter the names of multiple tables. You can use wildcards. For more information, see *Selecting Objects* in [Segment Management with BR\\*Tools](#).
5. In *Tablespace names* and *Table owner*, you can specify multiple objects but you cannot use wildcards. BRSPACE processes all the tables in the specified tablespace name(s) or all tables belonging to the specified table owner(s). But these entries are optional.
6. You can also specify the dump directory using the parameter [exp\\_dump\\_dir](#) in the initialization profile `init<DBSID>.sap`, but the entry you make on the screen overwrites the profile parameter.
- 7.
8. Choose *Continue*.  
BRGUI or BRTOOLS prompts you to start BRSPACE.
9. Choose *Continue* to start BRSPACE.

- o Command line:

Enter at least the following command:

```
brspace -f tbexport
```

You can enter more parameters, including the table names, if required. For more information, see [-BRSPACE -f -tbexport](#).

Whichever way you start the procedure – with BRGUI or BRTOOLS, or from the command line – you can use quick mode if you know the final object names, in this case the table names. For more information, see [How to Use BR\\*Tools](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#).
3. If you have already entered the table names, continue with step 5 (quick mode).

 Note

If you have entered multiple tables, BRSPACE displays as confirmation a *List of tables for export*. If you have not already made a final selection, you can make a selection from this list.

Continue with step 5 (quick mode).

BRSPACE displays the *Export tables main menu*.

4. Choose *Export tables*.

BRSPACE displays the table list:



List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Owner</i>	Table owner
<i>Table</i>	Table name
<i>Part.</i>	Partitioned
<i>Rows</i>	Number of rows
<i>Space [KB]</i>	Space occupied by the table
<i>Data [KB:%]</i>	Amount of data in the table: percentage of occupied space

5. Select a table or multiple tables.

 Example

These examples only apply to input in character mode:

To select the first three tables in the list, enter 1-3.

To select the first and third tables, enter 1, 3.

To select the first three tables and the fifth, enter 1-3, 5.

To select all tables, select 0.

BRSPACE displays the menu, *Main options for export of tables.*

6. Set the required options:

Menu Entry	Meaning
<i>Export utility (utility)</i> – display only	Oracle export utility to be used
<i>Tablespaces for export (tablespaces)</i> – display only	The tablespace that you selected above, if any
<i>Owners for export (owner)</i> – display only	The table owner that you selected above, if any
<i>Export table rows (rows)</i>	<a href="#">-f tbexport -r -rows</a>
<i>Export table indexes (indexes)</i>	<a href="#">-f tbexport -i -indexes</a>
<i>Export table constraints (constraints)</i>	<a href="#">-f tbexport -c -constraints</a>

Menu Entry	Meaning
Export table grants (grants)	<a href="#">-f tbexport -g grants</a>
Export table triggers (triggers)	<a href="#">-f tbexport -t triggers</a>

7. When you have set the required options, choose *Continue*

BRSPACE displays the menu, *Additional options for export of tables*.

Menu Entry	Meaning
Use direct path (direct)	<a href="#">-f tbexport -d direct</a>
Export buffer size in KB (buffer)	<a href="#">-f tbexport -b buffer</a>
Compress table extents (compress)	<a href="#">-f tbexport -m compress</a>
Consistent export (consistent)	<a href="#">-f tbexport -n consistent</a>
Parallel degree (parallel)	<a href="#">-f tbexport -p parallel</a>
Max. size of dump file in MB (filesize)	<a href="#">-f tbexport -z filesize</a>
Force table export (force)	<a href="#">-f tbexport -f force</a>
EXP command (command)	The EXP command that is to be executed using the current settings. For more information, see your Oracle documentation.

8. To start processing with the selected options, choose *Continue*.

## Result

Check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.tbe` displays the details.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).

These files are created during the export:

- The export dump file, `expdat.dmp`, is created in subdirectory `<encoded timestamp>.edd` of the dump directory.
- The parameter file `parfile.exp` is created in subdirectory `<encoded timestamp>` of the directory `$SAPDATA_HOME/sapreorg`. It contains the parameters for the Oracle export tool, EXP.

## More Information

[Special Export and Import Functions with BRSPACE](#)



### Importing Tables with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to import tables from an operating system file.

#### Note

This section describes how you import tables with BR\*Tools.

For more information on the approach to table import, see [Export/Import](#).

#### Note

Do not use this procedure for the transport of database objects between databases.

Do not use this procedure for restore.

## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:
  - o BRGUI or BRTOOLS:
    1. Choose **► Segment Management ► Import tables ◀**.

BRGUI or BRTOOLS displays the menu *BRSPACE options for import tables*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-pl-profile</a>
<i>Database user/password (user)</i>	<a href="#">-ul-user</a>
<i>BRSPACE export run / dump file (export)</i>	<a href="#">-f tbimport -x -export</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-cl-confirm</a>
<i>Extended output (output)</i>	<a href="#">-ol-output</a>

<b>Menu Entry</b>	<b>Equivalent BRSPACE Command Option</b>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f tbimport</a> command that is to be executed using the current settings.

3. Choose *Continue*.

BRGUI or BRTOOLS prompts you to start BRSPACE.

4. Choose *Continue* to start BRSPACE.

- o Command line:

Enter at least the following command:

```
brspace -f tbimport
```

You can enter more parameters, including the table names, if required. For more information, see [BRSPACE -f tbimport](#).

 **Note**

Whichever way you start the procedure - with BRGUI or BRTOOLS, or from the command line - you can use quick mode if you know the final object names, in this case the name of the BRSPACE export run or the export file name. For more information, see [How to Use BR\\*Tools](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#).
3. If you have already entered the BRSPACE export run or the export file name, continue with step 5 (quick mode).

BRSPACE displays the *Import tables main menu*.

4. Choose *Import tables*.

BRSPACE displays the list of BRSPACE exports for import:

<b>List Entry</b>	<b>Meaning</b>
<i>Pos.</i>	List sequence number
<i>Run</i>	BRSPACE export identifier - BRSPACE log name
<i>Date</i>	Export date

List Entry	Meaning
<i>Tables</i>	Number of tables in export
<i>Dumps</i>	Number of export dump files
<i>Size [KB]</i>	Total size of the export dump file(s)
<i>Util.</i>	Export utility used

5. Select an export run.

BRSPACE displays the menu *Main options for import*.

6. Set the required options:

Menu Entry	Meaning
<i>Import utility (utility)</i> – display only	<a href="#">-f tbimport -l -utility</a>
<i>Import type (type)</i>	<a href="#">-f tbimport -t -type</a>
<i>Owners for export (owner)</i>	<a href="#">-f tbimport -o -owner</a>
<i>Tables for import (tables)</i>	<a href="#">-f tbimport -t -tables</a>
<i>Import table rows (rows)</i>	<a href="#">-f tbimport -r -rows</a>
<i>Import table indexes (indexes)</i>	<a href="#">-f tbimport -i -indexes</a>
<i>Import table constraints (constraints)</i>	<a href="#">-f tbimport -c -constraints</a>
<i>Import table grants (grants)</i>	<a href="#">-f tbimport -g -grants</a>
<i>Import table triggers (triggers)</i>	<a href="#">-f tbimport -e -triggers</a>

7. When you have set the required options, choose *Continue*

BRSPACE displays the menu *Additional options for import*.

Menu Entry	Meaning
<i>Import buffer size in KB (buffer)</i>	<a href="#">-f tbimport -b -buffer</a>
<i>Commit after each array insert (commit)</i>	<a href="#">-f tbimport -m -commit</a>
<i>Ignore creation errors (ignore)</i>	<a href="#">-f tbimport -n -ignore</a>
<i>Table exists (action)</i>	<a href="#">-f tbimport -a -action</a>
<i>Parallel degree (parallel)</i>	<a href="#">-f tbimport -p -parallel</a>

Menu Entry	Meaning
<i>Max. size of dump file in MB (filesize)</i> – input only possible if the dump file is not from BRSPACE. For more information, see menu entry <i>BRSPACE export run / dump file</i> above.	<a href="#">-f tbimport -z -filesize</a>
<i>Force table import (force)</i>	<a href="#">-f tbimport -f -force</a>
<i>IMP command</i>	The IMP command that is to be executed using the current settings. For more information, see your Oracle documentation.

- To start processing with the selected options, choose *Continue*.

## Result

Check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.tbi` displays the details.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).

The parameter file, `parfile.imp` is created in subdirectory `<encoded timestamp>` of the directory `$$$APDATA_HOME/sapreorg`. It contains the parameters for the Oracle import tool, IMP.



## Altering Tables with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to alter tables as follows:

- Switch on table monitoring
- Switch off table monitoring
- Set parallel degree
- Shrink tables (Oracle 10g or higher)

### Note

This section describes how you alter tables with BR\*Tools.

For more information on the approach to altering tables, see [Segment Management](#).

## Procedure

- Start the procedure using BRGUI or BRTOOLS, or from the command line:


○ BRGUI or BRTOOLS:

1. Choose  *Segment Management*  *Alter tables.* 

BRGUI or BRTOOLS displays the menu *BRSPACE options for alter tables*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p profile</a>
<i>Database user/password (user)</i>	<a href="#">-u user</a>
<i>Alter table action (action)</i>	<a href="#">-f talter -a action</a>
<i>Tablespace names (tablespace)</i>	<a href="#">-f talter -s tablespace</a>
<i>Table owner (owner)</i>	<a href="#">-f talter -o owner</a>
<i>Table names (table)</i>	<a href="#">-f talter -t table</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c confirm</a>
<i>Extended output</i>	<a href="#">-o output</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s scroll</a>
<i>Message language (language)</i>	<a href="#">-l language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f talter</a> command that is to be executed using the current settings.

3.  Note
4. If required, in *Table names* you can enter the names of multiple tables. You can also use wildcards. For more information, see “Selecting Objects” in [Segment Management with BR\\*Tools](#).
5. In *Tablespace names* and *Table owner*, you can specify multiple objects but you cannot use wildcards. BRSPACE processes all the tables in the specified tablespace name(s) or all tables belonging to the specified table owner(s). But these entries and alter action entry are optional.
- 6.
7. Choose *Continue*.

BRGUI or BRTOOLS prompts you to start BRSPACE.

8. Choose *Continue* to start BRSPACE.

- Command line:

Enter at least the following command:

```
brspace -f tbalter
```

You can enter more parameters, including the table names and action, if required. For more information, see [BRSPACE -f tbalter](#).

 Note

Whichever way you start the procedure – with BRGUI or BRTOOLS, or from the command line – you can use quick mode if you know the final object names, in this case the table names. For more information, see [How to Use BR\\*Tools](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#).
3. BRSPACE displays the *Alter tables main menu*.
4. You can use quick mode as follows:
  - If you have already entered the action, continue with step 4.
  - If you have already entered the table names and action, continue with step 6.

 Note

If you have entered multiple tables, BRSPACE displays as confirmation a *List of tables for alter*. If you have not already made a final selection, you can make a selection from this list.

Continue with step 6 (quick mode).

5. Choose or confirm the required action:
  - *Switch on table monitoring*
  - *Switch off table monitoring*
  - *Set parallel degree*
  - *Shrink tables*
6. If you have already entered the table names, continue with step 6 (quick mode).

BRSPACE displays the table list:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Owner</i>	Table owner



List Entry	Meaning
<i>Table</i>	Table name
<i>Part.</i>	Partitioned
<i>Parts</i>	Number of partitions
<i>Monit.</i>	Table monitoring setting
<i>Paral.</i>	Parallel degree

 Note

BRSPACE only displays the tables that can be processed by your chosen action.

For example, if you choose *Switch off table monitoring*, only the tables that currently have table monitoring on are displayed.

7. Select a table or multiple tables.

 Example

These examples only apply to input in character mode.

To select the first three tables in the list, enter 1-3.

To select the first and third tables, enter 1, 3.

To select the first three tables and the fifth, enter 1-3, 5.

To select all tables, enter 0.

BRSPACE displays the menu *Options for alter of tables*.

8. Set the required options:

Menu Entry	Meaning
<i>Current monitoring status (currmonit)</i> – display only	Current monitoring status of the table
<i>Current parallel degree (currdeg)</i> – display only	Current parallel degree of the table
<i>Alter table action (action)</i> – display only	The action that you selected above

Menu Entry	Meaning
<i>New parallel degree (degree)</i>	<a href="#">-f tbalter -d -degree</a>
<i>SQL command line (command)</i>	The SQL command that is to be executed using the current settings. For more information, see your Oracle SQL documentation.

9. To start processing with the selected options, choose *Continue*.

## Result

Check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.tba` displays the details.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Altering Indexes with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to alter indexes as follows:




- Coalesce index
- Set degree of parallelism
- Shrink indexes (Oracle 10g or higher)

### Note


This section describes how you alter indexes with BR\*Tools.

For more information on the approach to altering indexes, see [Segment Management](#).

## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:
  - BRGUI or BRTOOLS:
    1. Choose  *Segment Management*  *Alter indexes*. .
    - BRGUI or BRTOOLS displays the menu *BRSPACE options for alter indexes*, where you specify the options with which you call BRSPACE.
    2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Alter index action (action)</i>	<a href="#">-f idalter -a -action</a>
<i>Tablespace names (tablespace)</i>	<a href="#">-f idalter -s -tablespace</a>
<i>Index owner (owner)</i>	<a href="#">-f idalter -o -owner</a>
<i>Table names (table)</i>	<a href="#">-f idalter -t -table</a>
<i>Index names (index)</i>	<a href="#">-f idalter -i -index</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Extended output</i>	<a href="#">-o -output</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f idalter</a> command that is to be executed using the current settings.

3.  Note
  4. If required, in *Table names* and *Index names* you can enter the names of multiple objects. You can also use wildcards. For more information, see "Selecting Objects" in [Segment Management with BR\\*Tools](#).
  5. In *Tablespace names* and *Table owner*, you can specify multiple objects but you cannot use wildcards. BRSPACE processes all the indexes in the specified tablespace(s) or all indexes belonging to the specified index owner(s) or table(s). But these entries and alter action entry are optional.
  - 6.
  7. Choose *Continue*.  
BRGUI or BRTOOLS prompts you to start BRSPACE.
  8. Choose *Continue* to start BRSPACE.
- Command line:

Enter at least the following command:

```
brspace -f idalter
```

You can enter more parameters, including the index names and action, if required. For more information, see [BRSPACE -f idalter](#).

 Note

Whichever way you start the procedure - with BRGUI or BRTOOLS, or from the command line - you can use quick mode if you know the object names, in this case the index names. For more information, see [How to Use BR\\*Tools](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#).
3. BRSPACE displays the *Alter indexes main menu*.
4. You can use quick mode as follows:
  - If you have already entered the action, continue with step 4.
  - If you have already entered the index names and action, continue with step 6.

 Note

If you have entered multiple indexes, BRSPACE displays as confirmation a *List of indexes for alter*. If you have not already made a final selection, you can make a selection from this list.

Continue with step 6 (quick mode).


5. Choose or confirm the required action:
  - *Coalesce index*
  - *Set parallel degree*
  - *Shrink indexes*
6. If you have already entered the index names, continue with step 6 (quick mode).

BRSPACE displays the index list:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Owner</i>	Index owner
<i>Table</i>	Table name
<i>Index</i>	Index name
<i>Part.</i>	Partitioned

List Entry	Meaning
<i>Parts</i>	Number of partitions
<i>Paral.</i>	Parallel degree

7. Select an index or multiple indexes.

 Example

These examples only apply to input in character mode.

To select the first three indexes in the list, enter 1-3.

To select the first and third indexes, enter 1, 3.

To select the first three indexes and the fifth, enter 1-3, 5.

To select all indexes, enter 0.

BRSPACE displays the menu, *Options for alter of indexes*.

8. Set the required options:

Menu Entry	Meaning
<i>Current parallel degree (currdeg)</i> – display only	Current parallel degree of the index
<i>Alter index action (action)</i> – display only	The action that you selected above
<i>New parallel degree (degree)</i>	<a href="#">-f idalter -dl-degree</a>
<i>SQL command line (command)</i>	The SQL command that is to be executed using the current settings. For more information, see your Oracle SQL documentation.

9. To start processing with the selected options, choose *Continue*.

## Result

Check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.ida` displays the details.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).

## Showing Tables with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to show information about tables.

### Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:
  - BRGUI or BRTOOLS:
    1. Choose **► Segment management ► Additional segment functions ► Show tables ◀**.

BRGUI or BRTOOLS displays the menu *BRSPACE main options for showing database information*, where you specify the options with which you call BRSPACE.
  - 2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Database tablespace (tablespace)</i>	<a href="#">-f dbshow -s -tablespace</a>
<i>Database owner (owner)</i>	<a href="#">-f dbshow -o -owner</a>
<i>Database table (table)</i>	<a href="#">-f dbshow -t -table</a>
<i>Create log file (log)</i>	<a href="#">-f dbshow -l -log</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Extended output (output)</i>	<a href="#">-o -output</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f dbshow -c tbinfo</a> command that is to be executed using the current settings.

3. If required, in *Database table* you can enter the names of multiple tables by using wildcards. For more information, see *Selecting Objects* in [Segment Management with BR\\*Tools](#).

4. In *Database tablespace* and *Database owner*, you can specify multiple objects but you cannot use wildcards. BRSPACE processes all the tables in the specified tablespace(s) or all tables belonging to the specified owner(s).

5. Choose *Continue*

BRGUI or BRTOOLS prompts you to start BRSPACE.

6. Choose *Continue* to start BRSPACE.

- o Command line:

Enter at least the following command:

```
brspace -f dbshow -c tbinfo
```

You can enter more parameters if required. For more information, see [BRSPACE -f dbshow](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#) if you set the option *Create log file* (parameter `-l|-log`).

3. If you started BRSPACE from the command line without the information class name (`-c|-class`) – that is, with `brspace -f dbshow` – BRSPACE displays the *Show database information main menu*.

Choose **► Show database segment information ► Show tables ◀**.


4. If you have already entered a single table name, continue with step 4.

BRSPACE displays the *List of database tables*:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Owner</i>	Table owner
<i>Table</i>	Table name
<i>Part.</i>	Partitioned: yes or no
<i>Parts</i>	Number of partitions for partitioned table
<i>Monit.</i>	Monitoring attribute
<i>Paral.</i>	Degree of parallelism
<i>Inds</i>	Number of indexes
<i>Tablespace</i>	Tablespace name
<i>Analyzed</i>	Date last analyzed by update statistics
<i>Rows</i>	Number of rows
<i>Space[KB]</i>	Total space allocated to the table

List Entry	Meaning
<i>Used</i> [KB:%] ]	Used space in table: percentage of total space used
<i>Data</i> [KB:%]	Amount of data in the table: percentage of total space for data

5. To see more information, select one or more tables.

 Example

These examples only apply to input in character mode.

To select the first three entries in the list, enter 1-3.

To select the first and third entries, enter 1, 3.

To select the first three entries and the fifth, enter 1-3, 5.

To select all entries, enter 0.

BRSPACE displays *Information about table*:

List Entry	Meaning
<i>Partitioned table (partitioned)</i>	Partitioned: yes or no
<i>Number of partitions (partitions)</i>	Number of partitions for partitioned table
<i>Monitoring attribute (monitoring)</i>	Monitoring attribute
<i>Parallel degree (parallel)</i>	Degree of parallelism
<i>Number of indexes (indexes)</i>	Number of indexes
<i>Tablespace name (tablespace)</i>	Tablespace name
<i>Last analyzed (analyzed)</i>	Date last analyzed by update statistics
<i>Sample size (sample)</i>	Sample size of update statistics
<i>Number of rows (rows)</i>	Number of rows
<i>Allocated space in KB (space)</i>	Total space allocated to the table
<i>Used space in KB / % (used)</i>	Used space in table / percentage of total space used
<i>Pure data in KB / % (data)</i>	Amount of data in the table / percentage of total space for data
<i>Number of chained rows (chained)</i>	Number of chained rows



List Entry	Meaning
<i>Next extent size in KB (next)</i>	Next extent size
<i>Maximum number of extents (maxexts)</i>	Maximum number of extents possible

- If you specified multiple tables, choose *Continue* to scroll through.

## Result

If you set the option *Create log file* (parameter `-l|-log`), check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.dbw` displays the details.

For more information about how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Showing Indexes with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to show information about indexes.

## Procedure

- Start the procedure using BRGUI or BRTOOLS, or from the command line:
  - BRGUI or BRTOOLS:
    - Choose **► Segment management ► Additional segment functions ► Show indexes ◀**.

BRGUI or BRTOOLS displays the menu *BRSPACE main options for showing database information*, where you specify the options with which you call BRSPACE.

- Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Database tablespace (tablespace)</i>	<a href="#">-f dbshow -s -tablespace</a>
<i>Database owner (owner)</i>	<a href="#">-f dbshow -o -owner</a>
<i>Database table (table)</i>	<a href="#">-f dbshow -t -table</a>

Menu Entry	Equivalent BRSPACE Command Option
<i>Database index (index)</i>	<a href="#">-f dbshow -i -index</a>
<i>Create log file (log)</i>	<a href="#">-f dbshow -l -log</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Extended output (output)</i>	<a href="#">-o -output</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f dbshow -c idinfo</a> command that is to be executed using the current settings.

3.  Note

4. If required, you can enter the names of multiple objects in *Database index* or *Database table* by using wildcards. For more information, see “Selecting Objects” in [Segment Management with BR\\*Tools](#).
5. In *Database tablespace* and *Database owner*, you can specify multiple objects but you cannot use wildcards.
6. BRSPACE processes all the indexes in the specified tablespace(s), or all indexes belonging to the specified owner(s) or table(s).
- 7.
8. Choose *Continue*  
BRGUI or BRTOOLS prompts you to start BRSPACE.
9. Choose *Continue* to start BRSPACE.

o Command line:

Enter at least the following command:

```
brspace -f dbshow -c idinfo
```

You can enter more parameters if required. For more information, see [BRSPACE -f dbshow](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#) if you set the option *Create log file* (parameter `-l|-log`).

3.  Note

4. If you started BRSPACE from the command line without the information class name (-c|-class) – that is, with `brspace -f dbshow` – BRSPACE displays the *Show database information main menu*.
5. Choose **Show database segment information** **Show indexes**.
- 6.
7. If you have already entered a single index name, continue with step 4.

BRSPACE displays the *List of database indexes*:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Owner</i>	Table owner
<i>Table</i>	Table name
<i>Index</i>	Index name
<i>Part.</i>	Partitioned (yes / no)
<i>Parts</i>	Number of partitions for partitioned table
<i>Paral.</i>	Degree of parallelism
<i>Tablespace</i>	Tablespace name
<i>Analyzed</i>	Date last analyzed by update statistics
<i>Rows</i>	Number of rows
<i>Space[KB]</i>	Total space allocated to the index
<i>Used[KB:%] ]</i>	Used space in index: percentage of total space used
<i>Data [KB:%]</i>	Amount of data in index: percentage of total space for data

To see more information, select one or more indexes.

#### Example

These examples only apply to input in character mode.

To select the first three entries in the list, enter 1-3.

To select the first and third entries, enter 1, 3.

To select the first three entries and the fifth, enter 1-3, 5.

To select all entries, enter 0.

BRSPACE displays *Information about index*:

List Entry	Meaning
<i>Unique index (unique)</i>	Unique index: yes or no
<i>Bitmap index (bitmap)</i>	Bitmap index: yes or no
<i>Partitioned index (partitioned)</i>	Partitioned index: yes or no
<i>Number of partitions (partitions)</i>	Number of partitions for partitioned index
<i>Parallel degree (parallel)</i>	Degree of parallelism
<i>Table name (table)</i>	Table name
<i>Tablespace name (tablespace)</i>	Tablespace name
<i>Last analyzed (analyzed)</i>	Date last analyzed by update statistics
<i>Sample size (sample)</i>	Sample size of update statistics
<i>Number of rows (rows)</i>	Number of rows
<i>Allocated space in KB (space)</i>	Total space allocated to the index
<i>Used space in KB / % (used)</i>	Used space in index: percentage of total space used
<i>Pure data in KB / % (data)</i>	Amount of data in index: percentage of total space for data
<i>Next extent size (next)</i>	Next extent size
<i>Maximum number of extents (maxextents)</i>	Maximum number of extents possible

8. If you specified multiple indexes, choose *Continue* to scroll through.

## Result

If you set the option *Create log file* (parameter `-l|-log`), check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.dbw` displays the details.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Showing Table Partitions with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to show information about table partitions.

## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:

- BRGUI or BRTOOLS:

1. Choose **► Segment management ► Additional segment functions ► Show table partitions ◀**.

BRGUI or BRTOOLS displays the menu *BRSPACE main options for showing database information*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Database tablespace (tablespace)</i>	<a href="#">-f dbshow -s -tablespace</a>
<i>Database owner (owner)</i>	<a href="#">-f dbshow -o -owner</a>
<i>Database table (table)</i>	<a href="#">-f dbshow -t -table</a>
<i>Create log file (log)</i>	<a href="#">-f dbshow -l -log</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Extended output (output)</i>	<a href="#">-o -output</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f dbshow -c tpinfo</a> command that is to be executed using the current settings.

3. If required, in *Database table* you can enter the names of multiple tables by using wildcards. For more information, see *Selecting Objects* in [Segment Management with BR\\*Tools](#).
4. In *Database tablespace* and *Database owner*, you can specify multiple objects but you cannot use wildcards. BRSPACE processes all the table partitions in the specified tablespace or all table partitions belonging to the specified owner(s) or table(s).

5. Choose *Continue*

BRGUI or BRTOOLS prompts you to start BRSPACE.

6. Choose *Continue* to start BRSPACE.

- o Command line:

Enter at least the following command:

```
brspace -f dbshow -c tpinfo
```

You can enter more parameters if required. For more information, see [BRSPACE -f dbshow](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#) if you set the option *Create log file* (parameter `-l|-log`).

3.  Note

4. If you started BRSPACE from the command line without the information class name (`-c|-class`) – that is, with `brspace -f dbshow` – BRSPACE displays the *Show database information main menu*.

5. Choose  *Show database segment information*  *Show table partitions* .

6.

7. If you have already entered a single table name, continue with step 4.

BRSPACE displays the *List of database table partitions*:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Owner</i>	Partition owner
<i>Table</i>	Table name
<i>Partition</i>	Partition name
<i>Tablespace</i>	Tablespace name
<i>Analyzed</i>	Date last analyzed by update statistics
<i>Rows</i>	Number of rows
<i>Space[KB]</i>	Total space allocated to the table partition
<i>Used[KB:%] ]</i>	Used space in table partition: percentage of total space used (“high-water mark”)
<i>Data [KB:%]</i>	Amount of data in table partition: percentage of total space for data

8. To see more information, select one or more table partitions.

 Example

To select the first three entries in the list, enter 1-3.

To select the first and third entries, enter 1, 3.

To select the first three entries and the fifth, enter 1-3, 5.

To select all entries, enter 0.

BRSPACE displays *Information about table partition*:

List Entry	Meaning
<i>Hash partition (hash)</i>	Hash partitions: yes or no
<i>Tablespace name (tablespace)</i>	Tablespace name
<i>Last analyzed (analyzed)</i>	Date last analyzed by update statistics
<i>Sample size (sample)</i>	Sample size of update statistics
<i>Number of rows (rows)</i>	Number of rows
<i>Allocated space in KB (space)</i>	Total space allocated to the table
<i>Used space in KB / % (used)</i>	Used space in table / percentage of total space used ("high-water mark")
<i>Pure data in KB / % (data)</i>	Amount of data in the table / percentage of total space for data
<i>Number of chained rows (chained)</i>	Number of chained rows
<i>Next extent size in KB (next)</i>	Next extent size
<i>Maximum number of extents (maxexts)</i>	Maximum number of extents possible

9. If you specified multiple table partitions, choose *Continue* to scroll through.

## Result

If you set the option *Create log file* (parameter `-l | -log`), check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.dbw` displays the details.

For more information about how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Showing Index Partitions with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to show information about index partitions.

### Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:
  - BRGUI or BRTOOLS:
    1. Choose *Segment management* *Additional segment functions* *Show index partitions* .

BRGUI or BRTOOLS displays the menu *BRSPACE main options for showing database information*, where you specify the options with which you call BRSPACE.
  - 2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p profile</a>
<i>Database user/password (user)</i>	<a href="#">-u user</a>
<i>Database tablespace (tablespace)</i>	<a href="#">-f dbshow -s tablespace</a>
<i>Database table (table)</i>	<a href="#">-f dbshow -t table</a>
<i>Database index (index)</i>	<a href="#">-f dbshow -i index</a>
<i>Create log file (log)</i>	<a href="#">-f dbshow -l log</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c confirm</a>
<i>Extended output (output)</i>	<a href="#">-o output</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s scroll</a>
<i>Message language (language)</i>	<a href="#">-l language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f dbshow -c ipinfo</a> command that is to be executed using the current settings.

3. Note



4. If required, in *Database table* you can enter the names of multiple tables or indexes by using wildcards. For more information, see “Selecting Objects” in [Segment Management with BR\\*Tools](#).
5. In *Database tablespace*, you can specify multiple objects but you cannot use wildcards.
6. BRSPACE processes all the index partitions in the specified tablespace(s), or the index partitions belonging to the specified owner(s), table(s), or index(es).
- 7.
8. Choose *Continue*

BRGUI or BRTOOLS prompts you to start BRSPACE.

9. Choose *Continue* to start BRSPACE.

- Command line:

Enter at least the following command:




```
brspace -f dbshow -c ipinfo
```

You can enter more parameters if required. For more information, see [BRSPACE -f dbshow](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#) if you set the option *Create log file* (parameter `-l|-log`).

3.  Note

4. If you started BRSPACE from the command line without the information class name (`-c|-class`) – that is, with `brspace -f dbshow` – BRSPACE displays the *Show database information main menu*.

5. Choose  *Show database segment information*  *Show index partitions* .

- 6.

7. If you have already entered a single index name, continue with step 4.

BRSPACE displays the *List of index partitions*:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Owner</i>	Partition owner
<i>Table</i>	Table name
<i>Index</i>	Index name
<i>Partition</i>	Partition name
<i>Tablespace</i>	Tablespace name
<i>Analyzed</i>	Date last analyzed by update

List Entry	Meaning
	statistics
Rows	Number of rows
NextExt[KB ]	Next extent size
MaxExts	Maximum number of extents possible

8. To see more information, select one or more index partitions.

 Example

These examples only apply to input in character mode.

To select the first three entries in the list, enter 1-3.

To select the first and third entries, enter 1, 3.

To select the first three entries and the fifth, enter 1-3, 5.

To select all entries, enter 0.

BRSPACE displays *Information about index partition:*

List Entry	Meaning
Hash partition (hash)	Hash partition: yes or no
Table name (table)	Table name
Tablespace name (tablespace)	Tablespace name
Last analyzed (analyzed)	Date last analyzed by update statistics
Sample size (sample)	Sample size of update statistics
Number of rows (rows)	Number of rows
Next extent size in KB (next)	Next extent size
Maximum number of extents (maxexts)	Maximum number of extents possible

9. If you specified multiple index partitions, choose *Continue* to scroll through.

## Result

If you set the option *Create log file* (parameter `-l|-log`), check the results in the [BRSPACE logs](#).

- The [summary log](#)space<DBSID>.log displays the return code.
- The [detail log](#)s<encoded timestamp>.dbw displays the details.

For more information about how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).

## Showing Segments with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to show information about database segments.

### Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:
  - BRGUI or BRTOOLS:
    1. Choose **► Segment management ► Additional segment functions ► Show segments ◀**.

BRGUI or BRTOOLS displays the menu *BRSPACE main options for showing database information*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Database tablespace (tablespace)</i>	<a href="#">-f dbshow -s -tablespace</a>
<i>Database owner (owner)</i>	<a href="#">-f dbshow -o -owner</a>
<i>Database table (table)</i>	<a href="#">-f dbshow -t -table</a>
<i>Database index (index)</i>	<a href="#">-f dbshow -i -index</a>
<i>Create log file (log)</i>	<a href="#">-f dbshow -l -log</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Extended output (output)</i>	<a href="#">-o -output</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language</i>	<a href="#">-l -language</a>

Menu Entry	Equivalent BRSPACE Command Option
(language)	
BRSPACE command line (command)	This shows you the <a href="#">BRSPACE -f dbshow -c sginfo</a> command that is to be executed using the current settings.

3.  Note

4. If required, in *Database table* or *Database index* you can enter the names of multiple tables or indexes by using wildcards. For more information, see "Selecting Objects" in [Segment Management with BR\\*Tools](#).
5. In *Database tablespace* and *Database owner*, you can specify multiple objects but you cannot use wildcards.
6. BRSPACE processes all the segments in the specified tablespace(s), or all segments belonging to the specified owner(s), table(s), or index(es).

7.

8. Choose *Continue*

BRGUI or BRTOOLS prompts you to start BRSPACE.

9. Choose *Continue* to start BRSPACE.

- o Command line:

Enter at least the following command:

```
brspace -f dbshow -c sginfo
```

You can enter more parameters if required. For more information, see [BRSPACE -f dbshow](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#) if you set the option *Create log file* (parameter `-l|-log`).

3.  Note

4. If you started BRSPACE from the command line without information class name (`-c|-class`) – that is, with `brspace -f dbshow` – BRSPACE displays the *Show database information main menu*.

5. Choose  *Show database segment information*  *Show segments* .


6.

7. BRSPACE displays the *List of database segments*:

List Entry	Meaning
Pos.	List sequence number

List Entry	Meaning
<i>Owner</i>	Segment owner
<i>Type</i>	Segment type: table, index, tabpart, indpart, lobseg, lobind, cluster, rollback, temp, cache
<i>Segment</i>	Segment name
<i>Partition</i>	Partition name if applicable
<i>Tablespace</i>	Tablespace name
<i>Size[KB]</i>	Segment size
<i>Extents</i>	Number of extents
<i>NextExt[KB]</i> <i>]</i>	Next extent size
<i>MaxExts</i>	Maximum number of extents possible

8. To see more information, select one or more segments.

 Example

These examples only apply to input in character mode.

To select the first three entries in the list, enter 1-3.

To select the first and third entries, enter 1, 3.

To select the first three entries and the fifth, enter 1-3, 5.

To select all entries, enter 0.

BRSPACE displays *Information about segment*:

List Entry	Meaning
<i>Segment type (type)</i>	Segment type: table, index, tabpart, indpart, lobseg, lobind, cluster, rollback, temp, cache
<i>Tablespace name (tablespace)</i>	Tablespace name
<i>Segment size in KB (size)</i>	Segment size
<i>Number of extents (extents)</i>	Number of extents
<i>Next extent size in KB (next)</i>	Next extent size
<i>Maximum number of</i>	Maximum number of extents possible

List Entry	Meaning
<i>extents (maxext)</i>	

- If you specified multiple segments, choose *Continue* to scroll through.

## Result

If you set the option *Create log file* (parameter `-l|-log`), check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.dbw` displays the details.

For more information about how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Showing Segment Extents with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to show information about segment extents.

### Procedure

- Start the procedure using BRGUI or BRTOOLS, or from the command line:
  - BRGUI or BRTOOLS:
    - Choose **► Segment management ► Additional segment functions ► Show segment extents ◀**.

BRGUI or BRTOOLS displays the menu *BRSPACE main options for showing database information*, where you specify the options with which you call BRSPACE.

- Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Database tablespace (tablespace)</i>	<a href="#">-f dbshow -s -tablespace</a>
<i>Database file (file)</i>	<a href="#">-f dbshow -f -file</a>
<i>Database owner (owner)</i>	<a href="#">-f dbshow -o -owner</a>
<i>Database table (table)</i>	<a href="#">-f dbshow -t -table</a>

Menu Entry	Equivalent BRSPACE Command Option
<i>Database index (index)</i>	<a href="#">-f dbshow -i -index</a>
<i>Create log file (log)</i>	<a href="#">-f dbshow -l -log</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Extended output (output)</i>	<a href="#">-o -output</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f dbshow -c seinfo</a> command that is to be executed using the current settings.

3.  Note

4. If required, in *Database table* or *Database Index* you can enter the names of multiple tables or indexes by using wildcards. For more information, see "Selecting Objects" in [Segment Management with BR\\*Tools](#).
5. In *Database tablespace*, *Database file*, and *Database owner*, you can specify multiple objects but you cannot use wildcards.
6. BRSPACE processes all segment extents for the specified objects.
- 7.
8. Choose *Continue*  
BRGUI or BRTOOLS prompts you to start BRSPACE.
9. Choose *Continue* to start BRSPACE.

o Command line:

Enter at least the following command:

```
brspace -f dbshow -c seinfo
```

You can enter more parameters if required. For more information, see [BRSPACE -f dbshow](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#) if you set the option *Create log file* (parameter `-l|-log`).

3.  Note

4. If you started BRSPACE from the command line without the information class name (-c|-class) – that is, with `brspace -f dbshow` – BRSPACE displays the *Show database information main menu*.
5. Choose **Show database segment information** **Show segment extents**.
- 6.
7. BRSPACE displays the *List of segment extents*:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Owner</i>	Segment owner
<i>Type</i>	Segment type: table, index, tabpart, indpart, lobseg, lobind, cluster, rollback, temp, cache
<i>Segment</i>	Segment name
<i>Partition</i>	Partition name if applicable
<i>Tablespace</i>	Tablespace name
<i>Extent Id</i>	Extent ID
<i>File Id</i>	File ID
<i>Block Id</i>	Block ID
<i>Size[KB]</i>	Extent size

8. To see more information, select one or more segment extents.

 Example

These examples only apply to input in character mode.

To select the first three entries in the list, enter 1-3.

To select the first and third entries, enter 1, 3.

To select the first three entries and the fifth, enter 1-3, 5.

To select all entries, enter 0.

BRSPACE displays *Information about extent of segment*.

List Entry	Meaning
<i>Segment type (type)</i>	Segment type: table, index, tabpart, indpart, lobseg, lobind, cluster, rollback, temp, cache
<i>Tablespace name (tablespace)</i>	Tablespace name



List Entry	Meaning
<i>Data file (file)</i>	Data file name
<i>Extent Id. (extent_id)</i>	Extent ID
<i>File Id. (file_id)</i>	File ID
<i>Relative file number (file_no)</i>	Relative file number
<i>Block Id. (block_id)</i>	Block ID
<i>Extent size in KB (size)</i>	Extent size

9. If you specified multiple segment extents, choose *Continue* to scroll through.

## Result

If you set the option *Create log file* (parameter `-l|-log`), check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.dbw` displays the details.

For more information about how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Showing Free Extents with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to show information about free extents.

## Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:
  - BRGUI or BRTOOLS:
    1. Choose **► Segment management ► Additional segment functions ► Show free extents ◀**.

BRGUI or BRTOOLS displays the menu *BRSPACE main options for showing database information*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-pl-profile</a>
<i>Database user/password (user)</i>	<a href="#">-ul-user</a>

Menu Entry	Equivalent BRSPACE Command Option
<i>Database tablespace (tablespace)</i>	<a href="#">-f dbshow -s tablespace</a>
<i>Database file (file)</i>	<a href="#">-f dbshow -f file</a>
<i>Create log file (log)</i>	<a href="#">-f dbshow -l log</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c confirm</a>
<i>Extended output (output)</i>	<a href="#">-o output</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s scroll</a>
<i>Message language (language)</i>	<a href="#">-l language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f dbshow -c feinfo</a> command that is to be executed using the current settings.

3.  Note

4. If required, in *Database tablespace* and *Database file*, you can specify multiple objects but you cannot use wildcards.
5. BRSPACE processes all free extents for the specified objects.
- 6.
7. Choose *Continue*.

BRGUI or BRTOOLS prompts you to start BRSPACE.

8. Choose *Continue* to start BRSPACE.

- o Command line:

Enter at least the following command:

```
brspace -f dbshow -c feinfo
```

You can enter more parameters if required. For more information, see [BRSPACE -f dbshow](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#) if you set the option *Create log file* (parameter `-l|-log`).

3.  Note


4. If you started BRSPACE from the command line without the information class name (`-c|-class`) – that is, with `brspace -f dbshow` – BRSPACE displays the *Show database information main menu*.

5. Choose  *Show database segment information*  *Show free extents* .

- 6.
7. BRSPACE displays the *List of free extents*:

List Entry	Meaning
<i>Pos.</i>	List sequence number
<i>Tablespace</i>	Tablespace name
<i>File Id</i>	File ID
<i>Block Id</i>	Block ID
<i>Size[KB]</i>	Extent size

8. To see more information, select one or more free extents.

 Example

These examples only apply to input in character mode.

To select the first three entries in the list, enter 1-3.

To select the first and third entries, enter 1, 3.

To select the first three entries and the fifth, enter 1-3, 5.

To select all entries, enter 0.

BRSPACE displays *Information about free extent*:

List Entry	Meaning
<i>Tablespace name (tablespace)</i>	Tablespace name
<i>Data file (file)</i>	Data file name
<i>File Id. (file_id)</i>	File ID
<i>Relative file number (file_no)</i>	Relative file number
<i>Block Id. (block_id)</i>	Block ID
<i>Extent size in KB (size)</i>	Extent size

9. If you specified multiple free extents, choose *Continue* to scroll through.

## Result

If you set the option *Create log file* (parameter `-l|-log`), check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.dbw` displays the details.

For more information about how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Backup and Database Copy with BR\*Tools

You can back up the database files and offline redo log files of your Oracle database with [BR\\*Tools](#). If you are performing an offline backup of the whole database, the temporary files and online redo log files are also backed up. If you have lost data and want to recover it, the backup files are essential. You can also make a database copy and [back up non-database files and directories](#).

You typically use BR\*Tools for a one-off backup of the:

- Database files, such as after a structure change to the database. For example, after [moving a data file](#), you can back up the database to make the recovery procedure easier.
- Offline redo log files. Oracle copies the online redo log files to the archiving directory, so creating the offline redo log files.



### Recommendation

For routine backups, we recommend one of the following:

- BA Planning Calendar to schedule a backup and then view its log
- A scheduler offered by a supplier of the BACKINT interface
- The scheduler `cron` for UNIX or `at` for Windows



### Caution

If you need to [recover your database](#), you must have access to all offline redo log files that have been written since the database backup. Otherwise you can only recover the database to the point in time of the last available redo log. Therefore, you must always back up the offline redo log files after a database backup, and immediately after an online backup.

In production systems, we strongly recommend you to:

- Run the database in ARCHIVELOG mode with automatic archival turned on. For more information, see [Setting Up Archiving](#).
- Make two copies of the offline redo log files. In test systems, one copy is often sufficient.

## Integration

- BRTOOLS normally calls the SAP tool BRBACKUP or BRARCHIVE to perform the backup. You can also perform a backup directly by calling BRBACKUP or BRARCHIVE from the command line.

## Recommendation

We recommend you to normally use BRTOOLS rather than BRBACKUP or BRARCHIVE. This is because the BRTOOLS menus simplify entry of the correct parameters.

- “Archiver stuck” problem

The Oracle database hangs if it is operated in ARCHIVELOG mode, but the archiving process cannot save the online redo log files because the archiving directory is full. This situation is called archiver stuck. If this occurs you must back up the offline redo log files and delete them from the archive directory as soon as possible.

To avoid archiver stuck, back up the offline redo log files regularly to tape. How often you do this depends on the amount of activity in your SAP System. If a lot of redo log entries are written, and the redo log files are frequently switched, be sure to constantly monitor the archive directory. When necessary, save and delete the offline redo log files.

For more information about avoiding archiver stuck see *SAP Note* [316642](#).

- Make sure the necessary parameters have been set for BRBACKUP and BRARCHIVE in the [Initialization Profile init<DBSID>.sap](#).
- You can back up the offline redo log files regardless of the current status of the database (open or closed) and the SAP system. If you have configured your system according to the SAP recommendations, the Oracle database system saves the online redo log files automatically as offline redo log files. Unless you have changed the standard profile `init<DBSID>.ora`, the offline redo log files are stored in the archive directory `<SAPDATA_HOME>/oraarch`. For more information, see [Setting Up Archiving](#).

## Features

You can perform the following backup functions with BR\*Tools:

- Database backup
- Archivelog backup (that is, backup of the offline redo log files)
- Database copy
- Non-database backup
- Backup of database disk backup
- Verification of database backup
- Verification of archivelog backup
- Additional functions
  - Update of compression rates
  - Preparation of RMAN backups
  - Deletion of database disk backups
  - Deletion of archivelog disk backups

- Controlling of BRARCHIVE run
- Initialization of BRBACKUP tape volumes
- Initialization of BRARCHIVE tape volumes

BRTOOLS calls [BRRESTORE](#) for the verification functions.

## Activities

1. You call the backup function in BRTOOLS and check the displayed backup parameters, changing them as required.

The default parameter values, which are set in the [initialization profile `init<DBSID>.sap`](#), are as follows:

- For database backup, the default is an offline whole database backup to a local tape device without file compression. This means that the online redo log files and control file are backed up as well as the data files.
- For archivelog (that is, offline redo log) file backup, the default is a first copy backup of offline redo log files without deletion to a local tape device without file compression.

### Note

BRTOOLS only lets you change certain parameters for the backup. If you have to make other changes, you must change the `init<DBSID>.sap` profile manually and then restart BRTOOLS.

2. If required, you change the default values for the backup parameters in the initialization profile `init<DBSID>.sap` and restart BRTOOLS.
3. If required, you choose **Backup and Database Copy** **Reset program status** to set the defaults used to the values in the initialization profile `init<DBSID>.sap`. For certain input values, there is no corresponding parameter in the initialization profile, in which case the default value from the BRTOOLS program is used.
4. You start the backup.

If the backup is being made locally or remotely to tapes or disks, then the backup is monitored and an estimation is made of the backup time, based on the elapsed time and the size of the files that still have to be backed up. You also see success or error messages.

5. You check the results of the backup in the [BRBACKUP logs](#) or [BRARCHIVE Logs](#).



## Backing Up the Database with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to back up the database. BRBACKUP performs the backup.

### Note

Remember that the SAP system is not available for production work during an offline backup, since the database is closed. However, you can perform an online backup when the database system and SAP system are running.

## Prerequisites

- Make sure the database is running in ARCHIVELOG mode. For more information, see [Setting Up Archiving](#).
- Make sure you have set the necessary BRBACKUP parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRBACKUP.
- Make sure you have the right backup volumes. When you back up to tape, make sure that you have read the notes on managing and initializing the tapes. For more information, see [Volume Management](#).
- Allow enough time for the backup. [Hardware compression](#) can halve the backup time.

## Procedure

1. Start BRGUI or BRTOOLS.
2. Choose **► Backup and database copy ► Database Backup ◀**.
3. Set the required options:

Menu Entry	Equivalent BRBACKUP Command Option
<i>BRBACKUP profile (profile)</i>	<a href="#">-p -profile</a>
<i>Backup device type (device)</i>	<a href="#">-d -device</a>
<i>Tape volumes for backup (volume)</i>	<a href="#">-v -volume</a>
<i>BACKINT/ Mount profile (parfile)</i>	<a href="#">-r -parfile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Backup type (type)</i>	<a href="#">-t -type</a>
<i>Files for backup (mode)</i>	<a href="#">m -mode</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Query mode (query)</i>	<a href="#">-q -query</a>
<i>Compression mode (compress)</i>	<a href="#">-k -compress</a>
<i>Verification mode (verify)</i>	<a href="#">-w -verify</a>
<i>Fill-up previous backups (fillup)</i>	<a href="#">-f -fillup</a>
<i>Parallel execution (execute)</i>	<a href="#">-e -execute</a>

Menu Entry	Equivalent BRBACKUP Command Option
<i>Additional output (output)</i>	<a href="#">-o -output</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRBACKUP command line (command)</i>	This shows you the <a href="#">BRBACKUP command</a> that is to be executed using the current settings.

4. To start the backup with the selected options, choose *Continue*.
5. Check the results in the [BRBACKUP logs](#).
  - o The [summary log](#) `back<DBSID>.log` displays the return code.
  - o The [detail log](#) `b<encoded timestamp>.<ext>` displays the progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).



## Backing Up the Offline Redo Log Files with BR\*Tools

You can use [BR\\*Tools](#) to back up the offline redo log files. BRARCHIVE performs the backup.



Note

The offline redo log files are called the archivelogs in the BRTOOLS menus to save space.

### Prerequisites

- Make sure the database is running in ARCHIVELOG mode. For more information, see [Setting Up Archiving](#).
- Make sure you have set the necessary BRARCHIVE parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRARCHIVE.
- Make sure you have the right backup volumes. When you back up to tape, make sure that you have read the notes on managing and initializing the tapes. For more information, see [Volume Management](#).
- Allow enough time for the backup. [Hardware compression](#) can halve the backup time.

### Procedure

1. Start BRGUI or BRTOOLS.
2. Choose **Backup and database copy** **Archivelog backup**.
3. Set the required options:



Menu Entry	Equivalent BRARCHIVE Command Option
<i>BRARCHIVE profile (profile)</i>	<a href="#">-p -profile</a>
<i>BRARCHIVE function (function)</i>	<a href="#">-s -sc -ds -dc -sd -scd -ss -ssd -cs -cds</a>
<i>Backup device type (device)</i>	<a href="#">-d -device</a>
<i>Tape volume for backup (volume)</i>	<a href="#">-v -volume</a>
<i>BACKINT/Mount profile (parfile)</i>	<a href="#">-r -parfile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Maximum number of files (number)</i>	<a href="#">-n -number</a>
<i>Back up disk backup (archive)</i>	<a href="#">-a -archive</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Query mode (query)</i>	<a href="#">-q -query</a>
<i>Compression mode (compress)</i>	<a href="#">-k -compress</a>
<i>Verification mode (verify)</i>	<a href="#">-w -verify</a>
<i>Fill mode, group size (fill)</i>	<a href="#">-f -fill</a>
<i>Modify mode, delay apply (modify)</i>	<a href="#">-m -modify</a>
<i>Additional output (output)</i>	<a href="#">-o -output</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRARCHIVE command line (command)</i>	This shows you the <a href="#">BRARCHIVE command</a> that is to be executed using the current settings.

4. To start processing with the selected options, choose *Continue*.
5. Check the results in the [BRARCHIVE logs](#).
  - o The [summary log](#) arch<DBSID>.log displays the backed up archive log files and the return code.
  - o The [detail log](#) a<encoded timestamp>.<ext> displays progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).



## Copying the Database with BR\* Tools

You can use [BR\\*Tools](#) for Oracle to copy the database. BRBACKUP performs the database copy.

### Prerequisites

Make sure that you have set the necessary BRBACKUP parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRBACKUP.

### Procedure

1. Start BRGUI or BRTOOLS.
2. Choose ► *Backup and database copy* ► *Database copy* ◀.
3. Set the required options:

Menu Entry	Equivalent BRBACKUP Command Option
<i>BRBACKUP profile (profile)</i>	<a href="#">-p -profile</a>
<i>Backup device type (device)</i>	<a href="#">-d -device</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Backup type (type)</i>	<a href="#">-t -type</a>
<i>Files for backup (mode)</i>	<a href="#">m -mode</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Query mode (query)</i>	<a href="#">-q -query</a>
<i>Verification mode (verify)</i>	<a href="#">-w -verify</a>
<i>Fill-up previous backups (fillup)</i>	<a href="#">-f -fillup</a>
<i>Parallel execution (execute)</i>	<a href="#">-e -execute</a>
<i>Additional output (output)</i>	<a href="#">-o -output</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRBACKUP command</i>	This shows you the <a href="#">BRBACKUP -d disk_copy or</a>

Menu Entry	Equivalent BRBACKUP Command Option
<i>line (command)</i>	<a href="#">stage_copy</a> command that is to be executed using the current settings.

4. To start the copy with the selected options, choose *Continue*.
5. Check the results in the [BRBACKUP logs](#).
  - o The [summary log](#) `back<DBSID>.log` displays the return code.
  - o The [detail log](#) `b<encoded timestamp>.<ext>` displays the progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).



## Non-Database Backup with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to perform a non-database backup of specified files. BRBACKUP performs the backup.

### Prerequisites

- Make sure you have set the necessary BRBACKUP parameters in the [initialization profile](#) `init<DBSID>.sap`, because BRTOOLS uses these when it calls BRBACKUP.
- Make sure you have the right backup volumes. When you back up to tape, make sure that you have read the notes on managing and initializing the tapes. For more information, see [Volume Management](#).
- Allow enough time for the backup. [Hardware compression](#) can halve the backup time.

### Procedure

1. Start BRGUI or BRTOOLS.
2. Choose **Backup and database copy** **Non-database backup**.
3. Set the required options:

Menu Entry	Equivalent BRBACKUP Command Option
<i>BRBACKUP profile (profile)</i>	<a href="#">-p -profile</a>
<i>Backup device type (device)</i>	<a href="#">-d -device</a>
<i>Tape volumes for backup (volume)</i>	<a href="#">-v -volume</a>
<i>BACKINT/ Mount profile (parfile)</i>	<a href="#">-r -parfile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>

Menu Entry	Equivalent BRBACKUP Command Option
Backup type (type)	<a href="#">-t -type</a>
Files for backup (mode)	<a href="#">m -mode</a>
Confirmation mode (confirm)	<a href="#">-c -confirm</a>
Query mode (query)	<a href="#">-q -query</a>
Compression mode (compress)	<a href="#">-k -compress</a>
Verification mode (verify)	<a href="#">-w -verify</a>
Fill-up previous backups (fillup)	<a href="#">-f -fillup</a>
Parallel execution (execute)	<a href="#">-e -execute</a>
Additional output (output)	<a href="#">-o -output</a>
Message language (language)	<a href="#">-l -language</a>
BRBACKUP command line (command)	This shows you the <a href="#">BRBACKUP -m command</a> that is to be executed using the current settings.

4. To start processing with the selected options, choose *Continue*.
5. Check the results in the [BRBACKUP logs](#).
  - o The [summary log](#) back<DBSID>.log displays the return code.
  - o The [detail log](#) b<encoded timestamp>.rsb displays the progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).



## Backing Up a Database Disk Backup

You can use [BR\\*Tools](#) for Oracle to back up the database. BRBACKUP performs the backup.

### Prerequisites

- Make sure the database is running in ARCHIVELOG mode. For more information, see [Setting Up Archiving](#).
- Make sure you have set the necessary BRBACKUP parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRBACKUP.
- Make sure you have the right backup volumes. When you back up to tape, make sure that you have read the notes on managing and initializing the tapes. For more information, see [Volume Management](#).

- Allow enough time for the backup. [Hardware compression](#) can halve the backup time.

## Procedure

1. Start BRGUI or BRTOOLS.
2. Choose ► *Backup and database copy* ► *Database Backup* ◀.
3. Select the BRBACKUP disk backup that you want to back up.
4. Set the required options:

Menu Entry	Equivalent BRBACKUP Command Option
<i>BRBACKUP profile (profile)</i>	<a href="#">-p -profile</a>
<i>Backup device type (device)</i>	<a href="#">-d -device</a>
<i>Tape volumes for backup (volume)</i>	<a href="#">-v -volume</a>
<i>BACKINT/ Mount profile (parfile)</i>	<a href="#">-r -parfile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Backup type (type)</i>	<a href="#">-t -type</a>
<i>Disk backup for backup (backup)</i>	<a href="#">-b -backup</a>
<i>Delete disk backup (delete)</i>	<a href="#">-bd -backup_delete</a>
<i>Files for backup (mode)</i>	<a href="#">m -mode</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Query mode (query)</i>	<a href="#">-q -query</a>
<i>Compression mode (compress)</i>	<a href="#">-k -compress</a>
<i>Verification mode (verify)</i>	<a href="#">-w -verify</a>
<i>Fill-up previous backups (fillup)</i>	<a href="#">-f -fillup</a>
<i>Parallel execution (execute)</i>	<a href="#">-e -execute</a>
<i>Additional output (output)</i>	<a href="#">-o -output</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>

Menu Entry	Equivalent BRBACKUP Command Option
<i>BRBACKUP command line (command)</i>	This shows you the <a href="#">BRBACKUP command</a> that is to be executed using the current settings.

5. To start the backup with the selected options, choose *Continue*.
6. Check the results in the [BRBACKUP logs](#).
  - o The [summary log](#) `back<DBSID>.log` displays the return code.
  - o The [detail log](#) `b<encoded timestamp>.<ext>` displays the progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).

## Verifying a Database Backup with BR\*Tools




You can use [BR\\*Tools](#) for Oracle to verify a database backup. BRRESTORE performs the verify.

For more information about database verify, see [Backup Verify](#).

### Prerequisites

- Make sure that you have set the necessary BRRESTORE parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRRESTORE.
- Make sure you have the correct volumes for the verify.

### Procedure

1. Start BRGUI or BRTOOLS.
2. Choose  *Backup and database copy*  *Verification of database backup* .
3. Select the BRBACKUP database backup that you want to verify.
4. Set the required options:

Menu Entry	Equivalent BRRESTORE Command Option
<i>BRRESTORE profile (profile)</i>	<a href="#">-p -profile</a>
<i>BRBACKUP backup run (backup)</i>	<a href="#">-b -backup b1 backup1</a>
<i>Verification device type (device)</i>	<a href="#">-d -device</a>
<i>BACKINT/ Mount profile (parfile)</i>	<a href="#">-r -parfile</a>
<i>Database user /password</i>	<a href="#">-u -user</a>

Menu Entry	Equivalent BRRESTORE Command Option
(user)	
Verification mode (verify)	<a href="#">-w -verify</a>
Files for verification (mode)	<a href="#">-m -mode</a>
Confirmation mode (confirm)	<a href="#">-c -confirm</a>
Query mode (query)	<a href="#">-q -query</a>
Compression mode (compress)	<a href="#">-k -compress</a>
Parallel execution (execute)	<a href="#">-e -execute</a>
Additional output (output)	<a href="#">-o -output</a>
Message language (language)	<a href="#">-l -language</a>
BRRESTORE command line (command)	This shows you the <a href="#">BRRESTORE -b -w</a> command that is to be executed using the current settings.

5. To start the verification with the selected options, choose *Continue*.
6. Check the results in the [BRRESTORE logs](#).
  - o The [summary log](#) `rest<DBSID>.log` displays the return code.
  - o The [detail log](#) `r<encoded timestamp>.rsb` displays progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).



## Verifying an Offline Redo Log Backup with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to verify an offline redo log backup, that is, an archive log backup. BRRESTORE performs the verify.

For more information about database verify, see [Backup Verify](#).


### Prerequisites

- Make sure that you have set the necessary BRRESTORE parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRRESTORE.
- Make sure you have the correct volumes for the verify.

## Procedure

1. Start BRGUI or BRTOOLS.
2. Choose ► *Backup and database copy* ► *Verification of archivelog backup* ◀.
3. Select the BRARCHIVE offline redo log file backup that you want to verify.
4. Set the required options:

Menu Entry	Equivalent BRRESTORE Command Option
<i>BRRESTORE profile (profile)</i>	<a href="#">-p -profile</a>
<i>First sequence number (first_seq)</i>	<a href="#">-a -archive -a1 -archive1 &lt;log_no1&gt;</a>
<i>Last sequence number (last_seq)</i>	<a href="#">-a -archive -a1 -archive1 &lt;log_no2&gt;</a>
<i>Verification device type (device)</i>	<a href="#">-d -device</a>
<i>BACKINT/ Mount profile (parfile)</i>	<a href="#">-r -parfile</a>
<i>Database user /password (user)</i>	<a href="#">-u -user</a>
<i>Verification mode (verify)</i>	<a href="#">-w -verify</a>
<i>Verify second copy (sec_copy)</i>	<a href="#">-a2 -archive2</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Query mode (query)</i>	<a href="#">-q -query</a>
<i>Compression mode (compress)</i>	<a href="#">-k -compress</a>
<i>Additional output (output)</i>	<a href="#">-o -output</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRRESTORE command line (command)</i>	This shows you the <a href="#">BRRESTORE -a -w</a> command that is to be executed using the current settings.

5.  Note
6. The following menu options refer to the sequence numbers of the first and last offline redo log file to be verified:
  - *First sequence number (first\_seq)*



- Last sequence number (*last\_seq*)

BRRESTORE sets these according to the BRARCHIVE run that you selected.

If required, you can set these manually. In this case, the corresponding backups must all have been made to the same device type.

7. To start processing with the selected options, choose *Continue*.
8. Check results in the [BRRESTORE logs](#).
  - The [summary log](#) `rest<DBSID>.log` displays the return code.
  - The [detail log](#) `r<encoded timestamp>.rsa` displays the progress.

For more information on how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).



## Additional Functions for Backup and Database Copy with BR\*Tools

The following sections describe additional functions for backup and database copy with BR\*Tools.



## Updating Compression Rates with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to update compression rates. BRBACKUP performs the update. If you use devices with [hardware compression](#), we recommend updating compression rates monthly, as this helps BRBACKUP to make the most efficient use of your tape volumes.

### Prerequisites

Make sure that you have set the necessary BRBACKUP parameters in the [initialization profile `init<DBSID>.sap`](#), because BRTOOLS uses these when it calls BRBACKUP.

### Procedure

1. Start BRGUI or BRTOOLS.
2. Choose **Backup and database copy** **Additional functions** **Update of compression rates**.
3. Choose the required options:

Menu Entry	Equivalent BRBACKUP Command Option
<i>BRBACKUP profile (profile)</i>	<a href="#">-pl-profile</a>

Menu Entry	Equivalent BRBACKUP Command Option
Database user/password (user)	<a href="#">-u -user</a>
BRBACKUP run type (type)	<a href="#">-t -type</a>
Files for compression (mode)	<a href="#">m -mode</a>
Confirmation mode (confirm)	<a href="#">-c -confirm</a>
Query mode (query)	<a href="#">-q -query</a>
Parallel execution (execute)	<a href="#">-e -execute</a>
Additional output (output)	<a href="#">-o -output</a>
Message language (language)	<a href="#">-l -language</a>
BRBACKUP command line (command)	This shows you the <a href="#">BRBACKUP -k only</a> command that is to be executed using the current settings.

4. To start processing with the selected options, choose *Continue*.
5. You check the results in the [BRBACKUP logs](#).
  - The [summary log](#) back<DBSID>.log displays the return code.
  - The [detail log](#) b<encoded timestamp>.cmb displays progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).



## Preparing RMAN Backups with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to prepare RMAN backups. BRBACKUP performs the preparation.

If you back up with the [Oracle Recovery Manager \(RMAN\)](#), we recommend that you perform this procedure once a month. It determines the optimal distribution of files to save sets for an RMAN backup.

### Prerequisites

Make sure that you have set the necessary BRBACKUP parameters in the [initialization profile](#) [init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRBACKUP.

## Procedure

1. Start BRGUI or BRTOOLS.
2. Choose **Backup and database copy** **Additional functions** **Preparation of RMAN backups**.
3. Set the required options:

Menu Entry	Equivalent BRBACKUP Command Option
<i>BRBACKUP profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>BRBACKUP run (type)</i>	<a href="#">-t -type</a>
<i>Files for preparation (mode)</i>	<a href="#">m -mode</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Query mode (query)</i>	<a href="#">-q -query</a>
<i>Parallel execution (execute)</i>	<a href="#">-e -execute</a>
<i>Additional output (output)</i>	<a href="#">-o -output</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRBACKUP command line (command)</i>	This shows you the <a href="#">BRBACKUP -d rman_prep</a> command that is to be executed using the current settings.

4. To start processing with the selected options, choose *Continue*.
5. You check the results in the [BRBACKUP logs](#).
  - o The [summary log](#) `back<DBSID>.log` displays the return code.
  - o The [detail log](#) `b<encoded timestamp>.rmp` displays progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).



## Deleting Database Disk Backups with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to delete database disk backups. BRBACKUP performs the deletion.

 Caution

Make sure that you copy disk backup files to another storage medium, such as tape, before you delete them with this procedure.

## Prerequisites

Make sure that you have set the necessary BRBACKUP parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRBACKUP.

## Procedure

1. Start BRGUI or BRTOOLS.
2. Choose **► Backup and database copy ► Additional functions ► Deletion of database disk backups ◄**.
3. Select the BRBACKUP backup that you want to delete.
4. Set the required options:

Menu Entry	Equivalent BRBACKUP Command Option
<i>BRBACKUP profile (profile)</i>	<a href="#">-pl-profile</a>
<i>Database user/password (user)</i>	<a href="#">-ul-user</a>
<i>BRBACKUP backup run (backup)</i>	<a href="#">-bl-backup</a>
<i>Files for delete (mode)</i>	<a href="#">-ml-mode</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-cl-confirm</a>
<i>Query mode (query)</i>	<a href="#">-ql-query</a>
<i>Additional output (output)</i>	<a href="#">-ol-output</a>
<i>Message language (language)</i>	<a href="#">-ll-language</a>
<i>BRBACKUP command line (command)</i>	This shows you the <a href="#">BRBACKUP -db</a> command that is to be executed using the current settings.

5. To start processing with the selected options, choose *Continue*.
6. You check the results in the [BRBACKUP logs](#).
  - The [summary log](#) back<DBSID>.log displays the return code.
  - The [detail log](#) b<encoded timestamp>.ddb displays progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).



## Deleting Offline Redo Log Backups on Disk with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to delete disk backups of the offline redo log files (that is, the archive logs). BRARCHIVE performs the deletion.

### Caution

Make sure that you copy disk backup files to another storage medium, such as tape, before you delete them with this procedure.

### Caution

BRARCHIVE deletes all available disk backups of offline redo log backups when you perform this procedure, unless you restrict this by setting the *Maximum number of files* (see the table below).

## Prerequisites

Make sure that you have set the necessary BRARCHIVE parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRARCHIVE.

## Procedure

1. Start BRGUI or BRTOOLS.
2. Choose **Backup and database copy** ► **Additional functions** ► **Deletion of archive log disk backups** ◀.
3. Set the required options:

Menu Entry	Equivalent BRARCHIVE Command Option
<i>BRARCHIVE profile (profile)</i>	<a href="#">-pl-profile</a>
<i>Database user/password (user)</i>	<a href="#">-uj-user</a>
<i>Backup device type (device)</i>	<a href="#">-dj-device</a>
<i>Delete after second copy (sec_copy)</i>	<a href="#">-dc</a>
<i>Maximum number of files (number)</i>	<a href="#">-nj-number</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-cj-confirm</a>

Menu Entry	Equivalent BRARCHIVE Command Option
Query mode (query)	<a href="#">-q query</a>
Additional output (output)	<a href="#">-o output</a>
Message language (language)	<a href="#">-l language</a>
BRARCHIVE command line (command)	This shows you the <a href="#">BRARCHIVE -a -ds dc</a> command that is to be executed using the current settings.

4.  Caution
5. Choose as device type the same device that you used for the backup of the disk backup of the offline redo log files (tape or Backint or stage). Do *not* use device type disk, because this deletes the original files in `oraarch` or `saparch`.
- 6.
7. To start processing with the selected options, choose *Continue*.
8. You check the results in the [BRARCHIVE logs](#).
  - o The [summary log](#) `arch<DBSID>.log` displays the return code.
  - o The [detail log](#) `a<encoded timestamp>.<ext>` displays progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).



## Controlling of BRARCHIVE Run with BR\*Tools

You can use [BR\\*Tools](#) to continuously back up the offline redo log files (that is, the archive log files). BRARCHIVE performs the controlling function.

### Prerequisites

Make sure you have set the necessary BRARCHIVE parameters in the [initialization profile `init<DBSID>.sap`](#), because BRTOOLS uses these when it calls BRARCHIVE.

### Procedure

1. Start BRGUI or BRTOOLS.
2. Choose  *Backup and database copy*  *Additional functions*  *Controlling of BRARCHIVE run* .
3. Set the required options:

Menu Entry	Equivalent BRARCHIVE Command Option
BRARCHIVE profile (profile)	<a href="#">-p profile</a>

Menu Entry	Equivalent BRARCHIVE Command Option
<i>Run control function (function)</i>	<a href="#">-f -fill stop suspend resume</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRARCHIVE command line (command)</i>	This shows you the <a href="#">BRARCHIVE -f command</a> that is to be executed using the current settings.

4. To start processing with the selected options, choose *Continue*.
5. You check the results in the [BRARCHIVE logs](#).
  - The [summary log](#) arch<DBSID>.log displays the return code.
  - The [detail log](#) a<encoded timestamp>.fst displays progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).



## Initializing BRBACKUP Tape Volumes with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to initialize BRBACKUP tape volumes. BRBACKUP performs the initialization.

### Prerequisites

- Make sure you have set the necessary BRBACKUP parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRBACKUP.
- Make sure you have the right backup volumes. When you back up to tape, make sure that you have read the notes on managing and initializing the tapes. For more information, see [Volume Management](#).

### Procedure

1. Start BRGUI or BRTOOLS.
2. Choose **Backup and database copy** **Additional Functions** **Initialization of BRBACKUP tape volumes**.
3. Set the required options:

Menu Entry	Equivalent BRBACKUP Command Option
<i>BRBACKUP profile (profile)</i>	<a href="#">-pl -profile</a>
<i>Initialization type (initialize)</i>	<a href="#">-il -initialize</a>

Menu Entry	Equivalent BRBACKUP Command Option
Number of volumes (number)	<a href="#">-n -number</a>
Confirmation mode (confirm)	<a href="#">-c -confirm</a>
Message language (language)	<a href="#">-l -language</a>
Tape volume names (volume)	<a href="#">-v -volume</a>
BRBACKUP command line (command)	This shows you the <a href="#">BRBACKUP -i</a> command that is to be executed using the current settings.

4. To start the initialization with the selected options, choose *Continue*.
5. You check the results in the [BRBACKUP logs](#).
  - o The [summary log](#) `back<DBSID>.log` displays the return code.
  - o The [detail log](#) `b<encoded timestamp>.tib` displays the progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).



## Initializing BRARCHIVE Tape Volumes with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to initialize BRARCHIVE tape volumes. BRARCHIVE performs the initialization.

### Prerequisites

- Make sure you have set the necessary BRARCHIVE parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRARCHIVE.
- Make sure you have the right backup volumes. When you back up to tape, make sure that you have read the notes on managing and initializing the tapes. For more information, see [Volume Management](#).

### Procedure

1. Start BRGUI or BRTOOLS.
2. Choose **Backup and database copy** **Additional Functions** **Initialization of BRARCHIVE tape volumes**.
3. Set the required options:

Menu Entry	Equivalent BRARCHIVE Command Option



Menu Entry	Equivalent BRARCHIVE Command Option
<i>BRARCHIVE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Initialization type (initialize)</i>	<a href="#">-i -initialize</a>
<i>Number of volumes (number)</i>	<a href="#">-n -number</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>Tape volume names (volume)</i>	<a href="#">-v -volume</a>
<i>BRARCHIVE command line (command)</i>	This shows you the <a href="#">BRARCHIVE -i</a> command that is to be executed using the current settings.

4. To start processing with the selected options, choose *Continue*.
5. You check the results in the [BRARCHIVE Logs](#).
  - o The [summary log](#) arch<DBSID>.log displays the return code.
  - o The [detail log](#) a<encoded timestamp>.tia displays the progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).



## Restore and Recovery with BR\*Tools

You can restore and recover your Oracle database with [BR\\*Tools](#).

This section discusses how to perform restore and recovery.

For more information on the approach to restore and recovery, see [Restore and Recovery](#).

### Integration

- BRTOOLS calls the SAP tool BRRECOVER. You can also perform [restore and recovery with SQLPLUS](#), but only if you are an expert.
- Make sure that you have set the necessary parameters for BRRECOVER in the [Initialization Profile init<DBSID>.sap](#).
- You have to have SYSDBA privilege to let BRRECOVER connect to the database. This means that you have to log on as ora<sid> (UNIX) or <sid>adm (Windows).
- The recovery is normally attended, that is, you have to enter parameters as the recovery runs, when prompted by BRRECOVER.

## Features

You can perform the following restore and recovery functions with BR\*Tools:

- Complete database recovery
- Database point-in-time (PIT) recovery
- Tablespace point-in-time (PIT) recovery
- Whole database reset
- Restore of individual backup files
- Restore and application of archivelog (that is, offline redo log) files
- Disaster recovery

## Activities

1. You call the restore and recovery function in BRTOOLS and check the displayed parameters, changing them as required.

The default parameter values, which are set in the [initialization profile `init<DBSID>.sap`](#), are as follows:

- [Recovery type](#) is complete database recovery.
- [Serial recovery](#) for applying offline redo logs, using the Oracle default settings
- [Interval](#) of 30 days for the displayed backups to be restored. This means that only backups from the last 30 days can be selected by default.
- [Scrolling](#) of 20 lines for the list menus, only relevant for character-mode menus

### Note

BRTOOLS only lets you change certain parameters for the backup. If you have to make other changes, you must change the `init<DBSID>.sap` profile manually and then restart BRTOOLS.

2. If required, you change the default values for the restore and recovery parameters in the initialization profile `init<DBSID>.sap` and restart BRTOOLS.
3. If required, you choose **► Restore and recovery ► Reset program status ◀** to set the defaults used to the values in the initialization profile `init<DBSID>.sap`. For certain input values, there is no corresponding parameter in the initialization profile, in which case the default value from the BRTOOLS program is used.
4. You start the restore or recovery.  
  
BRRECOVER takes you through the recovery step by step, displaying sub-menus as required.
5. You check the results in the [BRRECOVER logs](#).

# Complete Database Recovery with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to perform a complete database recovery. BRRECOVER performs the recovery.

## Note

This section discusses how to perform a complete database recovery.

For more information about how to approach a complete database recovery, see [Complete Database Recovery](#).

## Prerequisites

- Make sure you have set the necessary BRRECOVER parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRRECOVER.
- BRRECOVER calls BRRESTORE as necessary to perform restore functions.
- BRRECOVER calls SQLPLUS as necessary to apply offline redo log files.

## Process

1. You start BRGUI or BRTOOLS.
2. You choose **► Restore and recovery ► Complete database recovery ◀**.

You can do one of the following:

- Set the required entries now and run the recovery in unattended mode
- Wait for BRRECOVER to prompt you during the recovery, that is, run the recovery in attended mode

## Recommendation

We recommend you to run the recovery in attended mode.

3. If you want to run the recovery in unattended mode, you select *force* in *Confirmation mode (confirm)*.
4. To start the recovery, you choose *Continue*.

BRRECOVER starts the recovery. You perform the following steps with BRRECOVER.

## Note

If you are using attended mode, BRRECOVER guides you through the recovery, prompting you as necessary.

5. You [check the status of the database files](#).
6. You [select database backups](#).

7. You [restore data files](#).
8. If required, [you restore and apply an incremental backup](#).
9. You [restore and apply the online redo log – that is, archivelog – files](#).
10. You [open the database](#).
11. You check the results in the BRRECOVER and BRRESTORE logs:
  - [BRRECOVER logs](#):
    - The [summary log](#) `recov<DBSID>.log` displays the return code.
    - The [detail log](#) `v<encoded timestamp>.crv` displays the progress.
  - [BRRESTORE logs](#) for the restore functions:
    - The [summary log](#) `rest<DBSID>.log` displays the return code
    - The [detail log](#) `r<encoded timestamp>.<ext>` displays the progress.

For more information about how to view the logs, see [Showing Logs with BR\\*Tools](#).



## Database Point-In-Time Recovery with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to perform a database point-in-time (PIT) recovery. BRRECOVER performs the recovery.



Note

This section discusses how to perform a database PIT recovery.

For more information on how to approach a database PIT recovery, see [Database Point-In-Time Recovery](#).

### Prerequisites

- Make sure you have set the necessary BRRECOVER parameters in the [initialization profile](#) `init<DBSID>.sap`, because BRTOOLS uses these when it calls BRRECOVER.
- BRRECOVER calls BRRESTORE as necessary to perform restore functions.
- BRRECOVER calls SQLPLUS as necessary to apply offline redo log files.

### Process

1. You start BRGUI or BRTOOLS.
2. You choose **► Restore and recovery ► Database point-in-time recovery ◄**.

You can do one of the following:

- Set the command line now and run the recovery in unattended mode

- Wait for BRRECOVER to prompt you during the recovery, that is, run the recovery in attended mode

 Recommendation

We recommend you to normally run the recovery in attended mode.

If you are sure of the required entries – for example, for a routine database copy – you can run the recovery in unattended mode.

3. If you want to run the recovery in unattended mode, you select *force* in *Confirmation mode (confirm)*.
4. To start the recovery, you choose *Continue*.

BRRECOVER starts the recovery. You perform the following steps with BRRECOVER:

 Note

If you are using attended mode, BRRECOVER guides you through the recovery, prompting you as necessary.

5. You [set a point in time for the recovery](#).
  6. You [select database backups](#) or flashback.
  7. You [check the status of the database files](#).
- If you are using flashback database, continue with step 12 below.
8. If required, you [restore the control files](#).
  9. You [restore data files](#).
  10. If required, you [restore and apply an incremental backup](#).
  11. You [restore and apply the offline redo log – that is, archivelog – files](#)
  12. You [restore the offline redo log – that is, archivelog – files and perform the flashback database](#).
  13. You [open the database](#).

14. You check the results in the BRRECOVER and BRRESTORE logs:
  - [BRRECOVER logs](#):
    - The [summary log](#) `recov<DBSID>.log` displays the return code.
    - The [detail log](#) `v<encoded timestamp>.dpt` displays the progress.
  - [BRRESTORE logs](#) for the restore functions:
    - The [summary log](#) `rest<DBSID>.log` displays the return code
    - The [detail log](#) `r<encoded timestamp>.<ext>` displays the progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).

# Tablespace Point-In-Time Recovery with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to perform a tablespace point-in-time (PIT) recovery. BRRECOVER performs the recovery.

## Note

This section discusses how to perform a tablespace PIT recovery.

For more information on how to approach a tablespace PIT recovery, see [Tablespace Point-in-Time Recovery](#).

Tablespace PIT recovery is especially useful for Multiple Components in One Database (MCOB). It allows you to restore the tablespaces for a single component – for example, if a component upgrade has failed – without affecting the other components in the same database.

## Prerequisites

- Make sure you have set the necessary BRRECOVER parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRRECOVER.
- BRRECOVER calls BRRESTORE as necessary to perform restore functions.
- BRRECOVER calls SQLPLUS as necessary to apply offline redo log files.
- BRRECOVER calls the Oracle EXP and IMP tools to export and import the tablespaces that are not to be recovered.
- To start this procedure, you must be able to open the database.
- BRRECOVER opens the database in `RESTRICT` mode – that is, only the database administrator (DBA) can access the database. For SAP systems, including all other components installed in the same database, the database is unavailable.
- BRRECOVER always recovers the `SYSTEM` and `UNDO` tablespaces in addition to the tablespaces that you select.

## Process

1. You start BRGUI or BRTOOLS.
2. You choose  *Restore and recovery*  *Tablespace point-in-time recovery* .

You can do one of the following:

- Set the command line now and run the recovery in unattended mode
- Wait for BRRECOVER to prompt you during the recovery, that is, run the recovery in attended mode

## Recommendation

We recommend you to normally run the recovery in attended mode.

3. If you want to run the recovery in unattended mode, you select *force* in *Confirmation mode (confirm)*.
4. To start the recovery, you choose *Continue*.

BRRECOVER starts the recovery. You perform the following steps with BRRECOVER.

 Note

If you are running in attended mode, BRRECOVER guides you through the recovery, prompting you as necessary.

5. You [set a point in time and tablespaces for the recovery](#).
6. You [select database backups](#).
7. You [check the status of the tablespaces](#).
8. You [export the tablespaces that are not required for the recovery](#).
9. You [restore the data files](#).
10. If required, you [restore and apply an incremental backup](#).
11. You [restore and apply the online redo log - that is, archive log - files](#)
12. You [open the database and plug in the previously exported tablespaces](#).
13. You check the results in the BRRECOVER and BRRESTORE logs:
  - [BRRECOVER logs](#):
    - The [summary log](#) `recov<DBSID>.log` displays the return code.
    - The [detail log](#) `v<encoded timestamp>.tpt` displays the progress.
  - [BRRESTORE logs](#) for the restore functions:
    - The [summary log](#) `rest<DBSID>.log` displays the return code.
    - The [detail log](#) `r<encoded timestamp>.<ext>` displays the progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).

## Whole Database Reset with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to perform a whole database reset. BRRECOVER performs the reset.

 Note

This section discusses how to perform a whole database reset.

For more information on how to approach a whole database reset, see [Whole Database Reset](#).

## Prerequisites

- Make sure you have set the necessary BRRECOVER parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRRECOVER.
- BRRECOVER calls BRRESTORE as necessary to perform restore functions.
- BRRECOVER calls SQLPLUS as necessary to apply offline redo log files.

## Process

1. You start BRGUI or BRTOOLS.
2. You choose **► Restore and recovery ► Whole database reset ◀**.

You can do one of the following:

- Set the command line now and run the reset in unattended mode
- Wait for BRRECOVER to prompt you during the reset, that is, run the reset in attended mode

### Recommendation

We recommend you to normally run the reset in attended mode.

If you are sure of the required entries - for example, for a routine database copy - you can run the reset in unattended mode.

3. If you want to run the reset in unattended mode, you select *force* in *Confirmation mode (confirm)*.
4. To start the reset, you choose *Continue*.

BRRECOVER starts the reset. You perform the following steps with BRRECOVER.

### Note

If you are running in attended mode, BRRECOVER guides you through the reset, prompting you as necessary.

5. You [select a consistent database backup](#) or a restore point.
6. You [check the status of the database files](#).  
If you are using a restore point, continue with step 11 below.
7. You [restore the control files and the offline redo log files](#). You only restore the offline redo log files if you selected a consistent online backup.
8. You [restore data files](#).
9. If required, you [apply an incremental backup](#).
10. You [apply the offline redo log – that is, archive log – files](#) if they were restored in step 7.



11. You [restore offline redo log – that is, archivelog – files and flash back to the restore point](#).
12. You [open the database](#).
13. You check the results in the BRRECOVER and BRRESTORE logs:
  - [BRRECOVER logs](#):
    - The [summary log](#) `recov<DBSID>.log` displays the return code.
    - The [detail log](#) `v<encoded timestamp>.drs` displays the progress.
  - [BRRESTORE logs](#) for the restore functions:
    - The [summary log](#) `rest<DBSID>.log` displays the return code
    - The [detail log](#) `r<encoded timestamp>.<ext>` displays the progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).

## Restore of Individual Backup Files with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to restore individual backup files. BRRECOVER calls BRRESTORE to perform the restore.

### Caution

Only perform this procedure if you are an expert in Oracle database administration. Even as an expert, we recommend that you first try other types of recovery supported by BR\*Tools.

If you are not an expert, you might severely damage the database if you use this procedure. You might lose data and cause downtime to fix the problem.

### Note

This section discusses how to perform a restore of individual backup files.

For more information on how to approach restore of individual backup files, see [Restore of Individual Backup Files](#).

## Prerequisites

- Make sure you have set the necessary BRRECOVER parameters in the [initialization profile](#) `init<DBSID>.sap`, because BRTOOLS uses these when it calls BRRECOVER.
- Since this function is designed for experts, the usual safeguards are not present:
  - There is no database status check.
  - You have less guidance than otherwise:
    - BRRECOVER does not always use the BR\*Tools logs to guide you.

- BRRECOVER does not guide you step by step through restore and recovery.
  - You can choose individual actions independently.
  - The database is closed for the shortest time possible.
- You must meet the following prerequisites before starting the restore:

Procedure	Limitations	BRBACKUP logs
<i>Restore files from BRBACKUP backup</i>	You can only restore files saved by BRBACKUP, not by BRARCHIVE.	BRBACKUP summary and detail logs required
<i>Restore individual files from tape</i>	You cannot restore files from an RMAN backup.	No logs required
<i>Restore individual files from disk</i>	You cannot restore directories from a disk backup.	No logs required
<i>Restore individual files from backup utility</i>	You cannot restore files from an rman_util backup.	No logs required
<i>Restore and apply incremental backup</i>	You can only apply incremental backup, not offline redo log files.	BRBACKUP summary and detail logs required

- For all procedures, the backup medium with the required backup files must obviously be present.

## Process

1. You start BRGUI or BRTOOLS.
2. You choose **► Restore and recovery ► Restore of individual backup files ◀◀**.
3. You choose the required procedure from the list shown in the above table.

BRRECOVER starts the restore.

Processing now depends on which procedure you chose.

4. If you chose *Restore files from BRBACKUP backup* or *Apply incremental backup*:

1. You [select a single database backup](#).

If you are sure it contains the data that you require, you can select a backup that terminated with errors.

If you are performing *Restore files from BRBACKUP backup*, BRRECOVER displays a list of data files in the selected backup.

2. You select the files that you want to restore.

This does not apply to *Restore and apply incremental backup*, because changes to all data files are stored in one incremental save set.

3. You [restore data files](#) or [restore and apply incremental backup](#).

5. If you chose one of the remaining procedures:



Note

You need to know exactly which file to restore and where it is. With these procedures, you are effectively performing a copy at operating-system level.

You cannot use these procedures to restore an RMAN backup, except an RMAN backup to disk, which is effectively a one-to-one copy of database files.

1. You specify the location of the file:
  - The position on tape for *Restore individual files from tape*
  - The name of the backup file on disk for *Restore individual files from disk*
  - The file name and `back_id` for *Restore individual files from backup utility*

For all these procedure, you can also specify the destination for the restore.

If a disk or backup utility is involved in the restore, you must specify the full path to the file that you want to restore.

2. You perform the restore, which uses the following main BRRESTORE parameters:
  - [-n|-number](#) for *Restore individual files from tape*
  - [-n2|-number2](#) for *Restore individual files from disk*
  - [-b2|-back2](#) for *Restore individual files from backup utility*

6. You check the results in the BRRECOVER and BRRESTORE logs:
  - [BRRECOVER logs](#):
    - The [summary log](#) `recov<DBSID>.log` displays the return code.
    - The [detail log](#) `v<encoded timestamp>.rif` displays the progress.
  - [BRRESTORE logs](#) for the restore functions:
    - The [summary log](#) `rest<DBSID>.log` displays the return code
    - The [detail log](#) `r<encoded timestamp>.rif` displays the progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).

## Example

This example shows the contents of the BRRECOVER detail log file for *Restore files from BRBACKUP backup*.

```
BOR655I Choice menu 120 - please decide how to proceed
```

```
-----  
Restore of individual backup files  
1 = Restore files from BRBACKUP backup  
2 - Restore individual files from tape  
3 - Restore individual files from disk
```

- 4 - Restore individual files from backup utility
- 5 - Restore and apply incremental backup
- 6 - Exit program
- 7 - Reset program status

Standard keys: c - cont, b - back, s - stop, r - refr, h - help

-----  
 BR0662I Enter your choice:

BR0280I Time stamp 2003-01-29 19.05.25

BR0663I Your choice: 'c'

BR0259I Program execution will be continued...

BR0699I Reading log file /oracle/GC2/sapbackup/backGC2.log ...

BR0280I Time stamp 2003-01-29 19.05.25

BR0658I List menu 121 - please select one entry

-----  
 BRBACKUP database backups for restore

Pos. Log Start Type Files Device Rc

- 1 = bdjwhckx.ffd 2003-01-29 17.30.51 offline 17/17 disk 0
- 2 - bdjwhadu.fft 2003-01-29 17.05.14 offline 17/17 tape 1
- 3 - bdjwgyrq.fff 2003-01-29 16.48.42 offline 17/17 util\_onl 0
- 4 - bdjwgtj.fnt 2003-01-29 16.26.55 onl\_cons 17/17 tape 0
- 5 - bdjwgvvh.fnf 2003-01-29 16.16.29 onl\_cons 17/17 util\_onl 0
- 6 - bdjwcfgm.ffd 2003-01-28 17.48.54 offline 17/17 disk 0
- 7 - bdjvdblz.fff 2003-01-23 14.52.03 offline 17/17 util\_onl 0

.....

BR0280I Time stamp 2003-01-29 19.05.29

BR0663I Your selection: '5'

BR0699I Reading log file /oracle/GC2/sapbackup/bdjwgvvh.fnf ...

BR0280I Time stamp 2003-01-29 19.05.29

BR0659I List menu 122 + please select one or more entries

-----  
 Backup files for restore

Pos. Tablespace Id. Name

- 1 - DRSYS 3 /oracle/GC2/sapdata1/drsys\_1/drsys.data1

.....  
9 - PSAPTESTD 2 /oracle/GC2/sapdata6/testd\_1/testd.data1  
10 - PSAPTESTD 12 /oracle/GC2/sapdata6/testd\_2/testd.data2  
11 - PSAPTESTI 11 /oracle/GC2/sapdata5/testi\_1/testi.data1  
.....  
18 - 0 /oracle/GC2/sapbackup/cntrlGC2.dbf  
19 - archive\_log /oracle/GC2/saparch/1\_8.dbf  
20 - archive\_log /oracle/GC2/saparch/1\_9.dbf  
Standard keys: c - cont, b - back, s - stop, r - refr, h - help

-----  
BR0662I Enter your selection:  
BR0280I Time stamp 2003-01-29 19.05.36  
BR0663I Your selection: '9-11'  
BR0280I Time stamp 2003-01-29 19.05.36  
BR0657I Input menu 123 - please check/enter input values

-----  
BRRESTORE main options for restore from BRBACKUP backup  
1 - BRRESTORE profile (profile) ..... [initGC2.sap]  
2 - BRBACKUP backup run (backup) ..... [bdjwgvvh.fnf]  
3 - Fill-up previous restores (fillup) . [no]  
4 - Restore device type (device) ..... [util\_file]  
5 - BACKINT/Mount profile (parfile) .... [initGC2.utl]  
6 # Database user/password (user) ..... [system/\*\*\*\*\*]  
7 ~ Restore destination (rest\_dest) .... []  
8 - Files for restore (mode) ..... [2,11-12]  
Standard keys: c - cont, b - back, s - stop, r - refr, h - help

-----  
BR0662I Enter your choice:  
BR0280I Time stamp 2003-01-29 19.05.37  
BR0663I Your choice: 'c'  
BR0259I Program execution will be continued...  
BR0280I Time stamp 2003-01-29 19.05.37  
BR0657I Input menu 124 - please check/enter input values

```

-----
Additional BRRESTORE options for restore from BRBACKUP backup
1 - Confirmation mode (confirm) ..... [yes]
2 - Query mode (query) ..... [no]
3 # Compression mode (compress) ..... [no]
4 # Parallel execution (execute) ..... [0]
5 - Additional output (output) ..... [no]
6 - Message language (language) ..... [E]
7 - BRRESTORE command line (command) . [-p initGC2.sap -b
bdjwgvvh.fnf -d util_file -r /oracle/GC2/dbs/initGC2.utl -m 2,11-12 -
1 E]

```

Standard keys: c - cont, b - back, s - stop, r - refr, h - help

```

-----
BR0662I Enter your choice:
BR0280I Time stamp 2003-01-29 19.05.39
BR0663I Your choice: 'c'
BR0259I Program execution will be continued...
BR0342I Database instance GC2 is open
BR0064I Database instance GC2 will be shut down now
.....
BR291I BRRESTORE will be started with options '-p initGC2.sap -b
bdjwgvvh.fnf -d util_file -r /oracle/GC2/dbs/initGC2.utl -m 2,11-12 -
1 E'
BR0280I Time stamp 2003-01-29 19.05.49
BR0670I Enter 'c[ont]' to continue, 'b[ack]' to go back, 's[top]' to
abort:
BR0280I Time stamp 2003-01-29 19.06.04
BR0257I Your reply: 'c'
BR0259I Program execution will be continued...
=====
BR401I BRRESTORE 6.40 (0)
BR405I Start of file restore: rdjwhkwq.rsb 2003-01-29 19.06.04
BR0428W File /oracle/GC2/sapdata6/testd_1/testd.data1 will be
overwritten
BR0428W File /oracle/GC2/sapdata6/testd_2/testd.data2 will be
overwritten

```

```

BR0428W File /oracle/GC2/sapdata5/testi_1/testi.data1 will be
overwritten

BR0280I Time stamp 2003-01-29 19.06.04

BR0256I Enter 'c[ont]'' to continue, 's[top]'' to cancel the program:

BR0280I Time stamp 2003-01-29 19.06.05

BR0257I Your reply: 'c'

BR0259I Program execution will be continued...

.....

BR0280I Time stamp 2003-01-29 19.06.06

BR0229I Calling backup utility...

BR0280I Time stamp 2003-01-29 19.08.04

#FILE..... /oracle/GC2/sapdata6/testd_2/testd.data2

#RESTORED. 1043853566

BR0280I Time stamp 2003-01-29 19.08.04

#FILE..... /oracle/GC2/sapdata6/testd_1/testd.data1

#RESTORED. 1043853582

BR0280I Time stamp 2003-01-29 19.08.04

#FILE..... /oracle/GC2/sapdata5/testi_1/testi.data1

#RESTORED. 1043853591

BR0280I Time stamp 2003-01-29 19.08.04

BR0374I 3 of 3 files restored by backup utility

BR0230I Backup utility called successfully

BR0406I End of file restore: rdjwhkwq.rsb 2003-01-29 19.08.04

BR0280I Time stamp 2003-01-29 19.08.04

BR0403I BRRESTORE terminated successfully with warnings
=====

```

## Restore and Application of Offline Redo Log Files with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to restore and apply offline redo log files – that is, archivelog files. BRRESTORE restores offline redo log files and BRRECOVER applies the offline redo log files.

 Caution

Only perform this procedure if you are an expert in Oracle database administration. Even as an expert, we recommend that you first try other types of recovery supported by BR\*Tools.

If you are not an expert, you might severely damage the database with this procedure. You might lose data and cause downtime to fix the problem.

 Note

This section discusses how to perform a restore and apply of offline redo log files.

For more information about how to approach restore and apply of offline redo log files, see [Restore and Application of Offline Redo Log Files](#).

## Prerequisites

- Make sure you have set the necessary BRRECOVER parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRRECOVER.
- Since this function is designed for experts, the usual safeguards are not present:
  - There is no database status check.
  - You have less guidance than otherwise, since BRRECOVER does not guide you step by step through restore and recovery.
  - You can choose individual actions independently.
  - The database is closed for the shortest time possible.
  - You can overwrite existing offline redo log files if required.
- You must meet the following prerequisites before starting the restore or apply:

Procedure	Requirements
<i>Restore archivelog files</i>	<ul style="list-style-type: none"> <li>○ Check that there is enough free space in the restore directory</li> <li>○ The BRARCHIVE summary log exists</li> </ul>
<i>Apply archivelog files</i>	Identify the offline redo log sequence number that the database requires to start the apply.
<i>Open database</i>	Make sure that all database files are consistent before trying to open the database.

- BRRECOVER calls SQLPLUS to apply the offline redo log files.
- For *Apply archivelog files*, you can force application of the *online* redo logs – that is, the online redo logs that have been archived but not yet overwritten.

## Process


1. You start BRGUI or BRTOOLS.
2. You choose  *Restore and recovery*  *Restore and application of archivelog files* .
3. You choose the required procedure.



BRRECOVER starts the restore or apply.


- If you chose *Restore archivelog files*, you set the required options and choose Continue to start the restore

Menu Entry	Equivalent BRRESTORE Command Options
<i>BRRESTORE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Instance of archivelog thread (instance)</i> Oracle Real Application Cluster (RAC) only	<a href="#">-a -archive &lt;DBSID&gt;</a>
<i>First sequence number (first_seq)</i>	<a href="#">-a -archive &lt;log_no1&gt;-&lt;log_no2&gt;</a>
<i>Last sequence number (last_seq)</i>	<a href="#">-a -archive &lt;log_no1&gt;-&lt;log_no2&gt;</a>
<i>Restore device type (device)</i>	<a href="#">-d -device</a>
<i>BACKINT/Mount profile (parfile)</i>	<a href="#">-r -parfile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Restore second copy (sec_copy)</i>	<a href="#">-a2 -archive2</a>
<i>Destination directory (dest_dir)</i>	<a href="#">-a -archive &lt;log_no1&gt;-&lt;log_no2&gt; rest_dir</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Query mode (query)</i>	<a href="#">-q -query</a>
<i>Compression mode (compress)</i>	<a href="#">-k -compress</a>
<i>Additional output (output)</i>	<a href="#">-o -output</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRRESTORE command line (command)</i>	This shows you the <a href="#">BRRESTORE command</a> that is to be executed using the current settings.

-  Note
- Restore device type (device)* is taken from the device type used for the backup containing the selected offline redo log files.
- All selected offline redo log files must be backed up on the same backup device type.

8. *Restore second copy (sec\_copy)* lets you use the second copy of the restore if the first is damaged or unavailable.
- 9.
10. If you chose *Apply archivelog files*, you set the required options and choose Continue to start the apply:

Menu Entry	Meaning
<i>Archivelog directory (archive_dir)</i>	The source directory used by SQLPLUS to apply the redo log files
<i>Instance of archivelog thread (instance)</i>	Database instance that created the offline redo log files
<i>Apply from redo log group (apply_redo)</i>	Apply changes recorded in the online redo log group
<i>First sequence number (first_seq)</i>	The sequence number of the first offline redo log file to apply
<i>Last sequence number (last_seq)</i>	The sequence number of the last offline redo log file to apply
<i>Last system change number (last_scn)</i>	Specifies that the recovery finishes with an Oracle system change number (SCN)
<i>End point-in-time (end_pit)</i>	Specifies that the recovery finishes with a normal point in time, using the time stamp (that is, date, hours, minutes, and seconds)
<i>Use backup control file (back_ctl)</i>	Option for SQLPLUS RECOVER command
<i>Parallel recovery (degree)</i>	Specifies whether the recovery is serial or parallel
<i>SQLPLUS command (command)</i>	This shows you the SQLPLUS command that is to be executed using the current settings.

11.  Note
12. You *must* enter *First sequence number (first\_seq)* and *Last sequence number (last\_seq)*. In this case, all the offline redo logs in the sequence you specify are applied. If required, you can also enter *Last system change number (last\_scn)* or *End point-in-time (end\_pit)*. In this case, only changes in the logs up till the `last_scn` or `end_pit` that you specify are applied.
- 13.
14. If you chose *Open database*, you [open the database](#).
15. You check the results in the BRRECOVER and BRRESTORE logs:
  - o [BRRECOVER logs](#):
    - The [summary log](#) `recov<DBSID>.log` displays the return code.
    - The [detail log](#) `v<encoded timestamp>.rsa` displays the progress.

- [BRRESTORE logs](#) for the restore functions:
  - The [summary log](#) `rest<DBSID>.log` displays the return code
  - The [detail log](#) `r<encoded timestamp>alf` displays the progress.

For more information about how to view the logs, see [Showing Logs with BR\\*Tools](#).

## Example

This example shows the contents of the BRRECOVER detail log file for *Apply archivelog files*.

```
BR0655I Choice menu 131 - please decide how to proceed
-----

Restore and application of archivelog files

1 = Restore archivelog files
2 - Apply archivelog files
3 - Open database
4 - Exit program
5 - Reset program status

Standard keys: c - cont, b - back, s - stop, r - refr, h - help
-----

BR0662I Enter your choice:
BR0280I Time stamp 2003-01-29 19.08.51
BR0663I Your choice: '2'
BR0613I Database instance GC2 is shut down
BR0750I Database instance GC2 will be mounted now
.....
BR0280I Time stamp 2003-01-29 19.10.55
BR0657I Input menu 134 - please check/enter input values
-----

Apply archivelog files to database instance GC2

1 - Archivelog directory (archive_dir) ..... [/oracle/GC2/saparch]
2 # Instance of archivelog thread (instance) . []
3 - Apply from redolog group (apply_redo)..... []
4 - First sequence number (first_seq) ..... [9]
5 - Last sequence number (last_seq) ..... [10]
6 ~ Last system change number (last_scn) ..... []
```

```

7 ~ End point-in-time (end_pit) ..... []
8 - Use backup control file (back_ctl) ..... [no]
9 ~ Parallel recovery (degree) ..... []
10 - SQLPLUS command (command) ..... [recover from
'/oracle/GC2/saparch' database]

Standard keys: c - cont, b - back, s - stop, r - refr, h - help
-----

BR0662I Enter your choice:
BR0280I Time stamp 2003-01-29 19.10.56
BR0663I Your choice: 'c'
BR0259I Program execution will be continued...

BR0783I Archivelog files with sequence number 9-10 will be applied to
database GC2
BR0280I Time stamp 2003-01-29 19.10.56
BR0670I Enter 'c[ont]' to continue, 'b[ack]' to go back, 's[top]' to
abort:
BR0280I Time stamp 2003-01-29 19.10.58 BR0257I Your reply: 'c'
BR0259I Program execution will be continued...
BR0280I Time stamp 2003-01-29 19.10.58
BR0336I Applying offline redo log file /oracle/GC2/saparch/1_9.dbf
...
BR0336I Applying offline redo log file /oracle/GC2/saparch/1_10.dbf
...
BR0280I Time stamp 2003-01-29 19.10.58
BR0337I Offline redo log file /oracle/GC2/saparch/1_9.dbf applied
successfully
BR0280I Time stamp 2003-01-29 19.10.59
BR0337I Offline redo log file /oracle/GC2/saparch/1_10.dbf applied
successfully
=====

```

## Disaster Recovery with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to recover from a disaster. You use BRRECOVER to restore missing or damaged profiles and log files. BRRESTORE is *not* called by disaster recovery, because it always requires the [BRBACKUP Logs](#) or the [BRARCHIVE Logs](#), which might not be available in a disaster.

You use this process if either of the following is true:

- You have lost the entire database, including the profiles and the BRBACKUP logs or the BRARCHIVE logs.
- You have only lost the profiles and the BRBACKUP or BRARCHIVE logs.

This process does *not* actually restore data files or recover redo log files. It only restores the profiles and BRBACKUP or BRARCHIVE logs from copies that you made during previous backups. Therefore, it prepares the database for you to perform one of the following guided BR\*Tools options:

- [Database point-in-time \(PIT\) recovery](#)
- [Whole database reset](#)

Complete database recovery and database PIT recovery are not possible after disaster recovery because the current control file (required for complete database recovery) is missing and the database cannot be opened (required for tablespace PIT recovery).

 Caution

Only perform this process if you are an expert in Oracle database administration. Even as an expert, we recommend that you first try other types of recovery supported by BR\*Tools.

If you are not an expert, you might severely damage the database with this procedure. You might lose data and cause downtime to fix the problem.

 Note

This section discusses how to perform disaster recovery.

For more information on how to approach disaster recovery, see [Disaster Recovery](#).

## Prerequisites

- Since this function is designed for experts, the usual safeguards are not present:
  - There is no database status check.
  - You have less guidance than otherwise, since BRRECOVER does not guide you step by step through restore and recovery.
  - You can choose individual actions independently.
- For all procedures, the backup medium with the required backup files must obviously be present.
- You need to know exactly which file to restore and where it is. You are effectively performing a copy at operating-system level.
- BRRECOVER restores the profiles and logs to the standard directory.
- If you choose device type *Backup utility* for the restore, note the following:
  - The BACKINT repository with the latest backup must be available because the tapes are administered using this repository in the backup utility.

- The BACKINT parameter file must normally exist, depending on the specific implementation that you are using. If it is required but is unavailable, you must first try and recreate it before performing disaster recovery.
- BRRECOVER calls BACKINT to perform the restore.
- BACKINT performs the restore from the profiles or logs of the latest backup.

## Process

1. You start BRGUI or BRTOOLS.
2. You choose **► Restore and recovery ► Disaster recovery ◄**.
3. You choose the required procedure:
  - *Restore profiles and logs files from BRBACKUP backup*
  - *Restore profiles and logs files from BRARCHIVE backup*

BRRECOVER starts the restore and displays the menu *Device Type*.

4. You choose the device tape where the backups of the profiles or logs are stored.

BRRECOVER displays the parameters for restoring the profiles or logs, depending on what kind of device type you specified. The default parameters are taken from the current profile, [Initialization Profile init<DBSID>.sap](#), if available.

5. For a disk backup or a utility backup, you note the following:
  - For a disk backup, you can specify the backup directory. BRRECOVER looks in the sub-directory <DBSID> of the specified directory to find the summary log for BRBACKUP or BRARCHIVE. It uses the information there for the restore.
  - For a utility backup, check *Prerequisites* above.

6. You choose *Continue* to continue the restore with the displayed parameters.

BRRECOVER warns you that the profiles and logs might be overwritten.

BRRECOVER displays the restore menu where you can specify which profiles and logs to restore:

- If a log or profile already exists on disk, the recommendation is *No* to avoid overwriting it.
  - If a log or profile does not exist on disk, the recommendation is *Yes* to let you restore it.
7. If required, you change the recommended values for restoring the profiles and logs. You can select several profiles or logs to restore.
  8. You choose *Continue* to start restoring the selected profiles or logs.
  9. You check the results in the [BRRECOVER logs](#):
    - The [summary log](#) `recov<DBSID>.log` displays the return code.
    - The [detail log](#) `v<encoded timestamp>.drv` displays the progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).

## Example

This example shows the contents of the BRRECOVER detail log file for *Restore profiles and log files from BRBACKUP backup*, using a local tape.

```
BR0655I Choice menu 136 - please decide how to proceed
-----
Disaster recovery main menu
1 = Restore profiles and log files from BRBACKUP backup
2 - Restore profiles and log files from BRARCHIVE backup
3 - Exit program
4 - Reset program status
Standard keys: c - cont, b - back, s - stop, r - refr, h - help
-----
BR0662I Enter your choice:
BR0280I Time stamp 2003-01-31 18.18.27
BR0663I Your choice: 'c'
BR0259I Program execution will be continued...
BR0280I Time stamp 2003-01-31 18.18.27
BR0656I Choice menu 137 - please make a selection
-----
Device type for restoring profiles and log files from BRBACKUP backup
1 = Local tape
2 - Remote tape
3 - Local disk
4 - Remote disk
5 - Backup utility
Standard keys: c - cont, b - back, s - stop, r - refr, h - help
-----
BR0662I Enter your choice:
BR0280I Time stamp 2003-01-31 18.18.28
BR0663I Your choice: 'c'
BR0259I Program execution will be continued...
BR0280I Time stamp 2003-01-31 18.18.28
BR0657I Input menu 138 - please check/enter input values
```

-----  
Parameters for restoring profiles and log files from local BRBACKUP  
tape

- 1 - Backup profile (profile) .....  
[/oracle/GC2/dbs/initGC2.sap]
  - 2 - Tape drive with no-rewind (tape\_address) .. [/dev/rmt/1mn]
  - 3 - Tape drive with rewind (tape\_address\_rew) . [/dev/rmt/1m]
  - 4 - Tape rewind command (rewind) ..... [mt -f \$ rewind]
  - 5 - Tape positioning command (tape\_pos\_cmd) ... [mt -f \$ fsf \$]
  - 6 - Cpio input options (cpio\_in\_flags) ..... [-iuv -C 65536]
- Standard keys: c - cont, b - back, s - stop, r - refr, h - help

-----  
BR0662I Enter your choice:

BR0280I Time stamp 2003-01-31 18.18.29

BR0663I Your choice: 'c'

BR0259I Program execution will be continued...

.....

BR0280I Time stamp 2003-01-31 18.18.40

BR0657I Input menu 145 - please check/enter input values

-----  
Restore of profiles and log files from BRBACKUP backup

- 1 - BR\*Tools profile (init\_sap) ..... [no]
- 2 - Oracle profile (init\_ora) ..... [no]
- 3 - Oracle spfile (sp\_file) ..... [no]
- 4 # BACKINT/Mount profile (init\_utl) .. [no]
- 5 - Detail log (det\_log) ..... [yes]
- 6 - Summary log (sum\_log) ..... [no]
- 7 - BRSPACE summary log (space\_log) ... [no]
- 8 - BRSPACE structure log (struc\_log) . [no]
- 9 - BRSPACE parameter log (param\_log) . [no]
- 10 # Control file copy (control\_file) .. [no]
- 11 # Oracle wallet file (ora\_wallet) ... [no]

Standard keys: c - cont, b - back, s - stop, r - refr, h - help



BR0662I Enter your choice:  
BR0280I Time stamp 2003-01-31 18.18.42  
BR0663I Your choice: 'c'  
BR0259I Program execution will be continued...  
BR0746I File /oracle/GC2/sapreorg/reorgGC2.log will be restored from  
/dev/rmt/1mn  
BR0746I File /oracle/GC2/sapreorg/structGC2.log will be restored from  
/dev/rmt/1mn  
BR0280I Time stamp 2003-01-31 18.18.42  
BR0670I Enter 'c[ont]' to continue, 'b[ack]' to go back, 's[top]' to  
abort:  
BR0280I Time stamp 2003-01-31 18.18.44  
BR0257I Your reply: 'c'  
BR0259I Program execution will be continued...  
BR0210I Please mount BRBACKUP volume, if you have not already done so  
BR0280I Time stamp 2003-01-31 18.18.44  
BR0670I Enter 'c[ont]' to continue, 'b[ack]' to go back, 's[top]' to  
abort:  
BR0280I Time stamp 2003-01-31 18.18.46  
BR0257I Your reply: 'c'  
BR0259I Program execution will be continued...  
BR0280I Time stamp 2003-01-31 18.18.46  
BR0226I Rewinding tape volume in device /dev/rmt/1m ...  
BR0370I Directory /oracle/GC2/sapbackup/vdjwrcgu.1 created  
BR0280I Time stamp 2003-01-31 18.19.10  
BR0226I Rewinding tape volume in device /dev/rmt/1m ...  
BR0226I Winding tape volume in device /dev/rmt/1m ...  
BR0285I This function can take several seconds/minutes - be patient  
BR0351I Restoring  
/oracle/GC2/sapbackup/vdjwrcgu.1/reorg\_log+struct\_log  
BR0355I from /dev/rmt/1mn ...  
#FILE..... /oracle/GC2/sapbackup/vdjwrcgu.1/reorg\_log+struct\_log  
#RESTORED. reorgGC2.log,structGC2.log #0/26  
BR0370I Directory /oracle/GC2/sapbackup/vdjwrcgu created  
BR0202I Saving /oracle/GC2/sapreorg/reorgGC2.log

```

BR0203I to /oracle/GC2/sapbackup/vdjwrcgu/reorgGC2.log ...
BR0428W File /oracle/GC2/sapreorg/reorgGC2.log will be overwritten
BR0746I File /oracle/GC2/sapreorg/reorgGC2.log will be restored from
/oracle/GC2/sapbackup/vdjwrcgu.1/reorgGC2.log
BR0202I Saving /oracle/GC2/sapreorg/structGC2.log
BR0203I to /oracle/GC2/sapbackup/vdjwrcgu/structGC2.log ...
BR0428W File /oracle/GC2/sapreorg/structGC2.log will be overwritten
BR0746I File /oracle/GC2/sapreorg/structGC2.log will be restored from
/oracle/GC2/sapbackup/vdjwrcgu.1/structGC2.log
BR0668I Warnings or errors occurred - you can continue to ignore them
or go back to repeat the last action
BR0280I Time stamp 2003-01-31 18.22.32
BR0670I Enter 'c[ont]' to continue, 'b[ack]' to go back, 's[top]' to
abort:
BR0280I Time stamp 2003-01-31 18.22.35
BR0257I Your reply: 'c'
BR0259I Program execution will be continued...
BR0351I Restoring /oracle/GC2/sapreorg/reorgGC2.log
BR0355I from /oracle/GC2/sapbackup/vdjwrcgu.1/reorgGC2.log ...
#FILE..... /oracle/GC2/sapreorg/reorgGC2.log
#RESTORED. /oracle/GC2/sapbackup/vdjwrcgu.1/reorgGC2.log
BR0351I Restoring /oracle/GC2/sapreorg/structGC2.log
BR0355I from /oracle/GC2/sapbackup/vdjwrcgu.1/structGC2.log ...
#FILE..... /oracle/GC2/sapreorg/structGC2.log
#RESTORED. /oracle/GC2/sapbackup/vdjwrcgu.1/structGC2.log
BR0749I 2 files have been successfully restored
=====

```



## Managing Flashback Database with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to manage and display the flashback database status and restore points.

### Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:
  - BRGUI or BRTOOLS:

1. Choose **► Restore and Recovery ► Manage flashback database ◀**.

BRGUI or BRTOOLS displays the menu *BRSPACE main options for managing flashback database*, where you specify the options with which you call BRSPACE.

2. Set the required options:

Menu Entry	Equivalent BRSPACE Command Option
<i>BRSPACE profile (profile)</i>	<a href="#">-p -profile</a>
<i>Database user/password (user)</i>	<a href="#">-u -user</a>
<i>Flashback action (action)</i>	<a href="#">-f mfbck -a -action</a>
<i>Restore point (point)</i>	<a href="#">-f mfbck -p -point</a>
<i>Confirmation mode (confirm)</i>	<a href="#">-c -confirm</a>
<i>Scrolling line count (scroll)</i>	<a href="#">-s -scroll</a>
<i>Message language (language)</i>	<a href="#">-l -language</a>
<i>BRSPACE command line (command)</i>	This shows you the <a href="#">BRSPACE -f mfbck</a> command that is to be executed using the current settings.

3. Choose *Continue*

BRGUI or BRTOOLS prompts you to start BRSPACE.

4. Choose *Continue* to start BRSPACE.

- o Command line:

Enter at least the following command:

```
brspace -f mfbck
```

You can enter more parameters if required. For more information, see [BRSPACE -f mfbck](#).

2. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#).
3. If you have already entered the action, continue with step 4 (quick mode).
4. Choose or confirm the required action:

- o Switch on flashback database

For flashback database, the database must be in archivelog mode and the flash recovery area must already be configured, as described in SAP Note

[966073](#). You can specify the required retention period for the flashback data in *New retention time in min (retention)*.

- Switch off flashback database
- Create restore point

This creates a normal or a guaranteed restore point. The restore point name is restricted to 30 characters in BR\*Tools. We recommend you to use short names for ease of use.

- Drop restore point

This deletes a normal or a guaranteed restore point. You can choose from the displayed list of available restore points.

- Show flashback status

This shows the following information:

<b>Entry</b>	<b>Meaning</b>
<i>Flashback database status (status)</i>	On or off
<i>Oldest flashback time (oldest_time)</i>	Oldest flashback time in the flashback logs
<i>Oldest flashback SCN (oldest_scn)</i>	Oldest flashback system change number (SCN)
<i>Flashback retention time in min (retention)</i>	Retention period for the flashback data
<i>Current flashback size in KB (curr_size)</i>	Current size of the flashback data
<i>Estimated flashback size in KB (estim_size)</i>	Estimated size of the flashback data
<i>Location of recovery area (ra_location)</i>	Location of flash recovery area
<i>Used space in recovery area in KB (ra_space)</i>	Used space in flash recovery area
<i>Space limit in recovery area in KB (ra_limit)</i>	Maximum space in flash recovery area
<i>Reclaimable space in KB (ra_reclaim)</i>	Space available for reuse in flash recovery area
<i>Number of files in recovery area (ra_files)</i>	Number of files in flash recovery area

- Show restore points

This shows a list of available restore points, whether guaranteed or not, timestamp and system change number (SCN), and used storage in KB (for guaranteed restore points only).

5. Follow the prompts to perform your chosen action.

## Result

If you set the option *Create log file* (parameter `-l|-log`), check the results in the [BRSPACE logs](#).

- The [summary log](#) `space<DBSID>.log` displays the return code.
- The [detail log](#) `s<encoded timestamp>.mfb` displays the details.

For more information on how to view the logs with BR\*Tools, see [Showing Profiles and Logs with BR\\*Tools](#).



## Procedures for Restore and Recovery with BR\*Tools

For more information on how and when to use these procedures, see:

- [Complete database recovery](#)
- [Database point-in-time \(PIT\) recovery](#)
- [Tablespace point-in-time \(PIT\) recovery](#)
- [Whole database reset](#)
- [Restore of Individual Backup Files with BR\\*Tools](#)
- [Restore and Application of Offline Redo Log Files with BR\\*Tools](#)



## Setting Point In Time and Tablespaces for Recovery

You use this procedure to specify the:

- Point in time (PIT) when using BRRECOVER to perform one of the following:
  - [Database PIT recovery](#)
  - [Tablespace PIT recovery](#)
- Tablespaces to be recovered when using BRRECOVER to perform a tablespace PIT recovery

### Note

You can use this to select tablespaces from a single component if you are running Multiple Components in One Database (MCOB).

## Prerequisites

- You can specify the PIT as one of the following:
  - A normal point in time, using the time stamp (that is, date, hours, minutes, and seconds)
  - An Oracle system change number (SCN)
  - An Oracle log sequence number (SEQ)



BRRECOVER translates a normal point in time or an SCN into the corresponding sequence number (SEQ) of the last Oracle log to be applied.

- You can specify tablespaces by one of the following:
  - SAP owner of the tablespaces
  - Individual tablespace names



To select tablespaces from a single component if you are running Multiple Components in One Database (MCOB), enter the SAP owner of the tablespaces in the component that you want to recover. Each component has a different SAP owner.

See *Example* below for the contents of the log file.

## Procedure

1. To set the point in time for the recovery, enter one of the following:

Menu Entry	Equivalent BRRECOVER Command Option
<i>Database instance of archivelog thread (instance)</i>	<a href="#">-j -ins -instance</a>
<i>Last archivelog sequence to apply (last_seq)</i>	<a href="#">-n -seq -sequence</a>
<i>Last system change number to apply (last_scn)</i>	<a href="#">-g -scn -change</a>
<i>End point-in-time for recovery (end_pit)</i>	<a href="#">-n -pit -time</a>

2. If you are performing a tablespace PIT recovery, enter one of the following:

Menu Entry	Equivalent BRRECOVER Command Option
<i>Database owner for recovery (owner)</i>	<a href="#">-w -own -owner</a>

Menu Entry	Equivalent BRRECOVER Command Option
<i>Tablespaces for recovery (tablespace)</i>	<a href="#">-a tsp tablespace</a>

3. Choose *Continue* to continue processing with the selected PIT.
4. Check the results in the [BRRECOVER detail log](#), v<encoded timestamp>.<ext>.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).

## Example

This example shows the contents of the BRRECOVER detail log file for this procedure, using a tablespace PIT recovery.

```
BR0657I Input menu 107 - please check/enter input values
-----
Options for tablespace point-in-time recovery of database GC2
1 # Database instance of archivelog thread (instance) . []
2 ~ Last archivelog sequence to apply (last_seq) ..... []
3 ~ Last system change number to apply (last_scn) ..... []
4 ~ End point-in-time for recovery (end_pit) ..... []
5 ~ Database owner for recovery (sap_owner) ..... []
6 ~ Tablespaces for recovery (tablespace) ..... []
Standard keys: c - cont, b - back, s - stop, r - refr, h - help
-----
```

## Checking the Status of Database Files - I

You check the database status when using BRRECOVER to perform [Complete database recovery](#).

This procedure checks the database for critical errors that might prevent production operation.

### Prerequisites

BRRECOVER remounts – that is, closes and again mounts – the database to refresh the views V\$DATAFILE and V\$RECOVER\_FILE. It then checks the availability of the following files:

- Control files
- Redo log files
- Data files

BRRECOVER checks whether there are any data files not online. If so, it recommends to set them online.

See *Example* below for the contents of the log file.

## Procedure

1. Check the display and continue with the database recovery.

### Note

If there are no files that need to be recovered, BRRECOVER stops the recovery automatically.

2. Check the results in the [BRRECOVER detail log](#), v<encoded timestamp>.<ext>.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).

## Example

This example shows the contents of the BRRECOVER detail log file for the recovery phase "Check Status of Database Files".

```
BR0614I Database instance GC2 is mounted
BR0750I Database instance GC2 will be remounted now
BR0280I Time stamp 2003-01-29 19.12.25
BR0307I Shutting down database instance GC2 ...
BR0280I Time stamp 2003-01-29 19.12.31
BR0308I Shutdown of database instance GC2 successful
BR0280I Time stamp 2003-01-29 19.12.31
BR0330I Starting and mounting database instance GC2 ...
BR0280I Time stamp 2003-01-29 19.12.41
BR0331I Start and mount of database instance GC2 successful
BR0118I Tablespaces and data files

Tablespace Status File Status Id. Size Creation time Creation scn
Device Type Link
DRSYS UNKNOWN /oracle/GC2/sapdata1/drsys_1/drsys.data1 ONLINE 3
10493952 2002-07-31 16.55.55 5954 35651591 FILE NOLINK

EXAMPLE UNKNOWN /oracle/GC2/sapdata2/example_1/example.data1 ONLINE 4
126492672 2002-07-31 16.55.57 5973 35651591 FILE NOLINK

.....

BR0119I Redo log files

File Status Group Size First time First scn Device Type Link
/oracle/GC2/origlog/redo01m1.dbf INUSE 1 2097664 2003-01-29 19.04.28
5101811 35651591 FILE NOLINK
```



```
/oracle/GC2/mirrlog/redo01m2.dbf INUSE 1 2097664 2003-01-29 19.04.28
5101811 35651591 FILE NOLINK
```

.....

BR0120I Control files

File Size Reset time Reset scn Device Type Link

```
/oracle/GC2/sapdata1/cntrl/control01.ctl 4579328 2003-01-28 21.11.19
5040603 35651591 FILE NOLINK
```

```
/oracle/GC2/sapdata2/cntrl/control02.ctl 4579328 2003-01-28 21.11.19
5040603 35651591 FILE NOLINK
```

```
/oracle/GC2/sapdata3/cntrl/control03.ctl 4579328 2003-01-28 21.11.19
5040603 35651591 FILE NOLINK
```

BR0755I Checking the status of database files for instance GC2...

BR0758I Data file /oracle/GC2/sapdata5/stabd\_1/stabd.data1 not found  
- it will be restored/recreated and recovered

BR0758I Data file /oracle/GC2/sapdata6/stabi\_1/stabi.data1 not found  
- it will be restored/recreated and recovered

BR0760I Data file /oracle/GC2/sapdata6/testd\_1/testd.data1 not  
current - it will be recovered

BR0760I Data file /oracle/GC2/sapdata6/testd\_2/testd.data2 not  
current - it will be recovered

BR0760I Data file /oracle/GC2/sapdata5/testi\_1/testi.data1 not  
current - it will be recovered

## Selecting Database Backups

You select BRBACKUP database backups when using BRRECOVER to perform:

- [Complete database recovery](#)
- [Database point-in-time \(PIT\) recovery](#)
- [Tablespace point-in-time \(PIT\) recovery](#)
- [Whole database reset](#)
- [Restore of individual backup files](#)

## Prerequisites

- BRRECOVER only displays backups that finished with a return code of 0 or 1. The exception to this is for restore of individual backup files, where all backups are displayed.
- BRRECOVER displays complete, incremental, and partial backups.
- BRRECOVER also displays flashback database and restore points.

- BRRECOVER recommends the most recent successful backup as a default for the recovery. Unless you select another backup, it uses this for the recovery.
- For a complete database recovery, a database PIT recovery, or a tablespace PIT recovery, you can select multiple partial backups. BRRECOVER processes these intelligently. That is, it automatically detects overlapping partial backups and uses only the latest available data files.
- For whole database reset, BRRECOVER does not display partial backups for selection.
- For restore of individual backup files, you can only select one backup. In this case, BRRECOVER also displays backups with errors. If you are applying an incremental backup, BRRECOVER only displays incremental backups.

See *Example* below for the contents of the log file.

## Procedure

1. If required, select a backup that is different from the default recommended by BRRECOVER.

### Note

For whole database reset, you can only select the following types of database backup:

- Complete offline
- Complete online consistent
- Incremental offline
- Incremental online consistent

For more information, see [Database Backup Types](#).

2. Choose *Continue* to continue processing with the selected backup.

For complete or PIT recovery, BRRECOVER roughly checks the availability of offline redo log – that is, archivelog – files in the [BRARCHIVE summary log file](#), depending on the device type used for the backup:

- Tape or BACKINT
- Disk
- Stage (remote disk)

If the files are unavailable, BRRECOVER issues a warning. You can continue if you are sure that either of the following is true:

- You can find the required offline redo log files
- The offline redo log files will not be required

### Note

BRRECOVER identifies files that were added during or (only for PIT) after the selected backup. For example, a tablespace might have been extended

during an online backup. Such files are not contained in the backup but they might need to be recreated.

This does not apply to complete database recovery and restore individual backup files.

3. Check the results in the [BRRECOVER detail log](#), v<encoded timestamp>.<ext>.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).

## Example

This example shows the contents of the BRRECOVER detail log file for this procedure.

Database backups for complete database recovery

```
Pos. Log Start Type Mode Device Rc
1 = bdjwhckx.ffd 2003-01-29 17.30.51 offline full disk 0
2 - bdjwhadu.fft 2003-01-29 17.05.14 offline full tape 1
3 - bdjwgyrq.fff 2003-01-29 16.48.42 offline full util_onl 0
4 - bdjwgtj.fnt 2003-01-29 16.26.55 onl_cons full tape 0
5 - bdjwgvvh.fnf 2003-01-29 16.16.29 onl_cons full util_onl 0
```

## Checking the Status of Database Files - II

You check the database status when using BRRECOVER to perform one of the following:

- [Database Point-In-Time Recovery with BR\\*Tools](#)
- [Whole Database Reset with BR\\*Tools](#)

This procedure checks the availability and status of database files to determine how to handle them during restore and recovery.

### Prerequisites

- BRRECOVER checks whether the control files are available.  
If the control files are available, BRRECOVER remounts – that is, closes and again mounts – the database to refresh the views V\$DATAFILE and V\$RECOVER\_FILE.
- BRRECOVER checks the following files to identify which ones need to be overwritten:
  - Control files
  - Redo log files
  - Data files
- BRRECOVER identifies files that need to be later deleted. It identifies files that were added after the:
  - Selected backup - for whole database reset

- PIT – for database or tablespace PIT

#### Example

For example, a tablespace might have been extended after the selected backup or PIT. Such files need to be later deleted.

BRRECOVER writes the results of the status check to the BRRECOVER detail log file. See *Example* below for the contents of the log file.

## Procedure

1. Check the display and continue with the database recovery.
2. Check the results in the [BRRECOVER detail log](#), v<encoded timestamp>.<ext>.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).

## Example

This example shows the contents of the BRRECOVER detail log file for this procedure.

```
BR0614I Database instance GC2 is mounted
BR0750I Database instance GC2 will be remounted now
BR0280I Time stamp 2003-01-29 19.35.51
BR0307I Shutting down database instance GC2 ...
BR0280I Time stamp 2003-01-29 19.35.57
BR0308I Shutdown of database instance GC2 successful
BR0370I Directory /oracle/GC2/sapbackup/vdjwhnnh created
BR0202I Saving /oracle/GC2/sapdata1/cntrl/control01.ctl
BR0203I to /oracle/GC2/sapbackup/vdjwhnnh/control01.ctl ...
BR0280I Time stamp 2003-01-29 19.35.58
BR0330I Starting and mounting database instance GC2 ...
BR0280I Time stamp 2003-01-29 19.36.08
BR0331I Start and mount of database instance GC2 successful
BR0118I Tablespaces and data files

Tablespace Status File Status Id. Size Creation time Creation scn
Device Type Link

DRSYS UNKNOWN /oracle/GC2/sapdata1/drsys_1/drsys.data1 RECOVER 3
10493952 2002-07-31 16.55.55 5954 35651591 FILE NOLINK

EXAMPLE UNKNOWN /oracle/GC2/sapdata2/example_1/example.data1 RECOVER
4 126492672 2002-07-31 16.55.57 5973 35651591 FILE NOLINK

.....

BR0119I Redo log files
```

```

File Status Group Size First time First scn Device Type Link
/oracle/GC2/origlog/redo01m1.dbf INUSE 1 2097664 2003-01-29 19.04.28
5101811 35651591 FILE NOLINK

/oracle/GC2/mirrlog/redo01m2.dbf INUSE 1 2097664 2003-01-29 19.04.28
5101811 35651591 FILE NOLINK

.....

BR0120I Control files

File Size Reset time Reset scn Device Type Link
/oracle/GC2/sapdata1/cntrl/control01.ct1 4579328 2003-01-28 21.11.19
5040603 35651591 FILE NOLINK

/oracle/GC2/sapdata2/cntrl/control02.ct1 4579328 2003-01-28 21.11.19
5040603 35651591 FILE NOLINK

/oracle/GC2/sapdata3/cntrl/control03.ct1 4579328 2003-01-28 21.11.19
5040603 35651591 FILE NOLINK

BR0755I Checking the status of database files for instance GC2...

BR0428W File /oracle/GC2/sapdata1/drsys_1/drsys.data1 will be
overwritten

BR0428W File /oracle/GC2/sapdata2/example_1/example.data1 will be
overwritten

BR0428W File /oracle/GC2/sapdata4/indx_1/indx.data1 will be
overwritten

.....

BR0280I Time stamp 2003-01-29 19.36.09

BR0668I Warnings or errors occurred - you can continue to ignore them
or go back to repeat the last action

```

## Checking the Status of Tablespaces

You check the status of tablespaces when using BRRECOVER to perform [Tablespace Point-In-Time Recovery](#). After you have [selected the tablespaces to recover](#), this procedure identifies the corresponding database files.

### Prerequisites

- BRRECOVER checks that all files are online. This is normally true if the database is open.
- BRRECOVER identifies which data files need to be recovered, based on the tablespaces that you selected earlier.
- BRRECOVER checks if the tablespaces that you selected and the other data tablespaces are separate self-contained groups. This means that it checks whether there are references such as indexes or constraints that point from members of a group to members outside this group. If so, you cannot continue with the recovery.

BRRECOVER writes the results of the status check to the BRRECOVER detail log file. See *Example* below for the contents of the log file.

## Procedure

1. Check the display and continue with the tablespace recovery.
2. Check the results in the [BRRECOVER detail log](#), v<encoded timestamp>.<ext>.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).

## Example

```
BR0655I Control menu 106 - please decide how to proceed
-----

Tablespace point-in-time recovery main menu
1 + Set point-in-time and tablespaces for recovery
2 + Select database backup
3 = Check the status of tablespaces
4 * Export tablespaces not being recovered
5 * Restore required data files
6 # Restore and apply incremental backup
7 * Restore and apply archivelog files
8 * Open database and plug in exported tablespaces
9 - Exit program
10 - Reset program status

Standard keys: c - cont, b - back, s - stop, r - refr, h - help
-----

BR0662I Enter your choice:
BR0280I Time stamp 2003-01-31 18.36.35
BR0663I Your choice: 'c'
BR0259I Program execution will be continued...
BR0342I Database instance GC2 is open in RESTRICT mode
BR0118I Tablespaces and data files

Tablespace Status File Status Id. Size Creation time Creation scn
Device Type Link
DRSYS ONLINE /oracle/GC2/sapdata1/drsys_1/drsys.data1 ONLINE 3
10493952 2002-07-31 16.55.55 5954 35651591 FILE NOLINK
EXAMPLE ONLINE /oracle/GC2/sapdata2/example_1/example.data1 ONLINE 4
126492672 2002-07-31 16.55.57 5973 35651591 FILE NOLINK
```

```

.....

BR0119I Redo log files

File Status Group Size First time First scn Device Type Link

/oracle/GC2/origlog/redo01m1.dbf INUSE 1 2097664 2003-01-31 11.55.28
5217256 35651591 FILE NOLINK

/oracle/GC2/mirrlog/redo01m2.dbf INUSE 1 2097664 2003-01-31 11.55.28
5217256 35651591 FILE NOLINK

.....

BR0120I Control files

File Size Reset time Reset scn Device Type Link

/oracle/GC2/sapdata1/cntrl/control01.ctl 4579328 2003-01-28 21.11.19
5040603 35651591 FILE NOLINK

/oracle/GC2/sapdata2/cntrl/control02.ctl 4579328 2003-01-28 21.11.19
5040603 35651591 FILE NOLINK

/oracle/GC2/sapdata3/cntrl/control03.ctl 4579328 2003-01-28 21.11.19
5040603 35651591 FILE NOLINK

BR0755I Checking the status of database files for instance GC2...

BR0428W File /oracle/GC2/sapdata1/drsys_1/drsys.data1 will be
overwritten

BR0428W File /oracle/GC2/sapdata2/example_1/example.data1 will be
overwritten

BR0428W File /oracle/GC2/sapdata4/indx_1/indx.data1 will be
overwritten

.....

BR0668I Warnings or errors occurred - you can continue to ignore them
or go back to repeat the last action

BR0280I Time stamp 2003-01-31 18.36.35

BR0670I Enter 'c[ont]' to continue, 'b[ack]' to go back, 's[top]' to
abort:

BR0280I Time stamp 2003-01-31 18.36.36

BR0257I Your reply: 'c'

BR0259I Program execution will be continued...

BR0280I Time stamp 2003-01-31 18.36.37

BR0739I Checking if following tablespaces are self-contained:

DRSYS,EXAMPLE,INDX,PSAPRAWD,PSAPRAWI,

PSAPTESTI,TOOLS,USERS,XDB

BR0285I This function can take several seconds/minutes - be patient

```

BR0280I Time stamp 2003-01-31 18.36.52

BR0739I Checking if following tablespaces are self-contained:

PSAPSTABD, PSAPSTABI, PSAPTESTD, PSAPTTTTD

BR0285I This function can take several seconds/minutes - be patient

## Exporting the Tablespaces Not Being Recovered

You export tablespaces that do not need to be recovered when using BRRECOVER to perform [Tablespace Point-In-Time Recovery](#). During the [database status check](#), BRRECOVER identified which tablespaces it needs to export because they are not involved in the recovery.

### Prerequisites

- BRRECOVER uses the Oracle EXP tool to export the tablespaces that are not involved in the recovery.

#### Note

In fact, BRRECOVER calls EXP or EXPDP (Data Pump) to export only the metadata of these tablespaces. The tablespace data remains in the data files but is invisible to the database during the recovery. This means that the recovery does not affect these tablespaces.

- Before exporting these tablespaces, BRRECOVER sets them to status READ ONLY.
- BRRECOVER writes the results to the BRRECOVER detail log file. See *Example* below for the contents of the log file.

### Procedure

1. Check the display and continue with the database recovery.
2. Check the results in the [BRRECOVER detail log](#), v<encoded timestamp>.tpt.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).

### Example

This example shows the contents of the BRRECOVER detail log file for this procedure.

```
BR0655I Control menu 106 - please decide how to proceed
```

```
-----
```

```
Tablespace point-in-time recovery main menu
```

- ```
1 + Set point-in-time and tablespaces for recovery
2 + Select database backup
3 + Check the status of tablespaces
4 = Export tablespaces not being recovered
```



5 \* Restore required data files  
6 # Restore and apply incremental backup  
7 \* Restore and apply archivelog files  
8 \* Open database and plug in exported tablespaces  
9 - Exit program  
10 - Reset program status

Standard keys: c - cont, b - back, s - stop, r - refr, h - help

-----

BR0662I Enter your choice:  
BR0280I Time stamp 2003-01-31 18.37.06  
BR0663I Your choice: 'c'  
BR0259I Program execution will be continued...  
BR0342I Database instance GC2 is open in RESTRICT mode  
BR0767I Following tablespaces will be set READ ONLY:  
PSAPSTABD,PSAPSTABI,PSAPTESTD,PSAPTTTTD  
BR0280I Time stamp 2003-01-31 18.37.07  
BR0670I Enter 'c[ont]' to continue, 'b[ack]' to go back, 's[top]' to  
abort:  
BR0280I Time stamp 2003-01-31 18.37.10  
BR0257I Your reply: 'c'  
BR0259I Program execution will be continued...  
BR0768I Tablespace PSAPSTABD set READ ONLY  
BR0768I Tablespace PSAPSTABI set READ ONLY  
BR0768I Tablespace PSAPTESTD set READ ONLY  
BR0768I Tablespace PSAPTTTTD set READ ONLY  
BR0742I Metadata of the following tablespaces will be exported:  
PSAPSTABD,PSAPSTABI,PSAPTESTD,PSAPTTTTD  
BR0280I Time stamp 2003-01-31 18.37.10  
BR0670I Enter 'c[ont]' to continue, 'b[ack]' to go back, 's[top]' to  
abort:  
BR0280I Time stamp 2003-01-31 18.37.14  
BR0257I Your reply: 'c'  
BR0259I Program execution will be continued...  
BR0370I Directory /oracle/GC2/sapbackup/vdjwrdum created

```

BR0278I Command output of '/oracle/GC2/bin/exp':
Export: Release 9.2.0.1.0 - Production on Fri Jan 31 18:37:14 2003
Copyright (c) 1982, 2002, Oracle Corporation. All rights reserved.
Username:
Connected to: Oracle9i Enterprise Edition Release 9.2.0.1.0 -
Production
With the Partitioning option
JServer Release 9.2.0.1.0 - Production
Export done in WE8DEC character set and UTF8 NCHAR character set
Note: table data (rows) will not be exported
About to export transportable tablespace metadata...
For tablespace PSAPSTABD ...
. exporting cluster definitions
. exporting table definitions
. . exporting table DBABL
. . exporting table DBABD
. . exporting table DBAML
.....
. . exporting table SDBAH
For tablespace PSAPSTABI ...
. exporting cluster definitions
. exporting table definitions
For tablespace PSAPTESTD ...
. exporting cluster definitions
. exporting table definitions
. . exporting table SDBAH_1
For tablespace PSAPTTTTD ...
. exporting cluster definitions
. exporting table definitions
. . exporting table SDBAH_2
. exporting referential integrity constraints
. exporting triggers
. end transportable tablespace metadata export

```

Export terminated successfully without warnings.

BR0744I EXP called successfully for database instance GC2

## Restoring Control Files

You restore the control files when using BRRECOVER to perform:

- [Database point-in-time \(PIT\) recovery](#)
- [Whole database reset](#)

For a whole database reset, you also restore the offline redo log files, but *only* if you [selected](#) an online consistent backup.

Note

Whereas a whole database reset always restores the control files, a database PIT recovery only restores them if they:

- Are missing
- Do not match the selected backup

This procedure calls BRRESTORE to restore the control files and - if required for whole database reset - the offline redo log files.

## Prerequisites

You can repeat this phase if a BRRESTORE call has failed.

See *Example* below for the contents of the log file.

## Procedure

1. Set the required options:

| Menu Entry                             | Equivalent BRRESTORE Command Option      |
|----------------------------------------|------------------------------------------|
| <i>BRRESTORE profile (profile)</i>     | <a href="#">-p -profile</a>              |
| <i>BRBACKUP run (backup)</i>           | <a href="#">-b -backup b1 backup1</a>    |
| <i>Restore device type (device)</i>    | <a href="#">-d -device</a>               |
| <i>BACKINT/Mount profile (parfile)</i> | <a href="#">-r -parfile</a>              |
| <i>Files for restore (mode)</i>        | <a href="#">-m -mode 0 ,archive_logs</a> |
| <i>Confirmation mode (confirm)</i>     | <a href="#">-c -confirm</a>              |

| Menu Entry                              | Equivalent BRRESTORE Command Option                                                                     |
|-----------------------------------------|---------------------------------------------------------------------------------------------------------|
| <i>Query mode (query)</i>               | <a href="#">-q -query</a>                                                                               |
| <i>Compression mode (compress)</i>      | <a href="#">-k -compress</a>                                                                            |
| <i>Additional output (output)</i>       | <a href="#">-o -output</a>                                                                              |
| <i>Message language (language)</i>      | <a href="#">-l -language</a>                                                                            |
| <i>BRRESTORE command line (command)</i> | This shows you the <a href="#">BRRESTORE command</a> that is to be executed using the current settings. |

2. Note
3. *Restore device type (device)* is taken from the device type used for the selected backup.
4. *Files for restore (mode)* refers to the Oracle file ID or the keyword `archive_logs`. The files for restore varies:
  - For the control files restore, a dummy file ID, 0, is used.
  - For the restore of offline redo log files from an online consistent backup (only relevant for whole database reset, as described above), the keyword `archive_logs` is used.
5. To start the restore with the selected options, choose *Continue*.
6. Check the results in the [BRRESTORE logs](#):
  - The summary log `rest<DBSID>.log` displays the return code.
  - The detail log `r<encoded timestamp>.rsb` displays the progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).

## Example

The control file restore runs in a similar way to the example shown in [Restoring Data Files](#).

## Restoring Data Files

You use this procedure when using BRRECOVER to perform:

- [Complete database recovery](#)
- [Database point-in-time \(PIT\) recovery](#)
- [Tablespace point-in-time \(PIT\) recovery](#)
- [Whole database reset](#)
- [Restore of individual backup files](#)

This procedure calls BRRESTORE to restore files as follows:

| Type of Recovery                               | What is Restored                                                                                                                                                                                       |
|------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Complete database recovery                     | Data files that BRRECOVER identified as missing in <a href="#">Checking the Status of Database Files</a>                                                                                               |
| Database PIT recovery and whole database reset | All data files                                                                                                                                                                                         |
| Tablespace PIT recovery                        | Data files of the selected tablespaces plus the data files for the system and undo tablespaces                                                                                                         |
| Restore individual backup files                | <ul style="list-style-type: none"> <li>For <i>Restore files from BRBACKUP backup</i>:<br/>Selected data files</li> <li>For <i>Apply incremental backup</i>:<br/>Incremental backup save set</li> </ul> |

## Prerequisites

- BRRECOVER repeats this phase as required to restore all required files.
- BRRECOVER avoids duplicate restores by logging which files it has already restored.

See *Example* below for the contents of the log file.

## Procedure

- Set the required options:

| Menu Entry                                   | Equivalent BRRESTORE Command Option                                                                                                          |
|----------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| <i>BRRESTORE profile</i><br>(profile)        | <a href="#">-p -profile</a>                                                                                                                  |
| <i>BRBACKUP run</i><br>(backup)              | <a href="#">-b -backup b1 backup1</a>                                                                                                        |
| <i>Fill-up previous restores</i><br>(fillup) | <a href="#">-f -fillup</a>                                                                                                                   |
| <i>Restore device type</i><br>(device)       | <a href="#">-d -device</a>                                                                                                                   |
| <i>BACKINT/Mount profile</i><br>(parfile)    | <a href="#">-r -parfile</a>                                                                                                                  |
| <i>Database user/password</i><br>(user)      | <a href="#">-u -user</a>                                                                                                                     |
| <i>Restore destination</i><br>(rest_dest)    | Only for restore individual backup files, option <i>Restore files from BRBACKUP backup</i> :<br><a href="#">-m -mode = &lt;rest_dest&gt;</a> |

| Menu Entry                              | Equivalent BRRESTORE Command Option                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|-----------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <i>Files for restore (mode)</i>         | <p>B1-4</p> <ul style="list-style-type: none"> <li>○ Complete database recovery:<br/><a href="#">-m -mode &lt;file_ID1&gt;-&lt;file_ID2&gt;</a></li> <li>○ Database PIT recovery or whole database reset:<br/><a href="#">-m -mode all</a></li> <li>○ Tablespace PIT recovery:<br/><a href="#">-m -mode &lt;tablespace_list&gt;</a></li> <li>○ Restore individual backup files: <ul style="list-style-type: none"> <li>▪ <i>Restore files from BRBACKUP backup:</i><br/><a href="#">-m -mode &lt;file_ID1&gt;-&lt;file_ID2&gt;</a></li> <li>▪ <i>Apply incremental backup:</i><br/><a href="#">-m -mode incr</a></li> </ul> </li> </ul> |
| <i>Confirmation mode (confirm)</i>      | <a href="#">-c -confirm</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <i>Query mode (query)</i>               | <a href="#">-q -query</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <i>Compression mode (compress)</i>      | <a href="#">-k -compress</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <i>Parallel execution (execute)</i>     | <a href="#">-e -execute</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| <i>Additional output (output)</i>       | <a href="#">-o -output</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
| <i>Message language (language)</i>      | <a href="#">-l -language</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <i>BRRESTORE command line (command)</i> | This shows you the <a href="#">BRRESTORE command</a> that is to be executed using the current settings.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

2. Note
3. *Restore device type (device)* is taken from the device type used for the selected backup.
4. *Files for restore (mode)* refers to the Oracle file ID, a tablespace, or a keyword such as *incr* or *all*. The files for restore can vary, as described in the table above at the start of this procedure.
- 5.

6. To start the restore with the selected options, choose *Continue*.
7. Check the results in the [BRRESTORE logs](#):
  - o The summary log `rest<DBSID>.log` displays the return code.
  - o The detail log `r<encoded timestamp>.rsb` displays the progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).

## Example

This example shows the contents of the BRRECOVER detail log file for this procedure when executed as part of a complete database recovery:

### Note

This example shows files 9 and 10 to be restored. However, the files for restore can vary, as described in the table above at the start of this procedure.

```
BRRESTORE main options for restore of database files
1 - BRRESTORE profile (profile) ..... [initGC2.sap]
2 - BRBACKUP backup run (backup) ..... [bdjwhckx.ffd]
3 - Fill-up previous restores (fillup) . [no]
4 - Restore device type (device) ..... [disk]
5 # BACKINT/Mount profile (parfile) .... [dbs/initGC2.utl]
6 - Database user/password (user) ..... [system/*****]
7 - Files for restore (mode) ..... [9-10]

Standard keys: c - cont, b - back, s - stop, r - refr, h - help
-----

BR0280I Time stamp 2003-01-29 19.12.43

BR0134I Unattended mode with 'force' active - continuing processing
with default reply 'cont'

BR0280I Time stamp 2003-01-29 19.12.43

BR0657I Input menu 112 # please check/enter input values
-----

Additional BRRESTORE options for restore of database files
1 - Confirmation mode (confirm) ..... [force]
2 - Query mode (query) ..... [no]
3 - Compression mode (compress) ..... [no]
4 - Parallel execution (execute) ..... [0]
5 - Additional output (output) ..... [no]
```

```

6 - Message language (language) ..... [E]

7 - BRRESTORE command line (command) . [-p initGC2.sap -b
bdjwhckx.ffd -d disk -m 9-10 -c force -k no -e 0 -l E]

Standard keys: c - cont, b - back, s - stop, r - refr, h - help

-----

BR0280I Time stamp 2003-01-29 19.12.43

BR0134I Unattended mode with 'force' active - continuing processing
with default reply 'cont'

BR0291I BRRESTORE will be started with options '-p initGC2.sap -b
bdjwhckx.ffd -d disk -m 9-10 -c force -k no -e 0 -l E'

=====

BR0401I BRRESTORE 6.40 (0)

BR0405I Start of file restore: rdjwhlma.rsb 2003-01-29 19.12.44

BR0457I Probably the database must be recovered due to partial
restore

BR0280I Time stamp 2003-01-29 19.12.44

BR0407I Restore of database: GC2

BR0408I BRRESTORE action ID: rdjwhlma

BR0409I BRRESTORE function ID: rsb

BR0449I Restore mode: PARTIAL

BR0411I Database files for restore:

/oracle/GC2/sapdata5/stabd_1/stabd.data1

/oracle/GC2/sapdata6/stabi_1/stabi.data1

BR0419I Files will be restored from backup: bdjwhckx.ffd 2003-01-29
17.30.51

BR0416I 2 files found to restore, total size 12.016 MB

BR0424I Files will not be decompressed

BR0421I Restore device type: disk

BR0420I Files will be restored from directory:
/sapmnt/uw1030/b/backup/bdjwhckx

BR0134I Unattended mode with 'force' active - no operator
confirmation allowed

BR0351I Restoring /oracle/GC2/sapdata5/stabd_1/stabd.data1

BR0355I from /sapmnt/uw1030/b/backup/bdjwhckx/stabd.data1 ...

#FILE..... /oracle/GC2/sapdata5/stabd_1/stabd.data1

#RESTORED. /sapmnt/uw1030/b/backup/bdjwhckx/stabd.data1 #2/5

```



```

BR0280I Time stamp 2003-01-29 19.12.46

BR0418I 1 of 2 files restored - 8.008 MB of 12.016 MB done

BR0204I Percentage done: 66.64%, estimated end time: 19:12

BR0001I *****
BR0351I Restoring /oracle/GC2/sapdata6/stabi_1/stabi.data1
BR0355I from /sapmnt/uw1030/b/backup/bdjwhckx/stabi.data1 ...
#FILE..... /oracle/GC2/sapdata6/stabi_1/stabi.data1
#RESTORED. /sapmnt/uw1030/b/backup/bdjwhckx/stabi.data1 #2/11

BR0280I Time stamp 2003-01-29 19.12.46

BR0418I 2 of 2 files restored - 12.016 MB of 12.016 MB done

BR0204I Percentage done: 100.00%, estimated end time: 19:12

BR0001I *****

BR0406I End of file restore: rdjwhlma.rsb 2003-01-29 19.12.46

BR0280I Time stamp 2003-01-29 19.12.46

BR0402I BRRESTORE terminated successfully

=====

BR0292I Execution of BRRESTORE terminated with return code 0

```

## Restoring and Applying an Incremental Backup

You use this procedure when using BRRECOVER to perform:

- [Complete database recovery](#)
- [Database point-in-time \(PIT\) recovery](#)
- [Tablespace point-in-time \(PIT\) recovery](#)
- [Whole database reset](#)
- [Restore of Individual Backup Files with BR\\*Tools](#)

This procedure calls BRRESTORE to restore and apply incremental backups *only if* you specified this when you [selected a database backup](#).

### Prerequisites

- BRRECOVER lets you repeat this phase if there is an error.
- BRRECOVER checks whether there are any data files not online. If so, it recommends to set them online.

## Procedure

1. Set the required options:

| Menu Entry                              | Equivalent BRRESTORE Command Option                                                                     |
|-----------------------------------------|---------------------------------------------------------------------------------------------------------|
| <i>BRRESTORE profile (profile)</i>      | <a href="#">-p -profile</a>                                                                             |
| <i>BRBACKUP run (backup)</i>            | <a href="#">-b -backup b1 backup1</a>                                                                   |
| <i>Restore device type (device)</i>     | <a href="#">-d -device</a>                                                                              |
| <i>BACKINT/Mount profile (parfile)</i>  | <a href="#">-r -parfile</a>                                                                             |
| <i>Database user/password (user)</i>    | <a href="#">-u -user</a>                                                                                |
| <i>Files for restore (mode)</i>         | <a href="#">-m -mode incr</a>                                                                           |
| <i>Confirmation mode (confirm)</i>      | <a href="#">-c -confirm</a>                                                                             |
| <i>Query mode (query)</i>               | <a href="#">-q -query</a>                                                                               |
| <i>Additional output (output)</i>       | <a href="#">-o -output</a>                                                                              |
| <i>Message language (language)</i>      | <a href="#">-l -language</a>                                                                            |
| <i>BRRESTORE command line (command)</i> | This shows you the <a href="#">BRRESTORE command</a> that is to be executed using the current settings. |

2. Note
3. *Restore device type (device)* is taken from the device type used for the selected backup.
4.
  1. To start the apply and restore with the selected options, choose *Continue*.
  2. Check the results in the [BRRESTORE logs](#):
    - The [summary log](#) `rest<DBSID>.log` displays the return code.
    - The [detail log](#) `r<encoded timestamp>.rsb` displays the progress.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).

## Restoring and Applying Offline Redo Log Files

You use this procedure when using BRRECOVER to perform:

- [Complete database recovery](#)
- [Database point-in-time \(PIT\) recovery](#)
- [Tablespace point-in-time \(PIT\) recovery](#)
- [Whole database reset](#) – if you [selected an online backup](#)

This procedure uses BRRESTORE to restore and SQLPLUS to apply offline redo log files, that is, archivelog files.

## Prerequisites

- BRRESTORE restores the offline redo log files in ascending order of application. This means that the first logs to be restored are those that are needed to start the apply phase.
- This procedure restores and applies offline redo log files as follows:
  - Up to 100 files are processed as a single group.
  - All the files in a group must be from the same source. For example, files originally backed up with `util` cannot be restored and applied in the same group as files backed up with `rman`.
  - If there are two or more groups of files, they are processed in parallel with two concurrent processing threads.
  - For whole database reset, all offline redo log files that were restored in [Restoring Control Files](#) are applied in one group, which is generally small.
- If you perform this procedure as part of a PIT recovery, only the offline redo log files required to reach the specified point are applied. If you have entered a point in time (PIT) or an Oracle system change number (SCN) to specify the end point of the recovery, BRRECOVER translates this to the equivalent log sequence number (SEQ).
- BRRECOVER first remounts – that is, closes and again mounts – the database to refresh the views `V$DATAFILE`, `V$RECOVER_FILE`, and `V$RECOVERY_STATUS`.
- BRRECOVER checks whether there are any data files not online. If so, it recommends to set them online.
- BRRECOVER first displays a list of the redo log files to apply, as in the following example:

### Example

```
Archivelog files to apply for complete recovery

Pos. Seq. Status Apply From Disk Tape Util Rman Stage
1 - 11 in_arch yes arch
2 - 12 in_arch yes arch
3 - 13 in_arch yes arch
4 - 14 in_arch yes arch
```

```

5 - 15 redo_arch yes redo
6 - 16 redo_arch yes redo
7 - 17 in_redo yes redo

```

**Note**

Some files might be present on multiple media sources. For example, a file might be present on disk and on tape. BRRECOVER normally uses the most accessible version of the file, unless you specify otherwise. In this example, it would use the disk version of the file.

- If the first copy of the redo log files is missing, you can use the second copy, if available. To do this, choose one of the following:
  - Change the command line for restore near the end of this procedure
  - Perform [Restoring and Applying Offline Redo Log Files - Expert Mode](#) instead of this procedure.
- BRRECOVER recreates files that were added to the database during or (only for PIT) after the selected backup. It identified these during the procedure [Selecting a Database Backup](#).
- For a tablespace PIT recovery, BRRECOVER sets OFFLINE the data files of the [exported tablespaces](#) – that is, the tablespaces not involved in the recovery.

See *Example* below for the contents of the log file.

## Procedure

1. If required, change the default options for the sources that BRRECOVER uses when it restores or applies the offline redo log files:

| Menu Entry                               | Details of the Offline Redo Log File Group to be Changed                                       |
|------------------------------------------|------------------------------------------------------------------------------------------------|
| <i>First sequence number (first_seq)</i> | The sequence number of the first offline redo log file for which you want to change the source |
| <i>Last sequence number (last_seq)</i>   | The sequence number of the last offline redo log file for which you want to change the source  |
| <i>New source for applying (source)</i>  | The new source of the offline redo log files                                                   |

2. If required, you can specify a different source for each missing offline redo log file that is listed with multiple sources. You can also change the source for sequences of offline redo log files.
3. Choose *Continue*.  
  
If a restore is necessary – that is, if not all required offline redo log files are on disk – BRRECOVER displays the restore menu.
4. If a restore is necessary, set the required options:

| Menu Entry                                  | Equivalent BRRESTORE Command Options           |
|---------------------------------------------|------------------------------------------------|
| <i>BRRESTORE profile (profile)</i>          | <a href="#">-p -profile</a>                    |
| <i>Profile for cpio (prof_cpio)</i>         | <a href="#">-p -profile</a>                    |
| <i>Profile for dd (prof_dd)</i>             | <a href="#">-p -profile</a>                    |
| <i>Profile for rman (prof_rman)</i>         | <a href="#">-p -profile</a>                    |
| <i>Profile for rman_dd (prof_rman_dd)</i>   | <a href="#">-p -profile</a>                    |
| <i>Profile for rman_set (prof_rman_set)</i> | <a href="#">-p -profile</a>                    |
| <i>BACKINT/Mount profile (parfile)</i>      | <a href="#">-r -parfile</a>                    |
| <i>Database user/password (user)</i>        | <a href="#">-u -user</a>                       |
| <i>Destination directory (dest_dir)</i>     | <a href="#">-a -archive = &lt;rest_dir&gt;</a> |
| <i>Confirmation mode (confirm)</i>          | <a href="#">-c -confirm</a>                    |
| <i>Additional output (output)</i>           | <a href="#">-o -output</a>                     |
| <i>Message language (language)</i>          | <a href="#">-l -language</a>                   |

5. If the restore menu is displayed, choose *Continue*.

BRRECOVER displays the menu for applying offline redo log files to the database instance.

6. If required, change the default options for applying offline redo log files:

| Menu Entry                                | Equivalent BRRECOVER Command Options |
|-------------------------------------------|--------------------------------------|
| <i>First sequence number (first_seq)</i>  | First offline redo log to apply      |
| <i>Last sequence number (last_seq)</i>    | Last offline redo log to apply       |
| <i>Use backup control file (back_ctl)</i> | Yes or no                            |
| <i>Parallel recovery (degree)</i>         | <a href="#">-e -degree</a>           |

7. Recommendation
8. Unless you are an expert, we recommend that you only change the last entry, *Parallel recovery (degree)*, if required.
- 9.
10. Choose *Continue*.

11. If a restore is necessary, set the required options for restoring an offline redo log group:

| Menu Entry                              | Equivalent BRRESTORE Command Options for Restore of an Offline Redo Log Group                           |
|-----------------------------------------|---------------------------------------------------------------------------------------------------------|
| <i>Restore unattended (unattend)</i>    | <a href="#">-cl-confirm</a>                                                                             |
| <i>BRRESTORE command line (command)</i> | This shows you the <a href="#">BRRESTORE command</a> that is to be executed using the current settings. |

12. Recommendation

- We recommend that only experts change the BRRESTORE command line.
- If you are restoring large numbers of offline redo log files, you can choose unattended mode. Otherwise, BRRECOVER prompts you again for each group.

13. If the above restore menu is displayed, choose *Continue* to start the restore.

If necessary, BRRESTORE restores the offline redo log files.

14. Set the required options to apply a group of offline redo log files:

| Menu Entry                               | Details of the Offline Redo Log File Group to be Applied                              |
|------------------------------------------|---------------------------------------------------------------------------------------|
| <i>Apply unattended (unattend)</i>       | Log group applied in attended or unattended mode                                      |
| <i>First sequence number (first_seq)</i> | First offline redo log to apply                                                       |
| <i>Last sequence number (last_seq)</i>   | Last offline redo log to apply                                                        |
| <i>SQLPLUS command (command)</i>         | This shows you the SQLPLUS command that is to be executed using the current settings. |

15. Recommendation

- Unless you are an expert, we recommend that you only change the first entry, *Apply unattended (unattend)*, if required.
- If you are applying large numbers of offline redo log files, you can choose unattended mode. Otherwise, BRRECOVER prompts you again for each group.

BRRECOVER applies the offline redo log files.

- 16.

17. Note

18. If there are two or more groups of offline redo log files to process, BRRESTORE and BRRECOVER work in parallel to restore and apply the files.

- 19.

20. Note
21. If you process in attended mode, you see the above menus with the BRRESTORE or BRRECOVER command line before each group of offline redo log files is processed.
22. Therefore, if you are processing many groups of offline redo log files, we recommend that you process the first few groups in attended mode to check that processing is correct. Then you can switch to unattended mode to finish processing automatically.
23. If successful, BRRECOVER displays the message *Media recovery completed*.
- 24.
25. Check the results in the [BRRESTORE](#) and [BRRECOVER](#) logs.
  - o The [detail log](#) r<encoded timestamp>.rsa displays the progress for restoring the offline redo log files.
  - o The [detail log](#) v<encoded timestamp>.<ext> displays the progress for applying the offline redo log files.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).

## Example

This example shows the contents of the BRRECOVER detail log file for this procedure. No restore is required in this example because all the offline redo log files are on disk.

```
BR0280I Time stamp 2003-01-29 19.13.39
BR0657I Input menu 117 # please check/enter input values
-----
Apply archivelog files to database instance GC2
1 - First sequence number (first_seq) .. [11]
2 - Last sequence number (last_seq) .... [17]
3 # Use backup control file (back_ctl) . [no]
4 ~ Parallel recovery (degree) ..... []
Standard keys: c - cont, b - back, s - stop, r - refr, h - help
-----
BR0280I Time stamp 2003-01-29 19.13.39
BR0134I Unattended mode with 'force' active - continuing processing
with default reply 'cont'
BR0781I All archivelog files to be applied are on disk - no restore
required
BR0783I Archivelog files with sequence number 11-14 will be applied
to database GC2
BR0280I Time stamp 2003-01-29 19.13.39
BR0336I Applying offline redo log file /oracle/GC2/saparch/1_11.dbf
...
```

```

BR0336I Applying offline redo log file /oracle/GC2/saparch/1_12.dbf
...
BR0336I Applying offline redo log file /oracle/GC2/saparch/1_13.dbf
...
BR0336I Applying offline redo log file /oracle/GC2/saparch/1_14.dbf
...
BR0280I Time stamp 2003-01-29 19.13.40
BR0337I Offline redo log file /oracle/GC2/saparch/1_11.dbf applied
successfully
BR0280I Time stamp 2003-01-29 19.13.40
BR0337I Offline redo log file /oracle/GC2/saparch/1_12.dbf applied
successfully
BR0280I Time stamp 2003-01-29 19.13.40
BR0337I Offline redo log file /oracle/GC2/saparch/1_13.dbf applied
successfully
BR0280I Time stamp 2003-01-29 19.13.40
BR0337I Offline redo log file /oracle/GC2/saparch/1_14.dbf applied
successfully
BR0280I Time stamp 2003-01-29 19.13.40
BR0784I Media recovery completed

```

## Performing Flashback Database

You use this procedure when using BRRECOVER to perform flashback database during:

- [Database point-in-time \(PIT\) recovery](#)
- [Whole database reset](#)

### Prerequisites

- BRRECOVER first remounts – that is, closes and again mounts – the database to refresh the views V\$DATAFILE, V\$RECOVER\_FILE, and V\$RECOVERY\_STATUS.
- BRRECOVER checks whether there are any data files not online. If so, it recommends to set them online.
- BRRECOVER displays a list of the redo log files to be automatically applied by Oracle during the flashback database, as in the following example:

#### Example

Archivelog files to apply for complete recovery

Pos. Seq. Status Apply From Disk Tape Util Rman Stage

1 - 11 in\_arch yes arch



- 2 - 12 in\_arch yes arch
- 3 - 13 in\_arch yes arch
- 4 - 14 in\_arch yes arch
- 5 - 15 redo\_arch yes redo
- 6 - 16 redo\_arch yes redo
- 7 - 17 in\_redo yes redo

**Note**

Some files might be present on multiple media sources. For example, a file might be present on disk and on tape. BRRECOVER normally uses the most accessible version of the file, unless you specify otherwise. In this example, it would use the disk version of the file.

## Procedure

1. If required, change the default options for Oracle to automatically apply the offline redo log files:

| Menu Entry                               | Meaning                         |
|------------------------------------------|---------------------------------|
| <i>First sequence number (first_seq)</i> | First offline redo log to apply |
| <i>Last sequence number (last_seq)</i>   | Last offline redo log to apply  |

2. If a restore is necessary, set the required options:

| Menu Entry                                  | Equivalent BRRESTORE Command Options           |
|---------------------------------------------|------------------------------------------------|
| <i>BRRESTORE profile (profile)</i>          | <a href="#">-p -profile</a>                    |
| <i>Profile for cpio (prof_cpio)</i>         | <a href="#">-p -profile</a>                    |
| <i>Profile for dd (prof_dd)</i>             | <a href="#">-p -profile</a>                    |
| <i>Profile for rman (prof_rman)</i>         | <a href="#">-p -profile</a>                    |
| <i>Profile for rman_dd (prof_rman_dd)</i>   | <a href="#">-p -profile</a>                    |
| <i>Profile for rman_set (prof_rman_set)</i> | <a href="#">-p -profile</a>                    |
| <i>BACKINT/Mount profile (parfile)</i>      | <a href="#">-r -parfile</a>                    |
| <i>Database user/password (user)</i>        | <a href="#">-u -user</a>                       |
| <i>Destination directory (dest_dir)</i>     | <a href="#">-a -archive = &lt;rest_dir&gt;</a> |

| Menu Entry                           | Equivalent BRRESTORE Command Options |
|--------------------------------------|--------------------------------------|
| Confirmation mode ( <i>confirm</i> ) | <a href="#">-c -confirm</a>          |
| Additional output ( <i>output</i> )  | <a href="#">-o -output</a>           |
| Message language ( <i>language</i> ) | <a href="#">-l -language</a>         |

- Choose *Continue*.
- If required, alter the *Database flashback target* for the flashback:

| Menu Entry                                    | Equivalent BRRECOVER Command Options                                                                                        |
|-----------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------|
| Database flashback target ( <i>target</i> )   | <a href="#">-o -rpt -point</a>                                                                                              |
| Database flashback command ( <i>command</i> ) | The SQL command that is to be executed using the current settings. For more information, see your Oracle SQL documentation. |

- For whole database reset, you can change the *restore point*.
- For database point-in-time recovery, you can change the *timestamp*.
- Choose *Continue*.

BRRECOVER performs flashback database.

- Check the results in the [BRRESTORE](#) and [BRRECOVER](#) logs. The [detail log](#) `r<encoded timestamp>.rsa` displays the progress for restoring the offline redo log files.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).

## Opening the Database

You use this procedure when using BRRECOVER to perform:

- [Complete database recovery](#)
- [Database point-in-time \(PIT\) recovery](#)
- [Tablespace point-in-time \(PIT\) recovery](#)
- [Whole database reset](#)
- [Restore and application of offline redo log files](#)

This procedure opens the database.

For tablespace PIT recovery, it also plugs in the [exported tablespaces](#).

## Prerequisites

- BRRECOVER:

- Checks tablespaces and, if offline, recommends you to switch them online
- Checks data files and warns you if they are not online
- BRRECOVER deletes files that were added after the selected backup (for whole database reset) or the PIT (for database or tablespace PIT). It identified these during the [database status check](#) or [selection of database backups](#). This does not apply to complete database recovery.
- The `resetlogs` parameter to open the database is relevant for the following types of recovery:
  - Database PIT
  - Tablespace PIT
  - Whole database reset if the selected backup was online
  - Restore and application of offline redo log files

#### Note

`Resetlogs` reformats the online redo log files and resets their sequence number to 1. In addition, it updates the control files to reflect a new incarnation of the database.

- For tablespace PIT recovery, BRRECOVER reinstates the tablespaces that were exported because they were not required for the recovery. To do this, BRRECOVER does the following after it has reopened the database:
  - It temporarily drops the affected tablespaces.
  - It calls the Oracle IMP tool to import the affected tablespaces (to be precise, it imports the metadata of these tablespaces).
  - It sets the affected tablespaces to READ-WRITE status.

## Procedure

1. Select the required option for *Reset logs option (reset\_logs)*.

#### Recommendation

Unless you are an expert, we recommend you to accept the setting that BRRECOVER proposes.

If the `resetlogs` option is set, BRRECOVER warns you that no more logs can be applied after the database has been opened. Normally you can ignore this warning.

2. Choose **Yes** to continue opening the database.

BRRECOVER opens the database.

For a PIT recovery or a whole database reset from an online consistent backup, BRRECOVER recreates the missing temporary database files.

BRRECOVER then checks the tablespaces and data files, the redo log files, and the control files.

For a PIT recovery and whole database reset, BRRECOVER deletes files that are no longer used by the database.

For a tablespace PIT recovery, BRRECOVER reinstates the tables that were exported as described at the end of *Prerequisites* above.

If successful, BRRECOVER displays a message like the following, depending on the type of recovery:

*Database point-in-time recovery completed.*

3. Check the results in the [BRRECOVER detail log](#), v<encoded timestamp>.<ext>.

For more information on how to view the logs, see [Showing Logs with BR\\*Tools](#).

## Example

This example shows the contents of the BRRECOVER detail log file for this procedure, using a database PIT recovery.

```
BR0614I Database instance GC2 is mounted
BR0064I Database instance GC2 will be shut down now
BR0280I Time stamp 2003-01-29 19.44.28
BR0307I Shutting down database instance GC2 ...
BR0280I Time stamp 2003-01-29 19.44.35
BR0308I Shutdown of database instance GC2 successful
BR0280I Time stamp 2003-01-29 19.44.35
BR0657I Input menu 135 # please check/enter input values
-----
Options for opening database instance GC2
1 ~ Reset logs option (reset_logs) . [resetlogs]
2 * Open database command (command). [alter database open resetlogs]
Standard keys: c - cont, b - back, s - stop, r - refr, h - help
-----
BR0280I Time stamp 2003-01-29 19.44.35
BR0134I Unattended mode with 'force' active - continuing processing
with default reply 'cont'
BR0786I Database instance GC2 will be opened now with option
'resetlogs'
BR0787I No more archivelog files can be applied after database has
been opened
BR0675I Do you want to perform this action?
BR0126I Unattended mode active - continuing processing with default
reply 'yes'
```

```

BR0280I Time stamp 2003-01-29 19.44.35

BR0304I Starting and opening database instance GC2 ...

BR0280I Time stamp 2003-01-29 19.44.55

BR0305I Start and open of database instance GC2 successful

BR0280I Time stamp 2003-01-29 19.44.57

BR0789I Temporary database file
/oracle/GC2/sapdata3/temp_1/temp.data1 was recreated

BR0118I Tablespaces and data files

Tablespace Status File Status Id. Size Creation time Creation scn
Device Type Link

DRSYS ONLINE /oracle/GC2/sapdata1/drsys_1/drsys.data1 ONLINE 3
10493952 2002-07-31 16.55.55 5954 35651591 FILE NOLINK

EXAMPLE ONLINE /oracle/GC2/sapdata2/example_1/example.data1 ONLINE 4
126492672 2002-07-31 16.55.57 5973 35651591 FILE NOLINK

.....

BR0119I Redo log files

File Status Group Size First time First scn Device Type Link

/oracle/GC2/origlog/redo01m1.dbf INUSE 1 2097664 0000-00-00 00.00.00
0 35651591 FILE NOLINK

/oracle/GC2/mirrlog/redo01m2.dbf INUSE 1 2097664 0000-00-00 00.00.00
0 35651591 FILE NOLINK

.....

BR0120I Control files

File Size Reset time Reset scn Device Type Link

/oracle/GC2/sapdata1/cntrl/control01.ctl 4579328 2003-01-29 19.44.45
5101776 35651591 FILE NOLINK

/oracle/GC2/sapdata2/cntrl/control02.ctl 4579328 2003-01-29 19.44.45
5101776 35651591 FILE NOLINK

/oracle/GC2/sapdata3/cntrl/control03.ctl 4579328 2003-01-29 19.44.45
5101776 35651591 FILE NOLINK

BR0716I Database point-in-time recovery completed

```

## Check and Verification with BR\*Tools

You can check and verify your Oracle database with [BR\\*Tools](#).

You typically use BR\*Tools menus for a one-off check and verification. The aim of regular checks and verification is to detect and correct potentially critical problems before these lead to downtime of the production database.

For more information, see [Database System Check](#).

#### Recommendation

For routine check and verification, we recommend one of the following:

- DBA Planning Calendar to schedule a check or verification and then view its log
- The scheduler `cron` for UNIX or `at` for Windows

## Integration

- BRTOOLS normally calls the SAP tool BRCONNECT or BRBACKUP. You can also perform a check or verification directly by calling BRCONNECT or BRBACKUP from the command line.

#### Recommendation

We recommend you to normally use BRTOOLS rather than BRCONNECT or BRBACKUP. This is because the BRTOOLS menus simplify entry of the correct parameters.

- Make sure that you have set the necessary parameters for BRCONNECT and BRBACKUP in the [Initialization Profile `init<DBSID>.sap`](#).

## Features

You can perform the following check and verification functions with BR\*Tools:

- Database system check
- Validation of database structure
- Verification of database blocks

## Activities

1. You call the required function in BRTOOLS and check the displayed parameters, changing them as required.

The default, which is set in the [initialization profile `init<DBSID>.sap`](#), is to check or verify the entire database.

2. If required, you change the default values for the check or verification parameters in the initialization profile `init<DBSID>.sap` and restart BRTOOLS.
3. If required, you choose *Check and verification Reset program status* to set the defaults used to the values in the initialization profile `init<DBSID>.sap`. For certain input values, there is no corresponding parameter in the initialization profile, in which case the default value from the BRTOOLS program is used.
4. You start the check or verification.
5. You check the results of the check or verification in the [BRCONNECT logs](#) or the [BRBACKUP logs](#).

# Checking the Database System with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to check the database system.

## Prerequisites

- Make sure you have set the necessary BRCONNECT parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRCONNECT.
- The database must be open.

## Procedure

1. Start BRGUI or BRTOOLS.
2. Choose *Database check and verification Database system check*.
3. Set the required options:

| Menu Entry                                  | Equivalent BRCONNECT Command Option                                                                      |
|---------------------------------------------|----------------------------------------------------------------------------------------------------------|
| <i>BRCONNECT profile (profile)</i>          | <a href="#">-p -profile</a>                                                                              |
| <i>Database user/password (user)</i>        | <a href="#">-u -user</a>                                                                                 |
| <i>Use default check settings (default)</i> | <a href="#">-f check -d -default</a>                                                                     |
| <i>Database owner for check (owner)</i>     | <a href="#">-f check -o -output</a>                                                                      |
| <i>Ignore DBCHECKORA settings (ignore)</i>  | <a href="#">-f check -i -ignore</a>                                                                      |
| <i>Ignore DBSTATC settings (igndb)</i>      | <a href="#">-f check -n -igndb</a>                                                                       |
| <i>Exclude from check (exclude)</i>         | <a href="#">-f check -e -exclude</a>                                                                     |
| <i>Confirmation mode (confirm)</i>          | <a href="#">-c -confirm</a>                                                                              |
| <i>Query mode (query)</i>                   | <a href="#">-q -query</a>                                                                                |
| <i>Extended output (output)</i>             | <a href="#">-o -output</a>                                                                               |
| <i>Message language (language)</i>          | <a href="#">-l -language</a>                                                                             |
| <i>BRCONNECT command line (command)</i>     | This shows you the <a href="#">BRCONNECT -f check</a> that is to be executed using the current settings. |

1. To start processing with the selected options, choose *Continue*.

2. Check the results in the [BRCONNECT logs](#).
  - The [summary log](#) `conn<DBSID>.log` displays the return code.
  - The [detail log](#) `c<encoded timestamp>.chk` displays the progress.

For more information on how to view the logs with BR\*Tools, see [Showing Logs with BR\\*Tools](#).

## Validating the Database Structure with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to validate the database structure with BR\*Tools.

### Prerequisites

- Make sure you have set the necessary BRCONNECT parameters in the [initialization profile](#) `init<DBSID>.sap`, because BRTOOLS uses these when it calls BRCONNECT.
- The database must be open.

### Procedure

1. Start BRGUI or BRTOOLS.
2. Choose *Database check and verification Validation of database structure*
3. Set the required options:

| Menu Entry                                 | Equivalent BRCONNECT Command Option   |
|--------------------------------------------|---------------------------------------|
| <i>BRCONNECT profile (profile)</i>         | <a href="#">-p -profile</a>           |
| <i>Database user/password (user)</i>       | <a href="#">-u -user</a>              |
| <i>Validate mode (validate)</i>            | <a href="#">-f stats -v -validate</a> |
| <i>Database owner for validate (owner)</i> | <a href="#">-f stats -o -owner</a>    |
| <i>Exclude from validate (exclude)</i>     | <a href="#">-f stats -e -exclude</a>  |
| <i>Tables for validate (table)</i>         | <a href="#">-f stats -t -table</a>    |
| <i>Confirmation mode (confirm)</i>         | <a href="#">-c -confirm</a>           |
| <i>Query mode (query)</i>                  | <a href="#">-q -query</a>             |
| <i>Parallel degree (parallel)</i>          | <a href="#">-f stats -p -parallel</a> |
| <i>Time limit (limit)</i>                  | <a href="#">-f stats -l -limit</a>    |



| Menu Entry                              | Equivalent BRCONNECT Command Option                                                                                           |
|-----------------------------------------|-------------------------------------------------------------------------------------------------------------------------------|
| <i>Force time limit (force)</i>         | <a href="#">-f stats -f -force limit</a>                                                                                      |
| <i>Extended output (output)</i>         | <a href="#">-o -output</a>                                                                                                    |
| <i>Message language (language)</i>      | <a href="#">-l -language</a>                                                                                                  |
| <i>BRCONNECT command line (command)</i> | This shows you the <a href="#">BRCONNECT -f stats -v -validate</a> command that is to be executed using the current settings. |

1. To start processing with the selected options, choose *Continue*.
2. Check the results in the [BRCONNECT logs](#).
  - The [summary log](#) `conn<DBSID>.log` displays the return code.
  - The [detail log](#) `c<encoded timestamp>.vst` displays the progress.

For more information on how to view the logs with BR\*Tools, see [Showing Logs with BR\\*Tools](#).

## Verifying Database Blocks with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to perform a database block verification.

### Prerequisites

- Make sure you have set the necessary BRBACKUP parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRBACKUP.

### Procedure

1. Start BRGUI or BRTOOLS.
2. Choose *Database check and verification Verification of database blocks* .
3. Set the required options:

| Menu Entry                           | Equivalent BRBACKUP Command Option |
|--------------------------------------|------------------------------------|
| <i>BRBACKUP profile (profile)</i>    | <a href="#">-p -profile</a>        |
| <i>Database user/password (user)</i> | <a href="#">-u -user</a>           |
| <i>BRBACKUP run type (type)</i>      | <a href="#">-t -type</a>           |
| <i>Verification mode (verify)</i>    | <a href="#">-w -verify</a>         |
| <i>Files for verification (mode)</i> | <a href="#">-m -mode</a>           |

| Menu Entry                         | Equivalent BRBACKUP Command Option                                                                                 |
|------------------------------------|--------------------------------------------------------------------------------------------------------------------|
| Confirmation mode<br>(confirm)     | <a href="#">-c -confirm</a>                                                                                        |
| Query mode (query)                 | <a href="#">-q -query</a>                                                                                          |
| Parallel execution<br>(execute)    | <a href="#">-e -execute</a>                                                                                        |
| Extended output (output)           | <a href="#">-o -output</a>                                                                                         |
| Message language<br>(language)     | <a href="#">-l -language</a>                                                                                       |
| BRBACKUP command<br>line (command) | This shows you the <a href="#">BRBACKUP -w only dbv</a> command that is to be executed using the current settings. |

1. To start processing with the selected options, choose *Continue*.
2. Check the results in the [BRBACKUP logs](#).
  - The [summary log](#) backup<DBSID>.log displays the return code.
  - The [detail log](#) b<encoded timestamp>.dbv displays the progress.

For more information on how to view the logs with BR\*Tools, see [Showing Logs with BR\\*Tools](#).

## Database Statistics with BR\*Tools

You can maintain the statistics of your Oracle database with [BR\\*Tools](#).

You typically use BR\*Tools menus for a one-off update statistics. The aim of regular update statistics is to maintain optimal database performance. Out-of-date statistics reduce performance due to inappropriate access paths for database queries. For more information, see [Update Statistics](#).

Recommendation

For routine update statistics, we recommend one of the following:

- DBA Planning Calendar to schedule a check or verification and then view its log
- The scheduler `cron` for UNIX or `at` for Windows

## Integration

- BRTOOLS normally calls the SAP tool BRCONNECT to update statistics. You can also update statistics directly by calling BRCONNECT from the command line.

Recommendation

We recommend you to normally use BRTOOLS rather than BRCONNECT. This is because the BRTOOLS menus simplify entry of the correct parameters.

- Make sure that you have set the necessary parameters for BRCONNECT in the [Initialization Profile init<DBSID>.sap](#).

## Features

You can perform the following backup functions with BR\*Tools:

- Update database statistics
- Collect missing statistics
- Delete harmful statistics

## Activities

1. You call the required function in BRTOOLS and check the displayed parameters, changing them as required.

The default parameter, which is set in the [initialization profile init<DBSID>.sap](#), is to use the `ANALYZE` command (Oracle 9i) or the `DBMS_STATS` package (Oracle 10g) to sequentially update statistics for all database tables.

Note

BRTOOLS only lets you change certain parameters for update statistics. If you have to make other changes, you must change the `init<DBSID>.sap` profile manually and then restart BRTOOLS.

2. If required, you change the default values for the update statistics parameters in the initialization profile `init<DBSID>.sap` and restart BRTOOLS.
3. If required, you choose *Database Statistics Reset program status* to set the defaults used to the values in the initialization profile `init<DBSID>.sap`. For certain input values, there is no corresponding parameter in the initialization profile, in which case the default value from the BRTOOLS program is used.
4. You start update statistics.
5. You check the results in the [BRCONNECT Logs](#).

## Updating Database Statistics with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to update database statistics.

### Prerequisites

- Make sure you have set the necessary BRCONNECT parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRCONNECT.
- The database must be open.

### Procedure

1. Start BRGUI or BRTOOLS.

2. Choose *Database statistics Update database statistics* .
3. Set the required options:

| <b>Menu Entry</b>                          | <b>Equivalent BRCONNECT Command Option</b> |
|--------------------------------------------|--------------------------------------------|
| <i>BRCONNECT profile (profile)</i>         | <a href="#">-p -profile</a>                |
| <i>Database user/password (user)</i>       | <a href="#">-u -user</a>                   |
| <i>Collection method (method)</i>          | <a href="#">-f stats -m -method</a>        |
| <i>Sample size (sample)</i>                | <a href="#">-f stats -s -sample</a>        |
| <i>Bucket count (buckets)</i>              | <a href="#">-f stats -b -buckets</a>       |
| <i>DBMS_STATS parallel degree (degree)</i> | <a href="#">-f stats -g -degrees</a>       |
| <i>Change threshold (change)</i>           | <a href="#">-f stats -c -change</a>        |
| <i>Database owner for update (owner)</i>   | <a href="#">-f stats -o -owner</a>         |
| <i>Ignore DBSTATC settings (ignore)</i>    | <a href="#">-f stats -n -ignore</a>        |
| <i>Exclude from update (exclude)</i>       | <a href="#">-f stats -e -exclude</a>       |
| <i>Tables for update (table)</i>           | <a href="#">-f stats -t -table</a>         |
| <i>Confirmation mode (confirm)</i>         | <a href="#">-c -confirm</a>                |
| <i>Query mode (query)</i>                  | <a href="#">-q -query</a>                  |
| <i>Parallel threads (parallel)</i>         | <a href="#">-f stats -p -parallel</a>      |
| <i>Update history tables (history)</i>     | <a href="#">-f stats -h -history</a>       |
| <i>Retain old statistics (retain)</i>      | <a href="#">-f stats -r -retain</a>        |
| <i>Time limit in minutes (limit)</i>       | <a href="#">-f stats -l -limit</a>         |
| <i>Force options (force)</i>               | <a href="#">-f stats -f -force</a>         |
| <i>Time interval in minutes (interval)</i> | <a href="#">-f stats -i -interval</a>      |
| <i>Extended output (output)</i>            | <a href="#">-o -output</a>                 |

| Menu Entry                              | Equivalent BRCONNECT Command Option                                                                              |
|-----------------------------------------|------------------------------------------------------------------------------------------------------------------|
| <i>Message language (language)</i>      | <a href="#">-l -language</a>                                                                                     |
| <i>BRCONNECT command line (command)</i> | This shows you the <a href="#">BRCONNECT -f stats</a> command that is to be executed using the current settings. |

1. To start processing with the selected options, choose *Continue*.
2. Check the results in the [BRCONNECT logs](#).
  - The [summary log](#) `conn<DBSID>.log` displays the return code.
  - The [detail log](#) `c<encoded timestamp>.sta` displays the progress.

For more information on how to view the logs with BR\*Tools, see [Showing Logs with BR\\*Tools](#).

## Collecting Missing Statistics with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to collect missing database statistics.

### Prerequisites

- Make sure you have set the necessary BRCONNECT parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRCONNECT.
- The database must be open.

### Procedure

1. Start BRGUI or BRTOOLS.
2. Choose *Database statistics Collect missing statistics*.
3. Set the required options:

| Menu Entry                                 | Equivalent BRCONNECT Command Option  |
|--------------------------------------------|--------------------------------------|
| <i>BRCONNECT profile (profile)</i>         | <a href="#">-p -profile</a>          |
| <i>Database user/password (user)</i>       | <a href="#">-u -user</a>             |
| <i>Collection method (method)</i>          | <a href="#">-f stats -m -method</a>  |
| <i>Sample size (sample)</i>                | <a href="#">-f stats -s -sample</a>  |
| <i>Bucket count (buckets)</i>              | <a href="#">-f stats -b -buckets</a> |
| <i>DBMS_STATS parallel degree (degree)</i> | <a href="#">-f stats -g -degree</a>  |

| Menu Entry                                | Equivalent BRCONNECT Command Option                                                                                         |
|-------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------|
| <i>Change threshold (change)</i>          | <a href="#">-f stats -c change</a>                                                                                          |
| <i>Database owner for collect (owner)</i> | <a href="#">-f stats -o owner</a>                                                                                           |
| <i>Ignore DBSTATC settings (ignore)</i>   | <a href="#">-f stats -n ignore</a>                                                                                          |
| <i>Exclude from collect (exclude)</i>     | <a href="#">-f stats -e exclude</a>                                                                                         |
| <i>Tables for collect (table)</i>         | <a href="#">-f stats -t table</a>                                                                                           |
| <i>Confirmation mode (confirm)</i>        | <a href="#">-c confirm</a>                                                                                                  |
| <i>Query mode (query)</i>                 | <a href="#">-q query</a>                                                                                                    |
| <i>Parallel threads (parallel)</i>        | <a href="#">-f stats -p parallel</a>                                                                                        |
| <i>Update history tables (history)</i>    | <a href="#">-f stats -h history</a>                                                                                         |
| <i>Retain old statistics (retain)</i>     | <a href="#">-f stats -r retain</a>                                                                                          |
| <i>Time limit in minutes (limit)</i>      | <a href="#">-f stats -l limit</a>                                                                                           |
| <i>Force options (force)</i>              | <a href="#">-f stats -f force</a>                                                                                           |
| <i>Extended output (output)</i>           | <a href="#">-o output</a>                                                                                                   |
| <i>Message language (language)</i>        | <a href="#">-l language</a>                                                                                                 |
| <i>BRCONNECT command line (command)</i>   | This shows you the <a href="#">BRCONNECT -f stats -t missing</a> command that is to be executed using the current settings. |

1. To start processing with the selected options, choose *Continue*.
2. Check the results in the [BRCONNECT logs](#).
  - The [summary log](#) `conn<DBSID>.log` displays the return code.
  - The [detail log](#) `c<encoded timestamp>.sta` displays the progress.

For more information on how to view the logs with BR\*Tools, see [Showing Logs with BR\\*Tools](#).

## Deleting Harmful Statistics with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to delete harmful database statistics.

## Prerequisites

- Make sure you have set the necessary BRCONNECT parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRCONNECT.
- The database must be open.

## Procedure

1. Start BRGUI or BRTOOLS.
2. Choose *Database statistics Delete harmful statistics*.
3. Set the required options:

| Menu Entry                               | Equivalent BRCONNECT Command Option                                                                                            |
|------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------|
| <i>BRCONNECT profile (profile)</i>       | <a href="#">-p -profile</a>                                                                                                    |
| <i>Database user/password (user)</i>     | <a href="#">-u -user</a>                                                                                                       |
| <i>Database owner for delete (owner)</i> | <a href="#">-f stats -o -owner</a>                                                                                             |
| <i>Ignore DBSTATC settings (ignore)</i>  | <a href="#">-f stats -n -ignore</a>                                                                                            |
| <i>Exclude from delete (exclude)</i>     | <a href="#">-f stats -e -exclude</a>                                                                                           |
| <i>Tables for delete (table)</i>         | <a href="#">-f stats -t -table</a>                                                                                             |
| <i>Confirmation mode (confirm)</i>       | <a href="#">-c -confirm</a>                                                                                                    |
| <i>Query mode (query)</i>                | <a href="#">-q -query</a>                                                                                                      |
| <i>Force options (force)</i>             | <a href="#">-f stats -f -force</a>                                                                                             |
| <i>Extended output (output)</i>          | <a href="#">-o -output</a>                                                                                                     |
| <i>Message language (language)</i>       | <a href="#">-l -language</a>                                                                                                   |
| <i>BRCONNECT command line (command)</i>  | This shows you the <a href="#">BRCONNECT -f stats -t harmful -d</a> command that is to be executed using the current settings. |

1. To start processing with the selected options, choose *Continue*.
2. Check the results in the [BRCONNECT logs](#).
  - The [summary log](#) `conn<DBSID>.log` displays the return code.
  - The [detail log](#) `c<encoded timestamp>.dst` displays the progress.

For more information on how to view the logs with BR\*Tools, see [Showing Logs with BR\\*Tools](#).

## Managing Database Statistics with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to manage database statistics.

### Prerequisites

- Make sure you have set the necessary BRSPACE parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRSPACE.
- The database must be open.

### Procedure

1. Start the procedure using BRGUI or BRTOOLS, or from the command line:
  - BRGUI or BRTOOLS:

1. Choose *Database Statistics Manage database statistics*.

BRGUI or BRTOOLS displays the menu *BRSPACE main options for managing database statistics*, where you specify the options with which you call BRSPACE.

2. Set the required options:

| Menu Entry                           | Equivalent BRSPACE Command Option        |
|--------------------------------------|------------------------------------------|
| <i>BRSPACE profile (profile)</i>     | <a href="#">-p -profile</a>              |
| <i>Database user/password (user)</i> | <a href="#">-u -user</a>                 |
| <i>Statistics action (action)</i>    | <a href="#">-f mstats -a -action</a>     |
| <i>Statistics export Id (expid)</i>  | <a href="#">-f mstats -i -expid</a>      |
| <i>Tablespace names (tablespace)</i> | <a href="#">-f mstats -s -tablespace</a> |
| <i>Table owner (owner)</i>           | <a href="#">-f mstats -o -owner</a>      |
| <i>Table names (table)</i>           | <a href="#">-f mstats -t -table</a>      |
| <i>Confirmation mode (confirm)</i>   | <a href="#">-c -confirm</a>              |
| <i>Extended output (output)</i>      | <a href="#">-o -output</a>               |
| <i>Scrolling line count</i>          | <a href="#">-s -scroll</a>               |



| Menu Entry                            | Equivalent BRSPACE Command Option                                                                               |
|---------------------------------------|-----------------------------------------------------------------------------------------------------------------|
| <i>(scroll)</i>                       |                                                                                                                 |
| <i>Message language (language)</i>    | <a href="#">-l-language</a>                                                                                     |
| <i>BRSPACE command line (command)</i> | This shows you the <a href="#">BRSPACE -f mstats</a> command that is to be executed using the current settings. |

3. Choose *Continue*.

BRGUI or BRTOOLS prompts you to start BRSPACE.

4. Choose *Continue* to start BRSPACE.

- Command line:

Enter at least the following command:

```
brspace -f mstats
```

You can enter more parameters, including the action, if required. For more information, see [BRSPACE -f mstats](#).

2. Note
3. Whichever way you start the procedure – with BRGUI or BRTOOLS, or from the command line – you can use quick mode if you know the object name, in this case the table name. For more information, see [How to Use BR\\*Tools](#).
- 4.
5. BRSPACE starts and you see a message that includes *Start of BRSPACE processing*. From now on, BRSPACE writes a [detail log](#).
6. BRSPACE displays the *Manage database statistics main menu*.
7. You can use quick mode as follows:
  - If you have already entered the action, continue with step 4.
  - If you have already entered the table name and action, continue with step 6.

#### Note

If you have entered multiple table names, BRSPACE displays as confirmation a *List of tables for manage statistics*. You cannot make a selection from this list. If required, go back and make a new selection.

Continue with step 6 (quick mode).

8. Choose or confirm the required action:
  - Export table statistics
  - Import table statistics
  - Delete table statistics

- Restore table statistics
- Lock table statistics
- Unlock table statistics
- Delete statistics exports
- Show statistics exports
- Show statistics versions

9. If you have already entered the table name, continue with step 6 (quick mode).

BRSPACE displays the table or export list, depending on which action you chose:

- For actions export, delete, restore, lock, or unlock table statistics:

| <b>List Entry</b> | <b>Meaning</b>                          |
|-------------------|-----------------------------------------|
| <i>Pos.</i>       | List sequence number                    |
| <i>Owner</i>      | Table owner                             |
| <i>Table</i>      | Table name                              |
| <i>Part.</i>      | Partitioned status                      |
| <i>Vers.</i>      | Version                                 |
| <i>Lock</i>       | Lock status                             |
| <i>Analyzed</i>   | Date last analyzed by update statistics |
| <i>Rows</i>       | Number of rows                          |

- For actions import table statistics, delete statistic exports, or show statistics exports:

| <b>List Entry</b> | <b>Meaning</b>       |
|-------------------|----------------------|
| <i>Pos.</i>       | List sequence number |
| <i>Run</i>        | Export run           |
| <i>Date</i>       | Date of export       |
| <i>Export Id.</i> | Export ID            |
| <i>Owner</i>      | Export owner         |
| <i>Tables</i>     | Number of tables     |

Note

BRSPACE only displays entries that can be processed by your chosen action.

10. Select a table or export or select multiple tables or exports.

Example

These examples apply only to input in character mode.

To select the first three tables or exports in the list, enter 1-3.

To select the first and third tables or exports, enter 1, 3.

To select the first three tables or exports and the fifth, enter 1-3, 5.

To select all entries, enter 0.

For actions export, delete, restore, lock, and unlock, BRSPACE displays the menu *Options for manage statistics of tables*.

11. For actions export, delete, restore, lock, and unlock, set the required options:

| Menu Entry                                                           | Meaning                                                               |
|----------------------------------------------------------------------|-----------------------------------------------------------------------|
| <i>Last analyzed time stamp (analyzed)</i><br>– display only         | Date and time of the last statistics analysis for the selected tables |
| <i>Number of statistics versions (stats_vers)</i><br>– display only  | Number of statistics versions for the selected tables                 |
| <i>Current statistics lock status (stats_lock)</i><br>– display only | Current lock status for the selected tables                           |
| <i>Manage statistics action (action)</i>                             | The action that you selected above.                                   |
| <i>Statistics export Id (expid)</i>                                  | <a href="#">-f mstats -ij-expid</a>                                   |
| <i>Time for restore statistics (time)</i>                            | <a href="#">-f mstats -m time</a>                                     |

12. To start processing with the selected options, choose *Continue*.

## Result

Check the results in the [BRSPACE logs](#).

- The [summary log](#) space<DBSID>.log displays the return code.
- The [detail log](#) s<encoded timestamp>.mst displays the details.
- The [structure change log](#) struc<DBSID>.log logs all structure changes.

For more information about how to view the logs with BR\*Tools, see [Showing Logs and Profiles with BR\\*Tools](#).

## Additional Functions with BR\*Tools

You can perform additional functions for your Oracle database with . [BR\\*Tools](#).

### Integration

- BRTOOLS normally calls the SAP tool BRCONNECT. You can also perform these functions directly by calling BRCONNECT from the command line.

Recommendation

We recommend you to normally use BRTOOLS rather than BRCONNECT. This is because the BRTOOLS menus simplify entry of the correct parameters.

- Make sure that you have set the necessary parameters for BRCONNECT in the [Initialization Profile init<DBSID>.sap](#).

### Features

You can perform the following additional functions with BR\*Tools:

- Show profiles and logs
- Clean up DBA logs and tables
- Adapt NEXT extents
- Change password of database user
- Create or change synonyms for DBA tables

### Activities

1. You choose the required function in BRTOOLS and check the displayed parameters, changing them as required.

The default parameter values, which are set in the [initialization profile init<DBSID>.sap](#), are as follows:

- Clean up DBA logs that are older than 30 days
  - Clean up records in DBA tables that are older than 100 days
  - Adapt NEXT extents for all tables and indexes
2. If required, you change the default values for the function parameters in the initialization profile `init<DBSID>.sap` and restart BRTOOLS.
  3. If required, you choose *Additional functions Reset program status* to set the defaults used to the values in the initialization profile `init<DBSID>.sap`. For certain input values, there is no corresponding parameter in the initialization profile, in which case the default value from the BRTOOLS program is used.
  4. You start processing.

5. You check the results in the [BRCONNECT logs](#).

## Showing Profiles and Logs with BR\*Tools

You can use [BR\\*Tools](#) for Oracle to show the following logs and profiles generated by BR\*Tools:

- Oracle profile – see Oracle documentation
- [BR\\*Tools profile, that is, initialization Profile init<DBSID>.sap](#)
- [BRBACKUP logs](#)
- [BRARCHIVE logs](#)
- [BRRESTORE logs](#)
- [BRRECOVER logs](#)
- [BRCONNECT logs](#)
- [BRSPACE logs](#)

For more information, see [Profiles, Logs, Messages, and Return Codes for BR\\*Tools](#).

### Procedure

1. Start BRGUI or BRTOOLS.
2. Choose one of the following:
  - *Additional Functions Show Profiles and Logs Oracle profile*
  - *Additional Functions Show Profiles and Logs BR\*Tools profile*
  - *Additional Functions Show Profiles and Logs BRBACKUP logs*
  - *Additional Functions Show Profiles and Logs BRARCHIVE logs*
  - *Additional Functions Show Profiles and Logs BRRESTORE logs*
  - *Additional Functions Show Profiles and Logs BRRECOVER logs*
  - *Additional Functions Show Profiles and Logs BRCONNECT logs*
  - *Additional Functions Show Profiles and Logs BRSPACE logs*

BRTOOLS displays a list of available logs or profiles.

3. Select the log or profile that you want to show.

BRTOOLS indicates which file is to be shown, as in the following example:

Example

```
Following file will be displayed:  
/oracle/GC2/sapbackup/rdjxfcel.rsb
```

4. Choose *Continue* to show the log or profile.

# Cleaning Up DBA Logs and Tables with BR\*Tools

[BR\\*Tools](#) You can clean up the DBA logs and tables for your Oracle database with .

This function helps you to free disk space used by old DBA logs and protects your database against unnecessary growth of the DBA tables.

For more information, see [Clean Up Old Logs and Trace Files with BRCONNECT](#).

## Prerequisites

- Make sure you have set the necessary BRCONNECT parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRCONNECT.
- The database must be open.

## Procedure

1. Start BRGUI or BRTOOLS.
2. Choose *Additional functions Clean up DBA logs and tables* .
3. Set the required options:

| Menu Entry                                | Equivalent BRCONNECT Command Option     |
|-------------------------------------------|-----------------------------------------|
| <i>BRCONNECT profile (profile)</i>        | <a href="#">-p -profile</a>             |
| <i>Database user/password (user)</i>      | <a href="#">-u -user</a>                |
| <i>Database owner for cleanup (owner)</i> | <a href="#">-f cleanup -o -owner</a>    |
| <i>BRBACKUP logs (backup)</i>             | <a href="#">-f cleanup -b -backup</a>   |
| <i>BRARCHIVE logs (archive)</i>           | <a href="#">-f cleanup -a -archive</a>  |
| <i>BRRESTORE logs (restore)</i>           | <a href="#">-f cleanup -r -restore</a>  |
| <i>BRRECOVER logs (recover)</i>           | <a href="#">-f cleanup -v -recover</a>  |
| <i>BRCONNECT logs (connect)</i>           | <a href="#">-f cleanup -c -connect</a>  |
| <i>BRSPACE log (space)</i>                | <a href="#">-f cleanup -s -space</a>    |
| <i>BRBACKUP disk backups (diskback)</i>   | <a href="#">-f cleanup -k -diskback</a> |
| <i>BRARCHIVE disk backups (diskarch)</i>  | <a href="#">-f cleanup -i -diskarch</a> |
| <i>BRSPACE export dumps (expdump)</i>     | <a href="#">-f cleanup -e -expdump</a>  |

| Menu Entry                                        | Equivalent BRCONNECT Command Option                                                                                |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------|
| Oracle trace files ( <i>trace</i> )               | <a href="#">-f-function cleanup -tl-trace</a>                                                                      |
| Database logs in SDBAH and SDBAD ( <i>dblog</i> ) | <a href="#">-f-function cleanup -dl-dblog</a>                                                                      |
| XDB logs in DBA tables ( <i>xdblog</i> )          | <a href="#">-f-function cleanup -xl-xdblog</a>                                                                     |
| Check messages in DBCHECKORA ( <i>checkmsg</i> )  | <a href="#">-f-function cleanup -ml-checkmsg</a>                                                                   |
| Confirmation mode ( <i>confirm</i> )              | <a href="#">-c confirm</a>                                                                                         |
| Query mode ( <i>query</i> )                       | <a href="#">-q query</a>                                                                                           |
| Clean up selected logs only ( <i>limit</i> )      | <a href="#">-f-function cleanup -l limit</a>                                                                       |
| Extended output ( <i>output</i> )                 | <a href="#">-o output</a>                                                                                          |
| Message language ( <i>language</i> )              | <a href="#">-l language</a>                                                                                        |
| BRCONNECT command line ( <i>command</i> )         | This shows you the <a href="#">BRCONNECT -f cleanup</a> command that is to be executed using the current settings. |

1. To start processing with the selected options, choose *Continue*.
2. Check the results in the [BRCONNECT logs](#).
  - The [summary log](#) `conn<DBSID>.log` displays the return code.
  - The [detail log](#) `c<encoded timestamp>.cln` displays the progress.

For more information on how to view the logs with BR\*Tools, see [Showing Logs with BR\\*Tools](#).

## Adapting Next Extents with BR\*Tools

You can adapt the next extents for tables in your Oracle database with [BR\\*Tools](#). The aim is to avoid extent overflow, which causes transactions to abort.

For more information, see:

- [Adapt Next Extents with BRCONNECT](#).
- [Methods of Adapting Next Extent Size](#)

## Prerequisites

- Make sure you have set the necessary BRCONNECT parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRCONNECT.
- The database must be open.
- It only makes sense to adapt next extents for tablespaces that are dictionary-managed. If all tablespaces are locally managed, do *not* use this function.

## Procedure

1. Start BRGUI or BRTOOLS.
2. Choose *Additional functions Adapt next extents* .
3. Set the required options:

| Menu Entry                              | Equivalent BRCONNECT Command Option                                                                             |
|-----------------------------------------|-----------------------------------------------------------------------------------------------------------------|
| <i>BRCONNECT profile (profile)</i>      | <a href="#">-p -profile</a>                                                                                     |
| <i>Database user/password (user)</i>    | <a href="#">-u -user</a>                                                                                        |
| <i>Database owner for adapt (owner)</i> | <a href="#">-f next -o -owner</a>                                                                               |
| <i>Maximum extent size in KB (max)</i>  | <a href="#">-f next -m -max</a>                                                                                 |
| <i>Maximum extent count (limit)</i>     | <a href="#">-f next -l -limit</a>                                                                               |
| <i>Special NEXT extents (special)</i>   | <a href="#">-f next -s -special</a>                                                                             |
| <i>Exclude from adapt (exclude)</i>     | <a href="#">-f next -e -exclude</a>                                                                             |
| <i>Tables for adapt (table)</i>         | <a href="#">-f next -t -table</a>                                                                               |
| <i>Confirmation mode (confirm)</i>      | <a href="#">-c -confirm</a>                                                                                     |
| <i>Query mode (query)</i>               | <a href="#">-q -query</a>                                                                                       |
| <i>Force options (force)</i>            | <a href="#">-f next -f -force</a>                                                                               |
| <i>Additional output (output)</i>       | <a href="#">-o -output</a>                                                                                      |
| <i>Message language (language)</i>      | <a href="#">-l -language</a>                                                                                    |
| <i>BRCONNECT command line (command)</i> | This shows you the <a href="#">BRCONNECT -f next</a> command that is to be executed using the current settings. |



4. To start processing with the selected options, choose *Continue*.
5. Check the results in the [BRCONNECT logs](#).
  - o The [summary log](#) `conn<DBSID>.log` displays the return code.
  - o The [detail log](#) `c<encoded timestamp>.nxt` displays the progress.

For more information about how to view the logs with BR\*Tools, see [Showing Logs with BR\\*Tools](#).

## Methods of Adapting Next Extent Size

You can use the BRCONNECT function in BR\*Tools to adapt the next extent size (you cannot do this with BRSPACE because it is designed for the administration of locally managed tablespaces). Here we explain examples of the following two methods that you can use to change next extents with BRCONNECT:

- Delayed method

Enter the required nonstandard value in the [next\\_special](#) parameter of the initialization profile `init<DBSID>.sap`.

This parameter has the following syntax:

```
next_special = [<owner>.<table>:<size>[/<limit>]
| [<owner>.<index>:<size>[/<limit>] | (<object_size_list>)
```

You can enter `<size>` in KB; MB, or GB.

`<limit>` specifies the maximum number of extents, `MAX_EXTENTS`. 0 means unlimited.

Example

```
next_special = (SDBAH:400K, SAPR3.SDBAD:2M/300)
```

The first run of `brconnect -f next` after this parameter change permanently alters the next extent values.

Recommendation

We recommend you to use this method.

- You can use this method to quickly increase the next extent size of a table or index:

Caution

Only use this method if you want to set the next extent to a *higher* value than the standard value. If you set a lower value, the next run of `brconnect -f next` sets it back to the standard value. If you want to reduce the next extent, use the first method above.

1. Start BRGUI or BRTOOLS.

2. Choose *Additional functions Adapt next extents* .
3. Enter a value in *Special NEXT extents (special)* to set the option [-f next -s|-special](#)

Example

If you enter a value of 400k/300, this sets the next extent, NEXT\_EXTENT, to 400 KB and the maximum number of extents, MAX\_EXTENTS, to 300.

4. In the next menu, if you only want to change the value for the table and not for its indexes, set *force* to *nocasc*. Or, if you only want to change it for indexes, you can directly enter index names in the field *table*.
5. To start processing with the selected options, choose *Continue*.

## Changing the Password of the Database User with BR\*Tools

You can change the password of the SAP user in your Oracle database with [BR\\*Tools](#).

Caution

For security reasons, we strongly recommend that you do not use the standard password of the SAP user in a production system.

For more information, see [Additional BRCONNECT Functions](#).

### Prerequisites

The database must be open.

### Procedure

1. Start BRGUI or BRTOOLS.
2. Choose *Additional functions Change password of database user* .
3. Set the required options:

| Menu Entry                                       | Equivalent BRCONNECT Command Option |
|--------------------------------------------------|-------------------------------------|
| <i>BRCONNECT profile (profile)</i>               | <a href="#">-p -profile</a>         |
| <i>Database user/password (user)</i>             | <a href="#">-u -user</a>            |
| <i>Database owner to change password (owner)</i> | <a href="#">-f chpass -o -owner</a> |
| <i>Message language</i>                          | <a href="#">-l -language</a>        |

| <b>Menu Entry</b>                           | <b>Equivalent BRCONNECT<br/>Command Option</b>                                                                    |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------|
| <i>(language)</i>                           |                                                                                                                   |
| <i>BRCONNECT command<br/>line (command)</i> | This shows you the <a href="#">BRCONNECT -f chpass</a> command that is to be executed using the current settings. |

4. To start processing with the selected options, choose *Continue*.
5. Enter the new password when requested.
6. Check the results by viewing the messages displayed on the screen.

## Creating or Changing Synonyms for DBA Tables

[BR\\*Tools](#) You can create or change the synonyms for the DBA tables in your Oracle database with .

The results of DBA operations performed using BR\*Tools are stored in one component of a multi-component database. You can use this function to change the component where the results are to be stored and where they can be viewed.

For more information, see [Additional BRCONNECT Functions](#).

### Prerequisites

- Make sure you have set the necessary BRCONNECT parameters in the [initialization profile init<DBSID>.sap](#), because BRTOOLS uses these when it calls BRCONNECT.
- The database must be running.

### Procedure

1. Start BRGUI or BRTOOLS.
2. Choose *Additional functions Create/change synonyms for DBA tables* .
3. Set the required options:

| <b>Menu Entry</b>                              | <b>Equivalent BRCONNECT<br/>Command Option</b> |
|------------------------------------------------|------------------------------------------------|
| <i>BRCONNECT profile<br/>(profile)</i>         | <a href="#">-p -profile</a>                    |
| <i>Database user/password<br/>(user)</i>       | <a href="#">-u -user</a>                       |
| <i>Database owner for<br/>synonyms (owner)</i> | <a href="#">-f crsyn -o -owner</a>             |
| <i>Message language<br/>(language)</i>         | <a href="#">-l -language</a>                   |

| Menu Entry                              | Equivalent BRCONNECT<br>Command Option                                                                           |
|-----------------------------------------|------------------------------------------------------------------------------------------------------------------|
| <i>BRCONNECT command line (command)</i> | This shows you the <a href="#">BRCONNECT -f crsyn</a> command that is to be executed using the current settings. |

4. To start processing with the selected options, choose *Continue*.
5. Check the results by viewing the messages displayed on the screen.

## BR\*Tools in Detail

This section gives you detailed information on BR\*Tools, including the command options for calling the tools from the command line.

## BRBACKUP

This SAP tool for the Oracle database enables you to back up database files. For more information about features common to both tools, see [Common Features of BRBACKUP and BRARCHIVE](#).

The smallest unit that can be saved with BRBACKUP is a file. You can use BRBACKUP for backing up both files in the database and non-database files and directories. Use the [backup\\_mode](#) from the [Initialization Profile init<DBSID>.sap](#) or the command option `brbackup -m|-mode` for this purpose.

For more information, see:

- [Backing up Non-Database Files and Directories](#)
- [Backing Up Database Files](#)
- [BRTOOLS](#)
- [Hardware Compression for BRBACKUP](#)
- [Logging](#)
- [Completion of BRBACKUP Backups](#)

## Backing up Database Files

You can back up individual database files, tablespaces, or the entire database. BRBACKUP uses the procedures recommended by Oracle for performing online and offline database backups.

To specify database files, you can use the file ID, a generic name, or the full path name. By specifying ID intervals or a generic path, you can back up all the database files that meet these specifications.

- When you specify `all` or `full`, all the database data files (and therefore all the tablespaces) and control file are saved. For an offline backup, a member of every online redo log group is also saved.
- You can combine the value `all` with an `<object list>`. This enables you to back up other non-database files in addition to the database itself. However, we do not recommend this procedure. Whenever possible, save the database files and the non-database files in separate backup runs.
- The control file can only be addressed explicitly using the file ID 0. It is not usually necessary to back up this file, since it is always backed up automatically whenever at least one database data file is backed up.
- Online redo log files can only be addressed explicitly using the redo log group number, which must be assigned a leading zero (`0<n>`). To save all the online redo log files, specify the file ID 00. However, this might be required only for partial offline backup. It is not normally required because complete offline backups save these files automatically.
- All file IDs in the interval specified by `<file_ID1>-<file_ID2>` must be known in the database.
- If you use a generic path to define database data files, make sure that this path contains the `SAPDATA_HOME` directory and an additional generic specification (for example, `sapdata<n>` directory).

#### Note

If you want to back up, for example, only database files from `sapdata1` but not from `sapdata0`, specify one additional character to make the path name unique:

```
/oracle/c11/sapdata1
```

## More Information

[backup\\_mode](#) or

[-m|-mode](#).

## Backing Up Non-Database Files and Directories

Non-database files should only be backed up with BRBACKUP after an SAP upgrade or an Oracle upgrade. This backup method is *not* a replacement for a file system backup using operating system features.

- Directories and non-database files for backup must be specified with their complete paths.
- If you use BRBACKUP to save directories, only the files in that directory are saved; files of any existing subdirectories will not be saved. The use of parameter `sap_dir` or `ora_dir` is an exception, as it is possible to make a backup of all the non-database files of the SAP or Oracle environment with it.
- When you save directories with software compression (`compress = yes`), their contents are not compressed.

- If you want to back up a large number of non-database files and directories (all the SAP executables and profiles, for example), we recommend carrying out this backup separately.

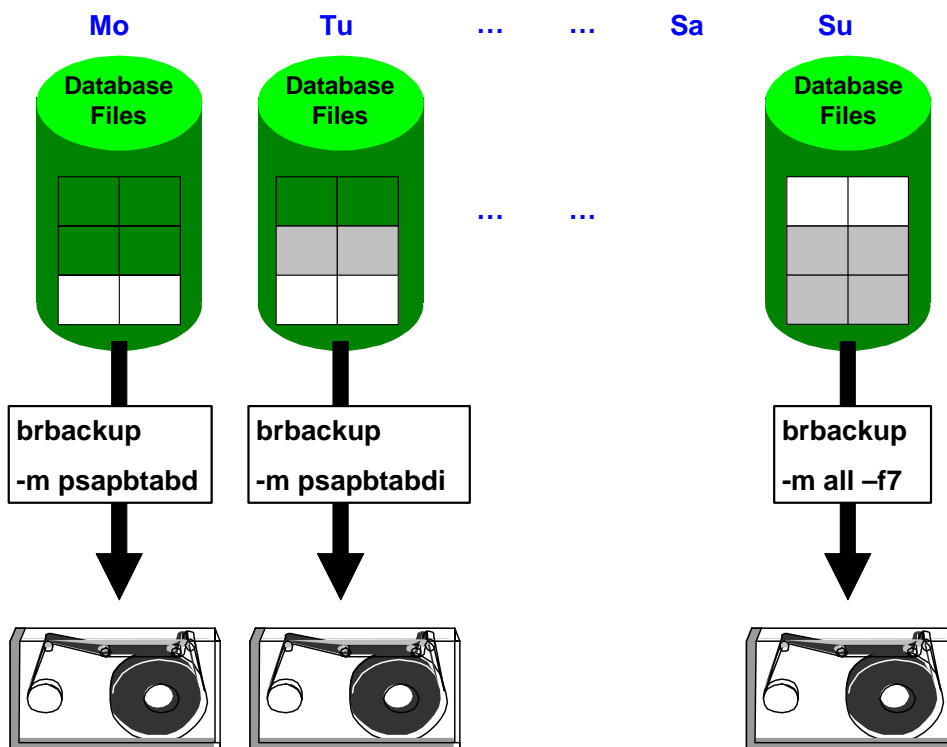
See also [backup\\_mode](#) or

[-m|-mode](#).

## Completion of BRBACKUP Backups

The completion of database backups is relevant in the following situations:

- A backup has terminated and you do not want to repeat it. Using the BRBACKUP option [-f|-fillup](#) you can simply continue the backup. BRBACKUP automatically determines the files that are still to be backed up: target set (defined with the parameter `backup_mode` or the option `-m`) minus set backed up (BRBACKUP detailed log). The completion of a backup can also refer to several terminated backups. In this case a new set of files to be backed up can be specified, differing from the original target volume set.
- You are making partial database backups and want to ensure, or monitor, that these will produce a complete backup. This is especially important for large databases, for which it is recommended to spread the backup over several days (for example, a week), in order to reduce the volume of data to be backed up every day. At the end of the week, to ensure that you have a complete backup, complete the partial backups on the last day of the cycle with the option `-f|-fill`. As above BRBACKUP automatically determines the files to be backed up.



Example

`brbackup -f b<timestamp>.pft`: the backup with the log name `b<timestamp>.pft` is completed

`brbackup -f last`: the last executed backup is completed

`brbackup -f 1`: the current day's backup is completed

`brbackup -f 7`: the backups of the last seven days (including the current day) are completed, as in the graphic above.

## More Information

[-f-fillup](#)

[Completion of BRRESTORE Runs](#)

[Partial Backups](#)

## Hardware Compression for BRBACKUP

When backup devices with hardware compression are used, BRBACKUP requires the current compression rates of the database files in order to determine the quantity of data to be saved after the hardware compression. Only in this manner can BRBACKUP ensure that the specified tape size is not exceeded and that the database files are correctly divided between the tapes.

BRBACKUP can only estimate the quantity of data that can be written to the volume, because BRBACKUP cannot directly determine the compression rates for hardware compression. BRBACKUP uses the software compression rates as an estimate. SAP therefore assumes that hardware and software compression return similar results. See [compress\\_cmd](#).

Before the first backup using tape devices with hardware compression, start a compression run to determine the compression rates:

```
brbackup -k only
```

This does not actually start a backup. It only determines the compression rates. The database files are only compressed (not saved) and the determined compression rates are stored in table SDBAD and in a detail log.

For more information, see [Logs for BR\\*Tools](#).

Caution

Repeat this activity at least once a month to update the compression rates.

Also repeat this activity after a reorganization or after the loading of a large amount of data.

## More Information

[Hardware Compression](#)

## Command Options for BRBACKUP

This section describes the command options for [BRBACKUP](#). If you start BRBACKUP without command options, the values in the [Initialization Profile init<DBSID>.sap](#) are used. Unless otherwise specified in the initialization profile `init<DBSID>.sap`, BRBACKUP starts an offline complete backup to a local tape device, with a storage capacity of 1200 MB and no file compression.

For more information about features common to both tools, see [Common Features of BRBACKUP and BRARCHIVE](#).

If you use BRBACKUP with command options, these override the corresponding values in the initialization profile. To use the options, you can specify either the letter indicated or the complete word.

This is the full command syntax:

### Syntax

```
brbackup
[-a|-archive [<bra_options>]]
[-b|-backup <log_name>|last [-bd|-backup_delete <log_name>|last]
[-c|-confirm [force]]
[-d|-device tape|disk|pipe|disk_copy|disk_standby|tape_auto|
pipe_auto|tape_box|pipe_box|util_file|util_file_online|
util_vol|util_vol_online|
stage|stage_copy|stage_standby|rman_prep|rman_util
|rman_disk|rman_stage]
[-db|-delete_backup <log_name>|last]
[-e|-execute <number>]
[-f|-fillup <log name1>[<log name2>, ..]
|<yyyy-mo-dd hh.mi.ss>|<days>|last]
[-h|-help [version]]
[-i|-initialize [force|show]]
[-k|-compress no|yes|hardware|only]
[-l|-language E|D]
[-m|-mode all|all_data|full|incr|<tablespace_name>|<file_ID>|
<file_ID1>-<file_ID2>|<generic_path>|<object
list>]|sap_dir|ora_dir|all_dir|partial|non_db]
[-n|-number <number of tapes>]
[-o|-output dist|time[,time|dist]]
[-p|-profile <profile>]
[-q|-query [check|nolog|split]]
[-r|-parfile <parameter_file>]
[-s|-saveaset 1|2|3|4|tsp|all]
[-t|-type online|online_cons|offline|offline_force|
```



offline\_standby|offline\_stop|online\_split|offline\_split|  
online\_mirror|offline\_mirror|online\_standby|offstby\_split|offstby\_  
mirror]  
[-u|-user [<user>[/<password>]]]  
[-v|-volume <volume>|<volume list>|SCRATCH]  
[-w|-verify [use\_dbv|only\_dbv|only\_conf|use\_rmv|only\_rmv]]  
[-V|-VERSION [ALL]]

End of the code.

*See also:*

[-a|-archive](#)

[-b|-backup](#)

[-c|-confirm](#)

[-d|-device](#)

[-e|-execute](#)

[-f|-fillup](#)

[-h|-help](#)

[-i|-initialize](#)

[-k|-compress](#)

[-l|-language](#)

[m|-mode](#)

[-n|-number](#)

[-o|-output](#)

[-p|-profile](#)

[-q|-query](#)

[-r|-parfile](#)

[-t|-type](#)

[-u|-user](#)

[-v|-volume](#)

[-w|-verify](#)

[-V|-VERSION](#)

## -a|-archive

This BRBACKUP command option starts BRARCHIVE at the end of a BRBACKUP run.

With this option BRARCHIVE is automatically started after the BRBACKUP backup of the data files. The offline redo log files, as well as all logs, are then copied by BRARCHIVE to the same tape(s) immediately after the backed up database files. This is also possible to disk with BACKINT.

The main advantage of this procedure is that you only have to start or schedule one program (usually BRBACKUP). The second program is started automatically after the first program has ended. If the backup is made to tape, you can also use the tape capacity optimally.

This procedure enables both an unattended backup and an effective usage of tape capacity.

You can also specify other options for BRARCHIVE backups. If not the appropriate defaults are used. Always place the option `-a` (including the additional options) at the end of the BRBACKUP command line call.

### Syntax

```
a|-archive [<bra_options>]
```

End of the code.

Default value: BRBACKUP and BRARCHIVE are called separately, the backup of the data files and the offline redo log files takes place in separate backups on separate tapes.

See: [Command Options for BRARCHIVE](#) and [Initialization Profile init<DBSID>.sap](#)

### Example

Unattended online backup of a database in parallel on two tape devices followed by a startup of BRARCHIVE, in order to create two copies of the offline redo log files in parallel on the same tapes.

```
brbackup -m all -t online -c -a -ssd -c
```

For more information, see:

[BRBACKUP and BRARCHIVE Backups in One Run](#)

## -b|-backup

This BRBACKUP command option backs up a disk backup.

### Syntax

```
-b|-backup [<log_name>|last]
```

End of the code.

Default value: `last`

Possible values:

- `<log_name>`: copies the BRBACKUP disk backup with this name to the current backup tape, remote disk, or backup utility (BACKINT)
- `last`: selects the last successful disk backup and copies this to the current backup tape, remote disk, or backup utility (BACKINT)

### Example

Backup of the last disk backup with the same parameters as defined in `init<DBSID>.sap`:

```
brbackup -b
```

See also:

[Two-Phase Backup](#)

## **-bd|-backup\_delete**

This BRBACKUP command option backs up then deletes a disk backup. After the backup the data is automatically deleted from the disk.

### Syntax

```
-bd|-backup_delete [ <log_name>|last ]
```

End of the code.

Default value: `last`

Possible values:

- `<log_name>`: copies the BRBACKUP backup to disk with this name to the current backup tape, or backup utility (BACKINT)
- `last`: selects the last successful backup to disk and copies this to the current backup tape, or backup utility (BACKINT)

### Example

Backup of the last disk backup with the same parameters as defined in `init<DBSID>.sap` followed by deletion:

```
brbackup -bd
```

See also:

[Two-Phase Backup](#)

## **-c|-confirm**

This BRBACKUP command option backs up in unattended mode. This option suppresses most of the confirmation messages normally displayed during the execution of a backup. This enables you to run the program in unattended mode.

### Syntax

```
-c|-confirm [force]
```

End of the code.

Default value: confirmation messages are issued and user entry is expected.

Possible values:

- `-c|-confirm`

This option suppresses *most* of the confirmation messages normally displayed during the execution of a backup. This enables you to run the program in unattended mode.

However, `-c|-confirm` does *not* suppress the following confirmation messages:

- Interactive password entry. To schedule a backup using CRON, use an appropriate script for entering the password. See [Unattended Backup](#).
- Prompt to mount the next volume, when more volumes are required than there are backup devices available. When a new volume must be mounted in the backup device during the backup, the utility responds as follows:
  - When a console is available, you can mount the next volume, make the entries required by the utility, and continue the backup.
  - When no console is available (the backup was scheduled using CRON or the CCMS, for example), the backup terminates.

#### Example

A backup is scheduled using CRON. Three tapes are required for the backup. Two tape devices are available and you have mounted two of the requested tapes. The backup is started, but is terminated once the two tapes have been written.

- Prompt for mounting the next tape when a `cpio` continuation tape is requested (see [cpio Continuation Tape](#)). The backup is started. When the physical end of tape is reached, the utility responds as follows:
  - When a console is available, you can mount the continuation tape, make the entries required by the utility, and continue the backup.
  - When no console is available (the backup was scheduled using CRON or the CCMS, for example), the backup terminates.

- `force`

When you specify the `-c force` option, you also suppress the following confirmation messages:

- Prompt to mount the next volume. When more volumes are required than there are backup devices available, the backup is not even started.
- Prompt to mount a `cpio` continuation tape. If a continuation tape is required, the backup is terminated at this point, and an appropriate `cpio` error message is displayed.

However, `-c force` does *not* suppress the following confirmation prompt:

Interactive password entry. To schedule a backup using CRON, use an appropriate script for entering the password. See [Unattended Backup](#).

#### Note

To run the SAP utility with, for example, the UNIX utility CRON, use the option `-c force`.

## **-db|-delete\_backup**

This BRBACKUP command option deletes a disk backup.

Syntax

```
-db|-delete_backup [<log_name>|last]
```

End of the code.

Default value: last

Possible values:

- `<log_name>`: deletes the BRBACKUP disk backup called `log_name` from disk
- `last`: BRBACKUP selects the last successful disk backup and deletes it from disk

## **-d|-device**

This BRBACKUP command option defines the backup device type.

Syntax

```
-d|-device tape|disk|pipe|disk_copy|disk_standby|tape_auto|  
pipe_auto|tape_box|pipe_box|util_file|util_file_online|  
util_vol|util_vol_online|stage|stage_copy|stage_standby|  
rman_prep|rman_util|rman_disk|rman_stage
```

End of the code.

Default value: tape

Possible values:

- `tape`: backs up to a local tape device
- `disk`: backs up to a local disk
- `pipe`: backs up to a tape device of a remote system (UNIX only)
- `disk_copy`: copies database files to a disk with an identical directory structure. The name of the new `SAP_Home` directory is defined in the parameter `new_db_home`.
- `disk_standby`: copies database files to a disk with an identical directory structure (compare `disk_copy`). To let you construct a standby database, a standby control file is generated and copied.
- `tape_auto` or `pipe_auto`: suppresses prompts for changing the tape. This is only useful when you use a tape device with automatic tape changing (tape changing device).
- `tape_box` or `pipe_box`: specifies jukeboxes or autoloader tape devices that can be addressed locally or remotely. The drivers for the data transfer (`cpio`, `dd`) are defined in the parameters `tape_address` or `tape_address_arch`, the drivers for rewinding are defined in the parameters `tape_address_rew` or `tape_address_rew_arch` and the drivers for mounting and dismounting the tapes are defined in the parameters `tape_address_ctl` or `tape_address_ctl_arch`.

- `util_file` or `util_file_online`: uses external backup programs for file-by-file backup. If you use this option, you might have to create a file containing the parameters required for that type of backup. If a parameter file of this type is required, you must specify the name of the file in the profile parameter `util_par_file` or with the option `-r`. Use parameter `util_file_online` for an online backup (if it is supported by the external backup program). This dynamically sets and ends the backup status of the tablespaces to be saved and thus greatly reduces the volume of offline redo log files during the backup.
- `util_vol` or `util_vol_online`: as for `util_file` or `util_file_online` but for volumes
- `stage|stage_copy|stage_standby`: backs up to remote disk. See [backup\\_dev\\_type](#)
- `rman_util|rman_disk|rman_stage|rman_prep`: backs up with the Oracle Recover Manager (RMAN) and an external backup tool. See [backup\\_dev\\_type](#)

See also:

Parameters in `init<DBSID>.sap`: [backup\\_dev\\_type](#).

## **-e|-execute**

This BRBACKUP command option executes the backup in parallel.

Syntax

```
-e|-execute <number of copy processes>
```

End of the code.

Default value: 0

Possible value:

`<number of copy processes>`: defines the maximum number of parallel copy processes. When set to the default value of 0, the number of parallel copy processes corresponds to the number of backup devices available (tape devices or disks). If the option `-k only` is used to determine the compression rates, the number of parallel copy processes corresponds to the number of disks (or logical volumes) on which the database files reside. The same is valid for `-w only_dbv|only_rmv` and `-d rman_prep`.

You can also define a different number `n` of copy processes. This causes the following to happen:

- Backup to tape
 

The value `n` should be less than or equal to the number of backup devices. If you define a value `n` less than the number of tape devices, this means that you can only use `n` of the available tape devices in parallel. Should a tape change to one of the tape devices used in parallel be necessary, there is an automatic change to the next free backup device and the backup continues there.
- Backup to disk
 

The number of parallel copy processes can be greater than the number of disks defined in `backup_root_dir|stage_root_dir` (but not greater than 255). In this case, one or more disks is written simultaneously by several processes. If you choose the number of copy processes `n` to be less than the number of disks, this means that

you can only use `n` of the available disks in parallel. If one of the disks used in parallel is full, there is an automatic change to the next unused disk and the backup continues there.

See also:

Parameters in `init<DBSID>.sap`: [exec\\_parallel](#).

## **-f|-fillup**

This BRBACKUP command option completes backup runs.

Syntax

```
-f|-fillup <log_name1>[,<log_name2>, ...] |<yyyy-mm-dd  
hh.mi.ss>|<days>|last
```

End of the code.

Default value: `last`

Possible values:

- `<log_name1>[,<log_name2>, ...]`: completes the named backups
- `<yyyy-mm-dd hh.mi.ss>`: completes the backups started since this date and time
- `<days>`: completes all backups in this number of preceding days
- `last`: completes the last backup to run

## **-g|-abort**

This BRBACKUP command option aborts backup runs. To abort BRBACKUP, you open a separate command window to enter the command shown below.

Syntax

```
brbackup -g|-abort stop|term|kill
```

End of the code.

Default value: `none`

Possible values:

- `stop`: causes a soft abort
- `term`: causes a normal abort
- `kill`: causes a hard abort

The abort generates a short log file in the relevant log file directory:

- **UNIX**: `<saparch>/<coded timestamp>.bab`
- **Windows**: `<saparch>\<coded timestamp>.bab`

The aborted backups are marked as faulty. However, you might be able to later use them for certain purposes (such as a fill-up), especially if the abort was soft.

## -h|-help

This BRBACKUP command option provides help information. You can use this option to obtain an overview of all applicable options for BRBACKUP.

Syntax

```
-h|-help [version]
```

End of the code.

Default value: no help

Possible value:

- `version`: displays detailed information about the versions of the program modules

## -i|-initialize

This BRBACKUP command option initializes tape volumes. You can use this option to initialize SAP volumes (that is, tapes) or non-SAP volumes. Only tapes of this type can be used for backup with BRBACKUP. See [Volume Management](#).

Syntax

```
-i|-initialize [force|show]
```

End of the code.

Default value: label checking before initialization

Possible values:

- `-i`: only for SAP volumes

You use this mainly to rename volumes. BRBACKUP suggests the volume name specified in `volume_backup`. Mount the volumes with the appropriate names or other SAP volumes with labels. BRBACKUP checks whether the expiration period has expired. If so, BRBACKUP initializes the volume and assigns it the specified name.

- `-i force`: initializes new or non-SAP volumes

The expiration period is *not* checked. It is also possible to initialize SAP volumes with `-i force`. However, we recommend you to only use this option when necessary, since the expiration period is not checked and the `tape_use_count` stored in the volume label is reset to one.

Note

For both the above options, you can also use in addition the option `-v`. The system then initializes the tapes with the names selected in `-v`.

You receive an error message from BRBACKUP and one of the following `cpio` messages if you want to start the backup program and want to use a non-initialized tape as a backup medium. Initialize this tape using the option `-i force` and then repeat the procedure.

The `cpio` message is platform-dependent and has, for example, the following texts



| Operating System | Message                                                                                  |
|------------------|------------------------------------------------------------------------------------------|
| Linux            | cpio: Bad header - checksum error. cpio: Not a cpio file.                                |
| AIX              | Cannot read from the specified input. Out of phase.                                      |
| HP-UX            | Out of phase -- get help. End of volume.                                                 |
| Solaris          | cpio: Bad header - checksum error. cpio: Not a cpio file, bad header.                    |
| Windows          | cpio: Read error on file...: No more data on tape.<br>cpio: Bad magic number in archive. |

- `-i show`: displays the volume label information

No initialization is executed. The volume remains unaltered.

## **-k|-compress**

This BRBACKUP command option sets the compression mode. If you specify this option, you can compress your data files *before* the backup runs.

Syntax

```
-k|-compress no|yes|hardware|only
```

End of the code.

Default value: `no` (no compression)

Possible values:

- `yes`: specifies software compression.
- `hardware`: specifies hardware compression. The prerequisites are that the tape device must support hardware compression and hardware compression must be active. See [Hardware Compression](#).
- `only`: determines the current compression rate of the individual data files. In this case, no backup is started. If you use tape devices with hardware compression, we recommend that you repeat this procedure about once a month. For more information, see [Hardware Compression for BRBACKUP](#).

See also [compress\\_cmd](#).

Corresponding parameter in `init<DBSID>.sap`: [compress](#)

## **-l|-language**

This BRBACKUP command option sets the message language.

Syntax

```
-l|-language E|D
```

End of the code.

Default value: E

#### Note

The default becomes invalid if you specify another value by setting the environment variable `BR_LANG` (language variable).

If you set option `-1`, the value specified with this option applies.

Possible values:

- D: German
- E: English

## **-m|-mode**

This BRBACKUP command option defines the file to be backed up. You can perform a complete database backup or back up specific tablespaces or files (whether part of the database or not). You can create object lists.

#### Syntax

```
-m|-mode all|all_data|full|incr|<tablespace>|<file_ID>|<file_ID1>-<file_ID2>|<generic_path>|<object_list>|sap_dir|ora_dir|all_dir|partial|non_db
```

End of the code.

Default: all

Possible values:

- `all`: backs up the whole database. This backup is *not* part of an incremental backup strategy.

In a [structure-retaining database copy](#) (`backup_dev_type = disk_copy` or `disk_standby`) you can retain the distribution of the `sapdata` directories to different drives (only for Windows).

#### Example

The files of the drive d are copied to drive k, the files of the drive e are copied to the drive l and the files of the drive f are copied to the drive m.

```
brbackup -d disk_copy -m all,d:=k:,e:=l:,f:=m:
```

If you do not specify a target drive, all files are copied to the directory defined in the parameter [new\\_db\\_home](#).

- `all_data`: backs up the files of all tablespaces, except for pure index tablespaces or empty tablespaces
- `full`: performs full database backup at level 0. See [Complete Backups](#). This backup can be part of an incremental backup strategy.

- `incr`: performs incremental backup with RMAN. See full database backup (level 0) in [Incremental Backup](#).
- `<tablespace>`: backs up the files of one tablespace
- `<file_ID>`: backs up a data file with the specified Oracle file ID as file ID. Control files can be addressed with the file ID 0. Online redo log files can be addressed using the file ID 0<n>, <n> is the redo log group number. To address all the online redo log files, use file ID 00. Temporary files are identified by negative numbers.
- `<file_ID1>-<file_ID2>`: backs up the files specified in the file ID interval. The specified file IDs must be known in the database.
- `<generic_path>`: backs up the required database file, non-database file, or directory when you enter a complete path. Enter a generic path to back up all the database data files whose name starts with that path. In this case, the path must contain at least the `SAPDATA_HOME` directory and an additional generic specification (for example, `sapdata<n>`) in the path.

#### Note

When you specify a directory to be backed up its contents and the names of the subdirectories are backed up. However the directory structure and the content of the subdirectories are not backed up.

- `<object list>`: backs up the listed objects. You can specify a list of tablespaces or files, or combine the key word `all` with an object list. The individual objects are separated by commas (not blanks).
- `sap_dir`: automatically determines and saves all the files of the SAP environment. This means that the following directory trees are saved: `/sapmnt/<SAPSID>`, `/usr/sap/<SAPSID>` and `/usr/sap/trans`. If possible, these directories should be backed separately. You can only use this option when saving to tape without verifying the backup.
- `ora_dir`: automatically determines and saves all the non-database files of the Oracle environment. This means that the directory trees are saved in `<ORACLE_HOME>` (except for the `sapdata<n>` and `origlog/mirrlog` directories). If possible, save these directories in a separate backup run. You can only use this option when saving to tape without verifying the backup.
- `all_dir`: performs a backup combining `sap_dir` and `ora_dir` (see above). It has the same effect as `-m sap_dir,ora_dir`.
- `partial`: selects all files of a partial disk backup without having to explicitly specify them. This is only used with the options `-b`, `-bd`, or `-db`.
- `non_db`: selects non-database files of a partial disk backup without having to explicitly select them. This is only used with the options `-b`, `-bd`, or `-db`.

#### Note

For UNIX systems: start BRBACKUP to save the SAP/Oracle environment using the following command as user `root` (otherwise you will not have the authorizations required for the directory to be saved):

```
brbackup -m sap_dir|ora_dir
```

Saving and restoring under `root` also has the advantage that you can be sure that the settings for the user and authorizations for the files and directories are kept after restoring.

Parameters in `init<DBSID>.sap`: [backup\\_mode](#).

If you want to repeatedly back up several tablespaces or files, it might be more effective to configure parameter `backup_mode` of the initialization profile accordingly.

## **-n|-number**

This BRBACKUP command option defines the number of tape volumes to be initialized.

You can only use this option together with one of the options `-i`, `-i force`, or `-i show`.

The default value is 10,000. The program processes all existing tapes – that is, in the `init<DBSID>.sap` parameter `volume_backup` or those defined with the option `-v`. To initialize only a specific number of tapes, change this value to meet your requirements.

Syntax

```
-n|-number <number of tapes>
```

End of the code.

Default value: 10,000

## **-o|-output**

This BRBACKUP command option prints additional information to the log file.

Syntax

```
-o|-output dist|time[,time|dist]
```

End of the code.

Default value: the BRBACKUP detail log is written in normal form. See [BRBACKUP Detail Log](#).

Possible values:

- `dist`: generates information about the distribution of the files for backup among the volumes (tapes or disks) to be used
- `time`: generates additional time stamps that enable you to determine the time required for the individual operations. Among other things, you can then determine the pure backup time for a file. After a successful backup, this information is included in list form for all backed-up files.

See [Log Supplements](#).

## **-p|-profile**

This BRBACKUP command option defines the profile name.

This profile is normally contained in directory `<ORACLE_HOME>/dbs` (UNIX) or `<ORACLE_HOME>\database` (Windows).

## Syntax

```
-p|-profile <profile>
```

End of the code.

Default value: `init<DBSID>.sap`

Possible value:

<profile>: specifies the name of the profile file. If this file is not in the standard directory, specify the complete path.

## -q|-query

This BRBACKUP command option sets query mode. You can use it to find out which volumes (for example, tapes) must be mounted for the backup process. In this case, backup is not started. Before you start a backup request with CRON, use this option to find out which volumes are required.

## Syntax

```
-q|-query [check|nolog|split]
```

End of the code.

Default value: the backup is started.

Possible values:

- `check`: checks whether the proper volumes have really been mounted in the backup devices. The backup is not started.

You can perform the preparation for an unattended backup to tapes as follows:

1. Enter `brbackup -q check` to query the required tapes.
2. Mount the required tapes in the tape devices.
3. Enter `cont` to start the check of the mounted tapes.

Once you have made these preparations, you can start an unattended backup on the same day, since you have already checked the validity of the tapes.

- `nolog`: does *not* create or update detail, summary, and database logs for the function

### Example

```
brbackup -u / -q nolog
```

- `split`: splits the disks. The database is stopped and the tablespaces are placed in backup status, with the result that the split is consistent.

## -r|-parfile

This BRBACKUP command option defines the BACKINT or mount parameter file.

If you want to perform a BRBACKUP backup using backup devices such as jukeboxes or autoloaders, you can define the name of a parameter file with this option or in the initialization profile, which contains the configuration parameters for the mount or dismount command.

When using non-SAP backup programs for the backup, you might have to store the additional information required in a parameter file when specifying the option `-d util_file|util_file_online`. You can enter the complete names of this file in the initialization profile or using the option `-r`. The contents of this file depend on the external backup program used. The SAP tools only pass on the information about the parameter file name that the external program can use to obtain the required information. To find out which parameters must appear in this file and the syntax of those parameters, contact the supplier of the external backup program.

#### Syntax

```
-r|-parfile <parameter_file>
```

End of the code.

Default value: no parameter file

Corresponding parameters in `init<DBSID>.sap`:

- [mount\\_par\\_file](#) for BRBACKUP backups
- [util\\_par\\_file](#) for external backup programs

## **-s|-save**set

This BRBACKUP command option defines the number of files in a save set.

#### Syntax

```
-s|-save
```

set 1|2|3|4|tsp|all

End of the code.

Default value: 1

Possible values:

- 1, 2, 3, 4: defines the number of files in a save set
- `tsp`: specifies that each save set contains all files of a tablespace
- `all`: specifies that only one save set with all database files is created

The SAP backup library helps to optimize the utilization of quick tape drives by combining multiple data files in save sets. Multiple file access (file multiplexing) maximizes the flow of data (streaming mode).

A save set can contain individual data files, all files of a tablespace, or the complete data backup. The size of the save sets for the backup must be selected according to the tape device. A fast data flow with a minimum save set size is the optimum.

We do *not* recommend large save sets, since in a restore the complete save set has to be imported, even if only one data file is required.

`saveset_members = all` is set as standard for an incremental backup with the SAP backup library so that only one "incremental save set" is created including all changed blocks.

For more information, see [RMAN Save-Set Grouping](#).

## **-t|-type**

This BRBACKUP command option defines the online or offline backup type.

## Syntax

```
-t|-type  
online|online_cons|offline|offline_force|offline_standby|online_st  
andby|offline_stop|online_split|offline_split|online_mirror|offlin  
e_mirror|offstby_split|offstby_mirror
```

End of the code.

Default value: `offline`

Possible values:

- `online`: performs the backup for the open database
- `online_cons`: keeps the database open during the backup. As well as the database files the offline redo log files generated during the backup are also copied to the same volume. This means that you have a logically consistent dataset available. This backup of the offline redo log files using BRBACKUP runs completely independently of other BRARCHIVE backups.
- `offline`: shuts down the database for the backup when the SAP system has also been shut down. Otherwise, the database is not shut down, and BRBACKUP terminates with an appropriate error message (BR0068E).
- `offline_force`: does not check whether an SAP system user is active. The database is shut down and an offline backup is performed.
- `offline_standby`: stops the standby database for the backup. This option is only relevant for the [standby database](#) configuration.
- `online_standby`: keeps the standby database mounted during the backup. This option is only relevant for the [standby database](#) configuration.
- `offline_stop`: performs the database backup in offline mode, followed by the migration of the saved database into `mount standby` status. This type of backup is only relevant when the production database is saved and then takes over the role of a standby database. The backup itself becomes a production system. See [Standby Database: BRBACKUP Backup of Database Files](#)
- `online_split`: performs the splitting and saving of the mirror disks in the split command scenario while the database is open. This option is only relevant for a [split mirror online backup](#).
- `offline_split`: stops the database for the splitting of the mirror disks in the split command scenario. The backup of the mirror disks is then done directly afterwards, while the database is open. This option is only relevant for a [split mirror offline backup](#).
- `online_mirror`: performs the splitting and saving of the mirror disks while the database is open in the SPLITINT scenario. BRBACKUP calls SPLITINT to perform the split. This option is only relevant for a [split mirror online backup](#).
- `offline_mirror`: stops the database for the splitting of the mirror disks in the SPLITINT scenario. The backup of the mirror disks is then done directly afterwards, while the database is open. BRBACKUP calls SPLITINT to perform the split. This option is only relevant for a [split mirror offline backup](#).
- `offstby_split`: stops the *standby* database for the splitting of the mirror disks in the split command scenario. For more information, see [Split Mirror Backup](#).

- `offstby_mirror`: stops the *standby* database for the splitting of the mirror disks in the SPLITINT scenario. The backup of the mirror disks is then done directly afterwards, while the standby database is mounted. BRBACKUP calls SPLITINT to perform the split. For more information, see [Split Mirror Backup](#).

For more information, see parameter [backup\\_type](#) in `init<DBSID>.sap`.

## **-u|-user**

This BRBACKUP command option defines the user name and password that BRBACKUP uses to log on to the database system.

Syntax

```
-u|-user [<user>[/<password>]]|/]
```

End of the code.

Default value: `system/<default password>`

If you only enter `-u`, BRBACKUP performs an interactive query of the user name and the password. You can enter the user name and the password separately (only enter the user name or the option `-u <user>`). BRBACKUP then prompts entry of the password. In this case, the password is not displayed during entry, and does not appear in the process list.

These measures are taken to protect the DBA password.

In shell scripts, you can structure the call as follows:

```
brbackup -c -u <<END <user>/<password> END
```

However, use this command only if the option `-c` is active and you are sure that the tape does not need to be changed.

Note

If you are working with an `OPSS` user, enter the following:

```
brbackup -u /
```

In this case, BRBACKUP tries to log on to the database as `OPSS` user. See the Oracle documentation and information in the SAP Service Marketplace at <http://service.sap.comnotes>.

The `OPSS` user must be defined in the database and have at least `SYSOPER` authorization and `SAPDBA` role. The user must also have `SYSDBA` authorization, if `RMAN` is to be used. With this method, it is not necessary to specify the password when calling BRBACKUP.

## **-v|-volume**

This BRBACKUP command option defines the tape volumes to be used.

Syntax

```
-v|-volume <volume>|<volume list>|SCRATCH
```

End of the code.

Default value: defined in `init<DBSID>.sap`. The length of the volume name is limited to 10 characters.



Possible values:

- `<volume>` or `<volume list>`: specifies the volume or volume list you want to use for backup. BRBACKUP checks whether the mounted volume (tape) has that name and whether the expiration period has expired. The individual objects in a volume list are separated by commas (no blanks).
- `SCRATCH`: specifies that you can mount any volume (for which the expiration period has expired) for the backup

If you want to use the standard volume(s) specified in parameter `volume_backup` of the initialization profile, do *not* use the parameter `SCRATCH`. In this case, the automatic volume management will cyclically select the volumes named in `volume_backup` and check whether their expiration period has expired. If a free volume (e.g. with expired expiration period) is found, you are prompted to mount it in the backup device (tape device). BRBACKUP checks the name of the mounted volume and makes sure the expiration period really has expired. The expiration period is configured with parameter `expir_period` in profile `init<DBSID>.sap`.

By using the reserved name `SCRATCH`, you can deactivate the automatic volume management. You can then mount any SAP volume. The program still makes sure that the expiration period has expired.

See [Volume Management](#).

Corresponding parameters in `init<DBSID>.sap`:

[volume\\_archive](#)

[volume\\_backup](#)

## **-w|-verify**

This BRBACKUP command option verifies the backup after the files have been backed up.

You can use this option to make sure that the backup is readable and complete. Once the backup phase is complete, all the saved files are restored from the volume (for example, tape) in sequence, decompressed (when `compress = yes` was used), read by the check program, and compared with the originals. See also [compress\\_dir](#).

During an offline backup, the file contents are compared in binary form. During an online backup, the sizes of the saved files are determined and checked.

See [Log Supplements](#).

Syntax

```
-w|-verify [use_dbv|only_dbv|only_conf|use_rmv|only_rmv]
```

End of the code.

Default values: no verification.

Possible values:

- `use_dbv`: performs a database backup followed by a restore to a temporary directory and a check of the Oracle block structure with the DBVERIFY tool
- `only_dbv`: calls DBVERIFY to check the internal block structure on the database files without a backup

- `only_conf`: calls the external backup utility only to confirm that the backup is known, not to verify the data
- `use_rmv`: restores and then verifies the successfully backed-up files using RMAN
- `only_rmv`: verifies the original database files using RMAN without starting a backup

Note that verification approximately *doubles* the required backup time.

For security reasons, we recommend using this option at least once within the volume expiration period for your complete backups – or even better, once a week. It enables you to detect and correct any possible hardware problems.

See also:

[Backup Verify](#)

## **-V|-VERSION**

This BRBACKUP command option displays detailed information on the program modules and patches.

Syntax

```
-V|-VERSION [ALL]
```

End of the code.

Possible value:

ALL: displays patch information for all BR\*Tools

## **BRBACKUP Logs**

For more information, see:

- [Names of the BRBACKUP Detail Logs](#)
- [BRBACKUP Detail Log](#)
- [Log Supplements](#)
- [BRBACKUP Summary Log](#)

## **Names of the BRBACKUP Detail Logs**

Every detail log contains a name with the following format:

```
b<encoded timestamp>.xyz
```

The first characters indicate the encoded time the backup was performed (action ID). The extension (function ID) indicates the type of backup. The logs are stored in the `sapbackup` directory.

Possible values for x:

- a: whole database backed up (`backup_mode = all|all_data`).

- **p**: one or more tablespaces or files backed up (that is, partial backup).
- **f**: full (level 0) database backup (`backup_mode = full`)
- **i**: incremental (level 1) database backup (`back_mode = incr`)

Possible values for *y*:

- **n**: backup performed online (`backup_type = online|online_cons|online_split|online_mirror|online_standby`).
- **f**: backup performed offline (`backup_type = offline|offline_force|offline_standby|offline_split|offline_mirror|offline_stop|offstby_split|offstby_mirror`).

Possible values for *z* (specification of backup devices):

- **t**: tape device (`backup_dev_type = tape|tape_auto|tape_box`).
- **d**: local disk (`backup_device_type = disk|disk_copy|disk_standby`).
- **p**: tape device on a remote system (`backup_device_type = pipe|pipe_auto|pipe_box`)
- **f**: external backup program used, backup performed file by file (`backup_device_type = util_file|util_file_online`).
- **v**: external backup program used, backup performed volume by volume (`backup_dev_typ = util_vol|util_vol_online`)
- **s**: remote disk (`backup_dev_type = stage|stage_copy|stage_standby`)
- **r**: backup with rman (`backup_dev_type = rman_util|rman_disk|rman_stage`)

Other function IDs:

- **tib**: BRBACKUP option `-i`, `-i force`, or `-i show` used to initialize a volume or display the information in the label.
- **qub**: BRBACKUP option `-q`, or `-q check|split` used to display which volumes are to be used for the backup, or to make sure that these volumes were actually mounted; no backup was started
- **cmb**: BRBACKUP option `-k only` used to only perform a software compression, but no backup was started. This can be used to determine the current compression rate of all files.
- **dbv**: BRBACKUP option `-w only_dbv` used to verify the internal database block structure with `DBVERIFY`, but no backup was started
- **rmp**: BRBACKUP option `-d rman_prep` used to prepare for backup with the Oracle Recovery Manager (RMAN), but no backup was started
- **ddb**: BRBACKUP option `-db` used to delete a disk backup, but no backup was started
- **bab**: BRBACKUP run was aborted using `brbackup -g|-abort`

# BRBACKUP Detail Log

This section describes the information contained in a detail log (a<encoded timestamp>.<ext>, see [Names of the BRBACKUP Detail Logs](#)).

The detail log file contains information about the status of the Oracle database at the time of the backup, and about the actions that were performed in the course of the backup.

- Displays the relevant parameters of initialization profile `init<DBSID>.sap` that were set during the BRBACKUP run.
- Information on Oracle archiving before starting BRBACKUP: Database mode (ARCHIVELOG, NOARCHIVELOG), status of the archiving process (enabled, disabled), archiving directory, oldest log sequence number of the online redo log files, next redo log files for archiving by Oracle, log sequence number of the current online redo log files, start SCN (system change number) of the current online redo log files
- Listing of data files of the tablespaces, the redo log files, and the control files:
  - Tablespaces and data files:  
Tablespace name, Oracle tablespace status (“\*” means: data tablespace), Oracle file status, file name, file size, Oracle file ID, disk volume ID, link directory at subdirectory level (usually for subdirectory <tablespace name>\_<file number>) or key word `NOLINK`, when no soft links were defined, file type `FILE` | `RAW`, maximum file size, file incremental size, and tablespace block size
  - Online redo log files:  
File name, file size, redo log group number, disk volume ID, Oracle file status, link directory or key word `NOLINK` when no soft links were defined, file type `FILE` | `RAW`
  - Control files:  
File name, file size, 0 (default for file ID), disk volume ID, link directory or key word `NOLINK` when no soft links were defined, file type `FILE` | `RAW`
- Listing of non-database files, if non-database files were saved:  
Disk volume ID, file size, file name
- Listing of directories and their contents, if directories were saved using this option:  
Directory name, disk volume ID, file category (`file` – file, `link` – soft link, `pipe` – named pipe, `dir` – directory, `spec` – special file), file size, file name

## Backup Flow

The log contains additional information when you start BRBACKUP with the option `-o dist|time` or `-w`. See [Log Supplements](#), [-o|-output](#) and [-w|-verify](#).

This information indicates which database file, non-database file, or directory, or which profile and log were saved and where.

- `#FILE`: the data file name as it appears in the control file. If `compress = only` is used, the compression rate is also displayed.

- #NDBF: name of a non-database file.
- #DIR: name of a directory.
- #ARCHIVE: name of the offline redo log files (only for `backup_type=online_cons`)
- #PFLOG: name of the relevant profile/log (only when archiving with BACKINT).
- #SAVED: this entry differs depending on the type of backup:
  - Backup on tape
    - #SAVED: file name, name of tape, position of the file on the tape, compression rate and the size of the compressed file (if compressed) or save set key (if RMAN backup)
  - Backup on disk
    - #SAVED: file name, symbolic volume name and file position (only important for BRRESTORE), compression rate and the size of the compressed file (only if compressed) or save set key and size (if RMAN set backup)
  - Backup using an external backup program
    - #SAVED: backup ID returned by the external backup program

This is followed by an archive log list that is created (and possibly updated) after the backup, as well as several closing messages.

## BRBACKUP Summary Log

You can display a brief entry for each backup in the summary log `back<DBSID>.log`. The entries in the file provide the following information about each backup using BRBACKUP:

- Action ID (encoded timestamp of the log name)
- Function ID (extension of the log name)
- Timestamp (date, time) specifying the start of the backup
- Timestamp (date, time) specifying the end of the backup
- Return code
- Total compression rate (appears only when compression was used)
- Total count of database files
- Number of saved database files
- Number of saved non-database files
- Begin redo log sequence number of the backup
- End redo log sequence number of the backup
- Begin Oracle system change number of the backup
- End Oracle system change number of the backup

- Type of backup:
  - `all[+]`: all database files
  - `all_data[+]`: all pure data tablespaces
  - `full[+]`: full database backup
  - `partial[+]`: partial backup
  - `incr[+]`: incremental database backup
  - `non_db`: only non-database files (for example, `sap_dir`, `ora_dir`)

“+” means that non-database files were also backed up.

- Value of the parameter [backup\\_type](#)
- Value of the parameter [backup\\_dev\\_type](#)
- Internal flags for the BRBACKUP command options
- BRBACKUP version

## BRARCHIVE

This SAP tool for the Oracle database enables you to archive offline redo log files. For more information about features common to both tools, see [Common Features of BRBACKUP and BRARCHIVE](#).

### Prerequisites

Make sure that the [initialization profile init<DBSID>.sap](#) is configured properly and use the appropriate [command options for BRARCHIVE](#).

### Features

- You can also start BRARCHIVE when the database is shut down.
- You should archive the offline redo log files on tape using BRARCHIVE.
- In contrast to [BRBACKUP](#), BRARCHIVE does not have its own management of tape continuation. When a tape is full, you must restart BRARCHIVE to write to the next volume. But this is not a real restriction due to the capacity of the tapes.
- For security reasons, we recommend using the option of archiving the offline redo log files to two backup devices in parallel (`brarchive -ss`, `brarchive -ssd`). You can also make this second copy serially (either by restarting BRARCHIVE with `brarchive -sc` or `brarchive -scd` or by using the option `-cs` or `-cds`). See [-sl-  
-sc|-ds|-dc|-sd|-scd|-ss|-ssd|-cs|-cds](#).
- Use the option [-f|-fill](#) to archive the offline redo log files permanently. In this way, you can make sure that the archiving directory does not fill up.
- To make sure that the offline redo log files are archived smoothly, the summary log must exist in readable form. Do not delete or change any entries in this log.

## More Information

- [Hardware Compression for BRARCHIVE](#)
- [Logging](#)

## Hardware Compression for BRARCHIVE

When you archive the offline redo log files with BRARCHIVE, performing software compression in advance is useless, since the previously determined compression rates cannot be used for archiving the next redo log files. If BRARCHIVE does not have any information on the compression rates, the default value of 1 is assumed. This means, for example, that when the tape size is 16,000 MB and the redo log files are 200 MB, up to 80 offline redo log files can be archived on a volume.

If, however, a larger number of offline redo log files were written on one day that actually fit on the tape, you can change the parameter `tape_size` accordingly.

We know from experience that during a hardware compression, the size of the offline redo logs shrinks by around a third. Therefore, you can increase the `init<DBSID>.sap` parameter `tape_size` by around 50% (to 24,000 MB in the above example) to reflect the actual compression rate. This enables you to archive many more (up to 120 in the example) offline redo log files with one BRARCHIVE call.

You can define a separate `init<DBSID>.sap` profile parameter, [tape\\_size\\_arch](#), for BRARCHIVE. If you do this, changes to individual parameters do *not* affect the next database backup with BRBACKUP.

See also:

[Hardware Compression](#)

## Command Options for BRARCHIVE

This section describes the command options for [BRARCHIVE](#). If you start BRBACKUP without command options, the values in the [initialization profile init<DBSID>.sap](#) are used. Unless otherwise specified in the initialization profile `init<DBSID>.sap`, BRARCHIVE starts archiving all the offline redo log files to tape, with a storage capacity of 1200 MB and no file compression.

For more information about features common to both tools, see [Common Features of BRBACKUP and BRARCHIVE](#).

If you use BRARCHIVE with command options, these override the corresponding values in the initialization profile. To use the options, you can specify either the letter indicated or the complete word.

This is the full command syntax:

Syntax

```
brarchive
[-a|-archive]
[-b|-backup [<brb_options>]]
[-c|-confirm [force]]
[-d|-device tape|disk|pipe|tape_auto|pipe_auto|tape_box|
```

```

pipe_box|util_file|stage|rman_util||util_disk|util_stage]
[-e|-execute <number>]
[-f|-fill [<number>|stop|suspend|resume]]
[-h|-help [version]]
[-i|-initialize [force|show]]
[-k|-compress no|yes|hardware|only]
[-l|-language E|D]
[-m|-modify [<delay>]]
[-n|-number <number of logs>|<number of tapes>]
[-o|-output dist|time[,time|dist]]
[-p|-profile <profile>]
[-q|-query [check|nolog]]
[-r|-parfile <parameter_file>]
[-s|-save|-sc|-second_copy|-ds|-delete_saved|-dc|-delete_copied|
-sd|save_delete|-scd|-second_copy_delete|-ss|-double_save|
-ssd|-double_save_delete|-cs|-copy_save|-cgs|-copy_delete_save]
[-u|-user [<user>[/<password>]]|/]
[-v|-volume <volume>|<volume list>|SCRATCH]
[-w|-verify [only_conf|use_rmv|first_rmv|only_rmv]]
[-V|-VERSION [ALL]]

```

End of the code.

See also:

[-a|-archive](#)

[-b|-backup](#)

[-c|-confirm](#)

[-d|-device](#)

[-e|-execute](#)

[-f|-fill](#)

[-h|-help](#)

[-i|-initialize](#)

[-k|-compress](#)

[-l|-language](#)

[-m|-modify](#)

[-n|-number](#)

[-o|-output](#)

[-p|-profile](#)



[-q|-query](#)

[-r|-parfile](#)

[-s|-sc|-ds|-dc|-sd|-scd|-ss|-ssd|-cs|-cds](#)

[-u|-user](#)

[-v|-volume](#)

[-w|-verify](#)

## **-a|-archive**

This BRARCHIVE command option backs up offline redo log files saved earlier on disk to tape, to remote disk, or to backup utility (BACKINT).

Syntax

```
-a|-archive
```

End of the code.

Default value: BRARCHIVE saves all those offline redo log files to tape that have not already been saved to tape. Both the copies saved directly to tape as well as those written to tape through a disk backup are taken into account. At the most two copies can be written to tape.

Example

A copy of the offline redo log files that were copied to disk in a previous backup is created in parallel on each of two different tapes:

```
brarchive -ss -a
```

See also:

[Two-Phase Backup](#)

## **-b|-backup**

This BRARCHIVE command option starts BRBACKUP at the end of BRARCHIVE processing.

When you enter the option -b, BRBACKUP is automatically started after the BRARCHIVE backup of the offline redo log files. After the offline redo log files, BRBACKUP copies the data files and all logs to the same tape or tapes (this is also possible to disk, and with BACKINT). The main advantage of this procedure is that you only have to start or schedule one program (usually BRBACKUP). The second program is started automatically after the first program has ended. If the backup is made to tape, you can also use the tape capacity optimally.

You can also specify other options for BRBACKUP backup. If you do not specify any options, the relevant default is used. Always place the option -b (including the additional options) at the end of the BRARCHIVE command line call.

See [Command Options for BRBACKUP](#).

Note

If you want to execute the BRBACKUP and BRARCHIVE backup procedure in one run, then we recommend that you perform the tape management under control of BRBACKUP. For this, you can use `brbackup -a|-archive`.

## Syntax

```
-b|-backup [<brb_options>]
```

End of the code.

Default value: BRBACKUP and BRARCHIVE are called separately, the backup of the data files and the offline redo log files are located in separate backups on separate tapes.

## Example

Unattended backup of the offline redo log files on a tape followed by a startup of an offline backup on the same tape.

```
brarchive -sd -c -b -m all -t offline -c
```

See also:

[BRBACKUP and BRARCHIVE Backups in One Run](#)

[Unattended Backup](#)

## **-c|-confirm**

This BRARCHIVE command option backs up in attended mode.

## Syntax

```
-c|-confirm [force]
```

End of the code.

Default value: confirmation messages are issued and user entry is expected.

See [-c|-confirm](#).

## **-d|-device**

This BRARCHIVE command option defines the backup device type.

## Syntax

```
-d|-device  
tape|disk|pipe|tape_auto|pipe_auto|tape_box|pipe_box|util_file|sta  
ge|rman_util|rman_disk|rman_stage
```

End of the code.

Default value: tape

Possible values:

- **tape:** local tape device  
Always archive offline redo log files finally to tape, if possible twice.
- **disk:** local disk. normally only use the option for archiving to disk in exceptional cases, such as two-step archiving. See [archive\\_copy\\_dir](#).
- **stage:** remote disk

- `pipe`: archiving on a tape device of a remote system (UNIX only)
- `tape_auto` or `pipe_auto`: the BRARCHIVE program generally only uses one tape for archiving (exception: archiving with `-ss` or `-ssd`). No continuation tapes are created. Therefore, specifying `tape_auto` or `pipe_auto` has no effect.
- `tape_box` or `pipe_box`: jukeboxes or autoloader tape devices that can be addressed locally or remotely. See [Backup with Automatic Tape Changers](#).
- `util_file`: archiving with BACKINT – see the corresponding option in [-dl-device](#).
- `rman_util|rman_disk|rman_stage`: archiving with Oracle Recover Manager (RMAN) and an external backup tool. See [RMAN Backup with an External Backup Library](#).

See also:

Parameters in `init<DBSID>.sap`: [backup\\_dev\\_type](#)

## **-e|-execute**

This BRARCHIVE command option is used to parallelize the verification of the backups of offline redo log files with BACKINT.

Syntax

```
-e|-execute <number of parallel processes>
```

End of the code.

Default value: 1

Possible value:

`<number of parallel processes>`: specifies the number of parallel processes running for the verification

## **-f|-fill**

This BRARCHIVE command option fills up a backup volume by waiting for the next offline redo log files.

When you select the `-f` option, BRARCHIVE waits for the next offline redo log files copied by Oracle to the archiving directory, and archives them on the volume as soon as they are created. This process continues until the volume is full or the specified number of offline redo log files for processing has been reached.

Syntax

```
-f|-fill [<number>|stop|suspend]|resume]
```

End of the code.

Default value: do not wait for the next offline redo log files to write them to the volume (tape). BRARCHIVE must be started afresh for the backup of the next offline redo log files.

Possible values:

- `<number>`: BRARCHIVE does not archive each offline redo log file individually, but waits until a certain `<number>` of files have accumulated in the archiving directory.

These offline redo log files are then saved to tape as a group. See [Grouping Offline Redo Log Files](#).

- `stop`: stops a BRARCHIVE archiving run that was started with `-f`. This operation might take some time (a few seconds or a few minutes). Inspect the log written by BRARCHIVE. See [Names of the BRARCHIVE Detail Logs](#).
- `suspend`: suspends BRARCHIVE processing that was started with `-f`. This can be useful if, for example, you want to perform an offline backup and later restart BRARCHIVE.
- `resume`: resumes processing suspended with `-f suspend`

## **-g|-abort**

This BRARCHIVE command option aborts archive runs. To abort BRARCHIVE, you open a separate command window to enter the command shown below.

Syntax

```
brarchive -g|-abort stop|term|kill
```

End of the code.

Default value: none

Possible values:

- `stop`: causes a soft abort
- `term`: causes a normal abort
- `kill`: causes a hard abort

The abort generates a short log file in the relevant log file directory:

- **UNIX**: `<saparch>/<coded timestamp>.aab`
- **Windows**: `<saparch>\<coded timestamp>.aab`

The aborted archives are marked as faulty. However, you might be able to later use them for certain purposes (such as a fill-up), especially if the abort was soft.

## **-h|-help**

This BRARCHIVE command option provides help information.

Syntax

```
-h|-help [version]
```

End of the code.

Default value: no help

See [-h|-help](#).

## **-i|-initialize**

This BRARCHIVE command option initializes tape volumes.

Use this option to initialize SAP volumes (tapes) or non-SAP volumes. Only tapes of this type can be used for archiving with BRARCHIVE. See [Volume Management](#).

Syntax

```
-i|-initialize [force|show]
```

End of the code.

Default value: no initialization.

If you specify the option `-i` to initialize volumes (and `-v` has not been specified), BRARCHIVE initializes the volumes specified in `volume_archive`.

See [-i|-initialize](#).

## **-k|-compress**

This BRARCHIVE command option sets compression mode.

Syntax

```
-k|-compress no|yes|hardware|only
```

End of the code.

Default value: `no` (no compression)

See [-k|-compress](#).

The option `-k only` has practically no meaning for BRARCHIVE as the information is not saved in the database. See [Hardware Compression for BBRACKUP](#). It can only be used to check the compression rate of offline redo log files.

## **-l|-language**

This BRARCHIVE command option sets the message language.

Syntax

```
-l|-language E|D
```

End of the code.

Default value: `E`

Possible values:

- `D`: German
- `E`: English

## **-m|-modify**

This BRARCHIVE command option applies the offline redo log files to a standby database.

Syntax

```
-m|-modify [<delay>]
```

End of the code.

Default value: no delay

`delay`: sends the created offline redo log files to the standby database before they are processed. There they can be applied with a delay time of `<delay>` minutes after creating the Oracle offline redo log file.

If there is a standby database, you must call `brarchive -m` to apply the offline redo log files.

For more information, see [Standby Database](#).

## **-n|-number**

This BRARCHIVE command option defines the number of offline redo log files or tape volumes to be processed.

The program processes practically all the existing offline redo log files or volumes for volume initialization – that is, the files in the `init<DBSID>.sap` parameter `volume_archive` or those defined with the option `-v`.

Syntax

```
-n|-number <number of logs>|<number of volumes>
```

End of the code.

Default value: 10,000

Possible values:

- `<number of logs>`: specifies the number of offline redo log files to archive and/or delete.
- `<number of volumes>`: specifies the number of volumes (tapes) to initialize, in combination with the option `-i`, `-i force` or `-i show`.

## **-o|-output**

This BRARCHIVE command option prints additional information to the log file.

Syntax

```
-o|-output dist|time[,time|dist]
```

End of the code.

Default value: the BRARCHIVE detail log is written in normal form. See [BRARCHIVE Detail Log](#).

See [-o|-output](#).

## **-p|-profile**

This BRARCHIVE command option defines the profile name.

Syntax

```
-p|-profile <profile>
```

End of the code.

Default value: `init<DBSID>.sap`

See [-p|-profile](#).

## **-q|-query**

This BRARCHIVE command option sets the query mode.

Syntax

```
-q|-query [check|nolog]
```

End of the code.

Default value: archiving is started.

Possible values:

- `check`: checks whether the proper volumes have really been mounted in the archive devices. The archive is not started.

You can perform the preparation for an unattended archive to tapes as follows:

1. Enter `brarchive -q check` to query the required tapes.
2. Mount the required tapes in the tape devices.
3. Enter `cont` to start the check of the mounted tapes.

Once you have made these preparations, you can start an unattended archive on the same day, since you have already checked the validity of the tapes.

- `nolog`: does *not* create or update detail, summary, and database logs for the function

Example

```
brarchive -u / -q nolog
```

See [-q|-query](#).

## **-r|-parfile**

This BRARCHIVE command option defines the BACKINT or mount parameter file.

Syntax

```
-r|-parfile <parameter_file>
```

End of the code.

Default value: no parameter file

See [-r|-parfile](#).

## **-s|-sc|-ds|-dc|-sd|-scd|-ss|-ssd|-cs|-cds**

This BRARCHIVE command option defines the BRARCHIVE function to be performed.

BRARCHIVE only performs the selected operation for the number of offline redo log files that you selected with the option `-n`. The program only deletes offline redo log files if they have already been successfully archived or copied.

If only one tape device exists, you can use the option `-cs` or `-cds` to ensure that BRARCHIVE creates a second copy of the offline redo log files in a run, deletes them if necessary and immediately continues with archiving. You can also achieve this by calling

BRARCHIVE first with the option `-s` and then with the option `-sc` or `-scd`. However, in this case two BRARCHIVE calls are required.

Parameters in `init<DBSID>.sap`: [archive\\_function](#).

### Syntax

```
-s|-save|-sc|-second_copy|-ds|-delete_saved|-dc|-delete_copied|  
-sd|save_delete|-scd|-second_copy_delete|-ss|-double_save|  
-ssd|-double_save_delete|-cs|-copy_save|-cds|-copy_delete_save
```

End of the code.

Default value: `-s`

Possible values:

- `-s|-save`: archives the offline redo log files
- `-sc|-second_copy`: creates a second copy of the offline redo log files that have already been archived
- `-ds|-delete_saved`: deletes offline redo log files that have been archived once
- `-dc|-delete_copied`: deletes offline redo log files that have been copied twice
- `-sd|-save_delete`: archives offline redo log files and then deletes these files
- `-scd|-second_copy_delete`: creates a second copy of the offline redo log files that have already been archived and then deletes these files



- `-ss|-double_save`: archives the offline redo logs to two backup devices (tape devices) in parallel
- `-ssd|-double_save_delete`: archives the offline redo logs to two backup devices (tape devices) in parallel and then deletes the files
- `-cs|-copy_save`: creates a second copy of offline redo log files that have already been archived and then archives the newly created offline redo log files
- `-cds|-copy_delete_save`: creates a second copy of offline redo log files that were already archived, then deletes them and starts archiving of the newly created offline redo log files

## **-u|-user**

This BRARCHIVE command option defines user name and password.

Syntax

```
-u|-user [<user>[/<password>]]/]
```

End of the code.

Default value: `system/<default_password>`

See [-u|-user](#).

## **-v|-volume**

This BRARCHIVE command option defines the tape volumes to be used.

Syntax

```
-v|-volume <volume>|<volume list>|SCRATCH
```

End of the code.

Default value: defined in `init<DBSID>.sap`

If the option `-v` is not used, BRARCHIVE uses the volumes (tapes) specified in parameter `volume_archive`. For more information, see [-vj-volume](#).

## **-w|-verify**

This BRARCHIVE command option verifies the backup after the offline redo log files were saved.

You can use this option to make sure that the backup is readable and complete. Once the archiving phase is complete, all the saved files are read from the volume (for example, tape) in sequence, decompressed (when `compress = yes` was used), read by the check program, and compared with their originals. When archiving to disk, verification is performed immediately after archiving the individual offline redo log files.

If the option `-s`, `-sc`, `-ss` or `-cs` is used, the file contents are compared in binary form. If the option `-sd`, `-scd`, `-ssd` or `-cds` is used, the sizes of the archived files are determined and checked.

See [Log Supplements](#).

Syntax

```
-w|-verify [only_conf|use_rmv|first_rmv|only_rmv]
```

End of the code.

Default value: no checks

Possible values:

- `only_conf`: BRARCHIVE calls the external backup utility only to confirm that the backup is known, not to verify the data.
- `use_rmv`: restores and then verifies the successfully backed-up files using RMAN
- `first_rmv`: verifies the original offline redo log files using RMAN before the backup starts
- `only_rmv`: verifies the original offline redo log files using RMAN without starting a backup

Note that verification approximately doubles the archiving time required.

For security reasons, we recommend using this option at least once within the volume expiration period for your archiving with `-s`, `-sc`, `-ss` or `-cs`. It enables you to detect any possible hardware problems and undertake the appropriate measures.

## **-V|-VERSION**

This BRARCHIVE command option displays detailed information on the program modules and patches.

Syntax

```
-V|-VERSION [ALL]
```

End of the code.

ALL: patch information is displayed for all BR\*Tools

# BRARCHIVE Logs

For more information, see:

- [Names of the BRARCHIVE Detail Logs](#)
- [BRARCHIVE Detail Log](#)
- [BRARCHIVE Summary Log](#)

## Names of the BRARCHIVE Detail Logs

Every detail log contains a name with the following format:

a<encoded timestamp>.<ext>

The first characters indicate the encoded time the archiving was performed (action ID). The extension (function ID) indicates the type of archiving. The logs are stored in the `saparch` directory.

Possible function IDs:

- `sve`: offline redo log files archived for the first time (option `-s`)
- `cpy`: offline redo log files archived a second time (option `-sc`)
- `svd`: offline redo log files archived and then deleted from the archiving directory (option `-sd`)
- `cpd`: offline redo log files archived a second time and then deleted from the archiving directory (option `-scd`)
- `dsv`: deleted archived redo log files (option `-ds`)
- `dcp`: offline redo log files that had been archived twice were deleted (option `-dc`)
- `ssv`: offline redo log files archived twice. Archiving was performed on two tape devices in parallel (option `-ss`).
- `ssd`: offline redo log files archived twice and then deleted. Archiving was performed on two tape devices in parallel (option `-ssd`).
- `cps`: archived offline redo log files were archived for a second time. Then continues with the archiving of the files newly included in the archiving directory (option `-cs`).
- `cds`: archived offline redo log files were archived for a second time and deleted. Then continues with the archiving of the files newly included in the archiving directory (option `-c ds`).
- `tia`: BRARCHIVE option `-i`, `-i force` or `-i show` was used to initialize a volume or display the information in the label
- `qua`: BRARCHIVE option `-q` or `-q check` was used to display which volumes are to be used for archiving or make sure that those volumes were actually mounted. No archiving was started.

- `cma`: BRARCHIVE option `-k only` was used to perform a software compression, but no archiving was started. This can be used to determine the current compression rate of all offline redo log files.
- `vra`: offline redo log file verification with RMAN
- `fst`: BRARCHIVE option `-f stop|suspend|resume` was used to control a BRARCHIVE run
- `aab`: BRARCHIVE run was aborted using `brarchive -g|-abort`

## BRARCHIVE Detail Log

This section describes the information contained in a detail log (`a<encoded timestamp>.<ext>`, see [Names of the BRARCHIVE Detail Logs](#)).

The detail log file contains information about the actions that were performed in the course of the archiving.

- Displays the relevant parameters of initialization profile `init<DBSID>.sap` that were set during the BRARCHIVE run
- The archiving flow

### Archiving Flow

The log will contain additional information when you start BRARCHIVE with the option `-o dist|time` or `-w|-verify`. See [Log Supplements for BRBACKUP and BRARCHIVE, -o|-output](#) and [-w|-verify](#).

This information indicates which file was saved where.

- `#ARCHIVE`: name of the relevant offline redo log files
- `#PFLOG`: name of the relevant profile/log (only when archiving with BACKINT)
- `#SAVED`: this entry varies depending on the type of archiving:
  - Archiving on Tape  
File name, name of tape, position of the file on the tape, compression rate and the size of the compressed file (if compressed) and save set key and size (if RMAN set backup)
  - Archiving on disk  
File name, symbolic volume name and file position (only important for BRRESTORE), compression rate and the size of the compressed file (only if compressed)
  - Archiving using an external backup program  
Backup ID returned by the external backup program

This is followed by several closing messages (for example, the total number of processed offline redo log files).

## Log Supplements

Using the option `-o dist|time [,time|dist]` or the option `-w` (or both) causes the detail log to be supplemented.

When you select `-o dist` and no compression rate is available yet, the compression rate 1:1 is selected for the offline redo log files (column `rate` in the display). Note that BRARCHIVE will only carry out archiving in parallel when started with one of the options `-ss` or `-ssd`.

See [Log Supplements for BRBACKUP and BRARCHIVE](#).

## BRARCHIVE Summary Log

You can display a brief entry for every archived offline redo log file in the summary log `arch<DBSID>.log`. The entries in the file provide the following information about each archiving run with BRARCHIVE:

- `#ARCHIVE`: log sequence number, name of the offline redo log files, creation time, file size, start SCN (system change number) of the offline redo log file, thread number
- `#SAVED`: this entry depends on the type of archiving for the offline redo log files:

- Archiving on tape

Action ID (encoded timestamp of the log name)

Function ID (extension of the log name)

Name of tape/position of the file on tape, starting with the following:

- `#` for `tape_copy_cmd = cpio`
- `$` for `tape_copy_cmd = dd`
- `+` for `tape_copy_cmd = rman`
- `-` for `tape_copy_cmd = rman_dd`
- `*` for `backup_dev_type = util_file`
- `&` for `backup_dev_type = rman_util`
- `@` for backup from disk backup with BACKINT

Timestamp (date, time) specifying the end of the archiving process

Compression rate and the size of the compressed file (if it was compressed) or save set key (if RMAN set backup)

- Archiving using an external backup program

Action ID (encoded timestamp of the log name)

Function ID (extension of the log name)

Backup ID returned by the external backup program (if the `util_file` option was used, an asterisk `**` or `@` appears before the backup ID)

Timestamp (date, time) specifying the end of the backup

- **#COPIED:** information as for **#SAVED** when the offline redo log files were archived a second time
- **#DELETED:** information (action ID, function ID, timestamp) about when the file was deleted
- **#\*:** information on the status of a BRARCHIVE run: **ORACLE\_SID**, device type (values like in **backup\_dev\_type** or **null** when offline redo log files were only deleted), action ID, function ID, timestamps (date, time) specifying the start and the end of the BRARCHIVE run, return code, the total compression rate (if compressed), first and last sequence number of saved offline redo log files, internal flag for the BRARCHIVE command options, and BRARCHIVE version
- **#DISK/#STAGE:** this information appears if you used the BRARCHIVE feature for archiving to local or remote disk:

Log sequence number, name of the offline redo log file, creation time, file size, start SCN (system change number) of the offline redo log file, thread number

- **#DISKSAV/#STAGESAV:** archiving to disk: action ID, function ID, name of the file on the local or remote disk, timestamp, compression rate and the size of the compressed file (if it was compressed), or save set key and size (if RMAN set backup)
- **#DISKDEL/#STAGEDDEL:** archiving to disk: information (action ID, function ID, timestamp), when the file was deleted from archive directory
- **#DELDISK:** a copy of an offline redo log file on disk was deleted: information (action ID, function ID, timestamp) when the file was deleted from the disk backup directory
- **#APPLIED:** offline redo log file was applied to the standby database
- **##:** you can enter user comments into the BRARCHIVE summary log with this prefix.

## BRRESTORE

This SAP tool enables you to restore an entire database backup or parts of it, when the backup was performed with [BRBACKUP](#). Any non-database files and directories you saved can also be restored. In the process, the subdirectories in `sapdata<n>` directories are automatically created, when necessary.

You can also restore the offline redo log files that were backed up with [BRARCHIVE](#). This operation can be performed at the same time as the restore of the corresponding backup.

- BRRESTORE can run unattended when option `-c force` is set. The option `-c` only suppresses the first confirmation prompts for mounting a volume.
- BRRESTORE uses the BRBACKUP logs and the summary log from BRARCHIVE to decide where to restore the requested file. You can manually specify a different directory as well.
- One or more incomplete BRRESTORE runs can be completed with the option `-f`. BRRESTORE automatically determines the files to be restored.

## Integration

BRRESTORE only restores the selected backup. It does *not* recover the database. To do this, start the recovery afterwards using one of the following:

- [BRRECOVER](#)
- The SQLPLUS tool from Oracle - see [Recovery with SQLPLUS](#)

For more information, see:

- [Restoring Files](#)
- [Examples of BRRESTORE Runs](#)

## Restoring Files

BRRESTORE can be called directly from the operating system command level. A list of the command options can be found in the [Command Options for BRRESTORE](#).

BRRESTORE also requires several parameters to be configured in the [initialization profile `init<DBSID>.sap`](#).

- The options `-d`, `-k`, `-m`, and `-r` can be preset using the appropriate BRRESTORE profile parameters. See [Effects of the Command Options](#).
- Only one of the options `-a`, `-b`, `-b2` or `-n`, `-n2` can be set. If you do not select any of these options, `-b last` is selected.

However, BRRESTORE can be started with option `-a` in parallel to BRRESTORE with option `-b` or `-n`.

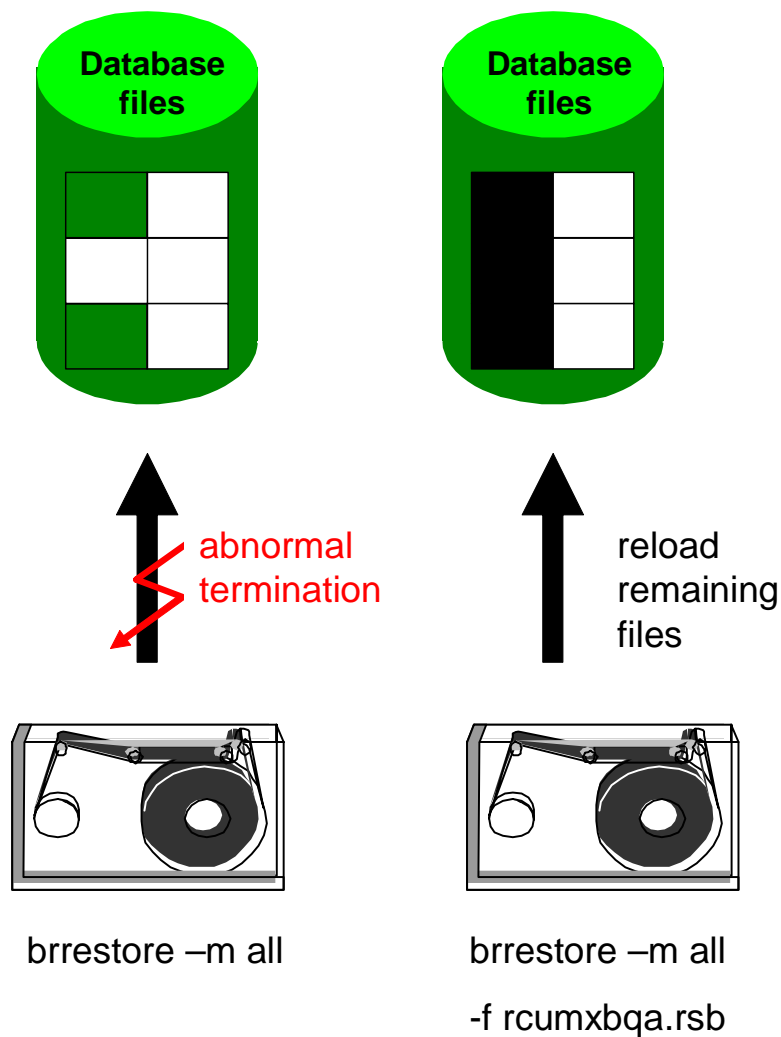
BRRESTORE can restore a database in parallel. BRRESTORE can also restore archived redo log files in parallel if they are located on several volumes. In both cases, several tape devices must be available, and the number of parallel copy processes must correspond to the number of tape devices. This is the standard setting with [exec\\_parallel](#).

- To restore a complete backup (reset the database), use the option `-m full`.
- The key word `all` can be combined with other options in an `<object list>` (for example, non-database files or directories). However, database files and non-database files are restored separately from each other, if they were backed up in separate backup runs, as recommended.
- Non-database files and directories must be defined with their complete path. Single files from the backed up directory can also be restored.
- Database data files can be defined with either a full or generic path. A generic path must contain the directory `SAPDATA_HOME` and a generic specification (for example, `sapdata<n>` directory).
- All the file IDs used in a specified interval `<file_ID1>-<file_ID2>` must be known in the database.
- Online redo log files can only be addressed directly by using the redo log group numbers with an additional leading zero (`0<n>`). To address all the online redo log files, specify file ID `00`.

- The control file can only be addressed directly by using file ID 0.
- When you start the restore of online redo log files or the control file, the mirror copies of these files are automatically recreated.
- Archived offline redo log files can be addressed using their log sequence numbers.

## Completion of BRRESTORE Runs

If a BRRESTORE restore is incomplete, for example, due to a termination of the BRRESTORE program you can complete the remainder of the files in another BRRESTORE run with the option `-f|-fillup`.



If you want to complete a terminated BRRESTORE run, start BRRESTORE with the option `-f|-fillup` and specify the log name of the BRRESTORE run to be completed. If you want to complete several runs, specify all log names with the option `-f`.

You can also use option `-f last` to complete the last BRRESTORE run, or option `-f <days>` to complete all BRRESTORE runs started in the last `<day>` days. For more information, see [-f|-fillup](#).



See also:

[Completion of BRBACKUP Backups](#)

## Examples of BRRESTORE Runs

- `brrestore -b last -m all`  
Restores all tablespaces without the control file and online redo log files from the last successful backup
- `brrestore -b bcnmhluz.aft -m full`  
Restores all the files from backup `bcnmhluz.aft`, including the control file and the online redo log files  
Restores the mirror copies of the control file and the online redo log files
- `brrestore -m /usr/sap/C11/SYS/profile`  
Restores the SAP profiles
- `brrestore -m /oracle/C11/sapdata1=/oracle/C11/sapdata5`  
Restores all the database data files that were originally stored in the subdirectories of `/oracle/C11/sapdata1` in directory `/oracle/C11/sapdata5`
- `brrestore -b last -m 1-10,01-04,0`  
Restores all the database data files with Oracle file IDs from 1 through 10, the four online redo log files, and the control file starting from the last successful backup  
Restores the mirror copies of the control file and the online redo log files
- `brrestore -m 0`  
Restores the control file. Restores the mirror copies of the control file
- `brrestore -b last -m /oracle/C11/sapdata2/ddicd_5/ddicd.data5`  
Restores a database data file starting from the last successful backup
- `brrestore -a 200-220`  
Restores the archived redo log files with the log sequence numbers from 200 through 220 into the archiving directory
- `brrestore -a 40-70=/oracle/C11/sapbackup 71-90=/oracle/C11/sapreorg`  
Restores the archived redo log files with the log sequence numbers from 40 through 70 in directory `sapbackup` and those with the log sequence numbers from 71 through 90 in directory `/oracle/C11/sapreorg`
- `brrestore -a 40-69,70-100=/oracle/C11/sapbackup`  
Restores the archived redo log files with the log sequence numbers from 40 through 69 in the archiving directory, and those with the log sequence numbers from 70 through 100 in directory `sapbackup`

- `brrestore -n det_log`

Restores a detail log to the local working directory

## Command Options for BRRESTORE

This section describes the command options for [BRRESTORE](#). If you start BRRESTORE without command options, the values in the [Initialization Profile `init<DBSID>.sap`](#) are used. Unless otherwise specified in the initialization profile `init<DBSID>.sap`, BRRESTORE restores the files of all tablespaces from the last successful backup of the database.

If you use BRRESTORE with command options (see below), these override the corresponding values in the initialization profile. To use the options, you can specify either the letter indicated or the complete word.

### Syntax

```
brrestore
[-a|-archive|-a1|-archive1
[<DBSID>,<log_no>[=<rest_dest>]]|
[<DBSID>,<log_no1>-<log_no2>[=<rest_dest>]]|
[<DBSID>,<log_no_list>[==<rest_dest>]]
[-a2|-archive2 [<DBSID>,<log_no>[=<rest_dest>]]|
[<DBSID>,<log_no1>-<log_no2>[=<rest_dest>]]|
[<DBSID>,<log_no_list>[==<rest_dest>]]
[-b|-backup|-b1|-backup1 <log name>|last]
[-b2|-backup2 <util_backup_id>|#NULL
[-c|-confirm [force]]
[-d|-device tape_disk|pipe|tape_auto|pipe_auto|tape_box|
pipe_box|util_file|util_vol|stage|rman_util|rman_disk|
rman_stage|rman]
[-e|-execute <number>]
[-f|-fillup <log_name1>[,<log_name2>,...]|
<yyyy-mm-dd hh.mi.ss>|<no. of days>|last]
[-h|-help [version]]
[-i|-interval <days>]
[-k|-compress no|yes|hardware]
[-l|-language E|D]
[-m|-mode all|all_data|full|incr|incr_all|incr_only|
incr_full|<tablespace>[=<rest_dest>]]|
<file_ID>[=<rest_dest>]|<file_ID1>-<file_ID2>[=<rest_dest>]|
|<generic_path>[=<rest_dest>]|<object list>|
archive_logs|partial|non_db[==<rest_dest>]]
[-n|-number <file_pos>|init_ora|spfile|init_sap|space_log|
det_log|sum_log|init_all|all_log|control_file[=<rest_dest>]]
[-n2|-number2<back_file>=<rest_dest>]
```

```
[-o|-output dist|time[,time|dist]]
[-p|-profile <profile>]
[-q|-query [check|nolog]]
[-r|-parfile <parameter_file>]
[-u|-user [<user>[/<password>]]|/]
[-w|-verify [use_dbv|only_conf|use_rmv]]
[-V|-VERSION [ALL]]
```

End of the code.

*See also:*

[-a|-archive|-a1|-archive1](#)

[-a2|-archive2](#)

[-b|-backup|-b1|-backup1](#)

[-b2|-backup2](#)

[-c|-confirm](#)

[-d|-device](#)

[-e|-execute](#)

[-h|-help](#)

[-i|-interval](#)

[-k|-compress](#)

[-l|-language](#)

[-m|-mode](#)

[-n|-number](#)

[-n2|-number2](#)

[-o|-output](#)

[-p|-profile](#)

[-q|-query](#)

[-r|-parfile](#)

[-u|-user](#)

[-w|-verify](#)

[-V|-VERSION](#)

## **-a|-archive|-a1|-archive1**

This BRRESTORE command option restores offline redo log files from the first copy.

If you use this option, BRRESTORE checks the BRARCHIVE summary log to see which volume contains the required archived redo log files (in this case, the first copy of the offline redo log files). Mount the requested volume in the backup device and enter `cont` to confirm that you want to start the restore.

You can also use this option when you restore archived redo log files from a disk.

#### Syntax

```
-a|-archive|-a1|-archive1 [<DBSID>,<log_no>[=<rest_dest>]] |
[<DBSID>,<log_no1>-<log_no2>[=<rest_dest>]] |
[<DBSID>,<log_no_list>[<rest_dest>]]
```

End of the code.

Default value: no restore of archived redo log files.

Possible values:

- `<DBSID>`: database instance ID, only required for Oracle Parallel Server (OPS)
- `<log_no>`: log sequence number to specify the requested first copy of the offline redo log files
- `<log_no1>-<log_no2>`: log sequence number interval to specify the requested first copies of the offline redo log files
- `<rest_dest>`: the restore directory where the archived redo log files are to be restored. If you do not specify a directory, the archiving directory (`<SAPDATA_HOME>/oraarch`) is selected. If you specify `==` this redirects all restored offline redo log files.
- `<log_no_list>`: you can combine any specifications for the log sequence intervals. Separate the individual names with commas. Do *not* use blanks.

Under certain circumstances, you can restore archived redo log files from several volumes simultaneously, in parallel. See [Restoring Files](#).

## **-a2|-archive2**

This BRRESTORE command option restores offline redo log files from the second copy.

If you use this option, BRRESTORE checks the BRARCHIVE summary log to see which volume contains the required archived redo log files (in this case, the second copy of the offline redo log files). Mount the requested volume in the backup device and enter `cont` to confirm that you want to start the restore.

You *cannot* use this option to restore archived redo log files from a disk.

#### Syntax

```
-a2|-archive2 [<DBSID>,<log_no>[=<rest_dest>]] |
[<DBSID>,<log_no1>-<log_no2>[=<rest_dest>]] |
[<DBSID>,<log_no_list> [==<rest_dest>]]
```

End of the code.

Default value: no restore of archived redo log files.

- `<DBSID>`: database instance ID, only required for Oracle Parallel Server (OPS)
- `<log_no>`: log sequence number to specify the requested second copy of the offline redo log files
- `<log_no1>--<log_no2>`: log sequence number interval to specify the requested second copies of the offline redo log files
- `<rest_dir>`, `<log_no_list>`: see [-a|-archive|a1|archive1](#).

## **-b|-backup|b1|backup1**

This BRRESTORE command option restores database files saved by BRBACKUP.

Syntax

```
-b|-backup|-b1|-backup1 <log name>|last
```

End of the code.

Default value: last successful backup of the database (`last`)

- `<log name>`: name of the detail log file `b<encoded timestamp>.<ext>` from a BRBACKUP backup. The requested objects are then restored from that database backup.
- `last`: the last successful database backup is used to restore the requested objects

## **-b2|-backup2**

This BRRESTORE command option restores individual files, calling backup tools via the BACKINT interface. With this option you can reload backups that were executed with an external backup tool via the interface BACKINT. Use option `-m` to define the files to be restored, using the full path.

Syntax

```
-b2|-backup2 <util_backup_id>|#NULL
```

End of the code.

Possible values:

- `<util_backup_id>`: backup ID of backup with an external tool
- `#NULL`: restore from the last BACKINT backup

See also:

[External Backup Programs](#)

## **-c|-confirm**

This BRRESTORE command option restores in unattended mode.

If you specify the option `-c`, confirmation messages that are output when the volume (for example, tape) is mounted are suppressed. In this case, BRRESTORE assumes that the correct volume has been mounted in the backup device (for example, tape device). All other BRRESTORE confirmation messages must be responded to.

## Syntax

```
-c|-confirm [force]
```

End of the code.

Default value: confirmation messages issued and user entry expected

Possible value:

*force*: all confirmation messages are suppressed. You can use this option when you regularly make database copies to have a current test system available, or when performing similar actions. See [Structure-Retaining Database Copy](#).

## Caution

Do *not* use `-c force` when recovering a database. Follow the BRRESTORE confirmation messages in this case.

## -d|-device

This BRRESTORE command option defines the restore device type. Depending on which backup you want to restore from, you can use this option to specify the backup media that was used.

## Syntax

```
-d|-device tape|disk|pipe|tape_auto|pipe_auto|tape_box  
tape_box|pipe_box|util_file|util_vol|stage|rman_util|  
rman_disk|rman_stage|rman
```

End of the code.

Default value: `tape`

Possible values:

- `disk`: local disk
- `tape`: local tape device
- `pipe`: tape device of a remote system (UNIX only)
- `stage`: remote disk
- `tape_auto` or `pipe_auto`: prompts for changing the tape will be suppressed. This is only useful when you use a tape device with automatic tape changing.
- `tape_box` or `pipe_box`: defines jukeboxes or autoloader tape devices that can be addressed locally or remotely
- `util_file`: use this option when you performed the backup file by file, using external backup programs. If a parameter file is required, specify its name in profile parameter `util_par_file` or with the option `-r`.
- `util_vol`: as for `util_file` but for volume by volume backups
- `rman_util|rman_disk|rman_stage|rman`: restores with Oracle Recover Manager (RMAN) and an external backup tool. See [RMAN Backup with an External](#)

[Backup Library](#). You can use `rman` as a replacement for the parameters `rman_util`, `rman_disk`, or `rman_stage` when restoring data files.

See also:

Parameters in `init<DBSID>.sap`: [backup\\_dev\\_type](#).

## **-e|-execute**

This BRRESTORE command option executes the restore in parallel.

When restoring, the maximum number of copy processes used corresponds to the number used for the backup. This means that you can only reduce the number of parallel copy processes by setting this option.

Syntax

```
-e|-execute <number of copy processes>
```

End of the code.

Default value: 0

See [-e|-execute](#).

## **-f|-fillup**

This BRRESTORE command option completes the restore run using the specified files.

Syntax

```
-f|-fillup <log_name>[,<log_name2>, ...] |<yyy-mm-dd hh-mi-ss> |<no.  
of days lost> |last
```

End of the code.

Default value: last

Possible values:

- `<log_name>[,<log_name2>, ...]`: one or more named BRRESTORE logs
- `<yyy-mm-dd hh-mi-ss>`: all restores started since the specified date and time
- `<no. of days lost>`: all restores in a defined number of preceding days
- `last`: the last restore run

See also:

[Completion of BRRESTORE Runs](#)

## **-g|-abort**

This BRRESTORE command option aborts restore runs. To abort BRRESTORE, you open a separate command window to enter the command shown below.

Syntax

```
brrestore -g|-abort stop|term|kill
```

End of the code.

Default value: none

Possible values:

- `stop`: causes a soft abort
- `term`: causes a normal abort
- `kill`: causes a hard abort

The abort generates a short log file in the relevant log file directory:

- **UNIX:** `<saparch>/<coded timestamp>.rab`
- **Windows:** `<saparch>\<coded timestamp>.rab`

The aborted restores are marked as faulty. However, you might be able to later use them for certain purposes (such as a fill-up), especially if the abort was soft.

## **-h|-help**

This BRRESTORE command option provides help information.

Syntax

```
-h|-help [version]
```

End of the code.

Default value: no help

See [-h|-help](#).

## **-i|-interval**

This BRRESTORE command option lets you override [recov\\_interval](#) from the initialization profile `init<DBSID>.sap`. BRRESTORE can use this command option to search for the last successful database backup or backup of the offline redo log files. If this value is too small, BRRESTORE might not find the required backup.

Syntax

```
-i|-interval <days>
```

End of the code.

Default value: 30 days

Example

To restore a full backup done in the last 100 days, enter the following command

```
brrestore -m full -i 100
```

## **-k|-compress**

This BRRESTORE command option sets compression mode. If the value for decompression does not agree with the compression type from the backup being used when you start BRRESTORE, an appropriate warning is issued. However, BRRESTORE always decompresses the files that were saved with software compression.



## Syntax

```
-k | -compress no|yes|hardware
```

End of the code.

Default value: no (no compression)

Possible values:

- `yes`: if you performed the backup with software compression and `compress = yes` is not set in the initialization profile, you should use this option.
- `hardware`: BRRESTORE does not distinguish internally between `no` and `hardware`.

Equivalent parameter in `init<DBSID>.sap`: [compress](#)

## -l|-language

This BRRESTORE command option sets the message language.

### Syntax

```
-l|-language E|D
```

End of the code.

Default value: E

Possible values:

- `D`: German
- `E`: English

## -m|-mode

This BRRESTORE command option defines the files to be restored.

### Syntax

```
-m|-modeall|all_data|full|incr|incr_only|incr_full|  
incr_all| <tablespace>[=<rest_dest>]|<file_ID>[=<rest_dest>]  
|<file_ID1>-<file_ID2>[=<rest_dest>]|<generic_path>[=<rest_dest>]|  
<object list>|archive_logs|partial|non_db[==<rest_dest>]
```

End of the code.

Default value: all

Possible values:

- `all`: the files in all tablespaces, but not the control files and online redo log files
- `all_data`: the files in all tablespaces that are not pure index tablespaces
- `full`: a complete backup, including any non-database files, directories, control files, online redo log files (complete offline backup) and offline redo log files (consistent online backup). The corresponding mirror copies of the control files and online redo log files are recreated.

- `incr`: an incremental backup with Oracle Recovery Manager (RMAN). See [RMAN Restore of Incremental Backups](#).
- `incr_only`: all files that were in the database at the time of the last full backup. See *Restoring Incremental Backups with Structural Changes* in [RMAN-Relevant Profile Parameters](#).
- `incr_full`: files that have been added to the database since the last full backup. See *Restoring Incremental Backups with Structural Changes* in [RMAN-Relevant Profile Parameters](#).
- `incr_all`: like `incr` plus all redo log, control, non-database files and directories, if any
- `<tablespace>`: the files of this tablespace
- `<file_ID>`: data file with the specified Oracle file ID. Control files have the file ID 0. Online redo log files have the file ID 0<n>, <n> is the redo log group number. To address all the online redo log files, use file ID 00. Temporary files are identified by negative numbers.
- `<file_ID1>-<file_ID2>`: the files specified by the file ID interval
- `<generic_path>`: the complete path to restore the required database file, non-database file, or directory. Specify a generic path to restore all the database data files whose name starts with that path. In this case, the path must contain at least the `SAPDATA_HOME` directory and an additional generic specification (for example, `sapdata<n>`) in the path. If the generic path is a directory, you can also restore one or more file(s) from this directory by additionally specifying one or more file names:  
  
`<directory_name>:<file_name1>[:<file_name2>...]`
- `<rest_dest>`: the restore directory to which the requested files is to be restored. If no directory is specified, the original directory from the backup log is selected. See also [new\\_db\\_home](#). If you specify `==` this redirects all restored files.
- `<object list>`: a list of tablespaces or files, or combine the key words `all` with an object list. When possible, always restore database files and non-database files in separate runs. The individual objects are separated by commas. Do not use blanks to separate the objects.
- `archive_logs`: the offline redo log files of a consistent BRBACKUP online backup (`backup_type=online_cons`).
- `partial`: all files from a partial backup without explicitly specifying them
- `non_db`: all non-database files and directories from a backup without explicitly specifying them

Parameters in `init<DBSID>.sap`: [backup mode](#).

## **-n|-number**

This BRRESTORE command option enables you to restore a file directly from a tape volume without having to specify the name of the backup log or the log sequence number.

Syntax

```
-n|-number <file_pos>|init_ora|spfile|init_sap|
space_log| det_log|sum_log|init_all|all_log|
control_file[=<rest_dest>]
```

End of the code.

Default value: none.

Possible values:

- `<file_pos>`: specifies the position of the file on the tape
- `init_ora|spfile`: restores the profiles `init<DBSID>.ora` and `spfile` are restored. These files are in position 2.
- `init_sap`: restores the initialization profile `init<DBSID>.sap`. This file is in position 3.
- `space_log`: restores the main log `space<DBSID>.log`, the structure log `struc<DBSID>.log`, and the parameter change log `param<DBSID>.log`. These files are in the third-to-last position.
- `det_log`: restores the detail BRBACKUP or BRARCHIVE log. This file is in the second-to-last position.
- `sum_log`: restores the summary BRBACKUP/BRARCHIVE log. This file is in the last position.
- `init_all`: restores `init<DBSID>.ora`, `spfile`, and `init<DBSID>.sap` in one run
- `all_log`: restores `reorg<DBSID>.log`, `struc<DBSID>.log`, `param<DBSID>.log`, detail and summary BRARCHIVE or BRBACKUP logs in one run.
- `<rest_dest>`: specifies the restore directory into which the requested files are restored. If no directory is specified, the files are restored to the current directory where BRRESTORE was started.
- `control_file`: restores the control file saved after a backup of offline redo log files with RMAN

## **-n2|-number2**

This BRRESTORE command option lets you restore a disk backup file to a specific restore destination.

Syntax

```
-n2|-number2 <back_file>=<rest_dest>
```

End of the code.

Default: none

## **-o|-output**

This BRRESTORE command option prints additional information to the log file.

Syntax

```
-o|-output dist|time[,time|dist]
```

End of the code.

Default value: the BRRESTORE detail log is written normally. See [BRRESTORE Detail Log](#).

See [-o|-output](#).

## **-p|-profile**

This BRRESTORE command option defines the profile name.

Syntax

```
-p|-profile <profile>
```

End of the code.

Default value: `init<DBSID>.sap`

See [-p|-profile](#).

## **-q|-query**

This BRRESTORE command option sets the query mode. You can find out which volumes (tapes) must be mounted for the restore process and which additional resources the program needs. In this case, restore is not started.

Syntax

```
-q|-query [check|nolog]
```

End of the code.

Default value: the restore process is started.

Possible values:

- `check`: checks whether the proper volumes have really been mounted in the backup devices. The restore is not started.
- `nolog`: does *not* create or update detail and summary logs for the function

Example

```
brrestore -u / -q nolog
```

## **-r|-parfile**

This BRRESTORE command option defines the BACKINT or mount parameter file.

Syntax

```
-r|-parfile <parameter_file>
```

End of the code.

Default value: no parameter file

See [-r|-parfile](#).

## -u|-user

This BRRESTORE command option defines user name and password for a restore with RMAN. It defines the user name and password used by BRRESTORE to log on to the database system.

### Syntax

```
-u|-user [<user>[/<password>]]|/]
```

End of the code.

Default value: system/<default\_password>

Possible values:

- If you only enter `-u`, an interactive query of the user name and the password is performed by the SAP utility. You can enter the user name and the password separately (only enter the user name or the option `-u <user>`). BRRESTORE then prompts entry of the password. In this case, the password is not displayed during entry and does not appear in the process list.
- If you enter `-u /` the Oracle OPSS mechanism is used.

### Note

These measures are taken to protect the DBA password.

## -w|-verify

This BRRESTORE command option verifies a backup of database files (BRBACKUP) or offline redo log files (BRARCHIVE).

### Syntax

```
-w|-verify [use_dbv|only_conf|use_rmv]
```

End of the code.

Default value: no verification

Possible values:

- `use_dbv`: restores files to [compress\\_dir](#), verifies them with DBVERIFY and then deletes them

Without `use_dbv`, files are read from backup media but *not* stored on disk. In either case, a normal restore is *not* performed.

- `only_conf`: BRBACKUP calls the external backup utility only to confirm that the backup is known, not to verify the data.
- `use_rmv`: restores and then verifies the successfully backed-up files using RMAN

### Example

This is an example of how to restore the last backup, followed by a check of the Oracle block structure using DBVERIFY:

```
brrestore -b last -w use_dbv
```

This is an example of how to restore the archived offline redo log files nos. 112- 250 and check them for readability:

```
brrestore -a 112-250 -w
```

See also:

[Backup Verify](#)

## **-V|-VERSION**

This BRRESTORE command option displays detailed information on the program modules and patches.

Syntax

```
-V|VERSION [ALL]
```

End of the code.

Possible value:

ALL: displays patch information for all BR\*Tools

## **BRRESTORE Logs**

For more information, see:

- [Names of the BRRESTORE Detail Logs](#)
- [BRRESTORE Detail Log](#)
- [BRRESTORE Summary Log](#)

## **Names of the BRRESTORE Detail Logs**

Every detail log contains a name with the following format:

```
r<encoded timestamp>.<ext>
```

The first characters indicate the encoded time the restore was performed (action ID). The extension (function ID) indicates the type of the restore. The logs are stored in the `sapbackup` directory.

Possible function IDs:

- `rsb`: restore from a BRBACKUP backup with option `-b` | `-backup` | `-b2`
- `rsa`: restore the offline redo log files specified by the log sequence numbers with the option `-a` | `-archive` | `-a1` | `-a2`
- `rsf`: restore a file characterized by its position on the backup volume with the option `-n` | `-number` | `-n2`
- `qur`: the BRRESTORE option `-q` or `-q check` was used to display which volumes are to be used for restore or to make sure that those volumes were actually mounted. No restore was started.

- `rab`: BRRESTORE run was aborted using `brrestore -g|-abort`

## BRRESTORE Detail Log

The detail log file contains the following information about the actions that were performed during the restore process

- The relevant parameters of initialization profile `init<DBSID>.sap` that were set during the BRRESTORE run
- Restore flow so that you can precisely monitor which backup was used to restore the files, which volumes were mounted, and so on
- `#FILE`: full path and name of the restored file
- `#NDBF`: full path and name of the restored non-database file
- `#DIR`: full name of the restored directory
- `#ARCHIVE`: full path and name of the restored archived redo log file
- `#RESTORED`: varies depending on which backup medium was used to restore the file:
  - Restore from tape
    - `#RESTORED`: name of the file on tape, volume name or file position as it was saved on the tape
  - Restore from disk
    - `#RESTORED`: complete name of the file on disk, as it was saved, and the symbolic volume name and file position
  - Restore using an external backup program
    - `#RESTORED`: backup ID returned by the external backup program when the file was backed up

### Log Supplements

Using the option `-o dist|time` to start BRRESTORE causes the detail log to be supplemented. The information about the distribution of files on the volumes (if you use `-o dist`) refers to the time when the files were saved. See [Log Supplements](#) and [-o|-output](#). The details of the restore times (`-o time`) refer to the restore process.

## BRRESTORE Summary Log

You can display a brief entry for each restore in the summary log `rest<DBSID>.log`. The entries in the file provide the following information about each restore using BRRESTORE:

- Action ID (encoded timestamp of the log name)
- Function ID (extension of the log name)
- Timestamp (date, time) specifying the start of the restore

- Timestamp (date, time) specifying the end of the restore
- Return code
- Total count of database files
- Number of restored database files
- Number of restored non-database files
- Value of [restore\\_mode](#)
- Value of [backup\\_dev\\_type](#)
- Internal flags for the BRRESTORE command options
- BRRESTORE version

## BRRECOVER

The SAP tool BRRECOVER for Oracle databases is used as a database administration tool to help you recover your database.

### Integration

You can use BRRECOVER from:

- The [command line](#)
- [BRTOOLS](#) with character-based menus or a GUI

### Prerequisites

- Make sure that the [initialization profile init<DBSID>.sap](#) is configured properly.
- Familiarize yourself with the [BRRECOVER command options](#).

### Features

You can use BRRECOVER to perform the following:

- Complete database recovery
- Database point-in-time (PIT) recovery
- Tablespace point-in-time (PIT) recovery
- Whole database reset
- Restore of individual backup files
- Restore and application of offline redo log files
- Disaster recovery

For more information on the approach to restore and recovery, see [Restore and Recovery](#).

BRRECOVER writes the following logs:



- [BRRECOVER detail log](#)
- [BRRECOVER summary log](#)

## Command Options for BRRECOVER

This section describes the command options for the BRRECOVER tool.

If you use BRRECOVER with command options, these override the corresponding values in the [initialization profile init<DBSID>.sap](#). To use the options, you can specify either letter indicated or the complete word.

The syntax of a BRRECOVER command is:

### Example

```
brrecover -t complete -p initGC2.sap
```

### Syntax

```
brrecover
[-a|-tsp|-tablespace <tsp_name>|<tsp_name_list>]
[-b|-backup [<log_name>|last]]
[-c|-confirm [force]]
[-d|-device tape|tape_auto|tape_box|pipe|pipe_auto|pipe_box|disk|
stage|util|util_file|util_vol|rman_util|rman_disk|rman_stage|rman]
[-e|-degree <number>]
[-g|-scn|-change <scn>]
[-h|-help [version]]
[-i|-interval <days>]
[-j|-ins|-instance <inst_name>]
[-l|-language E|D]
[-m|-pit|-time <yyy-mm-dd hh-mi-ss>]
[-n|-seq|-sequence <seq_nr>]]
[-n1|-seq1|-sequence1 <seq_nr>]]
[-p|-profile <profile>]
[-r|-parfile <parfile>]
[-s|-scroll <lines>]
[-t|-type complete|dbpit|tspit|reset|restore|apply|disaster]
[-u|-user [<user>[/<password>]]|/]
[-w|-own|-owner <own_name>|<own_name_list>]
[-V|-VERSION [ALL]]
```

End of the code.

See also:

[-a|-tsp|-tablespace](#)

[-b|-backup](#)  
[-c|-confirm](#)  
[-d|-device](#)  
[-e|-degree](#)  
[-g|-scn|-change](#)  
[-h|-help](#)  
[-i|-interval](#)  
[-j|-inst|-instance](#)  
[-l|-language](#)  
[-m|-pit|-time](#)  
[-n|-seq|-sequence](#)  
[-n1|-seq1|-sequence1](#)  
[-p|-profile](#)  
[-r|-parfile](#)  
[-s|-scroll](#)  
[-t|-type](#)  
[-u|-user](#)  
[-w|-own|-owner](#)  
[-V|-VERSION](#)

## **-a|-tsp|-tablespace**

This BRRECOVER command specifies the tablespaces to be recovered in a tablespace point-in-time (PIT) recovery.

Syntax

```
-a|-tsp|-tablespace <tsp_name>|<tsp_name_list>
```

End of the code.

Default value: none

Example

```
-tsp psapstabd,psapstabi
```

BRRECOVER recovers only the tablespace `psapstabd` and `psapstabi` for a tablespace PIT recovery.

## -b|-backup

This BRRECOVER command option specifies a BRBACKUP run from which to restore the database files.

Syntax

```
-b|-backup [<log_name>|last]
```

End of the code.

Default value: last

Possible values:

- `<log_name>`: restores database files from the BRBACKUP backup with the log name entered in `<log_name>` as `b<encoded_timestamp>.<function_id>`
- `last`: restores the files from the last successful database backup

## -c|-confirm

This BRRECOVER command option specifies whether the recovery is attended or unattended. In unattended mode, BRRECOVER only stops at menus and yes/no queries. At other prompts, it continues processing with the default value.

Syntax

```
-c|-confirm [force]
```

End of the code.

Default value: attended mode. You need to respond to the prompts and menus generated by BRRECOVER. You also have to check the default choices and input values suggested by BRRECOVER.

Possible value:

`force`: when you specify the option `-c force`, all confirmation messages are suppressed. In addition, BRRECOVER automatically selects default choices and accepts default input values in menus. You can use this option when you regularly make database copies for an up-to-date test system, or for similar actions where you are certain of the outcome.

Caution

Do not use the option `-c force` when recovering a production database. In this case, follow the BRRECOVER prompts and menus, and carefully check the default choices and input values suggested by BRRECOVER.

## -d|-device

This BRRECOVER command option defines the restore device type.

Syntax

```
-d|-device  
tape|tape_auto|tape_box|pipe|pipe_auto|pipe_box|disk|stage|  
util_file|util|util_vol|rman_util|rman_disk|rman_stage|rman
```

End of the code.

Default: tape

BRRECOVER supports the following backup media:

- tape: local tape device.
- pipe: tape device of a remote system
- tape\_auto or pipe\_auto:
  - suppresses prompts for changing the tape. This is only useful when you use
  - a tape device with automatic tape changing (tape changing device).
- tape\_box or pipe\_box:
  - jukeboxes or autoloader tape devices that can be addressed locally or remotely.
  - The drivers for the data transfer (cpio, dd)
  - are defined in the parameters tape\_address or tape\_address\_arch,
  - the drivers for rewinding are defined in the parameters tape\_address\_rew or tape\_address\_rew\_arch and the drivers for mounting and dismounting the tapes are defined in the
  - parameters tape\_address\_ctl or tape\_address\_ctl\_arch.
- disk: local disk. You have to use disk for
- disaster recovery if backups were done with rman\_disk.
- stage: restore from remote disk.
- You have to use stage for disaster recovery
- if backups were done with rman\_stage.

- `util_file|util:` for a recover created
- by external backup programs for file-by-file backup. If you use this option,
- you might have to create a file containing the parameters required for that
- type of recover. If a parameter file of this type is required, you must specify
- the name of the file in the profile parameter `util_par_file` or
- with the option `-r`. You have to use `util` for
- disaster recovery if backups were done with `rman_util`.
- `util_vol:` as for `util_file` but
- for volume by volume backups
- `rman_util|rman_disk|rman_stage|rman:`
- for a backup with the Oracle Recover Manager (RMAN). You can specify `rman` instead
- of `rman_util`, `rman_disk`,
- or `rman_stage` for restore of database files
- and offline redo log files.

See also:

[backup dev type](#)

## **-e|-degree**

This BRRECOVER command option instructs SQLPLUS to apply offline redo log files in parallel mode.

Syntax

```
-e|-degree <number>
```

End of the code.

Default value: Oracle default

Possible value:

<number>: specifies the number of Oracle recovery processes or threads running in parallel

This command corresponds to the parameter [recov\\_degree](#) in `init<DBSID>.sap`.

## **-g|-scn|-change**

This BRRECOVER command specifies the last Oracle system change number (SCN) for a point-in-time (PIT) recovery.

Syntax

```
-g|-scn|-change <scn>
```

End of the code.

Default value: none

Example

```
brrecover -scn 10401368920
```

BRRECOVER recovers the database to system change number 10401368920.

## **-h|-help**

This BRRECOVER command option provides help information and command line options about the version of BRRECOVER specified.

Syntax

```
-h|-help [version]
```

End of the code.

Default value: no help

## **-i|-interval**

This BRRECOVER command option specifies the interval in which BRRECOVER searches for backups.

Syntax

```
-i|-interval <days>
```

End of the code.

Default: 30

Example

```
brrecover -i 60
```

BRRECOVER searches for backups in the last 60 days.

This command corresponds to the parameter [recov\\_interval](#) in `init<DBSID>.sap`.

## **-j|-ins|-instance**

This BRRECOVER command option specifies the instance name of the Oracle Real Application Cluster (RAC), which the options [-sequence](#) and [-sequence1](#) refer to.

Syntax

```
-j|-ins|instance <inst_name>
```

End of the code.

Default: none

## **-l|-language**

This BRRECOVER command option sets the message language.

Syntax

```
-l|-language E|D
```

End of the code.

Default value: E

Note

The default becomes invalid if you specify another value by setting the environment variable `BR_LANG` (language variable).

If you set option `-l`, the value specified with this option applies.

Possible values:

- D: German
- E: English

## **-m|-pit|-time**

This BRRECOVER option specifies the point in time to which BRRECOVER recovers the database or tablespaces for a point-in-time (PIT) recovery.

Syntax

```
-m|-pit|-time <yyyy-mo-dd hh.mi.ss>
```

End of the code.

Default value: none

Example

```
brrecover -pit 2007-11-09 16.27.04
```

This command recovers the database to the state it was in on 9th November 2007 at 16.27.04.

## **-n|-seq|-sequence**

This BRRECOVER command specifies the sequence number of the last redo log file for a point-in-time (PIT) recovery.

Syntax

```
-n|-seq|-sequence <seq_nr>
```

End of the code.

Default value: none

Example

```
brrecover -seq 19094
```

BRRECOVER recovers the database to the redo log file with the sequence number 19094.

## **-n|-seq1|-sequence1**

This BRRECOVER command specifies the sequence number of the first redo log file for applying offline redo log files using the command option [-t apply](#).

Syntax

```
-n|-seq1|-sequence1 <seq_nr>
```

End of the code.

Default value: none

Example

```
brrecover -seq1 18487
```

BRRECOVER recovers the database from the redo log file with the sequence number 18487.

## **-o|-rpt|-point**

This BRRECOVER command specifies the restore point for the database reset with flashback database.

Syntax

```
-o|-rpt|-point <restore_point>|last
```

End of the code.

Default value: last restore point

Possible values:

- `<restore_point>`: specifies name of restore point
- `last`: uses last restore point



## **-p|-profile**

This BRRECOVER command option defines the profile name.

Syntax

```
-p|-profile <profile>
```

End of the code.

Default value: `init<DBSID>.sap`

## **-r|-parfile**

This BRRECOVER command option defines the BACKINT or mount parameter file.

Syntax

```
-r|-parfile <parameter_file>
```

End of the code.

Default value: no parameter file

## **-s|-scroll**

This BRRECOVER command option specifies the number of lines for scrolling in list menus. This option is not valid for BRGUI.

Syntax

```
-s|-scroll <lines>
```

End of the code.

Default value: 20

This command option corresponds to the parameter [scroll\\_lines](#) in `init<DBSID>.sap`.

## **-t|-type**

This BRRECOVER command specifies the type of recovery.

Syntax

```
-t|-type complete|dbpit|tspit|reset|restore|apply|disaster
```

End of the code.

Default value: `complete`

Possible values:

- `complete`: complete database recovery
- `dbpit`: database point-in-time recovery
- `tspit`: tablespace point-in-time recovery
- `reset`: whole database reset
- `restore`: restore of individual backup files

- `apply`: restore and apply offline redo log files (that is, archive logs)
- `disaster`: disaster recovery

This command corresponds to the parameter [recov\\_type](#) in `init<DBSID>.sap`.

## **-u|-user**

This BRRECOVER command option defines the user name and password with which BRRECOVER connects to the database. The user must have SYSDBA privileges.

Syntax

```
-u|-user [<user>[/<password>]]|/]
```

End of the code.

Default value: `system/<default_password>`

If you enter `-u /` the Oracle OPS\$ mechanism is used.

## **-V|-VERSION**

This BRRECOVER command option displays patch information for BRRECOVER.

Syntax

```
-V|-VERSION [ALL]
```

End of the code.

ALL: displays patch information for all BR\*Tools

## **-w|-own|-owner**

This BRRECOVER command specifies the SAP owner for a tablespace point-in-time (PIT) recovery.

Syntax

```
-w|-own|-owner <own_name>|<own_name_list>
```

End of the code.

Default value: none

## **BRRECOVER Logs**

For more information, see:

- [BRRECOVER Detail Log](#)
- [BRRECOVER Summary Log](#)

## **BRRECOVER Detail Log**

The detail log file contains full information about what happened during the recovery.

The file displays information about the:

- Relevant parameters from the [initialization profile init<DBSID>.sap](#) that were set during the BRRECOVER run
- Recovery type
- Menus that were displayed during the recovery and the options that you chose
- BRRECOVER commands used to perform each phase of the restore and recovery, and the results
- Remounting of the database
- Status of the tablespaces, data files, control files, and redo log files
- Names of the database files to be restored or recovered

## Structure

BRRECOVER detail logs have names of the following form:

v<encoded timestamp>.<ext>

The logs are stored in the `sapbackup` directory.

The name consists of:

- Action ID

This consists of the fixed character `v` and the `<encoded time>` that the recovery was performed.

- Function ID

The suffix `<ext>` indicates the restore type:

- `crv`: complete database recovery
- `dpt`: database point-in-time recovery
- `tpt`: tablespace point-in-time recovery
- `drs`: whole database reset
- `rif`: restore of individual backup files
- `alf`: restore and application of offline redo log files
- `drv`: disaster recovery

## Example

This is an example of the start of a BRRECOVER detail log for complete database recovery:

```
BR0701I BRRECOVER 6.40 (0)
```

```
BR0705I Start of database recovery: vdjwhllh.crv 2003-01-29 19.12.25
```

BR0101I Parameters

Name Value

oracle\_sid GC2

oracle\_home /oracle/GC2

oracle\_profile /oracle/GC2/dbs/initGC2.ora

sapdata\_home /oracle/GC2

sap\_profile /oracle/GC2/dbs/initGC2.sap

recov\_type complete

recov\_copy\_dir /oracle/GC2/sapbackup

recov\_interval 100

scroll\_lines 20

backup\_dev\_type tape

system\_info oragc2/oragc2 uw1030 SunOS 5.8 Generic\_108528-15 sun4u

make\_info sun OCI\_901 Jan 29 2003

command\_line brrecover -t complete -c force

BR0280I Time stamp 2003-01-29 19.12.25

BR0707I Recovery of database: GC2

BR0708I BRRECOVER action ID: vdjwhllh

BR0709I BRRECOVER function ID: crv

BR0710I Recovery type: complete

BR0134I Unattended mode with 'force' active - no operator confirmation allowed

BR0280I Time stamp 2003-01-29 19.12.25

BR0655I Control menu 101 # please decide how to proceed

-----

Complete database recovery main menu

1 = Check the status of database files

2 \* Select database backup

3 \* Restore data files

4 \* Restore and apply incremental backup

5 \* Restore and apply archivelog files

6 \* Open database and post-processing

7 - Exit program

8 - Reset program status

Standard keys: c - cont, b - back, s - stop, r - refr, h - help

```
-----  
BR0280I Time stamp 2003-01-29 19.12.25  
BR0134I Unattended mode with 'force' active - continuing processing  
with default reply '1'  
BR0614I Database instance GC2 is mounted  
BR0750I Database instance GC2 will be remounted now  
BR0280I Time stamp 2003-01-29 19.12.25  
BR0307I Shutting down database instance GC2 ...  
BR0280I Time stamp 2003-01-29 19.12.31  
BR0308I Shutdown of database instance GC2 successful  
BR0280I Time stamp 2003-01-29 19.12.31  
BR0330I Starting and mounting database instance GC2 ...  
BR0280I Time stamp 2003-01-29 19.12.41  
BR0331I Start and mount of database instance GC2 successful
```

## BRRECOVER Summary Log

Each recovery run has an entry in the summary log `recov<DBSID>.log`.

### Structure

The entries in the file provide the following information about each recovery using BRRECOVER:

- Action ID (encoded timestamp of the [detail log](#) name)
- Function ID (extension of the detail log name)
- Timestamp (date, time) specifying the start of the restore
- Timestamp (date, time) specifying the end of the restore
- Return code
- Value of [recov\\_type](#)
- BRRECOVER version

## BRSPACE

The SAP tool BRSPACE for Oracle databases enables you to manage the space in your database:

- Instance administration:
  - Start up database
  - Shut down database
  - Alter database instance
  - Alter database parameter
  - Recreate database
- Tablespace administration:
  - Extend tablespace
  - Create tablespace
  - Drop tablespace
  - Alter tablespace
  - Alter data file
  - Move data file
- Segment management:
  - Reorganize tables
  - Rebuild indexes
  - Export tables
  - Import tables
  - Alter tables
  - Alter indexes

## Integration

BRSPACE is one of the [BR\\*Tools for Oracle database administration](#). Therefore, it is fully integrated in your SAP system running with Oracle.

## Features

BRSPACE has many parameters, which you can specify in the [Initialization Profile `init<DBSID>.sap`](#).

## Activities

You can use BRSPACE from the command line or with BRGUI or BRTOOLS. For more information, see:

- [Command Options for BRSPACE](#)
- BRGUI or BRTOOLS:
  - [Database Instance Management with BR\\*Tools](#)

- [Space Management with BR\\*Tools](#)
- [Segment Management with BR\\*Tools](#)

## Command Options for BRSPACE

This section describes the command options for the BRSPACE tool.

If you use BRSPACE with command options, these override the corresponding values in the [initialization profile init<DBSID>.sap](#). To use the options, you can specify either the letter indicated or the complete word.

This is the schematic command syntax:

### Syntax

```
brspace [<main_options>] -f|-function <function>
      [<function_options>]
```

End of the code.

This is the full command syntax:

### Syntax

```
brspace
[-c|-confirm [force]]
[-h|-help [<function>| [version]]]
[-l|-language E|D]
[-o|-output [detail|process|summary| [, ]time]]
[-p|-profile <profile>]
[-q|-query [nolog]]
[-s|-scroll <lines>]
[-u|-user [<user>[/<password>]]|/]
[-V|-VERSION [ALL]]
-f|-function <function> [<function_options>]
```

End of the code.

### Example

```
brspace -f dbparam -p circuits -v 24
```

See also:

[-c|-confirm](#)

[-h|-help](#)

[-l|-language](#)

[-o|-output](#)

[-p|-profile](#)

[-q|-query](#)

[-s|-scroll](#)

[-u|-user](#)

[-V|-VERSION](#)

[-f|-function](#)

## **-c|-confirm**

This BRSPACE command option specifies whether processing is attended or unattended. In unattended mode, BRSPACE only stops at menus and yes/no queries. At other prompts, it continues processing with the default value.

Syntax

```
-c|-confirm [force]
```

End of the code.

Default value: attended mode. You need to respond to the prompts and menus generated by BRSPACE. You also have to check the default choices and input values suggested by BRSPACE.

Possible value:

*force*: suppresses all confirmation messages and automatically selects default choices and accepts default input values in menus

Caution

Do *not* use the option *-c force* unless you are sure of the outcome because it might produce unexpected results.

## **-f|-function**

This BRSPACE command option specifies the function to be performed. You should always enter a function.

Syntax

```
-f|-function  
dbstart|dbshut|dbalter|dbparam|dbcreate|dbshow|  
tsextend|tscreate|tsdrop|tsalter|dfalter|dfmove|tbreorg|  
idrebuild|tbexport|tbimport|tbalter|idalter|mstats
```

End of the code.

Default value: current status of database instance is displayed

### **Function Options**

- dbstart: [starts the database](#)
- dbshut: [shuts the database](#)
- dbalter: [alters the database instance](#)
- dbparam: [alters database parameters](#)



- dbcreate: [recreates the database](#)
- dbshow: [shows database information](#)
- tsextend: [extends a tablespace](#)
- tscreate: [creates a tablespace](#)
- tsdrop: [drops a tablespace](#)
- tsalter: [alters a tablespace](#)
- dfalter: [alters a data file](#)
- dfmove: [moves a data file](#)
- tbreorg: [reorganizes tables](#)
- idrebuild: [rebuilds indexes](#)
- tbexport: [exports tables](#)
- tbimport: [imports tables](#)
- tbalter: [alters tables](#)
- idalter: [alters indexes](#)
- mstats: [manages database statistics](#)

## **-f dbstart**

This BRSPACE [function](#) starts the database instance. For more information, see [Starting Up the Database with BR\\*Tools](#).

Function options:

- `-f|-force`: if the database is running, forces an immediate shutdown then restarts the instance, even if SAP users are connected

Default: does not force a restart.

- `-i|-instance`: defines which database instances are to be started

**Syntax:** `-i|-instance all_inst|all_down|<instance>|<instance_list>`

- `all_inst` starts or restarts all instances of an Oracle Real Application Cluster (RAC) database
- `all_down` starts all database instances that are currently down, that is, stopped
- `<instance>` starts the specified instance
- `<instance_list>` starts the specified instances

Default: the database instance defined by the `ORACLE_SID` environment variable

Note

You only need to specify database instances for Oracle RAC

- `-m|-mode`: defines the mode in which the database instance starts

Syntax: `-m|-mode normal|restrict|force`

- `normal` starts the database instance normally
- `restrict` starts the database in restricted mode for database administration only
- `force` forces a shutdown abort and then a restart

Default: `normal`

- `-s|-state`: defines the state of the database after startup

Syntax: `-s|-state open|mount|nomount|standby`

- `open` opens the database for normal user access
- `mount` associates the database with its instance and evaluates control files
- `nomount` builds up the database instance and allocates operating system resources
- `standby` mounts a standby database

For more information on database states, see the Oracle documentation.

Default: `open`

## **-f dbshut**

This BRSPACE [function](#) shuts the database instance. For more information, see [Shutting Down the Database with BR\\*Tools](#).

Function options:

- `-f|-force`: forces an immediate shutdown of the database instance, even if SAP users are connected

Default: does not force a shutdown.

- `-i|-instance`: defines which database instances are to be shut

Syntax: `-i|-instance all_inst|all_up|<instance>|<instance_list>`

- `all_inst` shuts all instances of an Oracle Real Application Cluster (RAC). With option `-f|-force`, first starts up stopped instances before shutting them down, enabling a clean shutdown of aborted instances.
- `all_up` shuts all database instances that are currently up, that is, running
- `<instance>` shuts the specified instance
- `<instance_list>` shuts the specified instances

Default: the database instance defined by the `ORACLE_SID` environment variable

## Note

You only need to specify database instances for Oracle RAC.

- `-m|-mode`: defines the mode in which the database instance shuts

**Syntax:** `-m|-mode immediate|normal|transactional|abort`

- `immediate` shuts the database immediately after ending all user sessions and stopping all transactions (open transactions are rolled back)
- `normal` shuts the database cleanly after waiting for all users to disconnect.
- `transactional` shuts the database cleanly after waiting for all transactions to finish
- `abort` shuts the database immediately without rolling back open transactions

For more information on shutdown modes, see the Oracle documentation.

**Default:** `immediate`

## -f dbalter

This BRSPACE [function](#) alters the state of the database instance. For more information, see [Altering the Database Instance with BR\\*Tools](#).

Function options:

- `-a|-action`: specifies the action to change the state of the database instance

**Syntax:** `-a|-action switchlog|checkpoint|archlog|noarchlog`

- `switchlog` switches the current online redo log file
- `checkpoint` performs a database checkpoint
- `archlog` turns on `archivelog` mode
- `noarchlog` turns off `archivelog` mode

### Caution

The database must normally run in `archivelog` mode.

Otherwise you have no record of database transactions, which means you cannot restore and recover the database in the event of a crash.

For more information, see the Oracle documentation.

**Default:** none, since you must always specify an action

- `-f|-force`: forces an immediate shutdown of the database instance to mount state, as required to change `archivelog` mode

**Default:** does not force a shutdown if SAP users are connected.

- `-i|-instance`: defines which database instances are to be altered

**Syntax:** `-i|-instance all_inst|<instance>|<instance_list>`

- o `all_inst` alters all instances of an Oracle Real Application Cluster (RAC) database
- o `<instance>` alters the state of the specified instance
- o `<instance_list>` alters the state of the specified instances

**Default:** the database instance defined by the `ORACLE_SID` environment variable

**Note**

You only need to specify database instances for Oracle RAC.

## **-f dbparam**

This BRSPACE [function](#) changes the value of database parameters. For more information, see [Altering Database Parameters with BR\\*Tools](#).

**Function options:**

- `-a|-action`: specifies the action to change the database parameter

**Syntax:** `-a|-action change|reset|create`

- o `change` changes a database parameter
- o `reset` resets a database parameter
- o `create` creates the `init<DBSID>.ora` file from the `spfile`

**Default:** None, since you must always specify an action

- `-c|-comment`: lets you enter a comment on the parameter change

**Default:** No comment

- `-i|-instance`: defines for which database instance the parameter change is to apply

**Syntax:** `-i|-instance <instance>`

`<instance>` changes the parameter of the specified instance

**Default:** changes the parameter for all instances

**Note**

You only need to specify database instances in an Oracle Real Application Cluster (RAC) system.

- `-p|-parameter`: defines which database parameter is to be changed

**Syntax:** `-p|-parameter <parameter>`

`<parameter>` changes the specified parameter

Default: none, since you must always specify a parameter to change

#### Note

You must specify this option if you want to set a special parameter (that is, a “\_” parameter). For more information, see SAP Note 744585.

- `-s|-scope`: defines the scope of the parameter change

Syntax: `-s|-scope memory|spfile|both`

- `memory` changes the parameter immediately for the specified running instances
- `spfile` changes the parameter in the Oracle `spfile` for the specified instances, but this takes effect only after the instance has been restarted
- `both` changes the parameter in both `memory` and `spfile` for the specified instances

Default: `both`

- `-v|-value`: defines the new parameter value

Syntax: `-v|-value <value>`

`<value>` specifies the new parameter value

Default: none, since you must always specify the new parameter value.

## **-f dbcreate**

This BRSPACE [function](#) lets you recreate a database.

Function options:

- `-aa|-aux_autoext no|yes`  
SYS\_AUX tablespace file autoextend mode  
Default: old value
- `-af|-aux_file <file>|<sapdata_dir>|[sapdata]<N>|sapraw`  
SYS\_AUX tablespace file name or `sapdata/sapraw` directory  
Default: old value
- `-ai|-aux_incrsize <size>`  
SYS\_AUX tablespace file increment size in MB  
Default: old value
- `-am|-aux_maxsize <size>`  
SYS\_AUX tablespace file maximum size in MB  
Default: old value

- `-ar|-aux_rawlink <link_target>`  
SYS AUX tablespace raw disk or link target  
Default: old value
- `-as|-aux_size <size>]`  
SYS AUX tablespace file size in MB  
Default: old value
- `-ls|-log_size <size>`  
Redo log file size in MB  
Default: old value
- `-md|-max_data <number>`  
Maximum number of data files  
Default: old value
- `-mh|-max_hist <size>`  
Maximum size of redo log history  
Default: old value
- `-mi|-max_inst <number>`  
Maximum number of instances  
Default: old value
- `-ml|-max_log <number>`  
Maximum number of redo log groups  
Default: old value
- `-mm|-max_memb <number>`  
Maximum number of redo log members  
Default: old value
- `-of[1|2|3|4|5|6|7|8|9|10]|-orig_file[1|2|3|4|5|6|7|8|9|10]  
<file> |<sapdata_dir>|[sapdata]<N>|sapraw`  
Original redo log file name or sapraw directory  
Default: old value
- `-or[1|2|3|4|5|6|7|8|9|10]|-orig_rawlink[1|2|3|4|5|6|7|8|9|10]  
<link_target>`  
Original redo log raw disk or link target  
Default: old value

- `-ps|-pass_sys <password>`  
**SYS user password**  
**Default:** `<default_sys_password>`
- `-pt|-pass_syst <password>`  
**SYSTEM user password**  
**Default:** `<default_system_password>`
- `-rf[1|2|3|4|5|6|7|8|9|10]|-mirr_file[1|2|3|4|5|6|7|8|9|10]`  
`<file>|<sapdata_dir>|[sapdata]<N>|sapraw`  
**Mirror redo log file name or sapraw directory**  
**Default:** old value
- `-rr[1|2|3|4|5|6|7|8|9|10]|-mirr_rawlink[1|2|3|4|5|6|7|8|9|10]`  
`<link_target>`  
**Mirror redo log raw disk or link target**  
**Default:** old value
- `-sa|-syst_autoext no|yes`  
**SYSTEM tablespace file autoextend mode**  
**Default:** old value
- `-sf|-syst_file <file>|<sapdata_dir>|[sapdata]<N>|sapraw`  
**SYSTEM tablespace file name or sapdata or sapraw directory**  
**Default:** old value
- `-si|-syst_incrsize <size>`  
**SYSTEM tablespace file increment size in MB**  
**Default:** old value
- `-sm|-syst_maxsize <size>`  
**SYSTEM tablespace file maximum size in MB**  
**Default:** old value
- `-sr|-syst_rawlink <link_target>`  
**SYSTEM tablespace raw disk or link target**  
**Default:** old value
- `-ss|-syst_size <size>`  
**SYSTEM tablespace file size in MB**  
**Default:** old value

- `-tf|-temp_file <file>|<sapdata_dir>|[sapdata]<N>|sapraw`  
Temporary tablespace file name or sapdata or sapraw directory  
Default: old value
- `-tr|-temp_rawlink <link_target>`  
Temporary tablespace raw disk or link target  
Default: old value
- `-ts|-temp_size <size>`  
Temporary tablespace file size in MB  
Default: old value
- `-tt|-temp_tsp <tablespace>`  
Temporary tablespace name  
Default: PSAPTEMP
- `-uf|-undo_file <file>|<sapdata_dir>|[sapdata]<N>|sapraw`  
Undo tablespace file name or sapdata or sapraw directory
- `-ur|-undo_rawlink <link_target>`  
Syntax: `-ur|-undo_rawlink <link_target>`  
Undo tablespace raw disk or link target  
Default: old value
- `-us|-undo_size <size>`  
Undo tablespace file size in MB  
Default: old value
- `-ut|-undo_tsp <tablespace>`  
Undo tablespace name  
Default: PSAPUNDO

## **-f dbshow**

This BRSPACE [function](#) shows information on objects in the database. For more information, see:

Function options:

- `-c|-class`: specifies the class of database information to be shown  
Syntax: `-c|-class <info_class>`  
<info\_class> can be:



- o dbstate [shows database instance status](#)
- o dbparam [shows database parameters](#)
- o dbowner [shows database owners](#)
- o fbstate **shows** [flashback database status](#)
- o tsinfo [shows tablespaces](#)
- o dfinfo [shows data files](#)
- o rfinfo [shows redo log files](#)
- o cfinfo [shows control files](#)
- o dvinfos [shows disk volumes](#)
- o tbinfos [shows tables](#)
- o idinfos [shows indexes](#)
- o tpinfos [shows table partitions](#)
- o ipinfos [shows index partitions](#)
- o sginfos [shows segments](#)
- o seinfos [shows segment extents](#)
- o feinfos [shows free extents](#)

Default: dbstate

- -f|-file: specifies the data file name(s)

**Syntax:** -f|-file <file>|<file\_id>|<file\_id1>-<file\_id2>|<file\_list>

Default: all files

- -i|-index: specifies the index name(s)

**Syntax:** -i|-index all|allsel| [<owner>.]<index>| [<owner>.] [<prefix>] % [<suffix>] | [<owner>.] [<prefix>] \* [<suffix>] |<index\_list>

where:

all or “%” means preselect

allsel or “\*” means final select:

For more information about selection wildcards, see *Selecting Objects* in [Segment Management with BR\\*Tools](#).

Default: all indexes

- -l|-log: logs database information shown

Default: do not log

- `-n|-instance`: specifies the database instance name(s)

Syntax: `-n|-instance <instance>|<instance_list>`

Default: all database instances

- `-o|-owner`: specifies the SAP owner of the tables or indexes

Syntax: `-o|-owner <owner>|<owner_list>`

`<owner>` specifies the name of the SAP owner

Default: all SAP owners

- `-p|-parameter`: specifies which database parameters are to be shown

Syntax: `-p|-parameter <parameter>|<parameter_list>`

`<parameter>` or `<parameter_list>` specifies the parameter(s)

Default: all parameters

- `-s|-tablespace`: specifies the tablespace name for the objects to be shown

Syntax: `-s|-tablespace <tablespace>|<tablespace_list>`

`<tablespace>` or `<tablespace_list>` specifies the tablespace(s)

Default: all tablespaces

- `-t|-table`: specifies the table name(s) of the objects to be shown

Syntax: `-t|-table all|allsel|[<owner>.]<table>|`

`[<owner>.] [<prefix>] % [<suffix>] |`

`[<owner>.] [<prefix>] * [<suffix>] |<table_list>`

where:

`all` or `"%"` means preselect

`allsel` or `"**"` means final select:

For more information about selection wildcards, see *Selecting Objects* in [Segment Management with BR\\*Tools](#).

Default: all tables

## **-f tsextend**

This BRSPACE [function](#) extends a tablespace. For more information, see [Extending a Tablespace with BR\\*Tools](#). You can add up to five files to a tablespace with this function.

Function options:

- `-a|-autoextend`: sets autoextend mode for the data file so that Oracle automatically extends the tablespace as it grows

**Syntax:** `-a|-autoextend[1|2|3|4|5]`

`[1|2|3|4|5]` specifies the sequence number of the file to be added to the tablespace

**Default:** as for the last data file added to the tablespace

- `-f|-file`: defines the data file to extend the tablespace

**Syntax:** `-f|-file[1|2|3|4|5] <file>|<sapdata_dir>|`

`sapdata<N>|<N>|sapraw`

`[1|2|3|4|5]` specifies the sequence number of the file to be added to the tablespace

`<file>` specifies the full name of the data file

`<sapdata_dir>` specifies the path name to the `sapdata` directory where the new file will be located

`sapdata<N>` or `<N>` specifies the name or number of the `sapdata` directory where the new file is located

`sapraw` specifies the directory for a softlink to raw disks

BRSPACE automatically extends the incomplete file name to the full name using the [SAP naming conventions](#).

**Default:** generated automatically according to SAP naming conventions (recommended) in the `sapdata` directory used by last file added to the tablespace

- `-i|-incrsize`: defines the increment by which the file is extended if `autoextend` is set

**Syntax:** `-i|-incrsize[1|2|3|4|5] <size>`

`[1|2|3|4|5]` specifies the sequence number of the file to be added to the tablespace

`<size>` specifies the size of the increment in MB

**Default:** as for the last data file added to the tablespace

- `-m|-maxsize`: defines the maximum file size to which the file can be extended if `autoextend` is set

**Syntax:** `-m|-maxsize[1|2|3|4|5] <size>`

`<size>` specifies the maximum file size in MB. 0 sets the maximum database size to unlimited (maximum 32 GB for SAP standard of 8 KB database block size).

`[1|2|3|4|5]` specifies the sequence number of the file to be added to the tablespace

**Default:** as for the last data file added to the tablespace

- `-r|-rawlink`: defines the destination of a raw disk or a link to a non-`sapdata` directory. This is defined as follows:
  - For a raw disk, the SAP convention is to specify a softlink in the directory `/oracle/<DBSID>/sapraw`

- For a link to a destination outside sapdata, such as to `/oracle/temp`, the new file is added as a symbolic link

**Syntax:** `-r|-rawlink[1|2|3|4|5] <link target>`

`[1|2|3|4|5]` specifies the sequence number of the file to be added to the tablespace

`<link target>` specifies the link target

Default: no raw link

- `-s|-size`: defines the data file size

**Syntax:** `-s|-size[1|2|3|4|5] <size>`

`[1|2|3|4|5]` specifies the sequence number of the file to be added to the tablespace

`<size>` specifies the data file size in MB

Default: as for the last file added to the tablespace

- `-t|-tablespace`: Defines the tablespace to be extended

**Syntax:** `-t|-tablespace[1|2|3|4|5] <tablespace>`

`[1|2|3|4|5]` specifies the sequence number of the file to be added to the tablespace

`<tablespace>` specifies the tablespace name

Default: none, since you must always define the tablespace name

## **-f tscreate**

This BRSPACE [function](#) creates a new tablespace with up to five files. For more information, see [Creating a Tablespace with BR\\*Tools](#).

Function options:

- `-a|-autoextend`: sets autoextend mode for the data file so that Oracle automatically extends the tablespace as it grows

**Syntax:** `-a|-autoextend[1|2|3|4|5]`

`[1|2|3|4|5]` specifies the sequence number of the file for the new tablespace

Default: as for the last data file added to the database

- `-c|-contents`: defines the contents of the new tablespace

**Syntax:** `-c|-contents data|temp|undo`

- `data`: normal data contents
- `temp`: temporary tablespace for certain large-scale actions (for example, sorts)
- `undo`: undo tablespace

For more information about `temp` and `undo`, see the Oracle documentation.

Default: `data`

- `-d|-data`: defines the type of data in the new tablespace, according to SAP convention

Syntax: `-d|-data table|index|both`

- `table`: table data
- `index`: index data
- `both`: both table and index data - this is the current SAP recommendation

Default: `both`

- `-f|-file`: defines the first data file of the new tablespace

Syntax: `-f|-file[1|2|3|4|5] <file>|<sapdata_dir>|`

`sapdata<N>|<N>|sapraw`

`[1|2|3|4|5]` specifies the sequence number of the file for the new tablespace

`<file>` specifies the full name of the data file

`<sapdata_dir>` specifies the path name to the `sapdata` directory where the new file is located

`sapdata<N>|<N>` specifies the name or number of the `sapdata` directory where the new file is located

`sapraw` specifies the directory for a softlink to raw disks

BRSPACE automatically extends the incomplete file name to the full name using the [SAP naming conventions](#).

Default: generated automatically according to SAP naming conventions recommended in the `sapdata` directory used by the last file added to the database

- `-i|-incrsize`: Defines the increment by which the file is extended if `autoextend` is set

Syntax: `-i|-incrsize[1|2|3|4|5] <size>`

`[1|2|3|4|5]` specifies the sequence number of the file for the new tablespace

`<size>` specifies the size of the increment in MB

Default: as for the last data file added to the database

- `-j|-join`: defines the tablespace joined to the tablespace already defined with the above parameters, if option `-d|-data` was set to `table` or `index`. For example, `PSAPSTABD` and `PSAPSTABI` are, according to the old tablespace naming conventions, a pair of joined tablespaces.

Syntax: `-j|-join <tablespace>`

`<tablespace>` specifies the joined tablespace name

Default: follows SAP conventions

- `-l|-class`: specifies the table data class for the tablespace

**Syntax:** `-l|-class all|<tab_class>|<tab_class_list>|`

`<old_tsp>|<old_tsp_list>|none`

- `all` specifies that the tablespace has the standard SAP table data classes except class `USER`
- `<tab_class>` specifies that the tablespace has the new customer table data class. The name must start with “Y” or “Z”.
- `<tab_class_list>` as `tab_class` but for multiple classes
- `<old_tsp>` specifies that the new tablespace has the same table data class as an existing tablespace, which is later dropped
- `<old_tsp_list>` as `old_tsp` but for multiple classes
- `none` specifies that no table data class is assigned

You must specify data classes beginning with the letters “Y” or “Z” (these are customer data classes) and with a maximum length of five characters. You use these data classes for customer development or when moving tables. If you do not set this option, `BRSPACE` automatically generates a customer data classes beginning with “U”.

- `-m|-maxsize`: defines the maximum file size to which the file can be extended if `autoextend` is set

**Syntax:** `-m|-maxsize[1|2|3|4|5] <size>`

`[1|2|3|4|5]` specifies the sequence number of the file for the new tablespace

`<size>` specifies the maximum file size in MB. 0 sets the maximum database size to unlimited (maximum 32 GB for SAP standard of 8 KB database block size).

**Default:** as for the last data file added to the database

- `-o|-owner`: defines the SAP owner of the tables or indexes that are located in the new tablespace – required in Multiple Components in One Database (MCO) configurations

**Syntax:** `-o|-owner <owner>|none`

- `<owner>` specifies the name of the SAP owner
- `none` specifies that there is no owner

**Default:** no default in MCO, otherwise the single SAP owner

- `-p|-space`: defines the type of Oracle segment space management in the tablespace

**Syntax:** `-p|-space auto|manual`

- `auto`: bitmap management
- `manual`: freelist management

**Default:** `auto`

- `-r|-rawlink`: defines the destination of a raw disk or a link to a non-sapdata directory. This is defined as follows:
  - For a raw disk, the SAP convention is to specify a softlink in the directory `/oracle/<DBSID>/sapraw`
  - For a link to a destination outside sapdata, such as to `/oracle/temp`, the new file is added as a symbolic link

**Syntax:** `-r|-rawlink[1|2|3|4|5] <link target>`

`[1|2|3|4|5]` specifies the sequence number of the file for the new tablespace

`<link target>` specifies the link target

**Default:** no raw link

- `-s|-size`: defines the tablespace file size

**Syntax:** `-s|-size[1|2|3|4|5] <size>`

`[1|2|3|4|5]` specifies the sequence number of the file for the new tablespace

`<size>` specifies the data file size in MB

**Default:** as for the last data file added to the database

- `-t|-tablespace`: defines the tablespace to be created

**Syntax:** `-t|-tablespace <tablespace>`

`<tablespace>` specifies the [tablespace name](#)

**Default:** none, since you must always define the tablespace name

- `-u|-uniform`: defines tablespaces with a uniform size, where the space is not automatically allocated

**Syntax:** `-u|-uniform <size>`

`<size>` specifies the uniform tablespace size in MB

**Default:** none

**Note**

The options below all refer to the joined index tablespace.

- `-xa|-xautoextend`: refers to the joined index tablespace – see `-a|-autoextend` above
- `-xf|-xfile`: refers to the joined index tablespace – see `-f|-file` above
- `-xi|-xincrsize`: refers to the joined index tablespace – see `-i|-incrsize` above
- `-xm|-xmaxsize`: refers to the joined index tablespace – see `-m|-maxsize` above
- `-xr|-xrawlink`: refers to the joined index tablespace – see `-r|-rawlink` above

- `-xs|-xsize`: refers to the joined index tablespace – see `-s|-size` above

## **-f tsdrop**

This BRSPACE [function](#) drops a tablespace. For more information, see [Dropping a Tablespace with BR\\*Tools](#).

Function options:

- `-f|-force`: forces the tablespace to be dropped, even if not empty

Default: tablespace only dropped if already empty

- `-t|-tablespace`: defines the tablespace to be dropped

Syntax: `-t|-tablespace <tablespace>`

`<tablespace>` specifies the tablespace name

Default: none, since you must always define the tablespace name

## **-f tsalter**

This BRSPACE [function](#) alters a tablespace. For more information, see [Altering a Tablespace with BR\\*Tools](#).

Function options:

- `-a|-action`: specifies the action to alter the tablespace

Syntax: `-a|-action online|offline|begback|endback|coalesce`

- `online` sets the tablespace online
- `offline` sets the tablespace offline
- `begback` sets backup status
- `endback` resets backup status
- `coalesce` coalesces free extents
- `rename` renames a tablespace (Oracle 10g or higher)

Default: none, since you must always define the action

- `-f|-force`: forces offline mode

Default: do not set offline if a SAP user is connected

- `-m|-mode`: defines the mode in which the tablespace is set offline

Syntax: `-m|-mode normal|immediate|temporary`

For more information on offline mode, see the Oracle documentation.

- `-n|-name`: defines the new tablespace name

Syntax: `-n|-name <name>`



Default: none, since you must always specify the new name

- `-t|-tablespace`: defines the tablespace to be altered

Syntax: `-t|-tablespace all_ts|<tablespace>|<tablespace list>`

`all_ts`: all eligible tablespaces

`<tablespace>` or `<tablespace list>` specifies the tablespace name(s)

Default: none, since you must always define the tablespace name

## **-f dfalter**

This BRSPACE [function](#) alters a data file. For more information, see [Altering a Data File with BR\\*Tools](#).

Function options:

- `-a|-action`: specifies the action to alter the tablespace

Syntax: `-a online|offline|autoext|drop|fixsize|rename|resize`

- `online` sets the data file online
- `offline` sets the data file offline
- `autoext` switches on or maintains autoextend
- `fixsize` switches off autoextend
- `resize` resizes the data file
- `rename` renames the data file
- `drop` drops the empty data file

Default: none, since you must always specify the action

- `-c|-force`: forces offline mode and stop database

Default: do not set offline or stop database if a SAP user is connected

- `-f|-file`: specifies the data file name

Syntax: `-f|-file all_df|<file>|<file_id>|<file_list>`

`all_df`: all eligible data files (for example, from the specified tablespaces)

`<file>|<file_id>|<file_list>` specifies the data file(s)

Default: none, since you must always define the file name

- `-i|-incrsize`: specifies the new data file increment size

Syntax: `-i|-incrsize <size>`

`<size>` specifies the size in MB

Default: none, since you must always specify the new file increment size

- `-m|-maxsize`: specifies the new maximum data file size  
**Syntax:** `-m|-maxsize <size>`  
`<size>` specifies the size in MB. 0 sets the maximum database size to unlimited (maximum 32 GB for SAP standard of 8 KB database block size).  
**Default:** none, since you must always specify the maximum file size
- `-n|-name`: specifies the new data file name  
**Syntax:** `-n|-name <new_name>`  
`<new_name>` specifies the new data file name, according to SAP conventions  
**Default:** none, since you must always specify the new file name
- `-s|-size`: specifies the new data file size  
**Syntax:** `-s|-size <size>`  
`<size>` specifies the size in MB  
**Default:** none, since you must always specify the new file size
- `-t|-tablespace`: specifies the tablespace(s) for which data files are to be altered  
**Syntax:** `-t|-tablespace <tablespace>|<tablespace_list>`  
**Default:** none, but you can specify data files explicitly with option `-f` (see above).

## **-f dfmove**

This BRSPACE [function](#) moves a data file. For more information, see [Moving a Data File with BR\\*Tools](#).

Function options:

- `-c|-force`: forces database shutdown for move  
**Default:** do not force shutdown if a SAP user is connected
- `-d|-destination`: specifies the destination for the move  
**Syntax:** `-d|-destination <sapdata_dir>|<sapraw_dir>`  
`<sapdata_dir>|<sapraw_dir>` specifies a `sapdata` or `sapraw` (UNIX only) directory  
**Default:** none, since you must always specify the move destination
- `-f|-file`: specifies the data file name  
**Syntax:** `-f|-file all_df|<file>|<file_id>| <file_id1>-<file_id2>|<file_list>`  
`all_df`: all eligible data files (for example, from the specified tablespace)  
**Default:** none, since you must always define the file name

- `-p|-parallel`: specifies how many copy processes run in parallel to move multiple files

Syntax: `-p|-parallel <count>`

`<count>` specifies the number of parallel copy processes

Default: 1, that is, a single process

- `-r|-rawlink`: defines the destination of a raw disk or a link to a non-sapdata directory. This is defined as follows:
  - For a raw disk, the SAP convention is to specify a softlink in the directory `/oracle/<DBSID>/sapraw`
  - For a link to a destination outside `sapdata`, such as to `/oracle/temp`, the new file is added as a symbolic link

Syntax: `-r|-rawlink <link target>`

`<link target>` specifies the link target

Default: no raw link

- `-t|-tablespace`: specifies the tablespace(s) for which data files are to be moved

Syntax: `-t|-tablespace <tablespace>|<tablespace_list>`

Default: none, but you can specify data files explicitly with option `-f` (see above)

## **-f tbreorg**

This BRSPACE [function](#) reorganizes tables. For more information, see [Reorganizing Tables with BR\\*Tools](#).

Function options:

- `-a|-action`: specifies the action for the table reorganization

Syntax: `-a|-action`

`reorg|check|cleanup|stop|suspend|resume|long2lob`

- `reorg` reorganizes tables in the standard way
- `check` checks tables for reorganization

This option identifies tables that cannot be reorganized because, for example, they have `LONG` columns.

### Caution

With this option, the processing logic looks exactly the same as for the standard reorganization. However, processing is terminated directly before the actual reorganization starts.

- `cleanup` cleans up tables after failure

This option cleans up after a reorganization has terminated unexpectedly. Only use this option if a "hard" termination was executed on the

reorganization and if BRSPACE was unable to connect to the database during the final phase (for example, due to a core dump).

In this case, perform the cleanup repeatedly until no more objects for deletion are found.

#### Caution

To make sure that all objects are checked and tidied up where necessary, simply select all tables for the cleanup function in list menu 352 or set the command option `-t "*"` .

- o `stop` stops reorganization

This option enables a “clean” premature termination of the reorganization without all tables being processed. However, the reorganization of individual tables is not cancelled in this case. Instead, the process is terminated before the next table is processed. Therefore this action can take a long time.

- o `suspend` suspends reorganization

This option lets you pause the reorganization without terminating it. However, the reorganization of individual tables is not cancelled in this case. Instead, the process is terminated before the next table is processed. Therefore this action can take a long time.

- o `resume` resumes reorganization

This option lets you resume the reorganization that was paused with the `suspend` option.

- o `long2lob` performs an online conversion of the `LONG` or `LONG RAW` fields to `LOB`. You can then perform an online reorganization of tables converted in this way. Prerequisite for this is Oracle 10g or higher and SAP kernel 7.00 or higher.

Default: `reorg`

#### Caution

You start the `stop`, `suspend`, and `resume` actions in parallel to a current reorganization, usually from a different shell or command prompt, for example:

```
brspace -c force -f tbreorg -a suspend
```

...

```
brspace -c force -f tbreorg -a resume
```

- `-d|-ddl`: specifies how the Data Definition Language (DDL) statements for tables to be reorganized are handled

Syntax: `-d|-ddl yes|no|first|only|only_tab|only_ind|only_dep`

- o `yes` is the default and specifies that the DDL statements are generated and saved immediately before a table is reorganized
- o `no` specifies that the DDL statements are not saved – we do not normally recommend this, although there is a small performance gain

- `first` specifies that all DDL statements are generated before the reorganization begins. Processing then stops before the reorganization so that you can change the statements if required.

**Caution**

If you change some attributes, make sure that they:

- Are syntactically correct
  - Do not contain any new fields
  - Are compatible with the SAP dictionary
- `only` specifies that only the DDL statements are generated and saved. There is no actual reorganization.
  - `only_tab` specifies that the DDL statements are generated and saved only for tables
  - `only_ind` specifies that the DDL statements are generated and saved only for indexes
  - `only_dep` specifies that the DDL statements are generated and saved only for dependent objects such as constraints, triggers, grants

Default: `yes`

- `-e|-degree`: specifies parallel degree used internally by the DBMS\_REDEFINITION package for the reorganization of individual tables BRSPACE resets it to the old value after the reorganization has finished.

Syntax: `-e|-degree <degree>`

Default: the parallel degree of the current table

- `-i|-indts`: specifies that a separate index tablespace is to be used to store the indexes

Syntax: `-i|-indts <tablespace>`

`<tablespace>` specifies the tablespace name

Default: current index tablespace if option `-n|-newts` is not specified. Otherwise, the new table tablespace, that is, no separate index tablespace

- `-l|-initial`: specifies the initial extent size of the reorganized table

Syntax: `-l|-initial <category>`

`<category>` specifies the size of the initial extent after reorganization and is restricted to the following possible values:

- 1: 16 KB
- 2: 64 KB
- 3: 1 MB
- 4: 8 MB

- 5: 64 MB
- 6: 1 GB

Default: none

If you do not specify `initial` at all, the reorganized table retains the initial extent size from before the reorganization.

- `-m|-mode`: lets you reorganize the tables offline using `ALTER TABLE (PARTITION) MOVE` (not the Oracle DBMS\_REDEFINITION package).

This method can be faster than an online reorganization but it locks the tables, so make sure that your SAP system is *not* running at reorganization time.

Syntax: `-m|-mode online|offline`

Default: online

- `-n|-newts`: specifies that the tables are reorganized to a new tablespace

Syntax: `-n|-newts <tablespace>`

`<tablespace>` specifies the tablespace name

Default: current tablespace where the tables are located

- `-o|-owner`: defines the SAP owner of tables- useful for Multiple Components in One Database (MCOB) configurations

Syntax: `-o|-owner <owner>|<owner_list>`

`<owner>` or `<owner_list>` specifies the name of the SAP owner or owners

Default: no default in MCOB, otherwise the single SAP owner

- `-p|-parallel`: specifies parallel processing for the reorganization

Syntax: `-p|-parallel <threads>`

`<threads>` specifies the number of processing threads running in parallel

Default: 1, that is, no parallel processing

- `-r|-sortind` specifies the index on which the table is sorted during reorganization

Syntax: `-r|-sortind first|<ind_id>|<ind_name>`

- `first`: sorts on the SAP primary index (that is, with ID 0), or on the first index (lexically), or on the only index
- `<ind_id>`: sorts on the index with the ID `<ind_id>`, for example: 0, 1, or 2
- `<ind_name>`: sorts on the index with the name `<ind_name>`

- `-s|-tablespace`: specifies the tablespace of the tables to be reorganized

Syntax: `-s|-tablespace <tablespace>|<tablespace list>`

`<tablespace>` or `<tablespace list>` specifies the tablespace name(s)

Default: none

- `-t|-table`: specifies the tables to be reorganized

**Syntax:** `-t|-table all|allsel| [<owner>.<table>|  
 [<owner>.] [<prefix>] % [<suffix>] |  
 [<owner>.] [<prefix>] * [<suffix>] | <table_list>`

where:

`all` or “%” means preselect

`allsel` or “\*” means final select:

For more information about wildcards for preselect and select, see *Selecting Objects* in [Segment Management with BR\\*Tools](#).

Default: none, since you must always specify tables for reorganization

## -f idrebuild

This BRSPACE [function](#) rebuilds indexes. For more information, see [Rebuilding Indexes with BR\\*Tools](#).

Function options:

- `-a|-action`: specifies the action for the index rebuild

**Syntax:** `-a|-action rebuild|stop|suspend|resume`

- `rebuild` rebuilds indexes in the standard way
- `stop` stops the rebuild

This option enables a “clean” premature termination of the rebuild without all indexes being processed. However, this does not cancel the rebuild of individual indexes. Instead, the process is terminated before the next index is processed. Therefore this action can take a long time.

- `suspend` suspends the rebuild

This option lets you suspend the rebuild without terminating it. However, this does not cancel the rebuild of individual indexes. Instead, the process is terminated before the next index is processed. Therefore this action can take a long time.

- `resume` resumes the rebuild

This option lets you resume the rebuild that was paused with the `suspend` option.

Default: `rebuild`

### Caution

You start the `stop`, `suspend`, and `resume` actions in parallel to a current rebuild, usually from a different shell or command prompt, for example:

```
brspace -c force -f tbrebuild -a suspend
```

...

```
brspace -c force -f tbrebuild -a resume
```

- `-e|-degree` specifies the parallel degree to be used internally by the `ALTER INDEX REBUILD` statement for the rebuild of individual indexes. It is reset to the old value by `BRSPACE` after the rebuild has finished.

Syntax: `-e|-degree <degree>`

Default: the parallel degree of the current index

- `-i|-index` specifies the indexes to be rebuilt

Syntax: `-i|-index all|allsel| [<owner>.]<index>|`

`[<owner>.] [<prefix>]% [<suffix>]|`

`[<owner>.] [<prefix>]* [<suffix>]|<index_list>`

where:

`all` or “%” means preselect

`allsel` or “\*” means final select:

For more information about wildcards for preselect and select, see *Selecting Objects* in [Segment Management with BR\\*Tools](#).

Default: none, since you must always specify indexes for rebuild

- `-l|-initial`: specifies the initial extent size of the reorganized index

Syntax: `-l|-initial <category>`

`<category>` specifies the size of the initial extent after reorganization and is restricted to the following possible values:

- 1: 16 KB
- 2: 64 KB
- 3: 1 MB
- 4: 8 MB
- 5: 64 MB
- 6: 1 GB

Default: none

If you do not specify `initial` at all, the reorganized index retains the initial extent size from before the reorganization.

- `-m|-mode`: lets you rebuild the indexes offline – that is, without the `ONLINE` clause of the `ALTER INDEX REBUILD` statement (this locks the index).

Syntax: `-m|-mode online|offline`

This method can be faster than an online rebuild but it locks the indexes, so make sure that your SAP system is *not* running at rebuild time.



Default: online

- `-n|-newts`: specifies that the indexes are rebuilt to a new tablespace

Syntax: `-n|-newts <tablespace>`

`<tablespace>` specifies the name of the new tablespace

Default: current tablespace that the indexes are located in

- `-o|-owner`: defines the SAP owner of the tables - useful for Multiple Components in One Database (MCOB) configurations

Syntax: `-o|-owner <owner>|<owner_list>`

`<owner>` or `<owner_list>` specifies the name of the SAP owner or owners

Default: no default in MCOB, otherwise the single SAP owner

- `-p|-parallel`: specifies parallel processing for the rebuild

Syntax: `-p|-parallel <threads>`

`<threads>` specifies the number of processing threads running in parallel

Default: 1, that is, no parallel processing

- `-s|-tablespace`: specifies the tablespace of the indexes to be rebuilt

Syntax: `-s|-tablespace <tablespace>|<tablespace list>`

`<tablespace>` or `<tablespace list>` specifies the tablespace name(s)

Default: none

- `-t|-table`: specifies the table(s) to be exported

Syntax: `-t|-table all|allsel| [<owner>.]<table>|`

`[<owner>.] [<prefix>]% [<suffix>] |`

`[<owner>.] [<prefix>]* [<suffix>] |<table_list>`

where:

`all` or `"%"` means preselect

`allsel` or `"**"` means here also preselect

For more information about wildcards for preselect and select, see *Selecting Objects* in [Segment Management with BR\\*Tools](#).

Default: none

## **-f tbexport**

This BRSPACE [function](#) exports tables into an operating system file. For more information, see [Exporting a Table with BR\\*Tools](#) and [Special Export and Import Functions with BRSPACE](#).

BRSPACE now supports the Oracle Data Pump tool for export, which is called “expdp” in this section. The conventional export is called “exp”.

#### Note

Since this function directly uses the Oracle table export tool EXP, see the Oracle documentation for more information on most of the parameters listed below.

#### Function options:

- **-b|-buffer:** specifies the export buffer size for exp  
Syntax: `-b|-buffer <size>`  
<size> specifies the size in KB  
Default: 10240
- **-c|-constraints:** specifies whether table constraints are exported  
Syntax: `-c|-constraints yes|no`  
Default: yes
- **-d|-direct:** specifies whether direct path is used for the export mode with exp  
Syntax: `-d|-direct yes|no`  
Default: yes
- **-e|-triggers:** specifies whether table triggers are exported  
Syntax: `-e|-triggers yes|no`  
Default: yes
- **-f|-force:** forces export, even if SAP users are connected  
Default: export is not forced if SAP user is connected
- **-g|-grants:** specifies whether table grants are exported  
Syntax: `-g|-grants yes|no`  
Default: yes
- **-i|-indexes:** specifies whether table indexes are exported  
Syntax: `-i|-indexes yes|no`  
Default: yes
- **-l|-utility:** specifies whether the export uses the Data Pump tool  
Syntax: `-l|-utility exp|expdp`  
Default: exp  
exp means that the export uses the exp tool

expdp means that the export uses the Data Pump tool

- `-m|-compress`: specifies whether table extents or data are compressed

**Syntax:** `-m|-compress yes|no|meta`

**Default:** `yes`

`yes` is only for exp

`meta` means that metadata is compressed, only for expdp

- `-n|-consistent`: specifies whether the export is consistent to a single point-in-time

**Syntax:** `-n|-consistent yes|no`

**Default:** `no`

**Caution**

If you use this option, you *cannot* use the OPS\$ mechanism to connect the database. For more information, see [-uj-user](#).

- `-o|-owner`: defines the SAP owner of the tables to be exported - useful in Multiple Components in One Database (MCOB) configurations

**Syntax:** `-o|-owner <owner>|<owner_list>|all|full`

`<owner>` or `<owner_list>` specifies the name of the SAP owner or owners

`all` specifies all SAP owners

`full` specifies full database export

**Default:** no default in MCOB, otherwise the single SAP owner

- `-p|-parallel`: specifies the degree of parallelism for the export with expdp

**Syntax:** `-p|-parallel <degree>`

**Default:** `1`

- `-r|-rows`: specifies whether table rows are exported

**Syntax:** `-r|-rows yes|no|only`

**Default:** `yes`

`only` means that only data is exported for expdp

- `-s|-tablespace`: specifies the tablespace of the tables to be exported

**Syntax:** `-s|-tablespace <tablespace>|<tablespace list>`

`<tablespace>` or `<tablespace list>` specifies the tablespace name

**Default:** `none`

- `-t|-tables`: specifies the table(s) to be exported

**Syntax:** `-t|-tables all|allsel| [<owner>.]<table> |`  
`[<owner>.] [<prefix>] % [<suffix>] |`  
`[<owner>.] [<prefix>] * [<suffix>] | <table_list>`

where:

`all` or “%” means preselect

`allsel` or “\*” means final select:

For more information about wildcards for preselect and select, see *Selecting Objects* in [Segment Management with BR\\*Tools](#).

**Default:** none, since you must always select a table

- `-u|-dumpdir`: specifies the export dump directory

**Syntax:** `-u|-dumpdir <dir>`

`<dir>` specifies the dump directory name

**Default:** work directory, usually `$SAPDTA_HOME/sapreorg`

- `-z|-filesize` specifies the dump file size

**Syntax:** `-z|-filesize <size>`

`<size>` specifies the maximal dump file size in MB

**Default:** 2000/20000. 2000 is the normal maximal dump file size. However, if your operating system has large file support, the default is 20000.

## -f tbimport

This BRSPACE [function](#) imports tables from an operating system file. For more information, see [Importing Tables with BR\\*Tools](#) and [Special Export and Import Functions with BRSPACE](#).

BRSPACE now supports the Oracle Data Pump tool for import, which is called “impdp” in this section. The conventional import is called “imp”.

### Note

Since this function directly uses the Oracle import tool IMP, see the Oracle documentation for more information on most of the parameters listed below.

### Function options:

- `-a|-action`: specifies the processing action for existing tables with impdp

**Syntax:** `-a|-action skip|append|truncate|replace`

`skip` leaves the table as it is and moves on to the next object.

`append` loads rows from the source and leaves existing rows unchanged.

`truncate` deletes existing rows and then loads rows from the source.

`replace` drops the existing table and then creates and loads it from the source.

This is only supported for `impdp`.

Default: `skip`

- `-b|-buffer`: specifies the import buffer size for `imp`  
Syntax: `-b|-buffer <size>`  
`<size>` specifies the size in KB  
Default: `10240`
- `-c|-constraints`: specifies whether table constraints are imported  
Syntax: `-c|-constraints yes|no`  
Default: `yes`
- `-e|-triggers`: specifies whether triggers are imported with `impdp`  
Syntax: `-e|-triggers yes|no`  
Default: `yes`
- `-f|-force`: forces import, even if SAP users are connected  
Default: import is not forced if SAP user is connected
- `-g|-grants`: specifies whether table grants are imported  
Syntax: `-g|-grants yes|no`  
Default: `yes`
- `-i|-indexes`: specifies whether table indexes are created  
Syntax: `-i|-indexes yes|no`  
Default: `yes`
- `-m|-commit`: specifies whether there is a commit after each array insert with `imp`  
Syntax: `-m|-commit yes|no`  
Default: `yes`
- `-n|-ignore`: specifies whether table creation errors are ignored with `imp`  
Syntax: `-n|-ignore yes|no`  
Default: `no`
- `-o|-owner`: defines the SAP owner of the tables to be imported - useful in Multiple Components in One Database (MCOB) configurations  
Syntax: `-o|-owner <owner>|<owner_list>`  
`<owner>` or `<owner_list>` specifies the name of the SAP owner or owners  
Default: no default in MCOB, otherwise the single SAP owner

- `-p|-parallel`: specifies the degree of parallelism for the import with impdp  
**Syntax:** `-p|-parallel <degree>`  
**Default:** 1
- `-r|-rows`: specifies whether table rows are imported  
**Syntax:** `-r|-rows yes|no|only`  
`only` specifies that only data is imported for impdp  
**Default:** yes
- `-t|-tables`: specifies the table(s) to be imported  
**Syntax:** `-t|-tables [<owner>.]<table|<table_list>`  
**Default:** none
- `-x|-export`: specifies the export file  
**Syntax:** `-x|-export <exp_run>|<exp_dump>|<exp_dump_list>`
  - `<exp_run>` specifies the BRSPACE run identifier where the export dump file was created, that is, the BRSPACE log name (`s<coded_timestamp>.tbr`)
  - `<exp_dump>` specifies the full path of the export dump file (not necessarily a BRSPACE export)
  - `<exp_dump_list>` specifies a list of export dump files (not necessarily BRSPACE export dump files)**Default:** last BRSPACE export run
- `-y|-type` specifies the type of import  
**Syntax:** `-y|-type full|tables|indexfile|show|sqlfile`
  - `full` imports everything
  - `tables` imports individual tables as specified in the `-t|-table` parameter (see above)
  - `indexfile` dummy import – the DDL statements are written to the text file `indfile.sql` in the working directory but there is no import. Only supported for imp.
  - `show` shows the DDL statements in the export dump file but there is no import. Only supported for imp.
  - `sqlfile` means that DDL statements from the objects in the export dump file are stored in the text file `sqlfile.sql` in the working directory. No data is imported. Only supported for impdp.**Default:** full
- `-z|-filesize` specifies the dump file size for imp  
**Syntax:** `-z|-filesize <size>`

<size> specifies the maximal dump file size in MB

Default: value from BRSPACE export

## **-f tbalter**

This BRSPACE [function](#) alters tables. For more information, see [Altering a Table with BR\\*Tools](#).

Function options:

- **-a|-action:** specifies the action to alter tables

Syntax: `-a|-action monit|nomonit|parallel|shrink`

- `monit` sets the monitoring attribute on
- `nomonit` sets the monitoring attribute off
- `parallel` sets the degree of parallelism for queries
- `shrink` shrinks table segments online (Oracle 10g or higher)

Default: none, since you must always specify the action

- **-d|-degree:** specifies the degree for parallel queries

Syntax: `-d|-degree <degree>`

1 specifies normal serial processing

>1 specifies parallel query

Default: 0 (use Oracle default)

- **-o|-owner:** defines the SAP owner of the tables to be altered - useful in Multiple Components in One Database (MCOB) configurations

Syntax: `-o|-owner <owner>|<owner_list>`

`<owner>` or `<owner_list>` specifies the name of the SAP owner or owners

Default: no default in MCOB, otherwise the single SAP owner

- **-s|-tablespace:** specifies the tablespace of the tables to be altered

Syntax: `-s|-tablespace <tablespace>|<tablespace list>`

`<tablespace>` or `<tablespace list>` specifies the tablespace name(s)

Default: none

- **-t|-table:** specifies the tables to be altered

Syntax: `-t|-table all|allsel|[<owner>.]<table>|`

`[<owner>.] [<prefix>] % [<suffix>] |`

`[<owner>.] [<prefix>] * [<suffix>] |<table_list>`

where:

all or “%” means preselect

allsel or “\*” means final select:

For more information about wildcards for preselect and select, see *Selecting Objects* in [Segment Management with BR\\*Tools](#).

Default: none, since you must always specify a table to alter

## -f idalter

### Procedure

This BRSPACE [function](#) alters indexes. For more information, see [Altering an Index with BR\\*Tools](#).

Function options:

- -a|-action: specifies the action to alter indexes

Syntax: -a|-action coalesce|parallel|shrink

- coalesce merges internal index data
- parallel sets the degree of parallelism for queries
- shrink shrinks the index segments online (Oracle 10g or higher)

Default: none, since you must always specify the action

- -d|-degree: specifies the degree for parallel queries

Syntax: -d|-degree <degree>

1 specifies normal serial processing

>1 specifies parallel query

Default: 0 (use Oracle default)

- -i|-index: specifies the indexes to be altered

Syntax:

Syntax: -i|-index all|allsel|[<owner>.]<index>|

[<owner>.] [<prefix>] % [<suffix>] |

[<owner>.] [<prefix>] \* [<suffix>] | <index\_list>

where:

all or “%” means preselect

allsel or “\*” means final select:

For more information about wildcards, see *Selecting Objects* in [Segment Management with BR\\*Tools](#).

Default: none, since you must always specify an index to alter



- `-o|-owner`: defines the SAP owner of the indexes to be altered - useful in Multiple Components in One Database (MCOB) configurations

Syntax: `-o|-owner <owner>|<owner list>`

`<owner>` or `<owner list>` specifies the name of the SAP owner

Default: no default in MCOB, otherwise the single SAP owner

- `-s|-tablespace`: specifies the tablespace of the indexes to be altered

Syntax: `-s|-tablespace <tablespace>|<tablespace list>`

`<tablespace>` or `<tablespace list>` specifies the tablespace name(s)

Default: none

- `-t|-table`: specifies the table of the index(es) to be altered

Syntax:

Syntax: `-t|-table all|allsel|[<owner>.]<table>|`

`[<owner>.] [<prefix>] % [<suffix>] |`

`[<owner>.] [<prefix>] * [<suffix>] | <table_list>`

where:

`all` or `"%"` means preselect

`allsel` or `"**"` means here also preselect:

For more information about wildcards for preselect and select, see *Selecting Objects* in [Segment Management with BR\\*Tools](#).

Default: none

## -f mstats

This BRSPACE [function](#) lets you manage database statistics.

Function options:

- `-a|-action`: specifies the type of action to manage the database statistics.

Input syntax: `-a|-action`

`delete|delexp|export|import|lock|restore|showexp|showver|unlock`

- `delete` deletes statistics of selected tables
- `delexp` deletes selected exports of table statistics
- `export` exports statistics of the selected tables to the schema-related STATTAB table. You must specify a statistics export ID to make it easier to identify the export. The only restriction on this ID is that it must be 20 characters or less. You specify the ID in the option `-i|-expid`, as described below.

Example

For an InfoCube statistics export, you might specify the ID as follows:

```
-f mstats -a export -i "InfoCubes Stats"
```

- o `import` imports statistics of selected tables from the schema-related table `STATTAB`
- o `lock` locks statistics of selected tables
- o `restore` restores statistics of selected tables to a previous version saved by Oracle
- o `showexp` displays the exports of table statistics created by BRSPACE
- o `showver` displays the previous versions of table statistics saved by Oracle
- o `unlock` unlocks statistics of selected tables

Default: none, since you must always specify the action

- `-i|-expid`: statistics export ID

Default: last statistics export for action import, otherwise none

- `-m|-time <yyyy-mm-dd hh.mi.ss>`: time stamp for restore statistics

Default: last version saved by Oracle

- `-o|-owner <owner>|<owner_list>`: database owner of tables

Default: none

- `-r|-exprun <exp_run_log>`: statistics export run log name

Default: last BRSPACE statistics export

- `-s|-tablespace <tablespace>|<tablespace_list>`: tablespace name

Default: none

- `-t|-table all|allsel|[<owner>.]<table>|`  
`[<owner>.] [<prefix>] % [<suffix>] |`  
`[<owner>.] [<prefix>] * [<suffix>] | <table_list>`

where:

`all` means preselection of all tables

`allsel` means selection of all tables

Wildcards: “%” and “\*” mean preselect and select

For more information about wildcards for preselect and select, see *Selecting Objects* in [Segment Management with BR\\*Tools](#).

Default: none

## -f mfbck

This BRSPACE [function](#) lets you manage the flashback database feature.

Function options:

- `-a|-action`: specifies the type of manage action for flashback database.

Input syntax: `-a|-action fbon|fboff|rpcreate|rpdrop|fbshow|rpshow`

- `fbon` switches on flashback database
- `fboff` switches on flashback database
- `rpcreate` creates restore point
- `rpdrop` drops restore point
- `fbshow` shows flashback status
- `rpshow` shows restore points

Default: none, since you must always specify the action

- `-f|-force`: forces database shutdown for `fbon` and creates a guaranteed restore point if flashback database is not on

Default: do not shut down if SAP user still connected

- `-g|-guarantee`: specifies guaranteed restore point

Default: normal restore point

- `-p|-point <name>|last`: specifies restore point:

- `<name>`: specifies restore point name
- `last`: drops or shows last restore point

Default: none

- `-r|-retention <minutes>`: specifies flashback retention time

0 means unlimited retention time

Default: 1440

## -h|-help

This BRSPACE command option provides help information including an overview of BRSPACE functions..

Syntax

```
-h|-help [<function>|version]
```

End of the code.

Default value: display help information about all BRSPACE functions.

Possible values:

- `<function>`: displays help information about main options and specified function only
- `version`: displays detailed information on the versions of the program modules

## -l|-language

This BRSPACE command option sets the message language.

Syntax

```
-l|-language E|D
```

End of the code.

Default value: E

Note

The default becomes invalid if you specify another value by setting the environment variable `BR_LANG` (language variable).

If you set option `-l`, the value specified with this option applies.

Possible values:

- D: German
- E: English

## -o|-output

This BRSPACE command option controls the information written to the detail log.

Syntax

```
-o|-output [detail|process|summary|[,]time]
```

End of the code.

Default value: process

- `detail`: writes detailed processing information with object details, resulting in comprehensive information that you can use to troubleshoot
- `process`: writes information from the processing phase (not the initialization phase)
- `summary`: writes function summaries and totals
- `time`: generates additional timestamps that enable you to determine the time required for the individual operations

You must insert this option *before* the option `-f|-function`.

Example

```
brspace -u / -c force -o -f tbreorg -t "DBA*"
```

This option only affects BRSPACE functions in segment administration, such as [tbreorg](#) and [idrebuild](#).

## **-p|-profile**

This BRSPACE command option defines the profile name.

This profile is contained in directory <ORACLE\_HOME>/dbs (UNIX) or <ORACLE\_HOME>\database (Windows).

If you want to use a different profile, specify the name of the profile file here. If this file is not in the standard directory <ORACLE\_HOME>/dbs , specify the complete path.

Syntax

```
-p|-profile <profile>
```

End of the code.

Default value: `init<DBSID>.sap`

## **-q|-query**

This BRSPACE command option lets you call the BRSPACE function menus without starting processing.

Syntax

```
-q|-query [nolog]
```

End of the code.

`nolog`: does *not* create or update detail, summary, and database logs for the function

Example

```
brspace -u / -q nolog -f tsextent
```

## **-s|-scroll**

This BRSPACE command option specifies the number of lines for scrolling in list menus. This option is *not* valid for BRGUI.

Syntax

```
-s|-scroll <lines>
```

End of the code.

Default value: 20

This option corresponds to the parameter [scroll\\_lines](#) in `init<DBSID>.sap`.

## **-u|-user**

This BRSPACE command option defines the user name and password used by the SAP tool to log on to the database. Since BRSPACE connects to the database as SYSDBA, you do not have to specify the user and password if the operating system user belongs to the `dba` group.

## Syntax

```
-u|-user [<user>[/<password>] | /]
```

End of the code.

Default value: `system/<default_password>`

If you only enter `-u`, BRSPACE performs an interactive query of the user name and the password. You can enter the user name and the password separately (only enter the user name or the option `-u <user>`). BRSPACE then prompts entry of the password. In this case, the password is not displayed during entry, and does not appear in the process list.

This protects the DBA password.

In shell scripts, you can structure the call as follows:

```
brspace -c force -u -f dbshut <<END <user>/<password> END
```

However, use this command only if the option `-c` is active.

## Note

If you are working with an `OPSS$` user, use the following call:

```
brspace -u / -c -f tsextend
```

In this case, BRSPACE tries to log on to the database as `OPSS$` user (see Oracle documentation and information in the SAP Service Marketplace). The `OPSS$` user must be defined in the database and have at least `SYSDBA` authorization. With this method, it is not necessary to specify the password when calling BRSPACE.

For setups with Oracle Real Application Cluster (RAC), you normally have to specify the database user and password. BRSPACE needs this to remotely manage the database instances.

## -V|-VERSION

This BRSPACE command option displays patch information for BRSPACE.

## Syntax

```
-V|-VERSION [ALL]
```

End of the code.

`ALL`: displays patch information for all BR\*Tools

## BRSPACE Logs

BRSPACE writes a series of logs to record what happens during processing:

- [BRSPACE Detail Log](#)
- [BRSPACE Summary Log](#)
- [BRSPACE Structure Change Log](#)
- [BRSPACE Parameter Change Log](#)

# BRSPACE Detail Log

The detail log file contains full information about what happened during the BRSPACE function.

The file displays information about the:

- Relevant parameters from the [initialization profile init<DBSID>.sap](#) that were set during the BRSPACE run
- Function name
- Menus that were displayed and the options that you chose
- BRSPACE commands and results

## Structure

BRSPACE detail logs have names of the following form:

s<encoded timestamp>.<ext>

The logs are stored in the `sapreorg` directory.

The name consists of:

- Action ID  
This consists of the fixed character `s` and the `<encoded time>` that the function was performed.
- Function ID

The suffix `<ext>` indicates the function type:

- `dbr`: start up database
- `dbsh`: shut down database
- `dba`: alter database instance
- `dbp`: alter database parameter
- `dbc`: recreate database
- `dbw`: show database information (default)
- `tse`: extend tablespace
- `tsc`: create tablespace
- `tsd`: drop tablespace
- `tsc`: create tablespace
- `tsc`: create tablespace
- `tsa`: alter tablespace
- `dfa`: alter data file
- `dfm`: move data file

- o tbr: reorganize tables
- o idr: rebuild indexes
- o tbe: export tables
- o tbi: import tables
- o tba: alter tables
- o ida: alter indexes
- o mst: manage database statistics
- o mfb: manage flashback database

## Example

This is an example of the start of a BRSPACE detail log for the function show database information:

```
BR1001I BRSPACE 6.40 (0)
BR1002I Start of BRSPACE processing: sdljsvqc.dbw 2003-08-26 10.23.50
BR0101I Parameters
Name Value
oracle_sid GC2
oracle_home /oracle/GC2
oracle_profile /oracle/GC2/dbs/initGC2.ora
sapdata_home /oracle/GC2
sap_profile /oracle/GC2/dbs/initGC2.sap
space_function dbshow
space_copy_dir /oracle/GC2/sapreorg
scroll_lines 20
system_info oragc2/oragc2 uw1030 SunOS 5.8 Generic_108528-15 sun4u
oracle_info GC2 9.2.0.1.0 8192 8856 17005445
sap_info 620 SAPR3
make_info sun OCI_901 Aug 26 2003
command_line brspace -f dbshow -c dbparam -l
BR0280I BRSPACE time stamp: 2003-08-26 10.23.50
BR1009I Name of database instance: GC2
BR1010I BRSPACE action ID: sdljsvqc
BR1011I BRSPACE function ID: dbw
BR1012I BRSPACE function: dbshow
```



BR1034I Class of information to be shown: dbparam

BR0280I BRSPACE time stamp: 2003-08-26 10.23.51

BR0659I List menu 257 + you can select one or more entries

-----

List of database parameters

Pos. Parameter Modif. Inst. Value

- 1 - active\_instance\_count spfile \* <null>
- 2 - aq\_tm\_processes both \* 1
- 3 - archive\_lag\_target both \* 0
- 4 - audit\_file\_dest spfile \* ?/saptrace/audit
- 5 - audit\_sys\_operations spfile \* FALSE
- 6 - audit\_trail spfile \* NONE
- 7 - background\_core\_dump spfile \* partial
- 8 - background\_dump\_dest both \* /oracle/GC2/saptrace/bdump
- 9 - backup\_tape\_io\_slaves both \* FALSE
- 10 - bitmap\_merge\_area\_size spfile \* 1048576

## BRSPACE Summary Log

Each executed BRSPACE function has an entry in the summary log space<DBSID>.log.

### Structure

The entries in the file provide the following information about each executed run of a BRSPACE function:

| Entry                                 | Example             |
|---------------------------------------|---------------------|
| BRSPACE action id (encoded timestamp) | sdkwjpbg            |
| BRSPACE function id                   | tbe                 |
| Timestamp for start of BRSPACE run    | 2003-08-28 18:05:24 |
| Timestamp for end of BRSPACE run      | 2003-08-28 18:05:24 |
| Return code                           | 0 (successful)      |
| BRSPACE version                       | 6.40 (0)            |
| Function                              | tbexport            |
| Number of objects successfully        | 4                   |

| Entry                                           | Example              |
|-------------------------------------------------|----------------------|
| processed                                       |                      |
| The following only applies to tablespace export |                      |
| Max size of export dump file (MB)               | 20000                |
| Number of export dump files                     | 1                    |
| Total size of export dump file (B)              | 425984               |
| Export dump directory                           | /oracle/GC2/sapreorg |

#### Note

The combination of action id and function id - in the above, example, sdkwipbg.tbe - represents the name of the BRSPACE [detail log](#).

## Example

This is an example of a BRSPACE summary log:

```

sdkxdoyh idr 2003-06-20 20.39.19 2003-06-20 20.39.42 0 6.40 (0)
idrebuild

sdkxdpvr tba 2003-06-20 20.49.27 2003-06-20 20.52.56 4 6.40 (0)
tbalter

sdkxdqes tba 2003-06-20 20.53.22 2003-06-20 20.54.00 0 6.40 (0)
tbalter

sdkxdqhu ida 2003-06-20 20.54.42 0000-00-00 00.00.00 6 6.40 (0)
idalter

sdkxrftz dbr 2003-06-23 15.17.07 2003-06-23 15.17.30 0 6.40 (0)
dbstart

sdkxdpaq tbe 2003-06-24 20.40.20 2003-06-24 20.41.14 1 6.40 (0)
tbexport 4 20000 1 425984 /oracle/GC2/sapreorg

sdkxdpdd tbi 2003-06-24 20.41.25 2003-06-24 20.44.22 1 6.40 (0)
tbimport 4

```

## BRSPACE Structure Change Log

The structure change log contains a history of all database structure changes that you perform with BRSPACE. Structure changes consist of the following functions:

- Extend tablespace
- Create tablespace
- Drop tablespace

- Alter datafile (rename, drop, change file size attributes)
- Rename tablespace
- Move datafile

The file helps you to keep track of database structure changes.

## Structure

The file is called `struc<DBSID>.log` and is located in the `$SAPDATA_HOME/sapreorg` directory.

The first line is standard but the following lines vary according to the function.

- Standard first line

| Line        | Entry                                                                      | Example                |
|-------------|----------------------------------------------------------------------------|------------------------|
| #BRSRU<br>N | Timestamp (date, time) for when you start BRSPACE for the structure change | 2003-08-01<br>14.47.18 |
|             | BRSPACE action id (encoded timespace)                                      | sdlfavte               |
|             | BRSPACE function ID                                                        | tse                    |
|             | Function name                                                              | tsextend               |
|             | Effective and real user name                                               | user=oragc2/oragc2     |

- Note
- The combination of action id and function id – in the above example, `sdlfavte.tse` – represents the name of the BRSPACE [detail log](#).
- 
- Extend tablespace

| Line         | Entry                                                         | Example                                        |
|--------------|---------------------------------------------------------------|------------------------------------------------|
| #TSPEXT      | Timestamp (date, time) for when the structure change finished | 2003-08-01 14.47.27                            |
|              | Tablespace name                                               | PSAP2222                                       |
| #FILEAD<br>D | New datafile name                                             | /oracle/GC2/sapdata4/2222_6/2222.data6         |
|              | Parameters for new datafile                                   | size=1M autoext=yes maxsize=10M<br>incrsize=1M |

- Create tablespace

| Line | Entry | Example |
|------|-------|---------|
|------|-------|---------|

| Line     | Entry                                                         | Example                                  |
|----------|---------------------------------------------------------------|------------------------------------------|
| #TSPCRT  | Timestamp (date, time) for when the structure change finished | 2003-08-01 14.50.35                      |
|          | Tablespace name                                               | PSAP3333D                                |
|          | Parameters for new tablespace                                 | conts=data assm=auto data=table          |
| #FILEADD | New datafile name                                             | /oracle/GC2/sapdata4/3333d/3333d.data1   |
|          | Parameters for new datafile                                   | size=1M autoext=yes maxsize=10M incrs=1M |

- Drop tablespace

|             |                                                               |                                          |
|-------------|---------------------------------------------------------------|------------------------------------------|
| #TSPDROP    | Timestamp (date, time) for when the structure change finished | 2003-08-01 14.52.41                      |
|             | Tablespace name                                               | PSAP4444I                                |
|             | Attributes of deleted tablespace                              | files=2 size=2M                          |
| #FILEDELETE | Deleted datafile name                                         | /oracle/GC2/sapdata4/4444i_1/4444i.data1 |
|             | Datafile parameters                                           | size=1M                                  |

- Rename tablespace

|            |                                                               |                     |
|------------|---------------------------------------------------------------|---------------------|
| #TSPRENAME | Timestamp (date, time) for when the structure change finished | 2003-08-01 14.52.41 |
|            | Old tablespace name                                           | PSAP4444I           |
|            | New tablespace name                                           | newname=PSAP5555I   |

- Alter datafile

- o Switch on autoextend

|           |                     |                                          |
|-----------|---------------------|------------------------------------------|
| #FILEAUTO | Datafile name       | /oracle/GC2/sapdata4/2222_1/2222.data1   |
|           | Datafile parameters | size=1M autoext=yes maxsize=22M incrs=2M |

- o Switch off autoextend

|          |               |                                          |
|----------|---------------|------------------------------------------|
| #FILEFIX | Datafile name | /oracle/GC2/sapdata1/testd_5/ttttd.data5 |
|----------|---------------|------------------------------------------|

|  |                                             |                    |
|--|---------------------------------------------|--------------------|
|  | Parameters to specify the affected datafile | size=2M autoext=no |
|--|---------------------------------------------|--------------------|

- Resize datafile

|           |                                             |                                          |
|-----------|---------------------------------------------|------------------------------------------|
|           | Datafile name                               | /oracle/GC2/sapdata6/testd_1/testd.data1 |
| #FILESIZE | Parameters for new and old size of datafile | size=8M oldsize=4M                       |

- Move datafile

|           |                                     |                                                  |
|-----------|-------------------------------------|--------------------------------------------------|
| #TSP      | Timestamp (date, time) for the move | 2003-08-01 17.55.14                              |
|           | Tablespace                          | PSAP1111D                                        |
| #FILEMOVE | New datafile name                   | /oracle/GC2/sapdata5/1111d_2/1111d.data2         |
|           | Old datafile name                   | oldfile=/oracle/GC2/sapdata2/1111d_3/1111d.data3 |
|           | Datafile size                       | size=3M                                          |

## Example

This is an example of part of a structure change log showing each of the functions described above:

```
#BRSRUN... 2003-08-01 14.47.18 sdlfavte tse func=tsextend
user=oragc2/oragc2

#TSPEXT... 2003-08-01 14.47.27 PSAP2222

#FILEADD.. /oracle/GC2/sapdata4/2222_6/2222.data6 size=1M autoext=yes
maxsize=10M incrsiz=1M

#

#BRSRUN... 2003-08-01 14.48.31 sdlfavvz tsc func=tscreate
user=oragc2/oragc2

#TSPCRT... 2003-08-01 14.50.35 PSAP3333D conts=data assm=auto
data=table

#FILEADD.. /oracle/GC2/sapdata4/3333d_1/3333d.data1 size=1M
autoext=yes maxsize=10M incrsiz=1M
```

```

#
#BRSRUN... 2003-08-01 14.52.29 sdlfawfd tsd func=tsdrop
user=oragc2/oragc2
#TSPDROP.. 2003-08-01 14.52.41 PSAP4444I files=2 size=2M
#FILEDEL.. /oracle/GC2/sapdata4/4444i_1/4444i.data1 size=1M
#FILEDEL.. /oracle/GC2/sapdata4/4444i_2/4444i.data2 size=1M
#
#BRSRUN... 2003-08-01 16.37.36 sdlfbfns dfa func=dfalter
user=oragc2/oragc2
#FILEAUTO. /oracle/GC2/sapdata4/2222_1/2222.data1 size=2M autoext=yes
maxsize=22M incrsiz=2M
#FILEAUTO. /oracle/GC2/sapdata4/2222_2/2222.data2 size=1M autoext=yes
maxsize=22M incrsiz=2M
#
#BRSRUN... 2003-08-01 16.48.18 sdlfbgmk dfa func=dfalter
user=oragc2/oragc2
#FILEFIX.. /oracle/GC2/sapdata1/testd_5/ttttd.data5 size=2M
autoext=no
#FILEFIX.. /oracle/GC2/sapdata3/ttttd_1/ttttd.data1 size=4M
autoext=no
#
#BRSRUN... 2003-08-01 21.06.20 sdlfcdjw dfa func=dfalter
user=oragc2/oragc2
#TSP..... 2003-08-01 21.06.49 PSAPTESTD
#FILESIZE. /oracle/GC2/sapdata6/testd_1/testd.data1 size=8M
oldsize=4M
#FILESIZE. /oracle/GC2/sapdata6/testd_2/testd.data2 size=8M
oldsize=4M
#
#BRSRUN... 2003-08-01 17.54.03 sdlfbmid dfm func=dfmove
user=oragc2/oragc2
#TSP..... 2003-08-01 17.55.14 PSAP1111D
#FILEMOVE. /oracle/GC2/sapdata5/1111d_2/1111d.data2
oldfile=/oracle/GC2/sapdata2/1111d_2/1111d.data2 size=3M
#FILEMOVE. /oracle/GC2/sapdata5/1111d_3/1111d.data3
oldfile=/oracle/GC2/sapdata2/1111d_3/1111d.data3 size=3M

```

## BRSPACE Parameter Change Log

The parameter change log contains a history of all database parameter changes that you perform with BRSPACE.

The file helps you to keep track of database parameter changes.

## Structure

The file is called `param<DBSID>.log` and is located in the `$SAPDATA_HOME/sapreorg` directory. It has the following structure:

| Line                                 | Entry                                                                      | Example                                                            |
|--------------------------------------|----------------------------------------------------------------------------|--------------------------------------------------------------------|
| #BRSRUN                              | Timestamp (date, time) for when you start BRSPACE for the parameter change | 2003-08-01 10.36.14                                                |
|                                      | BRSPACE action id (encoded timespace)                                      | sdlezzlu                                                           |
|                                      | BRSPACE function ID                                                        | dbp                                                                |
|                                      | Function name (always dbparam)                                             | dbparam                                                            |
|                                      | Effective and real user name                                               | user=oragc2/oragc2                                                 |
| #PARCHNG<br>– change parameter value | Timestamp (date, time) for when BRSPACE completes the parameter change     | 2003-08-01 10.36.14                                                |
|                                      | Name of parameter changed                                                  | control_file_record_keep_time                                      |
|                                      | Scope of change: in memory, spfile, or both                                | scope=both                                                         |
| #PARRST<br>– reset parameter value   | Timestamp (date, time) for when BRSPACE completes the parameter reset      | 2003-08-01 11.03.10                                                |
|                                      | Name of parameter changed                                                  | control_file_record_keep_time                                      |
|                                      | Scope – in memory, spfile, or both                                         | scope=both                                                         |
| #VALUE                               | New parameter value                                                        | 30                                                                 |
|                                      | Old value                                                                  | ddval=[10]                                                         |
|                                      | Value in spfile                                                            | spfile=[<same>]                                                    |
|                                      | Note                                                                       | The value [null] means that Oracle now takes the default value for |

| Line               | Entry          | Example                 |
|--------------------|----------------|-------------------------|
|                    | the parameter. |                         |
| #COMMENT(optional) | Your comment   | For incremental backups |

#### Note

The combination of action id and function id - in the above example, `sdlezzlu.dbp` - represents the name of the BRSPACE [detail log](#).

## Example

This is an example of part of a parameter change log:

```
#BRSRUN... 2003-08-01 10.36.14 sdlezzlu dbp func=dbparam
user=oragc2/oragc2

#PARCHNG.. 2003-08-01 10.37.09 control_file_record_keep_time
scope=both

#VALUE.... [32] oldval=[30] spfval=[<same>]

#

#BRSRUN... 2003-08-01 10.37.41 sdlezzpd dbp func=dbparam
user=oragc2/oragc2

#PARRST... 2003-08-01 11.03.10 control_file_record_keep_time
scope=both

#VALUE.... [null] oldval=[32] spfval=[<same>]

#

#BRSRUN... 2003-08-01 11.03.35 sdlfabwx dbp func=dbparam
user=oragc2/oragc2

#PARRST... 2003-08-01 11.05.30 control_file_record_keep_time
scope=both

#VALUE.... [null] oldval=[32]

#

#BRSRUN... 2003-08-01 11.35.07 sdlfaerr dbp func=dbparam
ser=oragc2/oragc2

#PARCHNG.. 2003-08-01 11.35.39 control_file_record_keep_time
scope=both

#VALUE.... [33] oldval=[32]

#

#BRSRUN... 2003-08-01 11.36.13 sdlfaeuf dbp func=dbparam
ser=oragc2/oragc2
```



```

#PARRST... 2003-08-01 11.38.26 control_file_record_keep_time
scope=both

#VALUE.... [null] oldval=[33] spfval=[<same>]

#

#BRSRUN... 2003-08-01 14.28.33 sdlfaubx dbp func=dbparam
user=oragc2/oragc2

#PARCHNG.. 2003-08-01 14.29.15 control_file_record_keep_time
scope=both

#VALUE.... [30] oldval=[7]

#COMMENT.. Activate incremental backups

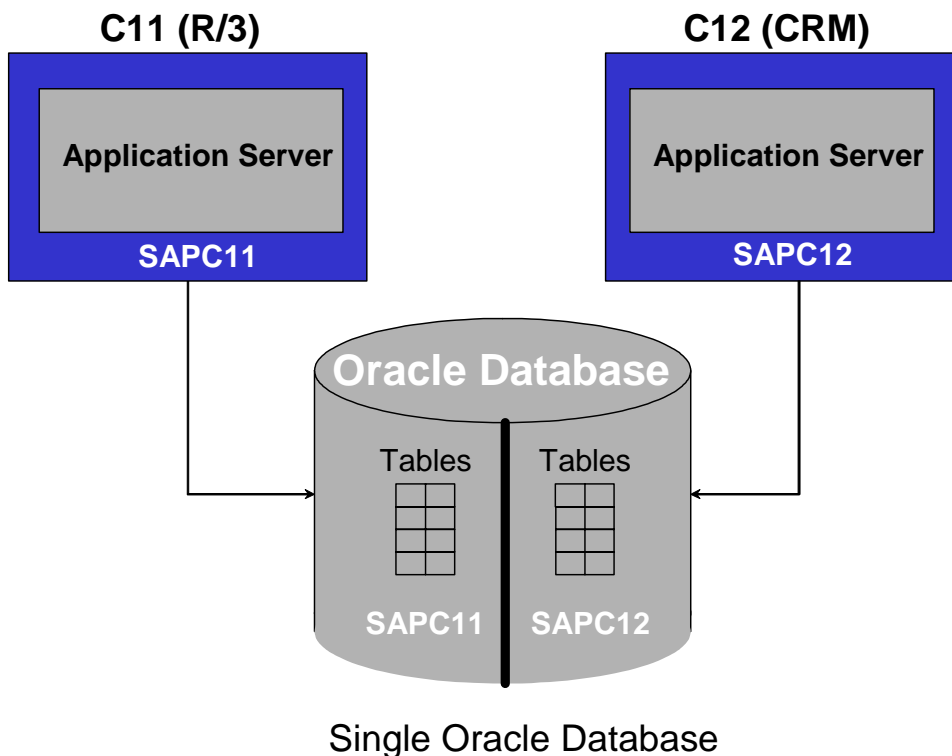
```

## BRCONNECT

The SAP tool BRCONNECT for Oracle databases is used as:

- A database administration tool, which you can call yourself from the command line
- A utility tool, which [BRBACKUP](#) calls in the background

BRCONNECT is specially designed to administer multi-schema databases, in which you have more than one SAP system in the same Oracle database:



## Features

- Database administration tool

As a database administration tool, BRCONNECT has the following [functions](#):

- Main functions, which write a [detailed](#) and a [summary](#) log
  - [Check database system, -f check](#)
  - [Adapt next extents, -f next](#)
  - [Update statistics, -f stats](#)
  - [Clean up old logs and trace files, -f cleanup](#)
- [Additional functions](#), which only write messages to the standard output device
  - [Change passwords of SAP database users, -f chpass](#)
  - Create global synonyms, [-f crsyn](#) – these are used by BR\*Tools
  - Start database, [-f dbstart](#)
  - Stop database, [-f dbshut](#)
  - Determine database state, [-f dbstate](#)
- Utility tool - monitor database status during a backup

BRBACKUP starts BRCONNECT during the backup to see if the status of the database corresponds to the backup mode. If you select backup mode online (`backup_type = online`), the database remains in this state during the backup.

If you select backup mode offline (`backup_type = offline` or `backup_type = offline_force`), the database is shut down and remains in the closed state during the backup.

If the state of the database changes unexpectedly during the backup, BRCONNECT terminates the backup and displays the messages BR0312E or BR0313E. After the backup, the database is always restored to its original status. This means that the database is left started, if it was running before the backup, or it is shut down, if it was shut down before the backup.

BRCONNECT has many parameters, which you can specify in the [Initialization Profile](#) `init<DBSID>.sap`.

## Activities

For more information about using BRCONNECT from the command line, see [Command Options for BRCONNECT](#).

## Database System Check with BRCONNECT

You can use [BRCONNECT](#) to check the Oracle database system. The aim is to prevent database problems that might lead to downtime for the database.

You can use BRCONNECT to check the following conditions:

- Database administration, such as configuration, space management, state of the database, and so on
- Database operations, such as backup results, failed operations, and so on

- Critical database messages in the Oracle alert file, such as `ORA-00600`
- Incorrectly set database profile parameters in the `init<DBSID>.ora` file, such as `log_archive_start = false`

When a critical situation is discovered, BRCONNECT writes an alert message to the [detail log](#) and to the results table DBMSGORA.

## Integration

The check conditions are specified in the control table DBCHECKORA. You can change these with transaction DB17. Detected alerts are also reported to the database monitor (transaction RZ20). For more information, see [Monitoring the Oracle Database](#). You can also view alerts with transaction DB16.

## Prerequisites

BRCONNECT can also use internal default conditions, which mostly correspond to the initial state of the DBCHECKORA table when we deliver it.

Note

For up-to-date information on the BRCONNECT default conditions, see SAP Note 435290.

## BRCONNECT Default Conditions for Database Administration

See [BRCONNECT Default Conditions for Database Administration](#).

## BRCONNECT Default Conditions for Database Operations

See [BRCONNECT Default Conditions for Database Operations](#).

## BRCONNECT Default Conditions for Database Messages

You can enter any Oracle error codes or error text as a condition name for this condition type. BRCONNECT searches the Oracle Alert log for corresponding Oracle error messages and might generate Alert messages. The Oracle error codes are taken into account for the standard test conditions (`-d` option).

See *Oracle Messages* in [Database Health Alerts](#).

Note

You can easily add any Oracle error codes as new test conditions for database messages using transaction DB17. You can also search any texts (character strings) in the Oracle Alert file by specifying the text in the `PARAM` field. Since the search in the Oracle Alert file takes the upper/lower case spelling into account and the entry in the `PARAM` field is always converted into upper case letters using DB17, the entry must be made using SQL with SQLPLUS, as in the following example:

Note

```
INSERT INTO DBCHECKORA (TYPE, PARAM, OBJECT, ACTIVE, SEVERITY, CHKOP,
CHKVAL, UNIT, CHKREP, REPUNIT, MODFLAG, MODDATE, MODUSER, REACTION,
CORRTYPE, CORRNAME, CHKDESC)
```

```
VALUES ('ORA', 'Checkpoint not complete', ' ', 'Y', 'W', ' ', ' ', ' ',
' ', ' ', ' ', ' ', ' ', ' ', ' ', 'D',
```

```
'Increase the size of online redo log files',
'Cannot switch to the next online redo log due to pending
checkpoint');
```

## BRCONNECT Default Conditions for Database Profile Parameters

These test conditions check the values of Oracle parameters. The standard test conditions for the database profile (-d option) correspond to the current SAP recommendations described in SAP Notes [124361](#) and 180605 (SAP BW) for Oracle 9i and SAP Note 830576 for Oracle 10g.

### Note

You can use transaction DB17 to easily adjust the test conditions for the database profile parameters, depending on the changed recommendations and for new Oracle releases.

## Activities

- You run the checks regularly (for example, daily). We recommend you to use the Database Planning Calendar in the SAP System for this. For more information, see [Database System Check](#).
- You use transaction DB16 to view alerts written by BRCONNECT to the results table DBMSGORA. For more information, see [Displaying Alert Messages from Database System Check](#).
- You use transaction DB17 to configure database system check. This includes activating or deactivating check conditions and changing the threshold and severity levels (that is, error, warning, or exception). For more information, see [Configuring Database System Check \(Oracle\)](#).
- You can exclude specified tables or indexes from the checks using the `check_exclude` parameter.

For more information on the command line options for the database checks, see [-f check](#).

## BRCONNECT Default Conditions for Database Administration

The check conditions for database administration are specified in the control table DBCHECKORA. For more information, see [Database System Check with BRCONNECT](#).

| Condition                                      | Severity | Description                                                                                                                                                                                                                                    |
|------------------------------------------------|----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| NOARCHIVELOG_MODE<br>(previously NOARCHIVELOG) | Error    | Checks whether the database is in NOARCHIVELOG mode, which is not allowed for production databases.<br><br>The OBJECT field is not specified for this condition. This condition does not have test operands, threshold values, or value units. |

| Condition                                            | Severity       | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|------------------------------------------------------|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p>ARCHIVER_STUCK<br/>(previously ARCHIVE_STUCK)</p> | <p>Warning</p> | <p>Checks the highest fill level of the archiving directory (by default oraarch).</p> <p>The OBJECT field is not specified for this condition.</p> <p>Possible test operands, threshold values, value units:</p> <ul style="list-style-type: none"> <li>• &gt;, &gt;= &lt;number&gt;</li> </ul> <p>The space used is larger than (or the same as) &lt;number&gt; % of the total space in the archiving directory.</p> <ul style="list-style-type: none"> <li>• &gt;, &gt;= &lt;number&gt;</li> </ul> <p>The space used is larger than (or the same as) &lt;number&gt; K M G bytes.</p> <ul style="list-style-type: none"> <li>• &lt;, &lt;= &lt;number&gt;</li> </ul> <p>The free space is smaller than (or the same as) &lt;number&gt; % of the total space in the archiving directory.</p> <ul style="list-style-type: none"> <li>• &lt;, &lt;= &lt;number&gt;</li> </ul> <p>The free space is smaller than (or the same as) &lt;number&gt; K M G bytes.</p> |
| <p>FILE_SYSTEM_FULL<br/>(previously FS_FULL)</p>     | <p>Warning</p> | <p>Checks the fill level of file systems on the database host. All file systems are checked by default against the same threshold value. However, you can define different threshold values for individual file systems by specifying the file system in the OBJECT field of the DBCHECKORA table (transaction DB17). Here you can use the following keywords for the database file systems:</p> <p>ORACLE_HOME, SAPDATA_HOME, SAPDATA1, SAPDATA2, ..., SAPARCH, ORIGLOG, MIRRLOG, SAPBACKUP, SAPCHECK, SAPREORG and SAPTRACE.</p> <p>Alternatively, you can specify any other file systems by entering the full path of a directory from this file system into the field (you can use upper/lower case spelling in DB17 only as of the Service</p>                                                                                                                                                                                                            |

| Condition                                                    | Severity       | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|--------------------------------------------------------------|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                              |                | <p>Packages mentioned in note 427673, otherwise you have to use native SQL).</p> <p>Possible test operands, threshold values, value units:</p> <ul style="list-style-type: none"> <li>• &gt;, &gt;= &lt;number&gt;</li> <p style="margin-left: 40px;">The space used is larger than (or the same as) &lt;number&gt; % of the total space in the file system.</p> <li>• &gt;, &gt;= &lt;number&gt;</li> <p style="margin-left: 40px;">The space used is larger than (or the same as) &lt;number&gt; K M G bytes.</p> <li>• &lt;, &lt;= &lt;number&gt;</li> <p style="margin-left: 40px;">The free space is less than (or the same as) &lt;number&gt; % of the total space in the file system.</p> <li>• &lt;, &lt;= &lt;number&gt;</li> <p style="margin-left: 40px;">The free space is smaller than (or the same as) &lt;number&gt; K M G bytes.</p> </ul> |
| <p>TABLESPACE_OFFLINE<br/>(previously TSP_OFFLINE)</p>       | <p>Error</p>   | <p>Checks whether there are tablespaces that are offline.</p> <p>The OBJECT field is not specified for this condition. This condition does not have test operands, threshold values, or value units.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| <p>TABLESPACE_IN_BACKUP<br/>(previously TSP_BACKUP_MODE)</p> | <p>Warning</p> | <p>Checks whether there are tablespaces that have the BACKUP status although BRBACKUP is not active</p> <p>The OBJECT field is not specified for this condition. This condition does not have test operands, threshold values, or value units.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <p>TABLESPACE_FULL<br/>(previously TSP_FULL)</p>             | <p>Warning</p> | <p>Checks the maximum level of tablespaces in the database. All tablespaces are checked by default against the same threshold value. However, you can define different threshold values for individual tablespaces by specifying the tablespace name in the OBJECT field of the DBCHECKORA table (transaction DB17).</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

| Condition                                                         | Severity       | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-------------------------------------------------------------------|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                                   |                | <p>Possible test operands, threshold values, value units:</p> <ul style="list-style-type: none"> <li>• &gt;, &gt;= &lt;number&gt;</li> </ul> <p>The space used is larger than (or the same as) &lt;number&gt; % of the total space in the tablespace.</p> <ul style="list-style-type: none"> <li>• &gt;, &gt;= &lt;number&gt;</li> </ul> <p>The space used is larger than (or the same as) &lt;test&gt; K M G bytes.</p> <ul style="list-style-type: none"> <li>• &lt;, &lt;= &lt;number&gt;</li> </ul> <p>The free space is less than (or the same as) &lt;number&gt; % of the total space in the tablespace.</p> <ul style="list-style-type: none"> <li>• &lt;, &lt;= &lt;number&gt;</li> </ul> <p>The free space is smaller than (or the same as) &lt;number&gt; K M G bytes.</p> |
| <p>DATA_FILE_MISSING<br/>(previously FILE_MISSING)</p>            | <p>Warning</p> | <p>Checks whether there are data files that no longer exist in the file system.</p> <p>The OBJECT field is not specified for this condition. This condition does not have test operands, threshold values, or value units.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <p>REDOLOG_FILE_MISSING<br/>(previously REDOLOG_MISSING)</p>      | <p>Error</p>   | <p>Checks whether there are online redo log files that no longer exist in the file system.</p> <p>The OBJECT field is not specified for this condition. This condition does not have test operands, threshold values, or value units.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| <p>CONTROL_FILE_MISSING<br/>(previously CONTROL_FILE_MISSING)</p> | <p>Error</p>   | <p>Checks whether there are control files that no longer exist in the file system.</p> <p>The OBJECT field is not specified for this condition. This condition does not have test operands, threshold values, or value units.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <p>DATA_FILE_MISMATCH<br/>(previously FILE_MISMATCH)</p>          | <p>Error</p>   | <p>Checks whether there are data files that are flagged as MISSING in Oracle control file.</p> <p>The OBJECT field is not specified for this condition. This condition does not have</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |

| Condition                                           | Severity | Description                                                                                                                                                                                                                                                                                                                                                                     |
|-----------------------------------------------------|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                     |          | test operands, threshold values, or value units.                                                                                                                                                                                                                                                                                                                                |
| INVALID_FILE_TYPE<br>(previously FILE_TYPE_UNKNOWN) | Error    | <p>Checks whether there are database files that have an illegal operating system type, for example, block raw files on Unix or compressed files on Windows.</p> <p>The OBJECT field is not specified for this condition.</p> <p>This condition does not have test operands, threshold values, or value units.</p>                                                               |
| REDOLOG_FILE_MIRROR<br>(previously REDOLOG_MIRROR)  | Error    | <p>Checks whether there are online redo log files that are not mirrored on the Oracle side.</p> <p>The OBJECT field is not specified for this condition. This condition does not have test operands, threshold values, or value units.</p>                                                                                                                                      |
| CONTROL_FILE_MIRROR<br>(previously CONTROL_MIRROR)  | Error    | <p>Checks whether there are control files that are not mirrored on the Oracle side.</p> <p>The OBJECT field is not specified for this condition. This condition does not have test operands, threshold values, or value units.</p>                                                                                                                                              |
| FILE_OFFLINE<br>(previously DF_OFFLINE)             | Error    | <p>Checks whether there are data files or online redo log files that are OFFLINE.</p> <p>The OBJECT field is not specified for this condition. This condition does not have test operands, threshold values, or value units.</p>                                                                                                                                                |
| CRITICAL_FILE                                       | Warning  | <p>Examines the data files with an activated autoextend feature. A check is made to see whether the file system might overflow due to the existing parameter setting (MAXSIZE and INCRSIZE) during the automatic file extension.</p> <p>The OBJECT field is not specified for this condition. This condition does not have test operands, threshold values, or value units.</p> |
| CRITICAL_TABLESPACE                                 | Warning  | <p>Enhances the CRITICAL_FILE check (see above) if there are several autoextend files of a tablespace on the same disk (that is, logical volume) by</p>                                                                                                                                                                                                                         |



| Condition                                                                        | Severity       | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------------------------------------------------------------------------|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                                                  |                | <p>checking whether there is sufficient free space on the hard disk to extend these files to the maximum size (<code>max_extn</code>).</p> <p>The <code>OBJECT</code> field is not specified for this condition. This condition does not have test operands, threshold values, or value units.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <p><code>TOO_MANY_EXTENTS</code><br/>(previously <code>MANY_EXTENTS</code>)</p>  | <p>Warning</p> | <p>Checks whether there are tables or indices, for which the number of allocated extents exceeds the specified threshold value. All tables or indices are checked by default against the same threshold value. However, you can define different threshold values for individual tablespaces by specifying the tablespace name in the <code>OBJECT</code> field of the <code>DBCHECKORA</code> table (transaction <code>DB17</code>).</p> <p>Possible test operands, threshold values, value units:</p> <ul style="list-style-type: none"> <li>• <code>&gt;, &gt;= &lt;number&gt;</code><br/>More than <code>&lt;number&gt;</code> % of the maximum number of extents was already allocated.</li> <li>• <code>&gt;, &gt;= &lt;number&gt;</code><br/>More than <code>&lt;number&gt;</code> extents were already allocated</li> <li>• <code>&lt;, &lt;= &lt;number&gt;</code><br/>Fewer than <code>&lt;number&gt;</code> % of the maximum number of extents can still be allocated.</li> <li>• <code>&lt;, &lt;= &lt;number&gt;</code><br/>Fewer than <code>&lt;number&gt;</code> extents can still be allocated.</li> </ul> |
| <p><code>CRITICAL_SEGMENT</code><br/>(previously <code>CRITICAL_SEGS</code>)</p> | <p>Warning</p> | <p>Checks whether there are tables or indexes that can bring the tablespace to overflow when up to 5 next extents are allocated. All tables or indices are checked by default against the same threshold value. However, you can define different threshold values for individual tablespaces by specifying the tablespace name in the <code>OBJECT</code> field of the <code>DBCHECKORA</code> table (transaction <code>DB17</code>).</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

| Condition                                                                      | Severity     | Description                                                                                                                                                                                                                                                                                                                                                                           |
|--------------------------------------------------------------------------------|--------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                                                |              | <p>Possible test operands, threshold values, value units:</p> <p>&lt;= &lt;number&gt;</p> <p>The tablespace overflows when you allocate up to 1, 2, 3, 4, or 5 next extents.</p>                                                                                                                                                                                                      |
| <p>IN_WRONG_TABLESPACE<br/>(previously<br/>TABLES_NOT_IN_TABLE_TABLESPACE)</p> | <p>Error</p> | <p>Checks whether there are tables that not in a table tablespace or indices, which are not in an index tablespace.</p> <p>For an MCODE system, checks whether table and indexes of an owner are in tablespaces belonging to the owner</p> <p>The OBJECT field is not specified for this condition. This condition does not have test operands, threshold values, or value units.</p> |
| <p>MISSING_INDEX<br/>(previously MISSING_INDEXES)</p>                          | <p>Error</p> | <p>Checks whether there are tables that do not have any indices and are not specified in the DBDIFF table exception.</p> <p>The OBJECT field is not specified for this condition.</p> <p>This condition does not have test operands, threshold values, or value units.</p>                                                                                                            |
| <p>MISSING_STATISTICS (previously<br/>NO_OPT_STATS)</p>                        | <p>Error</p> | <p>Checks whether there are tables or indices that do not have any statistics, although they should have these.</p> <p>The OBJECT field is not specified for this condition.</p> <p>This condition does not have test operands, threshold values, or value units.</p>                                                                                                                 |
| <p>HARMFUL_STATISTICS</p>                                                      | <p>Error</p> | <p>Checks whether there are tables or indices that have statistics, although they should not have these (for example, with ACTIVE flag set to N in DBSTATC).</p> <p>The OBJECT field is not specified for this condition. This condition does not have test operands, threshold values, or value units.</p>                                                                           |
| <p>PCTINCREASE_NOT_ZERO</p>                                                    | <p>Error</p> | <p>Checks whether there are tables or indexes for which the PCTINCREASE storage parameter is <i>not</i> equal to zero.</p>                                                                                                                                                                                                                                                            |

| Condition | Severity | Description                                                                                                                                                                                                                      |
|-----------|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|           |          | <p>This can lead to storage fragmentation and is not suitable for the SAP System.</p> <p>The OBJECT field is not specified for this condition. This condition does not have test operands, threshold values, or value units.</p> |

#### Note

Since the check conditions for database administration are hard-coded in BRCONNECT (known as built-in check conditions), you cannot add new conditions to the DBCHECKORA table. You can exclude individual tables and indexes or even complete tablespaces from certain checks that run at table or index level. To do this, specify the objects in the [check\\_exclude](#) parameter:

```
check_exclude = [<owner>.<table> | [<owner>.<index> | <tablespace>
| (<object_list>)
```

You can restrict the following check conditions in this way:

```
TOO_MANY_EXTENTS, CRITICAL_SEGMENT, IN_WRONG_TABLESPACE,
MISSING_INDEX, MISSING_STATISTICS, HARMFUL_STATISTICS,
PCTINCREASE_NOT_ZERO
```

## BRCONNECT Default Conditions for Database Operations

The check conditions for database operations are specified in the control table DBCHECKORA. For more information, see [Database System Check with BRCONNECT](#).

| Condition           | Severity | Description                                                                                                                                                                                                                                                                                                                                                                |
|---------------------|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LAST_ARCHIVE_FAILED | Warning  | <p>Checks whether the last backup of the offline redo log files with BRARCHIVE failed. BRCONNECT takes the following function IDs into account:</p> <p>sve, cpy, ssv, svd, cpd, ssd, cps, cds</p> <p>The OBJECT field is not specified for this condition. This condition does not have test operands, threshold values, or value units.</p>                               |
| LAST_BACKUP_FAILED  | Warning  | <p>Checks whether the last complete backup of the database with BRBACKUP failed. BRCONNECT takes the following function IDs into account:</p> <p>afd, afp, afr, afs, aft, and, anp, anr, ans, ant, ffd, ffp, ffr, ffs, fft, fnd, fnp, fnr, fns, fnt</p> <p>The OBJECT field is not specified for this condition. This condition does not have test operands, threshold</p> |

| Condition             | Severity | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-----------------------|----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                       |          | values, or value units.                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| LAST_STATS_FAILED     | Warning  | <p>Checks whether the last update of the optimizer statistics with BRCONNECT failed. BRCONNECT takes the following function IDs into account:</p> <p>sta, aly</p> <p>These refer to the entire database.</p> <p>The OBJECT field is not specified for this condition. This condition does not have test operands, threshold values, or value units.</p>                                                                                                                                        |
| LAST_OPERATION_FAILED | Warning  | <p>Checks whether the last DBA operation failed, which is identified by a function ID specified in the OBJECT field of the DBCHECKORA table.</p> <p>This condition does not have test operands, threshold values, or value units.</p> <p>In SAP Releases 4.0 and 4.5, where the OBJECT field is not yet defined in DBCHECKORA, you can define this condition by specifying the function ID in the PARAM field. You can also use this convention in later (including current) SAP releases.</p> |
| ARCHIVE_TOO_OLD       | Warning  | <p>Checks whether the last successful backup of the offline redo log files with BRARCHIVE is too old. BRCONNECT takes into account the following function ids:</p> <p>sve, cpy, ssv, svd, cpd, ssd, cps, cds</p> <p>The OBJECT field is not specified for this condition. Possible test operands, threshold values, value units:</p> <p>&gt;, &gt;= &lt;number&gt; D</p> <p>The last successful backup of the offline redo log files is older than &lt;number&gt; day(s).</p>                  |
| BACKUP_TOO_OLD        | Warning  | <p>Checks whether the last successful complete backup of the database with BRBACKUP is too old. BRCONNECT takes into account the following function IDs:</p> <p>afd, afp, afr, afs, aft, and, anp, anr, ans, ant, ffd, ffp, ffr, ffs, fft, fnd, fnp, fnr, fns, fnt</p> <p>The OBJECT field is not specified for this condition. Possible test operands, threshold values, value units:</p> <p>&gt;, &gt;= &lt;Number&gt; D</p> <p>The last successful complete backup of the database is</p>   |

| Condition         | Severity | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-------------------|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                   |          | older than <number> day(s).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| STATS_TOO_OLD     | Warning  | <p>Checks whether the last successful update of the Optimizer statistics with BRCONNECT is too old. BRCONNECT takes the following function ID into account:</p> <p>sta, aly</p> <p>The OBJECT field is not specified for this condition. Possible test operands, threshold values, value units:</p> <p>&gt;, &gt;= &lt;number&gt; D</p> <p>The last successful update of the optimizer statistics is older than &lt;number&gt; day(s).</p>                                                                                                                                                       |
| OPERATION_TOO_OLD | Warning  | <p>Checks whether the last successful DBA operation, identified by a function ID specified in the OBJECT field of the DBCHECKORA table, is too old.</p> <p>Possible test operands, threshold values, value units:</p> <p>&gt;, &gt;= &lt;number&gt; D</p> <p>The last successful DBA operation is older than &lt;number&gt; day(s).</p> <p>In SAP Releases 4.0 and 4.5, where the OBJECT field is not yet defined in DBCHECKORA, you can define this condition by specifying the function ID in the PARAM field. You can also use this convention in later (including current) SAP releases.</p> |

#### Note

Since the test conditions for database operations are programmed in a specific way in BRCONNECT (known as built-in test conditions), you cannot add new check conditions to the DBCHECKORA table. However, this is generally not necessary because operations can be monitored by the LAST\_OPERATION\_FAILED and OPERATION\_TOO\_OLD check conditions, or by specifying function IDs in the PARAM field.

## Adapt Next Extents with BRCONNECT

You can use [BRCONNECT](#) to adapt the next extents size. The aim is to avoid the structure of tablespaces deteriorating – that is, breaking up into a large number of small extents – because this reduces database performance.

#### Caution

This function is only relevant for dictionary-managed tablespaces. Do *not* use it if all tablespaces are locally managed.

## Prerequisites

The database tables and indexes are assigned to one of a number of categories. The standard size of the next extent for each category is defined in the tables `TGORA` (for tables) and `IGORA` (for indexes).

## Activities

When a table requires adapting, the size of the next extent is determined using an algorithm. This also makes sure that the value of `MAX_EXTENTS` for a table or index is not less than the value defined in the `TGORA` or `IGORA` table.

You can exclude specified tables or indexes from this function by using the `next_exclude` parameter.

You can specify individual values for `NEXT_EXTENT` or `MAX_EXTENTS` using the `next_special` parameter.

For more information on the command line options for adapting the next extents, see [-f next](#).

## More Information

[Internal Rules for Determining Next Extent Size](#)

[Methods of Adapting Next Extent Size](#)

## Internal Rules for Determining Next Extent Size

This algorithm is used by `BRCONNECT` to determine the next extent size when a table needs adapting. For more information, see [Adapt Next Extents](#).

1. `BRCONNECT` works out the value of 10% of the space currently allocated to the table or index.
2. Using this value, `BRCONNECT` selects the next smaller category from the table `TGORA` (for tables) or `IGORA` (for indexes). The tables are shown below at the end of this section.
3. `BRCONNECT` looks up the table in the `DD09L` table and selects the category given. The `DD09L` is a data dictionary table in which most SAP tables are entered. For partition and LOB segments, `BRCONNECT` takes 10% of the category value and, using this new value, again selects the next smaller category. If the table is not there, size category 0 is assumed.
4. `BRCONNECT` chooses the larger of the values from steps 2 and 3, if necessary reducing it to the value of the `init<DBSID>.sap parameter next_max_size`.
5. If the required space is greater than the remaining free space in the tablespace and no file in the affected tablespace permits an autoextend, `BRCONNECT` reduces the value from step 4 to the size of the largest free space segment in the tablespace.
6. `BRCONNECT` compares the value from step 5 with the current value of `NEXT_EXTENT` and chooses the larger value.
7. If the `next_special` parameter is defined for the table, `BRCONNECT` always uses this instead of the value determined so far.

8. BRCONNECT reduces the value from step 7 to the next smaller multiple of 5 times the database block size, to reduce free space wastage. The smallest possible value is 5 times the database block size.
9. BRCONNECT compares the value from step 8 with the current next extent size. If there is a difference, it changes the NEXT\_EXTENT storage parameter of the table or index to the newly determined next extent size.

#### NEXT Values in TORA/IGORA

| Size category | NEXT value for table (KB) | NEXT value for indexes (KB) |
|---------------|---------------------------|-----------------------------|
| 0             | 40                        | 40                          |
| 1             | 160                       | 80                          |
| 2             | 640                       | 160                         |
| 3             | 2560                      | 640                         |
| 4             | 10240                     | 2560                        |
| 5             | 20480                     | 5120                        |
| 6             | 40960                     | 10240                       |
| 7             | 81920                     | 20480                       |
| 8             | 163840                    | 40960                       |
| 9             | 327680                    | 81920                       |
| 10            | 655360                    | 163840                      |
| 11            | 1310720                   | 327680                      |
| 12            | 2621440                   | 655360                      |
| 13            | 5242880                   | 1310720                     |
| 14            | 10485760                  | 2621440                     |

## Update Statistics with BRCONNECT

You can use this [BRCONNECT](#) function to update the statistics on the Oracle database for the cost-based optimizer.

By running update statistics regularly, you make sure that the database statistics are up-to-date, so improving database performance. The Oracle cost-based optimizer (CBO) uses the statistics to optimize access paths when retrieving data for queries. If the statistics are out-of-date, the CBO might generate inappropriate access paths (such as using the wrong index), resulting in poor performance.

From Release 4.0, the CBO is a standard part of the SAP system. If statistics are available for a table, the database system uses the cost-based optimizer. Otherwise, it uses the rule-based optimizer.

BRCONNECT supports update statistics for the following:

- Partitioned tables, except where partitioned tables are explicitly excluded by setting the active flag in the `DBSTATC` table to `I`. For more information, see SAP Note [424243](#).
- [InfoCube tables](#) for the SAP Business Information Warehouse (SAP BW)

## Integration

You can update statistics using one of the following methods:

- DBA Planning Calendar in the Computing Center Management System (CCMS)

For more information, see [Update Statistics for the Cost-Based Optimizer in CCMS \(Oracle\)](#). The DBA Planning Calendar uses the BRCONNECT commands.

Recommendation

We recommend you to use this approach because you can easily schedule update statistics to run automatically at specified intervals (for example, weekly or even daily with Oracle 10g or higher).

- BRCONNECT, as described here

## Prerequisite

To use the CBO, make sure that the parameter `OPTIMIZER_MODE` in the [Oracle initialization profile](#) `init<DBSID>.ora` is set to `CHOOSE`.

## Features

BRCONNECT performs update statistics using a two-phase approach.

BRCONNECT:

1. Checks each table to see if the statistics are out-of-date
2. If required, updates the statistics on the table immediately after the check

For more information about how update statistics works, see [Internal Rules for Update Statistics](#).

## Activities

You can influence how update statistics works by using the `-force` options. For more information, see [-f stats](#).

Recommendation

Unless you have special requirements, we recommend you to perform the standard update statistics, using one of the following tools to schedule it on a regular basis (for example, daily or weekly):

- DBA Planning Calendar, as described above in *Integration*.



- A tool such as cron (UNIX) or at (Windows NT) to execute the following standard call:

```
brconnect -u / -c -f stats -t all
```

This is also adequate after an upgrade of the database or SAP system. It runs using the OPS\$ user without operator intervention.

The following are also standard commands that you can use to update statistics:

- Update statistics only for tables and indexes with missing statistics

```
brconnect -u / -c -f stats -t missing
```

- Check and update statistics for all tables defined in the DBSTATC table

```
brconnect -u / -c -f stats -t dbstatc_tab
```

## Example

For examples of how you can override the [internal rules for update statistics](#), see [-force with Update Statistics](#).

## -force with Update Statistics

This section gives examples of how you can use the `-force` options to override the [internal rules for update statistics](#). For more information about the `-force` options, see [-f stats](#).

### Caution

Only use these options in exceptional circumstances.

- Check and update statistics for all tables in tablespace PSAPBTAD

```
brconnect -u / -c -f stats -t psapbtabd -f allsel
```

- Update statistics without check for all tables relevant to the application monitor

```
brconnect -u / -c -f stats -t dbstatc_mon -f collect
```

- Update statistics for a pool table specified in the DBSTATC table with the `ACTIVE` flag set to `N`, to determine space usage, using method “estimate 10% rows”:

```
brconnect -u / -c -f stats -t atab -m EI -s P10 -f  
allsel,method,sample,collect,space
```

BRCONNECT stores the space usage data in tables DBSTATTORA (for tables) and DBSTATIORA (for indexes). Finally, the statistics can be immediately deleted - for example, when the `ACTIVE` flag is set to `N` in DBSTATC.

If you want to keep the statistics for such tables – for example, for test purposes – you must include the `keep` option in the command, as follows:

```
brconnect -u / -c -f stats -t rfblyg -m EI -s P10 -f  
allsel,method,sample,keep
```

### Caution

Option `-m EI` locks the relevant indexes.

- Check and update statistics according to the two-phase concept:

1. First phase: check statistics

```
brconnect -u / -c -f stats -t all -f nocoll
```

2. Second phase: update statistics on tables identified in the first phase

```
brconnect -u / -c -f stats -t all -f nocheck
```

## Deletion of Damaging Statistics

This section describes how BRCONNECT deletes damaging statistics for the cost-based optimizer of the Oracle database.

Pool and cluster tables (for Oracle 9i only) and tables that have the `ACTIVE` flag set to `N` or `R` in the `DBSTATC` control table should not normally have statistics, since such statistics can negatively affect database performance.

### Activities

In the standard update statistics run, using `brconnect -f stats -t all`, BRCONNECT checks whether such damaging statistics exist and deletes them if so.

You can delete such damaging statistics immediately with the following command:

```
brconnect -u / -c -f stats -t harmful -d
```

To delete statistics for other tables as well (only for test purposes), you can use the option `-f allsel`:

```
brconnect -u / -c -f stats -t sdbah,sdbad -d -f allsel
```

#### Caution

Starting with Oracle 10g, all tables generally have optimizer statistics.

## Verification of Table and Index Structure

This section describes how you can use BRCONNECT for the Oracle database to check the internal structure of table and index blocks. This is an alternative to DBVERIFY. However, the scope of the checks is not the same.

### Example

To check the structure of all tables and their indexes in the tablespace PSAPBTABD, enter the following command:

```
brconnect -u / -c -f stats -t psapbtabd -v cascade
```

#### Caution

Starting with Oracle 9, tables and indexes are no longer locked.

However, if you option `-v cascade_store` then the indexes are still locked.

## Internal Rules for Update Statistics

This algorithm is used by BRCONNECT to update statistics. For more information, see [Update Statistics with BRCONNECT](#).

1. BRCONNECT determines the working set of tables and indexes to be checked and updated. To do this, it uses:
  - Options `-t` | `-table` and `-e` | `-exclude`, as described in [-f stats](#) (these options take priority)
  - [stats\\_table](#) and [stats\\_exclude](#) parameters
2. If the working set contains pool, cluster (for Oracle 9i only) or other tables that have the `ACTIVE` flag in the `DBSTATC` table set to `N` or `R`, BRCONNECT immediately deletes the statistics for these tables. This is because they negatively affect database performance.
3. BRCONNECT checks statistics for the remaining tables in the working set, including tables that have the `ACTIVE` flag in the `DBSTATC` table set to `A` or `P`, as follows:
  - If the table has the `MONITORING` attribute set, BRCONNECT reads the number of inserted, deleted, and updated rows from the `DBA_TAB_MODIFICATIONS` table (this is standard in Oracle 10g).
  - Otherwise, BRCONNECT uses the standard method (see table below) to update statistics by using the unique index.

### Note

BRCONNECT uses the following standard method to check and update a table's statistics:

- Method and sample defined for the table in the `DBSTATC` table (has highest priority)
- Method and sample from the options `-m` | `-method` or `-s` | `-sample` of [-f stats -method](#) (takes priority) or the [stats\\_method](#) and [stats\\_sample\\_size](#) parameters
- Default method and sample (has lowest priority)

The following table describes the default method:

| Number of Rows in Table |                  | Analysis Method | Sample Size |
|-------------------------|------------------|-----------------|-------------|
|                         | Rows < 10,000    | C               |             |
| 10,000 <=               | Rows < 100,000   | E               | P30         |
| 100,000 <=              | Rows < 1,000,000 | E               | P10         |

| Number of Rows in Table                        | Analysis Method | Sample Size |
|------------------------------------------------|-----------------|-------------|
| 1,000,000 <= Rows < 10,000,000                 | E               | P3          |
| 10,000,000 <= Rows < 100,000,000               | E               | P1          |
| 100,000,000 <= Rows < 1,000,000,000            | E               | P.3         |
| 1,000,000,000 <= Rows < 10,000,000,000         | E               | P.1         |
| 10,000,000,000 <= Rows < 100,000,000,000       | E               | P.03        |
| 100,000,000,000 <= Rows < 1,000,000,000,000    | E               | P.01        |
| 1,000,000,000,000 <= Rows < 10,000,000,000,000 | E               | P.003       |
| 10,000,000,000,000 <= Rows                     | E               | P.001       |

Analysis method C means compute the statistics exactly. Analysis method E means estimate the statistics using the sample size specified.

For example, "E P10" means that BRCONNECT takes an estimated sample using 10% of rows.

For the CH, CX, EH, and EX methods, histograms are created.

For the CI, CX, EI and EX methods, the structure of indexes is validated in addition to collecting statistics. However, this *locks the indexes*.

4. BRCONNECT uses the number of new rows for each table in the working set, as derived in the previous step, to see if either of the following is true:
  - If table MONITORING is used (standard in Oracle 10g):
    - $\#old\ rows + \#inserted\ rows \geq \#old\ rows * (100 + threshold) / 100$
    - $\#old\ rows + \#updated\ rows \geq \#old\ rows * (100 + threshold) / 100$
    - $\#old\ rows - \#deleted\ rows \leq \#old\ rows * 100 / (100 + threshold)$
  - If table MONITORING is not used:
    - Number of new rows is greater than or equal to number of old rows \*  $(100 + threshold) / 100$
    - Number of new rows is less than or equal to number of old rows \*  $100 / (100 + threshold)$

The standard threshold is 50, but the value defined in [-f stats -change](#) or the [stats\\_change\\_threshold](#) parameter is used if specified.

5. BRCONNECT immediately updates statistics after checking for the following tables:
  - Tables where either of the conditions in the previous step is true
  - Tables from the DBSTATC table with either of the following values:
    - ACTIVE field U (unconditional)

- `ACTIVE` field `R` or `N` and `USE` field `A` (relevant for the application monitor) and the last update statistics was at least 30 days ago
6. `BRCONNECT` writes the results of update statistics to the `DBSTATTORA` table and also, for tables with the `DBSTATC` history flag or usage type `A`, to the `DBSTATHORA` table.
  7. For tables with update statistics using methods `EI`, `EX`, `CI`, or `CX`, `BRCONNECT` validates the structure of all associated indexes and writes the results to the `DBSTATIORA` table. `BRCONNECT` also does this for tables with the `DBSTATC` history flag or usage type `A`, writing the results to the `DBSTAIHORA` table.
  8. `BRCONNECT` immediately deletes the statistics that it created in this procedure for tables with the `ACTIVE` flag set to `N` or `R` in the `DBSTATC` table.

For more information about special rules for updating statistics of individual table partitions, see *SAP Notes* [744483](#) and [865366](#).

## Update Statistics for InfoCube Tables

The InfoCube tables used in SAP Business Information Warehouse (SAP BW) and Advanced Planner and Optimizer (APO) need to be processed in a special way when the statistics are being updated. Usually, statistics should be created using histograms, as described in SAP Note [129252](#).

Statistics for the InfoCube tables can be updated, together with other tables in a run. In this case, the statistics for the InfoCube tables are always created with histograms. You can specify which tables are to be handled as InfoCube tables using the `init<DBSID>.sap` parameter [stats\\_info\\_cubes](#). However, this is normally not needed.

### Prerequisites

The control table `RSNSPACE` for Business Information Warehouse (BW) and Advanced Planner and Optimizer (APO) dynamically determines which tables are to be handled as InfoCube tables. The content of the control table might change in future.

### Features

By default, tables with names starting with the following prefixes are processed by `BRCONNECT` as InfoCube tables:

```
/BIC/F*, /BIC/A9F*,
/BI0/F*, /BI0/A9F*,
/BIC/E*, /BIC/A9E*,
/BI0/E*, /BI0/A9E*,
/BIC/D*, /BIC/A9D*,
/BI0/D*, /BI0/A9D*,
/BIC/S*, /BIC/A9S*,
/BI0/S*, /BI0/A9S*,
/BIC/X*, /BIC/A9X*,
```

```

/BIO/X*, /BIO/A9X*,
/BIC/Y*, /BIC/A9Y*,
/BIO/Y*, /BIO/A9Y*,
/BIC/I*, /BIC/A9I*,
/BIO/I*, /BIO/A9I*,
/BIC/P*, /BIC/A9P*,
/BIO/P*, /BIO/A9P*,
/BIC/Q*, /BIC/A9Q*,
/BIO/Q*, /BIO/A9Q*
/BIO/02*, BIO/06*

```

The above list provides the default value of the `init<DBSID>.sap` parameter `stats_info_cubes`, that you can use to include the following kinds of tables in the list of InfoCube tables:

- Groups with names starting with a certain prefix and suffix, such as `<prefix>*<suffix>`
- Individual tables in the list of InfoCube tables

#### Caution

To include the above default list, specify the keyword `DEFAULT` in the first position, as in the following example:

```
stats_info_cubes = (DEFAULT, XYZ*)
```

However, we do *not* recommend this, since the contents of `RSNSPACE` do not always have to correspond to the standard. To take account of this, always use the following in this case:

```
stats_info_cubes = (RSNSPACE_TAB XYZ*)
```

To suppress special handling of the InfoCube tables completely, use the keyword `NULL`:

```
stats_info_cubes = NULL
```

If certain prefixes are omitted in the parameter definition, the corresponding tables are not to be handled as InfoCube tables.

However, we do not recommend you to do this. Normally you do not need to set the parameter at all.

For additional, special handling of InfoCube tables, you can use the keyword `INFO_CUBES` for the following:

- BRCONNECT with the [-f stats function](#) using the `-t|-table` and `-e|-exclude` options
- `init<DBSID>.sap` parameters:
  - [stats\\_table](#)

- [stats\\_exclude](#)
- [stats\\_dbms\\_stats](#)

The function of this keyword is to ensure that only InfoCube tables are processed in accordance with the selected parameter settings.

## Example

- `brconnect -u / -c -f stats -t info_cubes`

Statistics are only checked for InfoCube tables and updated, if required

- `brconnect -u / -c -f stats -t all -e info_cubes`

Statistics are checked for all tables besides InfoCube tables and updated, if necessary.

- `stats_dbms_stats = INFO_CUBES:R:4 brconnect -u / -c -f stats -t all`

Statistics are checked for all tables and updated, if necessary. New statistics for InfoCube tables are created with the DBMS\_STATS package using row sampling and an internal parallel degree of 4.

For more information, see [stats\\_dbms\\_stats](#).

- `brconnect -u / -c -f all`

This is the default. Statistics are checked for all tables and updated, if necessary. If InfoCube tables are present and selected following the update check, statistics are generated for them using histograms. This is the recommended standard call.

## Sample Sizes for Update Statistics

The sample size that BRCONNECT uses when updating statistics was 1% or more in the past. Now it is possible to use much smaller sample sizes, which improves the performance of update statistics on large tables. You can now set sample sizes of 0.001% to 100%. This applies to the following ways of setting the sample size:

- [-f stats -s|sample](#)
- Parameter [stats\\_sample\\_size](#) in `init<DBSID>.sap`
- Field OPTIO (sample size) in the DBSTATC field

In all cases, you can set

Example

To set the sample size to 0.05%, enter a value of `P.05` in the `brconnect -f stats -s` command, the `stats_sample_size` parameter in `init<DBSID>.sap`, or the OPTIO field of the DBSTATC table.

The BRCONNECT standard formula to determine the sample size is as follows:

| Number of Rows                        | Sample Size in % |
|---------------------------------------|------------------|
| 0 - 9,999                             | 100              |
| 10,000 - 99,999                       | 30               |
| 100,000 - 999,999                     | 10               |
| 1,000,000 - 9,999,999                 | 3                |
| 10,000,000 - 99,999,999               | 1                |
| 100,000,000 - 999,999,999             | 0.3              |
| 1,000,000,000 - 9,999,999,999         | 0.1              |
| 10,000,000,000 - 99,999,999,999       | 0.03             |
| 100,000,000,000 - 999,999,999,999     | 0.01             |
| 1,000,000,000,000 - 9,999,999,999,999 | 0.003            |
| > 9,999,999,999,999                   | 0.001            |

## Changing Database User Passwords with BRCONNECT

You can use BRCONNECT to change the password of database users, such as SAPR3 or SAP<SAPSID>.

### Caution

Do not leave the passwords for the database user set to the default values.

## Procedure

For more information, see [-f chpass](#).

## Result

BRCONNECT stores the new password as follows:

| User  | Table                          | Encryption (from Release 4.5B) |
|-------|--------------------------------|--------------------------------|
| SAPR3 | "OPS\$<ORACLE_SID>ADM".SAPUSER | Stored as SAPR3-CRYPT          |



| User        | Table                      | Encryption (from Release 4.5B) |
|-------------|----------------------------|--------------------------------|
| SAP<SAPSID> | "OPS\$<SAPSID>ADM".SAPUSER | Stored as SAP<SAPSID>-CRYPT    |

#### Note

In the case of the SAP<SAPSID> user, <SAPSID> refers to the SAP System ID. For example, if the SAP System ID is C11, the SAP<SAPSID> user is SAPC11, stored in the table OPS\$C11ADM.SAPUSER. The first character of <SAPSID> must be a letter and there is no distinction between uppercase and lowercase.

Database users other than SAPR3 and SAP<SAPSID> are not stored in SAPUSER database tables.

## Clean Up Old Logs and Trace Files with BRCONNECT

You can use [BRCONNECT](#) to clean up old log files, disk backups, export files, trace files and database logs. The aim is to avoid unnecessary use of disk space by deleting files that are no longer required.

### Features

BRCONNECT cleans up the following files:

- Detailed BRARCHIVE logs in the `saparch` directory
- Detailed BRBACKUP logs in the `sapbackup` directory
- Detailed BRCONNECT logs in the `sapcheck` directory
- Detailed BRRESTORE logs in the `sapbackup` directory
- Detailed BRRECOVER logs in the `sapbackup` directory
- Detailed BRSPACE logs in the `sapreorg` directory
- BRBACKUP disk backups of the database files
- BRARCHIVE disk backups of the offline redo log files
- BRSPACE export dumps and export parameter files in export dump directories
- Oracle trace and audit files
- Log records in the SDBAH and SDBAD tables
- Log records in the XDB tables
- Database check results in DBMSGORA table

## Activities

Using the cleanup parameters in the [Initialization Profile init<DBSID>.sap](#) you can determine how old the objects are before they are deleted. For example, see [cleanup\\_brarchive\\_log](#).

For more information on the command line options for cleaning up the log files, see [-f cleanup](#).

## Additional BRCONNECT Functions

These additional [BRCONNECT](#) functions only write messages to the standard output device if the option `-l|-log` is not specified.

### Features

- Change Passwords of Database Users

You can use this function to specify a new password for all database users.

At the same time, the encrypted password in the SAPUSER table is changed is for SAP users. The work processes of the application server use this password to connect to the database.

For more information, see [-f chpass](#).

- Setting Up SAP DBA Synonyms

You can use this function to specify the SAP System for database administration activities in a multi-schema database. You can then start and monitor database actions in the corresponding SAP System.

For more information, see [-f crsyn](#).

- Starting the Database Instance

You can start the database instance using `-f dbstart`.

For more information, see [-f dbstart](#).

- Stopping the Database Instance

You can use this function to stop the database instance. BRCONNECT first checks whether an SAP System is still running.

For more information, see [-f dbshut](#).

- Determine Database State

You can use function `-f dbstate` to determine the database state. The return codes have the following meanings:

- 0 - Database is running
- 1 - Database is stopped
- 2 - Database is in nomount or mount state
- 3 - Error, database status cannot be determined

For more information, see [-f dbstate](#).

## Command Options for BRCONNECT

This section describes the command options for the BRCONNECT tool.

If you use BRCONNECT with command options, these override the corresponding values in the [initialization profile init<DBSID>.sap](#). To use the options, you can specify either the letter indicated or the complete word.

This is the schematic command syntax:

### Syntax

```
brconnect [<command_options>] -f|-function <function>
[<function_options>]
```

End of the code.

This is the full command syntax:

### Syntax

```
brconnect
[-c|-confirm [|force]]
[-h|-help [<function>][|version]]
[-l|-language E|D]
[-o|-output detail|process|summary|[, ]time]
[-p|-profile <profile>]
[-q|-query [check|nolog]]
[-s|-sapsid <sid>|<sid_list>]
[-u|-user [<user>[/<password>]]]
[-V|-VERSION [ALL]]
-f|-function <function> [<function_options>]
```

End of the code.

### Example

```
brconnect -output detail -function check -default
```

See also:

[-c|-confirm](#)

[-h|-help](#)

[-l|-language](#)

[-o|-output](#)

[-p|-profile](#)

[-q|-query](#)

[-u|-user](#)

[-V|-VERSION](#)

[-f|-function](#)

## **-c|-confirm**

This BRCONNECT command option activates processing in unattended mode.

Syntax

```
-c|-confirm [force]
```

End of the code.

Default value: confirmation required for processing to be started

Use this option if BRCONNECT is started by an automatic scheduler such as `cron` (UNIX) or `at` (Windows).

Possible value:

`force` suppresses all confirmation messages. Normally, this is unnecessary because `-c` on its own is sufficient.

## **-f|-function**

This BRCONNECT command option specifies the function to be performed. You must always enter a function.

Input syntax: `-f|function`

```
chpass|crsyn|dbshut|dbstart|dbstate|check|cleanup|next|stats
```

Default value: None, since you must always specify a function option

### **Function Options**

- `check`: [checks the database system](#)
- `chpass`: [changes the password of database users](#)
- `cleanup`: [cleans up database logs](#)
- `crsyn`: [creates public synonyms for tables](#) used by BR\*Tools
- `dbshut`: [shuts down the database](#)
- `dbstart`: [starts up the database](#)
- `dbstate`: [checks the database state](#)
- `next`: [adapts next extents](#)
- `stats`: [updates optimizer statistics](#)

## **-f check**

This BRCONNECT [function](#) checks the database system. For more information, see [Database System Check](#).

Function options:

- `-d|-default`: uses internal BRCONNECT default settings to check the database system

Input syntax: `-d|-default`

Default value: uses settings from the control table DBCHECKORA

- `-e|-exclude`: defines tables and indexes to be excluded from the check

Input syntax:

`-e|-exclude [<owner>.]<table>| [<owner>.]<index>|`

`[<owner>.] [<prefix>]* [<suffix>]|<tablespace>|`

`<object_list>|non_sap|all_part|null`

`non_sap` means that non-SAP objects (for example, Oracle dictionary objects) are excluded from the check.

`all_part` means that SAP partitions (such as in Business Information Warehouse and Advanced Planner and Optimizer) are excluded from the check.

`null` invalidates the exclusion list defined by the [check\\_exclude](#) parameter. This means that no tables are excluded from processing.

Default value: no exclusion

You can use this option to exclude tables or indexes with exceptional space parameters or statistics handling.

This option overrides the [check\\_exclude](#) parameter.

- `-i|-ignore`: specifies that BRCONNECT ignores the settings in the control table DBCHECKORA. Instead, it uses the check conditions from the [check\\_cond](#) parameter of the initialization profile `init<DBSID>.sap` or the standard check conditions.

Default value: none

- `-n|-ignndbs`: specifies that BRCONNECT ignores the settings in the control table DBSTATC. Instead, it uses the check conditions from the [stats\\_special](#) parameter of the initialization profile `init<DBSID>.sap` or the standard check conditions.

- `-o|-owner`: defines the database owner of tables and indexes to be checked

Input syntax: `-o|-owner <owner1>[,<owner2>,...]`

Default value: all SAP owners in a multi-schema database or `SAPR3/SAP<SID>` in a standard SAP database

This option overrides the [check\\_owner](#) parameter.

## **-f chpass**

This BRCONNECT [function](#) changes the database user password.

Function options:

- `-l|-log <log_file>`: defines the name of the file containing logging information

Default value: no log file

- `-o|-owner`: defines for which database owner (that is, user) the password is to be changed

Input syntax: `-o|-owner <owner1>[,<owner2>, ...]`

Default value: `SAPR3/SAP<SID>` in a standard SAP database or all SAP owners in a multi-schema database

- `-p|-password`: defines the password for the database owner (that is, user)

Input syntax: `-p|-password <password>`

Default value: interactive entry of the password

## **-f cleanup**

This BRCONNECT [function](#) cleans up the database logs. For more information, see [Clean Up Old Logs and Trace Files](#).

Function options:

- `-a|-archive`: defines the retention period in days for BRARCHIVE detail log files

Input syntax: `-a|-archive <days>`

Default value: 30

This option controls which BRARCHIVE log files are deleted by the BRCONNECT cleanup function. It overrides the [cleanup\\_brarchive\\_log](#) parameter.

- `-b|-backup`: defines the retention period in days for BRBACKUP detail log files

Input syntax: `-b|-backup <days>`

Default value: 30

This option controls which BRBACKUP log files are deleted by the BRCONNECT cleanup function. It overrides the [cleanup\\_brbackup\\_log](#) parameter.

- `-c|-connect`: defines the retention period in days for BRCONNECT detail log files

Input syntax: `-c|-connect <days>`

Default value: 30

This option controls which BRCONNECT log files are deleted by the BRCONNECT cleanup function. It overrides the [cleanup\\_brconnect\\_log](#) parameter

- `-d|-dblog`: defines the retention period in days for records in the tables SDBAH and SDBAD

Input syntax: `-d|-dblog <days>`

Default value: 100

This option controls which records in the tables SDBAH and SDBAD are deleted by the BRCONNECT cleanup function. It overrides the [cleanup\\_db\\_log](#) parameter.

- `-e|-expdump`: defines the retention period in days for BRSPACE export dumps and scripts

Input syntax: `-e|-expdump <days>`

Default value: 30

This option controls which directories and their contents for BRSPACE export dumps and parameter files are deleted by the BRCONNECT cleanup function. It overrides the [cleanup\\_exp\\_dump](#) parameter.

- `-i|-diskarch`: defines the retention period for offline redo log files saved on disk

Input syntax: `-i|-diskarch <days>`

Default value: 30

This option controls which offline redo log files backed up on disk are deleted by the BRCONNECT cleanup function. It overrides the [cleanup\\_disk\\_archive](#) parameter.

- `-k|-diskback`: defines the retention period in days for database files saved on disk

Input syntax: `-k|-diskback <days>`

Default value: 30

This option controls which database files backed up on disk are deleted by the BRCONNECT cleanup function. It overrides the [cleanup\\_disk\\_backup](#) parameter.

- `-l|-limit`: defines that only objects explicitly specified by other function options are to be cleaned up

Input syntax: `-l|-limit`

Default value: Clean up all database logs

- `-m|-checkmsg`: defines the retention period in days for the alert messages from the database check runs

The messages are deleted from the DBMSGORA table when the retention period has expired.

Input syntax: `-m|-checkmsg <days>`

Default value: 100

This option overrides the [cleanup\\_check\\_msg](#) parameter.

- `-o|-owner`: defines the database owner of SDBAH, SDBA, DBMSGORA, and XDB tables to be processed by BRCONNECT cleanup function

Input syntax: `-o|-owner <owner1>[,<owner2>, ...]`

Default value: `SAPR3/SAP<SAPSID>` in a standard SAP database or all SAP owners in a multi-schema database

This option overrides the [cleanup\\_owner](#) parameter.

- `-r|-restore`: defines the retention period in days for BRRESTORE detail log files

Input syntax: `-r|-restore <days>`

Default value: 30

This option controls which BRRESTORE log files are deleted by the BRCONNECT cleanup function. It overrides the [cleanup\\_brrestore\\_log](#) parameter.

- `-s|-space`: defines the retention period in days for BRSPACE log files

Input syntax: `-s|-space <days>`

Default value: 30

This option controls which BRSPACE log files are deleted by the BRCONNECT cleanup function. It overrides the [cleanup\\_orspace\\_log](#) parameter.

- `-t|-trace`: defines the retention period in days for Oracle trace and audit files

Input syntax: `-t|-trace <days>`

Default value: 30

This option controls which Oracle trace and audit files are deleted by the BRCONNECT cleanup function. It overrides the [cleanup\\_ora\\_trace](#) parameter.

- `-v|-recover`: defines the retention period in days for BRRECOVER detail log files

Input syntax: `-v|-recover <days>`

Default value: 30

This option controls which BRRECOVER log files are deleted by the BRCONNECT cleanup function. It overrides the [cleanup\\_brrecover\\_log](#) parameter.

- `-x|-xdb`: defines the retention period in days for records in the XDB tables

Input syntax: `-x|-xdb <days>`

Default value: 100

This option controls which records in the XDB tables are deleted by the BRCONNECT cleanup function. It overrides the [cleanup\\_xdb\\_log](#) parameter.

## **-f crsyn**

This BRCONNECT [function](#) creates public synonyms for the SAP tools.

Function options:

- `-l|-log <log_file>`: defines the name of the file containing logging information

Default value: no log file

- `-o|-owner`: defines the database owner pointed to by the public synonyms used by BR\*Tools

Input syntax: `-o|-owner <owner>`

Default value: SAPR3/SAP<SID>

## **-f dbshut**

This BRCONNECT [function](#) shuts down the database in immediate mode.



Function options:

- `-f|-force`: shuts down the database, whether or not the SAP user is connected

Input syntax: `-f|-force`

Default value: if the SAP user is connected to the database, then it is not shut down.

- `-l|-log <log_file>`: defines the name of the file containing logging information

Default value: no log file

## **-f dbstart**

This BRCONNECT [function](#) lets you start up the database.

Function options:

`-l|-log <log_file>`: specifies the name of the file containing logging information

Default value: no log file

## **-f dbstate**

This BRCONNECT [function](#) lets you check the database state

Function options:

`-l|-log <log_file>`

where:

`<log_file>` specifies the name of the file containing logging information

Default value: no log file

## **-f next**

This BRCONNECT [function](#) adapts the next extents of database tables. For more information, see:

- [Adapt Next Extents](#)
- [Algorithm for Determining Next Extent Size](#)

Function options:

- `-e|-exclude`: defines tables and indexes to be excluded from adapting next extents

Input syntax:

`-e|-exclude [<owner>.]<table>| [<owner>.]<index>|`

`[<owner>.] [<prefix>]* [<suffix>] |<tablespace>|<object_list>`

`|all_part|null`

Default value: no exclusion, process all selected objects

`all_part` means that SAP partitions (such as in Business Information Warehouse and Advanced Planner and Optimizer) are excluded from the check.

`null` invalidates the exclusion list defined by the [next\\_exclude](#) parameter. This means that no tables or indexes are excluded from processing.

You can use this option to exclude from processing tables or indexes with exceptional space parameters. This option overrides the [next\\_exclude](#) parameter.

- `-f|-force`: forces the next extent size of all selected tables and indexes to be reduced to the maximum free extent size in the tablespace or to the maximum next extent size (`next_max_size` parameter) if required

Input syntax: `-f|-force free|max|both|nocasc`

Default value: `both`

The next extent size is not normally reduced, but only rounded down to the next multiple of five times the database block size.

- `free`: reduces the next extent size to the maximum free extent size in the tablespace
- `max`: reduces the next extent size to the value of parameter `next_max_size`
- `both`: reduces the next extent size to the smaller of the values `free` and `max`
- `nocasc`: With this option, you can prevent the check and, if necessary, the adjustment of NEXT extents from being implicitly performed for all indexes for the selected tables. This option is only provided for exceptional situations.

- `-l|-limit`: defines the maximum number of next extents (`MAX_EXTENTS`)

Input syntax: `-l|-limit <count>`

Default value: settings from tables `TGORA` and `IGORA`

0 means unlimited number of extents

This option overrides the [next\\_limit\\_count](#) parameter.

- `-m|-max`: defines the maximum size for next extents, which must not be exceeded when a next extent is adapted

Input syntax: `-m|-max <size>`

Default value: `2 GB - 5 * <database block size>`

0 means unlimited next extent size

You can specify this option in KB, MB, or GB. This option overrides the [next\\_max\\_size](#) parameter.

- `-o|-owner`: defines the database owner of tables and indexes to be processed

Input syntax: `-o|-owner <owner1>[,<owner2>, ...]`

Default value: `SAPR3/SAP<SID>` in a standard database or all SAP owners in a multi-schema database.

This option overrides the [next\\_owner](#) parameter.

- `-s|-special`: defines special sizes of next extent and maximum number of next extents for exceptional tables and indexes

Input syntax:

```
-s|-special [<owner>.<table>:<size>[/<limit>]|
[<owner>.<index>:<size>[/<limit>]|
[<owner>.] [<prefix>]* [<suffix>]:<size>[/<limit>]|
<special_list>| [all_sel:]<size>[/<limit>]|null
```

Default value: derived from [internal rules for determining next extent size](#)

This option overrides the [next\\_special](#) parameter.

- `<size>`: size of next extent for specified table or index ( )
  - `<limit>`: maximum number of next extents for specified table or index (MAX\_EXTENTS)
  - `all_sel`: sets NEXT\_EXTENT and MAX\_EXTENTS attributes to a certain value for all the database objects selected using the `-t` function option or the `next_table` parameter (see below). This option is provided for exceptional situations. This is the default when the table or index name is not specified.
  - `null`: invalidates the special list defined by the [next\\_special](#) parameter. It means that no tables or indexes are to be processed in a special way.
- `-t|-table`: defines the database objects (that is, tables, indexes, or tablespaces) to be processed

Input syntax:

```
-t|-table all|all_ind|special| [<owner>.<table>|
[<owner>.<index>
| [<owner>.] [<prefix>]* [<suffix>]|<tablespace>|<object_list>
```

Default value: all objects of selected owners

- `all`: All SAP tables and indexes
- `all_ind`: All SAP indexes
- `special`: Only tables and indexes defined in the [next\\_special](#) parameter

This option overrides the [next\\_table](#) parameter.

Note

To specify all SAP tables with indexes enter `-t all -f nocase`.

## **-f stats**

This BRCONNECT [function](#) updates optimizer statistics. For more information, see [Update Statistics](#).

Function options:

- **-b|-buckets:** defines the number of buckets in histograms

**Input syntax:** `-b|-buckets <count> auto|skewonly|repeat`

**Default value:** 75

This option overrides the [stats\\_bucket\\_count](#) parameter.

- `count:` number of buckets
- `auto:` creates histograms based on data distribution and use
- `skewonly:` creates histograms on columns based on data distribution
- `repeat:` creates histograms only for columns that already have histograms

For more information, see the Oracle documentation.

- **-c|-change:** changes threshold for the percentage of inserted, updated, or deleted rows causing update statistics

**Input syntax:** `-c|-change <threshold>`

**Default value:** 50

This option overrides the [stats\\_change\\_threshold](#) parameter.

- **-d|-delete:** deletes only damaging table and index statistics

**Default value:** Collect outdated and delete damaging statistics

You can use this option to delete statistics for pool and cluster tables (only in Oracle 9i), and for tables specified in the `DESTATC` control table with the active flags set to `N` or `R`.

- **-e|-exclude:** defines tables and indexes to be excluded from update statistics

**Input syntax:**

```
-e|-exclude [<owner>.]<table>| [<owner>.] [<prefix>]* [<suffix>]|
[<owner>.]<index><tablespace>|<object_list>|all_part|info_cubes
|null
```

`all_part` excludes check and update statistics for partitioned tables and indexes

`info_cubes` excludes check and update statistics for InfoCube tables.

`null` invalidates the exclusion list defined by the [stats\\_exclude](#) parameter. This means that no tables or indexes are excluded from processing.

**Default value:** no exclusions

You can use this option to exclude tables or indexes with exceptional statistics handling. This option overrides the [stats\\_exclude](#) parameter.

- **-f|-force:** forces a specific action while updating statistics

**Input syntax:**

```

-f|-force
[achist][,allcol][,alldef][,allsel][,autoall][,autocasc][,autogran]

[,autoinv][,buckets][,chkunb][,collect][,dchist][,degree][,delete][,dimcol]

[,globgran][,history][,ichist][,indcol][,keep][,limit][,lock][,locked][,method]

[,monit][,nocasc][,nocheck][,nocoll][,noinval][,onlyind][,pchart][,precision]

[,redcol][,sample][,space][,unlock]

```

Default value: [Internal rules determine the update statistics method](#)

- achist: creates histograms for all columns
- allcol: checks whether all columns have statistics
- alldef: uses all DBMS\_STATS defaults
- allsel: updates statistics for all selected objects (option `-t` or the `stats_table` parameter), including pool and cluster tables (relevant in Oracle 9i), or deletes statistics for selected objects, including non-pool and non-cluster tables
- autoall: uses all DBMS\_STATS auto options
- autocasc: uses auto-cascade on indexes for new table statistics
- autogran: uses auto-granularity on partitioned tables
- autoinv: uses auto-invalidation of dependent cursors
- buckets: uses option `-b` for number of buckets (overrides DBSTATC setting)
- chkunb: checks unbalanced indexes
- collect: updates statistics without checking them first
- dchist: reduces columns for histograms
- degree: uses option `-g` for degree of parallelism (overrides DBSTATC setting)
- delete: deletes statistics before recreating them
- dimcol: reduces update of column statistics

`dimcol` creates statistics only on indexed columns when one of the following conditions is met:

- The table has at least 100,000,000 rows and column statistics are no older than 30 days
- The table has at least 10,000,000 but not more than 99,999,999 rows, and column statistics are no older than 20 days

- The table has at least 1,000,000 but not more than 9,999,999 rows, and column statistics are no older than 10 days
  - `globgran`: uses global granularity for partitioned tables
  - `history`: stores the results of update statistics in the history tables `DBSTATHORA` and `DBSTAIHORA`, also for tables specified in the `DBSTATC` control table, where the history flag is not set there
  - `ichist`: creates histograms only for indexed columns
  - `indcol`: creates statistics only for indexed columns
  - `keep`: does not delete statistics after updating them for pool and cluster tables (relevant in Oracle 9i) (option `-f allsel`) or for tables with the active flag set to `N` or `R` in `DBSTATC`
  - `limit`: forces hard processing time limit defined in option `-l` or [stats\\_limit\\_time](#) parameter. Working threads are aborted.
  - `lock`: locks statistics after check or collect
  - `locked`: creates new statistics for tables with locked statistics
  - `method`: uses the method defined in option `-m` or [stats\\_method](#) parameter, also for tables specified in `DBSTATC`. See [stats\\_method](#) parameter
  - `monit`: `BRCONNECT` automatically sets the `MONITORING` attribute for all tables without this attribute and forces update statistics for them.
  - `nocasc`: prevents update statistics from being implicitly performed for all indexes of the selected tables. Only use this option in exceptional situations.
  - `nocheck`: does not check statistics. Instead, determine which tables to update statistics for by using the check results from the previous `BRCONNECT` run with `-f nocoll`. This is part of the two-phase concept for update statistics.
  - `nocoll`: checks statistics only by analyzing the primary index. Statistics are updated for tables that need new statistics in the next `BRCONNECT` run with `-f nocheck`. This is part of the two-phase concept for update statistics.
  - `noinval`: no invalidation of dependent cursors
  - `onlyind`: creates statistics only for indexes
  - `pchist`: creates histograms only for partitioning columns
  - `precision`: forces minimum precision (that is, sample size) defined in option `-s` or [stats\\_sample\\_size](#) parameter for all tables if statistics are collected with method `E`.
  - `redcol`: reduces update of column statistics
- `redcol` creates statistics only on indexed columns except:
- BW master, operational data store (ODS), or temporary tables, where statistics are created for all columns
  - Partitioned InfoCube tables (except for master tables), where statistics are also created for partitioning columns

- o `sample`: uses sample size defined in option `-s` or [stats\\_sample\\_size](#) parameter, also for tables specified in the control table `DBSTATC`
  - o `space`: collects space statistics, taking into account space allocated to LOB segments
  - o `unlock`: unlocks table statistics before check and collect
- `-g|-degree`: defines the degree of parallelism used by `DBSM_STATS` for update statistics

Input syntax: `-g|-degree <number>|auto|default|null`

Default value: `null`

This setting is valid for all tables, for which there is no parallelism setting in [stats\\_dbms\\_stats](#). However, note that the setting `-f degree` (see above), if used, takes precedence over the setting in `stats_dbms_stats`. You can also set a table-specific degree of parallelism using the table `DBSTATC`.

- o `auto`: auto degree
  - o `default`: Oracle default degree
  - o `null`: table degree
- `-h|-history`: stores the results of updating statistics in the history tables `DBSTATHORA` and `DBSTAIHORA` for tables not specified in `DBSTATC`

Input syntax: `-h|-history`

Default value: no history records are saved

The history data in the tables `DBSTATHORA` and `DBSTAIHORA` is used by the application monitor.

- `-i|-interval`: interval for collecting system (CPU, I/O) statistics

Input syntax: `-i|-interval <minutes>`

Default value: 10

You can also determine the interval for collecting system statistics using the `init<SID>.sap` parameter [stats\\_system\\_interval](#).

- `-l|-limit`: defines the processing time limit in minutes for updating statistics

Input syntax: `-l|-limit <minutes>`

Default value: 0, no limit

You can use this parameter to terminate long-running update statistics jobs after a certain period of time. The processing terminates after statistics have been collected for the current table or index (this is the “soft limit”). If you set the option `-f limit` (see above), processing terminates immediately (this is the “hard limit”).

This option overrides the [stats\\_limit\\_time](#) parameter.

- `-m|-method`: defines the method for updating statistics for tables that are not specified in the control table `DBSTATC`

**Input syntax:** `-m|-method E|EH|EI|EX|C|CH|CX|A|AH|AI|AX|`

`E|=C|=H|=I|=X|+H|+I`

**Default value:** [Internal rules](#) determine the update statistics method

This option overrides the [stats\\_method](#) parameter.

**E:** estimates

**EH:** estimates with histograms

**EI:** estimates with index validation

**EX:** estimates with histograms and index validation

**C:** computes

**CH:** computes with histograms

**CI:** computes with index validation

**CX:** computes with histograms and index validation

**A:** estimates with auto-sample size (DBMS\_STATS)

**AH:** estimates with auto-sample size (DBMS\_STATS) and histograms

**AI:** estimates with auto-sample size (DBMS\_STATS) and index validation

**AX:** estimates with auto-sample size (DBMS.stats) and histograms and index validation

**E=:** forces estimate for all specified tables, including tables in DBSTATC control table. Option `-f method` must be set

**C=:** forces compute for all specified tables, including tables in DBSTATC control table. Option `-f method` must be set

**=H:** forces collect statistics with histograms for all specified tables, including tables in DBSTATC control table. Option `-f method` must be set

**=I:** forces collect statistics with index validation for all specified tables, including tables in DBSTATC control table. Option `-f method` must be set

**=X:** forces collect statistics with histograms and index validation for all specified tables, including tables in DBSTATC control table. Option `-f method` must be set

**+H:** forces collect statistics with histograms for all tables, including tables in DBSTATC control table in addition to histograms, if specified in DBSTATC control table. Option `-f method` must be set.

**+I:** forces collect statistics with index validation for all tables, including tables in DBSTATC control table in addition to index validation, if specified in DBSTATC control table. Option `-f method` must be set.

- `-m|-ignore:` specifies that BRCONNECT ignores the settings in control table DBSTATC. Instead, it uses settings from the [stats\\_special](#) parameter of profile `init<DBSID>.sap`, if available
- `-o|-owner:` defines the database owner of tables and indexes for updating statistics



Input syntax: `-o|-owner <owner1>[,<owner2>, ...]`

Default value: `SAPR3/SAP<SID>` in a standard SAP database or all SAP owners in a multi-schema database

This option overrides the [stats\\_owner](#) parameter.

- `-p|-parallel`: defines the number of parallel threads for updating statistics

Input syntax: `-p|-parallel <number>`

Default value: 1

For example, you can set this parameter to the number of CPUs to speed up update statistics.

This option overrides the [stats\\_parallel\\_degree](#) parameter.

- `-r|-retain`: skips check and update of statistics of tables for which statistics were checked or updated in the specified time period

Input syntax: `-r|-retain <days>|last`

Default value: `last`

- `<days>`: skips tables and indexes for which statistics were checked or updated in the last `<days>` days
- `last`: skips tables and indexes for which statistics were checked or updated in the last BRCONNECT run. You can use this option to restart an aborted BRCONNECT run of update statistics.

- `-s|-sample`: defines the sample size in percentage or thousands of rows for updating statistics with method E for tables that are not specified in the DBSTATC control table

Input syntax: `-s|-sample P<p>|R<r>`

Default value: [Internal rules](#) determine the update statistics method

This option overrides the [stats\\_sample\\_size](#) parameter.

- `P<p>`: percentage of rows, where `<p>` is from 1 to 100 and from .001 to .999.
- `R<r>`: number of thousand rows

#### Note

While using `DBMS_STATS`, you can set the percentage of rows to less than 1, which is useful to improve performance on very large tables. For more information, see [Sample Sizes for Update Statistics](#).

For example, to set the sample size to 0.05%, enter `-s P.05`.

- `-t|-table`: defines the objects to be processed by update statistics

Input syntax:

`-t|-table all|all_ind|all_part|missing|harmful|dbstatc_tab|dbstatc_mon|dbstatc_mona| [<owner>.]<table>| [<owner>.]<index>|`

```
[<owner>.] [<prefix>]* [<suffix>] | <tablespace> |  
<object_list> | info_cubes | locked | system_stats |  
oradict_stats | oradict_tab
```

This option overrides the [stats\\_table](#) parameter.

- o all: all SAP tables and indexes
- o all\_ind: processes all indexes only. For example, you can use this to create the space statistics for all indexes.
- o all\_part: processes all partitioned tables and indexes
- o missing: only tables and indexes with missing statistics
- o harmful: processes all tables and indexes with damaging statistics
- o dbstatc\_tab: only tables specified in the DBSTATC control table
- o dbstatc\_mon: only tables specified in the DBSTATC control table that are relevant for the application monitor
- o dbstatc\_mona: only application tables specified in the DBSTATC control table that are relevant for the application monitor
- o info\_cubes: checks statistics only for InfoCube tables and updates them if necessary
- o locked: all tables with locked statistics
- o system\_stats: collects system (CPU, I/O) statistics using the DBMS\_STATS package

For more information, see the `init<SID>.sap` parameter [stats\\_table](#) and *SAP Note 601395*.

- o oradict\_stats: collects statistics for Oracle dictionary objects using DBMS\_STATS package

For more information, see the `init<SID>.sap` parameter [stats\\_table](#) and *SAP Note 863811*.

#### Caution

To collect system statistics, you need to start BRCONNECT with the SYSTEM database user because the package procedure DBMS\_STATS.GATHER\_SYSTEM\_STATS need DBA authorization.

- o oradict\_tab: validates structure for Oracle dictionary objects

#### Note

You must use the `-v` option with `oradict_tab`:

```
brconnect -u / -c -f stats -t oradict_tab -v
```

- `-u|-run`: lets you stop or suspend the current run of update statistics or validation of table or index structures

Input syntax: `-u|-run stop|suspend|resume`

Default value:

- `stop`: cleanly stops processing. The table or index currently being processed is completed but no further tables or indexes are processed. It might take some time to finish processing the current table or index if it is large.
- `suspend`: suspends processing. It might take some time to finish processing the current table or index if it is large.
- `resume`: resumes processing that has been suspended

You normally perform these actions in parallel in a separate command window from the current run of update statistics or validation of table or index structures.

- `-v|-validate`: validates a table or index structure, but no update statistics occurs

Input syntax: `-v|-validate table|index|cascade|index_store|cascade_store`

Default value: `cascade`

- `table`: validates internal structure of table blocks. Tables are not locked.
- `index`: validates internal structure of index blocks. Indexes are not locked.
- `cascade`: validates internal structure of table and index blocks, including relation between index and data rows. Tables and indexes are not locked.
- `index_store`: same as `index`, but also stores the statistical values, determined during the validation of the index structures, in the `DBSTATIONORA` table, so that they are available for the application monitor (ST07). Indexes are locked.
- `cascade_store`: same as `cascade`, but also stores the statistical values, determined during the validation of the index structures, in the `DBSTATIONORA` table, so that they are available for the application monitor (ST07). Tables and indexes are locked.

We recommend performing it when there is little processing on the database. However, in Oracle 9 or higher it does not lock tables and indexes for the attributes `table`, `index`, and `cascade`, but using attributes `index_store` and `cascade_store` locks the tables and indexes.

## **-h|-help**

This BRCONNECT command option provides help information, including an overview of BRCONNECT functions.

Syntax

`-h|-help [<function>|version]`

End of the code.

Default value: displays help information about all BRCONNECT functions.

Possible values:

- `<function>`: displays help information about main options and specified functions only
- `version`: displays detailed information on the versions of the program modules

## **-l|-language**

This BRCONNECT command option sets the message language.

Syntax

```
-l|-language E|D
```

End of the code.

Default value: E

Note

The default becomes invalid if you specify another value by setting the environment variable `BR_LANG` (language variable).

If you set option `-l`, the value specified with this option applies.

Possible values:

- D: German
- E: English

## **-o|-output**

This BRCONNECT command option controls the information written to the detail log.

Syntax

```
-o|-output [detail|process|summary] [,time]
```

End of the code.

Default value: `process`

- `detail`: writes detailed processing information to the log file. The log file is then comprehensive, which can be helpful to investigate problems.
- `process`: writes detailed information to the log file, including the analysis methods and sample sizes for updating the index statistics while statistics are being checked. These are used to determine whether tables need new statistics.
- `summary`: writes only summary information and total counts to the log file. This can be useful for creating initial statistics to avoid a large log file.
- `time`: generates additional time stamps that enable you to determine the time required for the individual operations

## -p|-profile

This BRCONNECT command option defines the profile name.

Syntax

```
-p|-profile <profile>
```

End of the code.

Default value: `init<DBSID>.sap`

This profile is contained in directory `<ORACLE_HOME>/dbs` (UNIX) or `<ORACLE_HOME>\database` (Windows).

If you want to use a different profile, specify the name of the profile file here. If this file is not in the standard directory `<ORACLE_HOME>/dbs`, specify the complete path.

## -q|-query

This BRCONNECT command option sets the query mode. No processing is started.

Syntax

```
-q|-query [check|nolog]
```

End of the code.

Default value: start processing

With this option, BRCONNECT displays information about the work to be done (for example, the number of database objects to be processed) by the selected function.

- `check`: displays objects that would be changed by the function (for example, the objects for which `NEXT` extents would be adapted by the `-f next` function)
- `nolog`: does *not* create or update detail, summary, and database logs for the function

Example

```
brconnect -u / -q check -f stats
```

## -u|-user

This BRCONNECT command option defines the user name and password used by the SAP tool to log on to the database.

Syntax

```
-u|-user [<user>[/<>password>]]|/]
```

End of the code.

Default value: `system/<default_password>`

If you only enter `-u`, BRCONNECT performs an interactive query of the user name and the password. You can enter the user name and the password separately (only enter the user name or the option `-u <user>`). BRCONNECT then prompts entry of the password. In this case, the password is not displayed during entry, and does not appear in the process list.

This protects the DBA password.

In shell scripts, you can structure the call as follows:

```
brconnect -c -u -f stats <<END <user>/<password> END
```

However, use this command only if the option `-c` is active.

#### Note

If you are working with an `OPSS$` user, use the following call:

```
brconnect -u / -c -f stats -t all
```

In this case, BRCONNECT tries to log on to the database as `OPSS$` user (see Oracle documentation and information in the SAP Service Marketplace). The `OPSS$` user must be defined in the database and have at least `SYSOPER` authorization and `SAPDBA` role. With this method, it is not necessary to specify the password when calling BRCONNECT.

## -V|-VERSION

This BRCONNECT command option displays detailed information on the program version and patches.

#### Syntax

```
-V|-VERSION [ALL]
```

End of the code.

ALL: displays patch information for all BR\*Tools

## BRCONNECT Logs

For more information, see:

- [Names of the BRCONNECT Detail Logs](#)
- [BRCONNECT Detail Log](#)
- [BRCONNECT Summary Log](#)

## Names of the BRCONNECT Detail Logs

Every BRCONNECT detail log contains a name with the following format:

```
c<encoded timestamp>.<ext>
```

The first characters indicate the encoded time the restore was performed (action ID). The extension (function ID) indicates the type of processing. The logs are stored in the `sapcheck` directory.

BRCONNECT only writes logs for the functions [check](#), [cleanup](#), [next](#) and [stats](#).

Possible function IDs:

- `.chk`: [check database system](#), function [-f check](#)

- `.cln`: clean up database log, function [-f cleanup](#)
- `.nxt`: adapt next extents, function [-f next](#)
- `.sta`: check and update statistics, function [-f stats](#)
- `.dst`: delete damaging statistics, function [-f stats -d](#)
- `.opt`: check statistics only, function [-f stats -f nocoll](#)
- `.aly`: collect statistics for tables with outdated statistics, function [-f stats -f nocheck](#)
- `.vst`: verify table and index structure, function [-f stats -v](#)
- `.quc`: determine objects to be processed for a given function, command option [-ql-query](#)

## BRCONNECT Detail Log

The detail log file contains information about the actions that were performed by BRCONNECT:

- The relevant parameters of initialization profile `init<DBSID>.sap` that were set during the BRRESTORE run
- Information about the numbers of objects to be processed
- For the check function, the conditions to be checked
- Processing details:
  - [Check function](#): alert conditions detected
  - [Cleanup function](#): database logs that were deleted
  - [Next function](#): tables and indexes for which the next extent was adapted
  - [Stats function](#): tables and indexes for which statistics were collected
- Summary information and total counts of objects processed

## BRCONNECT Summary Log

You can display a brief entry for each restore in the summary log `conn<DBSID>.log`. The logs are stored in the `sapcheck` directory. The entries in the file provide the following information about each function using BRCONNECT:

- Action ID (encoded timestamp of the log name)
- Function ID (extension of the log name)
- Timestamp (date, time) specifying the start of the function
- Timestamp (date, time) specifying the end of the function
- Return code

- BRCONNECT version
- BRCONNECT function and object for `next` and `stats`

## BRTOOLS

You can use BRTOOLS as a tool:

To display the menus for the [BR\\*Tools user interface](#) using a character-based interface

That is started internally by BRBACKUP, BRARCHIVE, and BRRESTORE

Note

Distinguish between the following:

[BR\\*Tools](#) is the program package containing BRSPACE, BRBACKUP, BRARCHIVE, BRRESTORE, BRRECOVER, BRCONNECT, and BRTOOLS.

BRTOOLS is the program that displays the character-based menus from which the other BR programs are called. It works together with BRGUI to generate a graphical user interface.

## Features

BRTOOLS as an internal tool started by BRBACKUP, BRARCHIVE, and BRRESTORE has the following features:

- Backup verification
- Saved files are compared with their originals

For more information, see [-w|-verify](#).

The option `-c force`

This denies the commands that are started (such as `cpio`) access to the console.

For more information, see [-c|-confirm](#).

[Parallel backup](#) to a remote host (`backup_dev_file = pipe`).

## Command Options for BRTOOLS

This section describes the command options for [BRTOOLS](#).

If you use BRTOOLS with command options, these override the corresponding values in the [initialization profile `init<DBSID>.sap`](#). To use the options, you can specify either the letter indicated or the complete word.

This is the schematic command syntax:

Syntax

```
brtools [<options>]
```

End of the code.

This is the full command syntax:



## Syntax

```
brtools  
[-c|-confirm]  
[-h|-help [version]]  
[-i|-interval]  
[-l|-language E|D]  
[-p|-profile <profile>]  
[-s|-scroll <lines>]  
[-u|-user [<user>[/<password>]]|/]  
[-w|-show <days>]  
[-V|-VERSION [ALL]]
```

End of the code.

See also:

[-c|-confirm](#)

[-h|-help](#)

[-i|-interval](#)

[-l|-language](#)

[-p|-profile](#)

[-s|-scroll](#)

[-u|-user](#)

[-w|-show](#)

[-V|-VERSION](#)

## **-c|-confirm**

This BRTOOLS command option specifies whether processing is attended or unattended. In unattended mode, BRTOOLS only stops at menus and yes/no queries. At other prompts, it continues processing with the default value.

### Syntax

```
-c|-confirm
```

End of the code.

Default value: attended mode. You need to respond to the prompts and menus generated by BRSPACE. You also have to check the default choices and input values suggested by BRSPACE.

## **-h|-help**

This BRTOOLS command option provides help information and command line options about the version of BRTOOLS specified.

### Syntax

```
-h|-help [version]
```

End of the code.

Default value: no help

Possible value:

`version`: displays detailed information on versions of the program modules

## **-i|-interval**

This BRTOOLS option sets the recovery interval. It is used to select backups for delete and verification. It is also passed to BRRECOVER.

Syntax

```
-i|-interval <days>
```

End of the code.

Default value: 30

This option corresponds to the `init<DBSID>.sap` parameter [recov\\_interval](#).

## **-l|-language**

This BRTOOLS command option sets the language for messages.

Syntax

```
-l|-language E|D
```

End of the code.

Default value: E

Note

The default becomes invalid if you specify another value by setting the environment variable `BR_LANG` (language variable).

If you set option `-l`, the value specified with this option applies.

Possible values:

- D: German
- E: English

## **-p|-profile**

This BRTOOLS command option defines the profile name.

Syntax

```
-p|-profile <profile>
```

End of the code.

Default value: `init<DBSID>.sap`

This profile is normally contained in the standard directory <ORACLE\_HOME>/dbs (UNIX) or <ORACLE\_HOME>\database (Windows).

If you want to use a different profile, specify the name of the profile file here. If this file is not in the standard directory, specify the complete path.

## **-s|-scroll**

This BRTOOLS command specifies the number of lines for scrolling in list menus. This option is not valid for BRGUI.

Syntax

```
-s|-scroll <lines>
```

End of the code.

Default value: 20

This option corresponds to the parameter [scroll\\_lines](#) in `init<DBSID>.sap`.

## **-u|-user**

This BRTOOLS command option defines the user name and password for RMAN to connect to the database.

Syntax

```
-u [<user>[/<password>]]|/]
```

End of the code.

Default value: `system/<default_password>`

If you only enter `-u`, an interactive query of the user name and the password is performed by the SAP tool. You can enter the user name and the password separately (only enter the user name or the option `-u <user>`). The tool then prompts entry of the password. In this case, the password is not displayed during entry and does not appear in the process list.

These measures are taken to protect the DBA password.

If you enter `-u /` the Oracle OPS\$ mechanism is used.

## **-w|-show**

This BRTOOLS command option specifies the time period for which BRTOOLS displays log files.

Syntax

```
-w|-show <days>
```

End of the code.

Default: 30

Possible values:

- 0: shows all available log files
- 1: shows log files created today

- >1: shows log files created in the previous specified number of days

This command corresponds to the parameter [show\\_period](#) in `init<DBSID>.sap`.

## **-V|-VERSION**

This BRTOOLS command option displays patch information of BRTOOLS.

Syntax

```
-V|-VERSION [ALL]
```

End of the code.

ALL: displays patch information for all BR\*Tools

## **Profiles, Logs, Messages, and Return Codes for BR\*Tools**

This section deals with the [profiles, logs, messages and return codes](#) for BR\*Tools.

### **Initialization Profile `init<DBSID>.sap`**

The initialization profile `init<DBSID>.sap` contains parameters that influence how the SAP tools perform various functions. It is usually stored in directory `<ORACLE_HOME>/dbs` (UNIX) or `<ORACLE_HOME>\database` (Windows).

To configure the SAP tools BRBACKUP, BRARCHIVE, BRRESTORE, BRRECOVER, BRCONNECT, BRSPACE, and BRTOOLS, you must use the initialization profile `init<DBSID>.sap`. You can edit the file with a text editor. If you do not make any changes, the SAP tools use the default values for the parameters.

Before you use one of the SAP tools, find out exactly which parameters you have to configure. Pay particular attention to parameters without default values and parameters that have device-specific information or require special platform-specific commands.

### **Structure**

The parameters and values in profile `init<DBSID>.sap` look as follows:

```
<Parameter> = <value>|(<value_list>)
```

```
where value_list = <value_1>,<value_2>
```

You separate the individual values in a value list by commas, and enclose the entire list in parentheses. You can use blanks between any symbols of such commands. If necessary, you can continue the parameter values can be continued on the next line (the line break is then treated as a blank). If a parameter value contains special characters such as space or \$, you must enclose the value in double quotes, as in the following example:

```
rewind = "mt -f $ rewind"
```

If you are unsure, compare your input format with the format in the sample profile, `/usr/sap/<SAPSID>/SYS/exe/run/initSID.sap`, or the initialization profile, `<ORACLE_HOME>/dbs/init<DBSID>.ora`.

## Integration

You can override many of the parameters in `init <DBSID>.sap` by using a command option when you call BRBACKUP, BRARCHIVE, BRCONNECT, or BRRESTORE. For more information, see:

- [Effects of the Command Options](#)
- [Command Options for BRBACKUP](#)
- [Command Options for BRARCHIVE](#)
- [Command Options for BRCONNECT](#)
- [Command Options for BRRESTORE](#)

Changes to parameter values do not take effect until you call the corresponding tool.

## archive\_copy\_dir

This parameter defines the directory used by BRARCHIVE to back up the offline redo log files to a local disk.

Syntax

```
archive_copy_dir = <dir>
```

End of the code.

Default value: first value of the parameter [backup\\_root\\_dir](#)

Required value:

<dir>: a directory in which the offline redo log files are to be backed up

This is generally only required when you are working with a two-phase backup procedure, which is similar to a [Two-Phase Backup](#). The offline redo log files are first collected in a directory on a disk and then written to tape using [BRARCHIVE option -a](#) or external means. If external tools are used, the user is responsible for backing up to tape and for restoring from the tape to the disk, if this become necessary in connection with a recovery.

Caution

- Only use BRARCHIVE backing up to disk in the situations mentioned above. In all other cases, always back up the offline redo log files to tape. Backing up to disk is *not* a substitute for backing up the offline redo log files to tape.
- You *cannot* combine back up to disk and tape during a single BRARCHIVE run.
- When you back up the offline redo log files to disk, you can only use the BRARCHIVE options `-s`, `-sd`, and `-ds`.

## archive\_dupl\_del

This parameter defines a second, or "duplex", copy of the offline redo log files created by BRARCHIVE.

### Syntax

```
archive_dupl_del = only|yes|no|check
```

End of the code.

Default value: `only`

Possible values:

- `only`: deletes offline redo log files created by the Oracle parameter `log_archive_duplex_dest`, but does *not* delete those created with the Oracle parameter `log_archive_dest_2`
- `no`: does *not* delete offline redo log files created by either the Oracle parameter `log_archive_duplex_dest` or the Oracle parameter `log_archive_dest_2` (this is the same as setting the environment variable `BR_NDD`)
- `yes`: deletes offline redo log files created by either the Oracle parameter `log_archive_duplex_dest` or the Oracle parameter `log_archive_dest_2`
- `check`: deletes offline redo log files created by either the Oracle parameter `log_archive_duplex_dest` or the Oracle parameter `log_archive_dest_2` *only when* a binary compare with the first copy of the offline redo log file has completed successfully. This protects you against data corruption.

## archive\_function

This parameter defines the type of BRARCHIVE backup.

### Syntax

```
archive_function = save|second_copy|delete_saved|  
deleted_copied|save_delete|second_copy_delete|  
double_save|double_save_delete|copy_save|copy_delete_save
```

End of the code.

Default value: `save`

Possible values:

- `save`: backs up the offline redo log files
- `second_copy`: creates a second copy of offline redo log files that were already backed up
- `delete_saved`: deletes offline redo log files that were backed up once
- `deleted_copied`: deletes offline redo log files that were copied a second time
- `save_delete`: backs up the offline redo log files and then deletes these files
- `second_copy_delete`: creates a second copy of offline redo log files that were already backed up and then deletes these files

- `double_save`: backs up the offline redo log files on two backup devices (tape devices) in parallel
- `double_save_delete`: backs up the offline redo log files on two backup devices (tape devices) in parallel and then deletes the files
- `copy_save`: creates a second copy of offline redo log files that were already backed up and then back up the offline redo log files that have been created in the meantime
- `copy_delete_save`: creates a second copy of offline redo log files that were already backed up and then deletes these files. The offline redo log files that have been created in the meantime are backed up.

If there is only one tape device, you can use parameter `archive_function = copy_save` or `copy_delete_save` to ensure that BRARCHIVE creates a second copy of the offline redo log files in one run, deletes it if necessary and continues backing up immediately. This can also be done by first calling BRARCHIVE with `archive_function = save` and then with `archive_function = second_copy` or `second_copy_delete`. However, two BRARCHIVE calls with modified parameters are necessary in this case. In practice, we recommend you to only use the second possibility with the corresponding BRARCHIVE command option call if at all.

## archive\_stage\_dir

This parameter identifies the directory used by BRARCHIVE to back up the offline redo log files to a remote disk. This parameter corresponds to [archive\\_copy\\_dir](#) for a backup to a local disk.

### Syntax

```
archive_stage_dir = <dir>
```

End of the code.

Default value: first value of the parameter [stage\\_root\\_dir](#)

Required value:

<dir>: a backup directory for the offline redo log files

### Example

```
archive_stage_dir = $SAPDATA_HOME/sapbackup
```

See also:

[Backup to a Remote Disk](#)

## backup\_dev\_type

This parameter specifies the backup medium that you want to use.

### Syntax

```
backup_dev_type = disk|tape|pipe|tape_auto|
pipe_auto|tape_box|pipe_box|disk_copy|disk_standby|
util_file|util_file_online|util_vol|util_vol_online|stage|
stage_copy|stage_standby|rman_prep|rman_util|rman_disk|rman_stage
```

End of the code.

Default value: `tape`

Possible values:

- `disk`: database backups to disk  
  
The directory that should be used for the backups is defined in parameter `backup_root_dir` or `archive_copy_dir`.
- `tape`: uses one or more local tape devices.
- `pipe`: backs up to a remote system using the commands entered in the profile parameters [remote\\_host](#), [remote\\_user](#), [copy\\_in\\_cmd](#), and [copy\\_out\\_cmd](#). All the functions that are available for a local backup on tape are also available here, but only on UNIX.
- `tape_auto`: uses a local tape device with a tape changer. The prompts for changing the volumes are suppressed.
- `pipe_auto`: uses a tape device with a tape changer on a remote system. The prompts for changing the volumes are suppressed. When you use this parameter specification, do not forget to set the parameters `copy_in_cmd`, `copy_out_cmd`, `remote_host` and `remote_user` appropriately.

The parameters `tape_auto` and `pipe_auto` generally have no effect on BRARCHIVE, since that program does not support continuation tapes.

- `tape_box`: uses jukeboxes and autoloader. The accompanying tape devices must be locally accessible.
- `pipe_box`: uses jukeboxes and autoloader. The accompanying tape devices must be remotely accessible (`remote_host`, `remote_user`).

For all tape device types, the drivers defined in the parameter `tape_address` or `tape_address_arch` are used for the data transfer (`cpio`, `dd`). For the rewind those defined in the parameter `tape_address_rew` or `tape_address_arch_rew` are used.

- `disk_copy`: copies database files to a disk with an identical directory structure. The name of the new `Oracle_Home` directory is defined in the parameter `new_db_home`. See [Structure-Retaining Database Copy](#).
- `disk_standby`: copies database files to a disk with an identical directory structure (compare `disk_copy`). To let you construct a standby database, a standby control file is generated and copied. See [Standby Database Configuration](#).
- `util_file`: backs up or restores file-by-file using the backup program specified by the BACKINT interface program

#### Note

Since backups at disk-volume level are not valid for backups of archive log files, BRARCHIVE ignores `util_vol` and `util_vol_online` (see below) and instead always uses `util_file`.

- `util_file_online`: (if supported by the manufacturer of the external backup program) backs up file-by-file using an external backup program addressed by the



BACKINT interface. The backup status is also set and ended dynamically for the tablespaces to be saved in an online backup. This value can also be set for an offline backup, in which case the database is not stopped before calling BACKINT. Instead, it is stopped when the first file is about to be backed up by BACKINT and started again after the last file has been saved.

- `util_vol`: backs up or restores volume-by-volume using the backup program specified by the BACKINT interface program. The resulting BACKINT call is `-t volume`.
- `util_vol_online`: (if supported by the manufacturer of the external backup program) backs up volume-by-volume using an external backup program addressed by the BACKINT interface. The backup status is also set and ended dynamically for the tablespaces to be saved in an online backup. This value can also be set for an offline backup, in which case the database is not stopped before calling BACKINT. Instead, it is stopped when the first volume is about to be backed up by BACKINT and started again after the last volume has been saved. The BACKINT call is `-t volume online`.

Only BRBACKUP accepts `util_vol_online`. BRRESTORE ignores it and uses `util_vol` instead.

#### Note

The parameters `util_file` or `util_vol` let you use other storage media (for example, optical media), provided the supplier provides corresponding backup programs and a BACKINT interface. In such cases, the SAP utilities can call the external backup program for the physical backup or restore of the corresponding files. If you use parameter `util_file_online` or `util_vol_online`, the volume of offline redo log files is also significantly reduced during an online backup. See [External Backup Programs](#).

Ask the supplier of the non-SAP backup programs and the interface BACKINT for any additionally required parameters for the backup program call. If necessary, store this information in an appropriately maintained parameter file, which you should also enter in the parameter `util_par_file` or the command option `-r|-parfile` of the SAP tools.

- `stage`: backs up to a remote disk. This can be used for standard backups with BRBACKUP, for incremental backups while using the RMAN functions. See [Backup to a Remote Disk](#).
- `stage_copy`: copies database files to a remote disk with an identical directory structure. The name of the new `Oracle_Home` directory is defined in the parameter `stage_db_home`.
- `stage_standby`: copies database files to a remote disk with an identical directory structure (compare `disk_copy`). To let you construct a standby database, this also generates and copies a standby control file.
- `rman_prep`: determines the best distribution of the files to save sets before a RMAN backup to tape with file multiplexing. See [RMAN Save-Set Grouping](#).
- `rman_util`: performs an RMAN backup in combination with a backup library and the backup tool of another manufacturer. BACKINT provides an interface to the external backup tool and is also used to back up profiles, log files and the control file. See [RMAN Backup with an External Backup Library](#).

- `rman_disk`: backs up with an external backup library and RMAN, but without BACKINT. Copies profiles and log files to local disk.
- `rman_stage`: backs up with an external backup library and RMAN, but without BACKINT. Copies profiles and log files to remote disk.

## backup\_mode

This parameter is used by BRBACKUP to determine the scope of the backup activity.

### Syntax

```
backup_mode = all|all_data|full|incr|<tablespace>|
<file_ID>|<file_ID1>-<file_ID2>|<generic_path>|
sap_dir|ora_dir|all_dir|<object_list>
```

End of the code.

Default value: all

Possible values:

- `all`: performs whole database backup using BRBACKUP
- `all_data`: saves the files of all tablespaces that are not pure index tablespaces or empty
- `full`: full database backup (level 0). See [Incremental Backup](#).
- `incr`: incremental backup with RMAN. See [Incremental Backup](#).
- `<tablespace>`: BRBACKUP backs up the file of the specified tablespaces.
- `<file_ID>`: backs up the file with this file ID. For data files, this is the Oracle file ID. Control files can be addressed with the file ID 0. Online redo log files can be addressed using the file ID 0<n>, <n> is the redo log group number. Specify file ID 00 to back up all existing online redo log files. Temporary files are identified by negative numbers.
- `<file_ID1>-<file_ID2>`: backs up the files in this interval
- `<generic_path>`: by entering a full path, you can save database files, non-database files, or the specified directory. By entering a generic path, you can save database data files whose name starts with that path. In this case, the path must contain at least the `SAPDATA_HOME` directory and an additional generic specification (for example, `sapdata<n>`) in the path.
- `sap_dir`: with this option, you can automatically determine and save all the files of the SAP environment. This means that the following directory trees are saved: `/sapmnt/<SAPSID>`, `/usr/sap/<SAPSID>`, `/usr/sap/trans` This option could, for example be used after an SAP upgrade. Saving with the `sap_dir` option should not replace regular backups of the file systems using operating system tools.

### Note

You can only use this option when saving to tape and when not performing a verification of the backup.

- `ora_dir`: with this option, you can automatically determine and save all the non-database files of the Oracle environment. This means that the directory trees under `<ORACLE_HOME>` (except for the directories `sapdata<n>` and `saplog` or `origlog/mirrlog`) are saved. You could, for example, use this option after an Oracle migration. Saving with the option `ora_dir` should not replace regular backups of the file systems with operating system tools.

Note

You can only use this option when saving to tape and when not carrying out a verification of the backup.

- `all_dir`: combines `sap_dir` and `ora_dir`. It means the same as `-m sap_dir,ora_dir`.
- `<object list>`: specifies an object list, as described in [Initialization Profile init<DBSID>.sap](#). This list can also include the key word `all`. However, we recommend processing database files and non-database files separately.

Note

For UNIX systems: Start BRBACKUP to save the SAP or Oracle environment (`backup_mode = sap_dir | ora_dir`) under user `root`, as otherwise you will not have the authorizations required for the directory to be saved.

Saving and restoring under `root` also has the advantage that you can be sure that the settings for the user and authorizations for the files and directories will be kept after restoring.

Note

The `root` user must have the environment of the corresponding `ora<sid>` user for BRBACKUP to start successfully

## backup\_root\_dir

This is the parameter that is used only by BRBACKUP to identify the directories in which database backups to disk are performed. In exceptions – for example, parameter `archive_copy_dir` is not defined – BRARCHIVE also uses the directories defined in this parameter if disk backup is required. If you enter more than one directory, you must enclose the names in parentheses and separate them by commas.

If you do not have enough storage space in one of the directories on your disk, make sure that you provide directories on other disks and add their names in `<dir_list>`. BRBACKUP then uses these directories for database backup.

Syntax

```
backup_root_dir = <dir>|(<dir_list>)
```

End of the code.

Default value: `<SAPDATA_HOME>/sapbackup`

Required value:

<dir>| (<dir\_list>: directory or directories where the disk backups are to be written

Note

See also:

[Backup to Multiple Disks](#)

## backup\_type

This BRBACKUP parameter identifies the type of the database backup.

Syntax

```
backup_type = online|online_cons|offline|
offline_force|offline_standby|online_standby|
online_split|offline_split|offline_stop|
online_mirror|online_mirror|
offstby_split|offstby_mirror
```

End of the code.

Default value: offline

Possible values:

- **online**: database backup in online mode, in other words, with the database running
- **online\_cons**: database backup in online mode. As well as the database files, the offline redo log files generated during the backup are copied to the same volume. You then have a logically consistent dataset available. This backup of the offline redo log files with BRBACKUP runs completely independently of other BRARCHIVE backups.
- **offline**: database backup in offline mode, in other words, the database is shut down during backup. When you select this parameter, BRBACKUP checks that no SAP system users are connected to the database. If an SAP System is active, the database is not shut down and BRBACKUP terminates the process with an error message (message number BR0068E).
- **offline\_force**: database backup in offline mode, in other words, the database is shut down during backup. BRBACKUP always shuts down the database, even if the SAP System is active.
- **offline\_standby**: data backup of a standby database in offline mode; in other words, the standby database is shut down during the backup. This backup mode is only relevant for the standby database configuration. See [Standby Database](#).
- **online\_standby**: data backup of a standby database in online mode, which means that the standby database remains mounted during the backup. This backup mode is only relevant for the standby database configuration. See [Standby Database](#).
- **online\_split**: the mirror disks are split and backed up while the database is running. The tablespaces to be backed up are only placed in BACKUP status during the split.

This backup mode is only relevant for the split command scenario of [Split-Mirror Disk Backup](#).

- `offline_split`: the database is only shut down to split the mirror disks. The backup of the mirror disks can take place whilst the database is running again. The SAP system is running during the entire split mirror backup. However, no transactions can be performed during the short period of time that the database is shut down.

This backup mode is only relevant for the split command scenario of [Split Mirror Backup](#).

- `online_mirror`: the mirror disks are split and backed up while the database is running. The tablespaces to be backed up are only placed in BACKUP status during the split. BRBACKUP calls SPLITINT to perform the split.

This backup mode is only relevant for the SPLITINT scenario of [Split Mirror Backup](#).

- `offline_mirror`: the database is only shut down to split the mirror disks. The backup of the mirror disks can take place whilst the database is running again. The SAP system is running during the entire split mirror backup. However, no transactions can be performed during the short period of time that the database is shut down. BRBACKUP calls SPLITINT to perform the split.

This backup mode is only relevant for the SPLITINT scenario of [Split Mirror Backup](#).

- `offline_stop`: database backup in offline mode without a consequent startup of the database. After its backup the database can be transferred directly into the mount standby status. This type of backup is only relevant in the following case: The productive database is saved and then takes over the role of a standby database. The backup itself becomes a productive system. For more information, see [Standby Database](#).
- `offstby_split`: in the split command scenario, the *standby* database is stopped for the splitting of the mirror disks. For more information, see [Split Mirror Backup](#).
- `offstby_mirror`: in the SPLITINT scenario, the *standby* database is stopped for the splitting of the mirror disks. The backup of the mirror disks is then done directly afterwards, while the standby database is mounted. BRBACKUP calls SPLITINT to perform the split. For more information, see [Split Mirror Backup](#).

See also:

[Online and Offline Backup](#)

## check\_cond

This parameter lets you emulate the table DBCHECKORA if you want to run check database in a Java database, in which the table DBCHECKORA does not exist or when you have deliberately decided to ignore DBSTATC using [-f check -ij-ignore](#). You can use `check_cond` to customize the check conditions.

### Syntax

```
check_cond =
("<type>:<cond>:<active>:<sever>:[<chkop>] :
[<chkval>]:[<unit>]",...)
```

End of the code.

Possible values:

- `<type>`: check condition

- DBA: database administration
- DBO: database operations
- ORA: database messages
- PROF: database profile
- <cond>: name of the check condition
- <active>: active flag of the proof condition
  - Y: yes (active)
  - N: no (inactive)
- <sever>: severity of the check condition
  - E: error
  - W: warning
  - A: exception
- <chkop>: check condition operator:
  - ==: equal
  - <>: not equal
  - <: smaller
  - <=: smaller or equal
  - >: bigger
  - >=: bigger or equal
  - ><: bigger or smaller
- <chkval>: check value of the check condition
- <unit>: unit of the check value
  - K: kilo
  - M: mega or minutes
  - G: giga
  - P: percent
  - S: seconds
  - H: hours
  - D: days
  - R: rate

Example

You can use this example as a template for your own settings.

This example corresponds to the standard settings plus the profile conditions for check conditions with Oracle 10g:

```
check_cond =
"DBA:ARCHIVER_STUCK:Y:W:>:90:P:",
"DBA:CONTROL_FILE_MIRROR:Y:E::::",
"DBA:CONTROL_FILE_MISSING:Y:E::::",
"DBA:CRITICAL_FILE:Y:W::::",
"DBA:CRITICAL_SEGMENT:Y:W:<=:2:",
"DBA:CRITICAL_TABLESPACE:Y:W::::",
"DBA:DATA_FILE_MISMATCH:Y:E::::",
"DBA:DATA_FILE_MISSING:Y:E::::",
"DBA:FILE_OFFLINE:Y:E::::",
"DBA:FILE_SYSTEM_FULL:Y:W:>:99:P:",
"DBA:HARMFUL_STATISTICS:Y:E::::",
"DBA:INVALID_FILE_TYPE:Y:E::::",
"DBA:IN_WRONG_TABLESPACE:Y:W::::",
"DBA:MISSING_INDEX:Y:E::::",
"DBA:MISSING_STATISTICS:Y:E::::",
"DBA:NOARCHIVELOG_MODE:Y:E::::",
"DBA:PCTINCREASE_NOT_ZERO:Y:W::::",
"DBA:REDOLOG_FILE_MIRROR:Y:E::::",
"DBA:REDOLOG_FILE_MISSING:Y:E::::",
"DBA:TABLESPACE_FULL:Y:W:>:95:P:",
"DBA:TABLESPACE_IN_BACKUP:Y:W::::",
"DBA:TABLESPACE_OFFLINE:Y:E::::",
"DBA:TOO_MANY_EXTENTS:Y:W:>:90:P:",
"DBO:ARCHIVE_TOO_OLD:Y:W:>:10:D:",
"DBO:BACKUP_TOO_OLD:Y:W:>:10:D:",
"DBO:LAST_ARCHIVE_FAILED:Y:W::::",
"DBO:LAST_BACKUP_FAILED:Y:W::::",
"DBO:LAST_OPERATION_FAILED:Y:W::::chk",
"DBO:LAST_OPERATION_FAILED:N:W::::",
```

"DBO:LAST\_STATS\_FAILED:Y:W::::",  
"DBO:OPERATION\_TOO\_OLD:Y:W:>:10:D:chk",  
"DBO:OPERATION\_TOO\_OLD:N:W:>:10:D:",  
"DBO:STATS\_TOO\_OLD:Y:W:>:10:D:",  
"ORA:00060:Y:W::::",  
"ORA:00272:Y:E::::",  
"ORA:00376:Y:E::::",  
"ORA:00600:Y:E::::",  
"ORA:01113:Y:E::::",  
"ORA:01114:Y:E::::",  
"ORA:01115:Y:E::::",  
"ORA:01122:Y:E::::",  
"ORA:01135:Y:E::::",  
"ORA:01149:Y:E::::",  
"ORA:01555:Y:W::::",  
"ORA:01562:Y:W::::",  
"ORA:01578:Y:E::::",  
"ORA:03113:Y:E::::",  
"ORA:07445:Y:E::::",  
"ORA:Checkpoint not complete:Y:W::::",  
"PROF:COMPATIBLE:Y:E:<>:10.2.0:",  
"PROF:CONTROL\_FILE\_RECORD\_KEEP\_TIME:Y:W:<:30:",  
"PROF:DB\_BLOCK\_SIZE:Y:E:<>:8192:",  
"PROF:DB\_FILES:Y:W:<:254:",  
"PROF:DB\_FILE\_MULTIBLOCK\_READ\_COUNT:Y:W::::",  
"PROF:FILESYSTEMIO\_OPTIONS:Y:E:<>:SETALL:",  
"PROF:LOG\_ARCHIVE\_START:Y:W::::",  
"PROF:LOG\_BUFFER:Y:W:><:4096,512:K:",  
"PROF:LOG\_CHECKPOINTS\_TO\_ALERT:Y:W:<>:TRUE:",  
"PROF:MAX\_DUMP\_FILE\_SIZE:Y:W:><:100000,10000:",  
"PROF:OPEN\_CURSORS:Y:W:><:2000,800:",  
"PROF:OPTIMIZER\_FEATURES\_ENABLE:Y:W::::",  
"PROF:OPTIMIZER\_INDEX\_COST\_ADJ:Y:W::::",



```

"PROF:OPTIMIZER_MODE:Y:W::::",
"PROF:PARALLEL_EXECUTION_MESSAGE_SIZE:Y:W:<>:16384,4096::",
"PROF:PARALLEL_THREADS_PER_CPU:Y:W:<>:2,1::",
"PROF:QUERY_REWRITE_ENABLED:Y:E:<>:FALSE::",
"PROF:RECYCLEBIN:Y:E:<>:OFF::",
"PROF:RECYCLEBIN:Y:E:<>:OFF::",
"PROF:REMOTE_OS_AUTHENT:Y:E:<>:TRUE::",
"PROF:REPLICATION_DEPENDENCY_TRACKING:Y:E:<>:FALSE::",
"PROF:SHARED_POOL_SIZE:Y:W:<:400:M:",
"PROF:STAR_TRANSFORMATION_ENABLED:Y:W:<>:TRUE::",
"PROF:STATISTICS_LEVEL:Y:W::::",
"PROF:TIMED_STATISTICS:Y:W::::",
"PROF:TRACE_ENABLED:Y:W::::",
"PROF:UNDO_MANAGEMENT:Y:E:<>:AUTO::",
"PROF:UNDO_TABLESPACE:Y:W:<>:PSAPUNDO::")

```

## check\_exclude

This parameter defines the database objects (that is, tables, indexes, and tablespaces) to be excluded from the [BRCONNECT check function](#).

### Syntax

```

check_exclude = [<owner>.<table>|<owner>.<index>|
<owner>.<prefix>*<suffix>|<tablespace>|
(<object_list>)|non_sap|all_part

```

End of the code.

Default value: no exclusion, all SAP tables and indexes are checked

Possible values:

- `non_sap`: excludes non-SAP objects from the check (for example, Oracle dictionary objects)
- `all_part`: excludes SAP partitions (such as in Business Information Warehouse and Advanced Planner and Optimizer) from the check

You can use this parameter to exclude tables or indexes with exceptional space parameters or statistics handling from the checks.

## check\_owner

This parameter defines the database owner of tables and indexes to be checked by the [BRCONNECT check function](#). You can use this parameter to limit the check to objects of selected SAP owners (that is, systems).

Syntax

```
check_owner = <owner>|(<owner_list>)
```

End of the code.

Default value: SAPR3/SAP<SID> in a standard SAP database or all SAP owners in a multi-schema database

## cleanup\_brarchive\_log

This parameter defines the retention period in days for BRARCHIVE detail log files, which are excluded for the specified period from the [BRCONNECT cleanup function](#).

Syntax

```
cleanup_brarchive_log = <days>
```

End of the code.

Default value: 30

## cleanup\_brbackup\_log

This parameter defines the retention period in days for BRBACKUP detail log files, which are excluded for the specified period from the [BRCONNECT cleanup function](#).

Syntax

```
cleanup_brbackup_log = <days>
```

End of the code.

Default value: 30

## cleanup\_brconnect\_log

This parameter defines the retention period in days for BRCONNECT detail log files, which are excluded for the specified period from the [BRCONNECT cleanup function](#).

Syntax

```
cleanup_brconnect_log
```

End of the code.

Default value: 30

## cleanup\_brrecover\_log

This parameter defines the retention period in days for BRRECOVER detail log files, which are excluded for the specified period from the [BRCONNECT cleanup function](#).

Syntax

```
cleanup_brrrecover_log = <days>
```

End of the code.

Default value: 30

## cleanup\_brrestore\_log

This parameter defines the retention period in days for BRRESTORE detail log files, which are excluded for the specified period from the [BRCONNECT cleanup function](#).

Syntax

```
cleanup_brrestore_log = <days>
```

End of the code.

Default value: 30

## cleanup\_brspcace\_log

This parameter defines the retention period in days for BRSPACE detail log files, which are excluded for the specified period from the [BRCONNECT cleanup function](#).

Syntax

```
cleanup_brspcace_log = <days>
```

End of the code.

Default value: 30

## cleanup\_check\_msg

This parameter defines the retention period in days for the alert messages in the DBMSGORA table from the database check runs using [BRCONNECT check function](#). The messages are excluded for the specified period from the [BRCONNECT cleanup function](#).

Syntax

```
cleanup_check_msg = <days>
```

End of the code.

Default value: 100

## cleanup\_db\_log

This parameter defines the retention period in days for records in the SDBAH and SDBAD tables, which are excluded for the specified period from the [BRCONNECT cleanup function](#).

Syntax

```
cleanup_db_log = <days>
```

End of the code.

Default value: 100

## cleanup\_disk\_archive

This parameter defines the retention period in days for offline redo log files saved on disk, which are excluded for the specified period from the [BRCONNECT cleanup function](#).

### Syntax

```
cleanup_disk_archive = <days>
```

End of the code.

Default: 30

## cleanup\_disk\_backup

This parameter defines the retention period in days for database files saved on disk, which are excluded for the specified period from the [BRCONNECT cleanup function](#).

### Syntax

```
cleanup_disk_backup = <days>
```

End of the code.

Default value: 30

## cleanup\_exp\_dump

This parameter defines the retention period in days for BRSPACE export dumps and parameter files, which are excluded for the specified period from the [BRCONNECT cleanup function](#).

### Syntax

```
cleanup_exp_dump = <days>
```

End of the code.

Default value: 30

## cleanup\_ora\_trace

This parameter defines the retention period in days for Oracle trace and audit files, which are excluded for the specified period from the [BRCONNECT cleanup function](#).

### Syntax

```
cleanup_ora_trace = <days>
```

End of the code.

Default value: 30

## cleanup\_owner

This parameter defines the database owner of tables (that is, SDBAH, SDBAD, DBMSGORA, and XDB tables) to be excluded from the [BRCONNECT cleanup function](#). You can use this parameter to limit the cleanup to objects of selected SAP owners (that is, systems).

### Syntax

```
cleanup_owner = <owner>|(<owner_list>)
```

End of the code.

Default value: SAPR3/SAP<SID> in a standard SAP database or all SAP owners in a multi-schema database

## cleanup\_xdb\_log

This parameter defines the retention period in days for records in the XDB tables, which are excluded for the specified period from the [BRCONNECT cleanup function](#).

### Syntax

```
cleanup_xdb_log = <days>
```

End of the code.

Default value: 100

## compress

This parameter defines whether files are to be compressed (software or hardware compression).

### Syntax

```
compress = no|yes|hardware|only
```

End of the code.

Default value: no

Possible values:

- `no`: no file compression
- `yes`: software compression of files during backup
- `hardware`: for tape units supporting hardware compression. BRRESTORE handles this setting as `compress = no`. Just setting this parameter does not activate hardware compression. It is merely information for BRBACKUP/BRARCHIVE. You also have to configure your backup device accordingly. This value is also supported for backups on disk with hardware-compressing file systems on Windows NT and AIX operating systems.
- `only`: software compression, but no backup of files is started. This setting is not used by BRRESTORE.

### compress = only: Software Compression Without Starting a Backup

If you use tape units that support hardware compression, SAP recommends that you perform software compression of the entire database with parameter `compress = only` at least once a month, so that the current compression rate can be determined for each of the data files. When you use this parameter, a backup is not started.

If you do not want to compress the entire database, we recommend that you at least compress those database files where a lot of changes take place. To determine these files, compare the compression rates of all files in two subsequent compression operations. The compression rates for files that have not changed will probably remain constant in the future, as well.

After extreme database changes (reorganization, release upgrade, transfer of large quantities of data), always start the software compression to determine the compression rates for the entire database.

The setting of parameter `backup_dev_type` is *not* relevant for this activity, since no backup is started.

Check the setting of option `-b 12` in command [compress\\_cmd](#).

## compress\_cmd

This parameter defines the command to be used for software file compression (if activated using the compression parameter such as: `compress = yes`).

### Syntax

```
compress_cmd = <cmd>
```

End of the code.

Default value: none

If you want to use the software compression, you must enter a command. The command must contain two \$ characters which stand for the file to be compressed and the compressed file.

You can enter the following command, for example:

```
compress_cmd = "compress -c $ > $"
```

Make sure that you adhere to the syntax rules when you enter parameters composed of several values. In particular, the double quotes are essential.

Compression is performed on disk. You can specify the directory in which compression is to be performed in parameter `compress_dir`.

BRBACKUP replaces the first variable in the command with the source file which you want compressed. The second variable is replaced with the name of the compressed file. The second file name is assigned the extension ".Z".

### Note

Ensure that the compression command you enter does not delete the original of the compressed file. SAP therefore recommends that you always use the option `-c` of the compress command (when available) so that the original is not deleted.

### Note

If you use compression with the parameter `compress = only` or option `-k only`, SAP recommends using option `-b 12` of the compress command:

```
compress_cmd = "compress -b 12 -c $ > $"
```

The compression rates determined in this manner correspond much more closely to the actual hardware compression rates, and therefore enable optimized file distribution and volume load balancing.

## compress\_dir

This parameter defines the directories where file compression occurs. Since file compression occurs on disk, you must provide enough space to compress the largest file in your database.

### Syntax

```
compress_dir = <dir>|(<dir_list>)
```

End of the code.

Default value: the directories specified in parameter `backup_root_dir`

#### Note

If you want a parallel backup with software compression on local or remote tape devices, `<n>` copy processes are generally started in parallel, where `<n>` is normally the number of defined backup devices that are connected. To be able to use this parallel copy effectively also for the compression, define in directory `compress_dir` as many directories as there are copy processes. See also [exec\\_parallel](#).

#### Note

If you check the readability and completeness of the backup using `-w|-verify`, the files are restored back to the directories defined in `compress_dir`. To perform this process in parallel as effectively as possible, define as many directories as there are copy processes. The number of copy processes generally corresponds to the number of connected tape devices, see [exec\\_parallel](#).

## copy\_in\_cmd

This parameter defines the command to read data from remote tape devices.

#### Syntax

```
copy_in_cmd = <cmd>
```

End of the code.

Default value: none

You must enter a value before you switch to “remote piping”, that is, `backup_dev_type = pipe|pipe_auto|pipe_box`.

This parameter specification is only useful in combination with parameters `remote_host` and `remote_user` for reading from remote systems.

#### Example

```
copy_in_cmd = "dd bs=50k if=$",
```

However, you can also use higher blocking to improve performance. The `$` character is replaced by the device address. Do *not* forget to enclose the parameter specifications composed of several values in double quotes.

## copy\_out\_cmd

This parameter defines the command to write data to remote tape devices.

#### Syntax

```
copy_out_cmd = <cmd>
```

End of the code.

Default value: none

You must enter a value before you switch to “remote piping”.

This parameter specification is only useful in combination with the parameter `remote_host` and `remote_user` for performing backups on remote systems.

#### Example

```
copy_out_cmd = "dd bs=50k of=$",
```

You can also use higher blocking to improve performance. The \$ character is replaced by the device address. Do *not* forget to enclose the parameter specifications composed of several values in double quotes.

## cpio\_disk\_flags

This parameter defines the flags for `cpio` commands that BRBACKUP uses to copy directories to disk.

#### Syntax

```
cpio_disk_flags = <flags>
```

End of the code.

Default value: `-pdcu`

#### Example

The call `brbackup -d disk -m /sapmnt/C11/bin` makes a backup copy of directory `/sapmnt/C11/bin` on disk. You must set parameter `cpio_disk_flags` for this.

## cpio\_flags

This parameter defines flags for `cpio` commands that the SAP tools use to write files to a tape volume.

#### Syntax

```
cpio_flags = <flags>
```

End of the code.

Default value: `-ovB`

The flag `B` causes a block size of 5 KB to be used. You can also use a higher block size, such as 64 KB (if `cpio` permits this for your hardware platform) to increase performance when writing to tape.

The block size in `cpio_flags` and `cpio_in_flags` must be the same.

## cpio\_in\_flags

This parameter defines flags for `cpio` commands that the SAP tools use for reading files from a tape volume.

#### Syntax

```
cpio_in_flags = <flags>
```



End of the code.

Default value: `-iuvB`

The flag `B` causes a block size of 5 KB to be used. You can also use a higher block size, such as 64 KB (if `cpio` permits this for your hardware platform), to increase performance when writing to tape.

The block size in `cpio_flags` and `cpio_in_flags` must be the same.

## db\_services

This parameter supports the stopping and starting of Oracle cluster services, Cluster Ready Services (CRS).

Syntax

```
db_services = no|yes
```

End of the code.

Default value: `no`

Possible values:

- `no`: no database service handling by BR\*Tools
- `yes`: activates database service handling by BR\*Tools

## dd\_flags

Defines the `dd` options required for writing to tape.

This parameter is required if you are working with raw devices. To access raw devices, you use the `dd` command. See [Raw Devices with BRBACKUP and BRRESTORE](#).

Syntax

```
dd_flags = <flags>
```

End of the code.

Default value: `"obs=16k"`

If you define the default value, the data is written to tape in blocks of 16 KB, but you can also use a higher value, for example 64 KB.

## dd\_in\_flags

This parameter defines the `dd` options required for reading from tape. It is required if you are working with raw devices. To access raw devices, you use the `dd` command. See [Raw Devices with BRBACKUP and BRRESTORE](#).

Syntax

```
dd_in_flags = <flags>
```

End of the code.

Default value: `"ibs=16k"`

If you define the default value, the data is read from tape in blocks of 16 KB, but you can also use a higher value, for example 64 KB.

## disk\_copy\_cmd

This parameter specifies the copy command to be used to copy files to local disks.

Syntax

```
disk_copy_cmd = copy|dd|ocopy|rman|copy_gnu|  
dd_gnu|rman_gnu|rman_set|rman_set_gnu
```

End of the code.

Default value: `copy`

Possible values:

- `copy`: `cp` (UNIX) or `copy` (Windows) command copies data to disk
- `dd`: `dd` command copies data to disk.
- `ocopy`: Oracle OCOPI tool copies data to a local disk – available only on Windows systems
- `rman`: Oracle Recovery Manager (RMAN) copies data to disk (for one-to-one copies of data files). See [RMAN Backup with an External Backup Library](#).
- `rman_set`: Oracle RMAN creates save sets for data files on disk. This enables the use of RMAN compress for disk backups. For more information, see [.rman\\_compress](#).

Note

The suffix `_gnu` lets you use the GNU tools to copy files. GNU `cpio` in particular is used to copy directories to disk

## dismount\_cmd

This parameter defines the dismount command for the automatic dismounting of a tape. For `backup_dev_type = tape_box` the command is called locally; for `backup_dev_type = pipe_box` it is called on a machine defined in the parameter `remote_host`.

Syntax

```
dismount_cmd = <cmd>
```

End of the code.

Default value: none. For `backup_dev_type = tape_box | pipe_box` this parameter *must* be defined in the initialization profile.

The dismount command uses its own options:

```
dismount_cmd = "<dismount_cmd> $ $ [ $ ] "
```

<dismount\_cmd>: Command name

The command name might be, for example, `dismount.csh`.

Do *not* forget to enclose the parameter specifications composed of several values in double quotes. The \$ characters stand, in the following order, for:

1. The name of the database to be backed up
2. The addresses of the tape devices
3. Optional: name of a file for additional configuration parameters (parameter: mount\_par\_file)

#### Caution

The dismount or mount command must be created by the user in the form of a program, a shell script or a batch file.

If the dismount command has been performed successfully the exit code 0 is displayed. No other output is displayed. The only possible outputs not recognized as errors are messages beginning with the characters #INFO.

*See also:*

[Backup with Automatic Tape Changers](#)

[Mount and Dismount Commands](#)

[mount\\_cmd](#)

[mount\\_par\\_file](#)

## exec\_parallel

This parameter specifies the maximum number of parallel copy processes.

#### Syntax

```
exec_parallel = <number>
```

End of the code.

Default value: 0

The number of parallel copy processes corresponds to the number of backup devices available (tape devices or disks) in this case. If you use option `-k only` to determine the compression rates, `-w only_dbv|only_rmv` to verify the database, or `-d rman_prep` to prepare for RMAN backups, the number of parallel copy processes corresponds to the number of disks (or logical volumes) on which the database files reside. If the number of disks is larger than 16, then 16 parallel copy processes are used.

- Backup to tape

The value `n` should be less than or equal to the number of backup devices. If you define a value `n` less than the number of tape devices, this means that you can only use `n` of the available tape devices in parallel. If a tape change is required in one of the tape devices used in parallel, there is an automatic change to the next free backup device and the backup continues there.

- Backup to disk

The number of parallel copy processes can be greater than the number of disks defined in `backup_root_dir` or `stage_root_dir` (but not greater than 255). One or more disks are then written by several processes at the same time.

If you choose the number of copy processes `n` to be less than the number of disks, this means that you can only use `n` of the available disks in parallel. If one of the disks used in parallel is full, there is an automatic change to the next disk which has not been used and backup continues there.

- Restoring

The maximum number of parallel copy processes that BRRESTORE can restore is the number of parallel copy processes used in the backup. The number of copy processes can be reduced.

## **exp\_dump\_dir**

This parameter specifies the directory for the BRSPACE export dump file.

Syntax

```
exp_dump_dir = <directory name>
```

End of the code.

Default value: `$SAPDATA_HOME/sapreorg`

Example

```
exp_dump_dir = $SAPDATA_HOME/sapreorg
```

## **exp\_table**

This parameter specifies the tables for an export using BRSPACE.

Syntax

```
exp_table = [<owner>.<table> | (<table_list>)
```

End of the code.

Default value: none

Example

```
exp_table = (SDBAH, SAPR3.SDBAD)
```

## **expir\_period**

This parameter specifies the expiration period for the tape volumes.

Syntax

```
expir_period = <days>
```

End of the code.

Default value: 30

Required values: indicate the period (in days) for which the tapes to be used for the backup should be locked by entering a whole number.

Before BRBACKUP or BRARCHIVE start the backup to a volume, the system checks whether the expiration period set by `expir_period` for this volume has expired.

See also:

[Volume Expiration Period.](#)

## imp\_table

This parameter specifies the tables for an import using BRSPACE.

Syntax

```
imp_table = [<owner>.<table> | (<table_list>)
```

End of the code.

Default value: none

Example

```
imp_table = (SDBAH, SAPR3.SDBAD)
```

## mount\_cmd

This parameter defines the mount command for the automatic mounting of a tape. The command is called locally for `backup_dev_type = tape_box`; for `backup_dev_type = pipe_box` it is called on a machine defined in parameter `remote_host`.

Syntax

```
mount_cmd = <cmd>
```

End of the code.

Default value: none

If `backup_dev_type = tape_box | pipe_box` is defined, then you *must* define this parameter in the initialization profile. For more information, see [backup\\_dev\\_type](#).

The mount command has its own options:

```
mount_cmd = "<mount_cmd> $ $ $ [$]"
```

<mount\_cmd>: Command name

For example, Command name might be `mount.csh`.

Do *not* forget to enclose the parameter specifications composed of several values in double quotes. The \$ characters stand, in the following order, for:

1. Name of the database to be backed up
2. Addresses of the tape devices
3. Tape names
4. Optional: name of a file for additional configuration parameters (parameter: `mount_par_file`)

Caution

The user must create the `dismount` or `mount` command in the form of a program, a shell script or a batch file.

If the `mount` command has been performed successfully the exit code 0 is displayed. No other output is displayed. The only possible outputs not recognized as errors are messages beginning with the characters `#INFO`.

See also:

[Backup with Automatic Tape Changers](#)

[Mount and Dismount Commands](#)

[dismount\\_cmd](#)

[mount\\_par\\_file](#)

## mount\_par\_file

This parameter is used in conjunction with the `mount` or `dismount` commands for automatic mounting or dismounting of tapes during backups with jukeboxes or autoloaders. The name of a parameter file is defined in `mount_par_file` or by the corresponding command option `-r|-parfile`. This file contains additional configuration parameters for the `mount` or `dismount` commands.

Syntax

```
mount_par_file = <file>
```

End of the code.

See also:

[-r|-parfile](#)

[Mount and Dismount Commands](#)

## new\_db\_home

This parameter defines the name of the `sapdata` home directory of the database copy. You must set this parameter if you want to make a database copy on local disks using `BRBACKUP`, that is, [backup\\_dev\\_type](#) = `disk_copy|disk_standby`.

Syntax

```
new_db_home = <dir>
```

End of the code.

Default value: none

Possible value:

<dir> is the new SAP database directory:

- UNIX: `/oracle/<NEW_SID>`
- Windows: `<drive>:\oracle\<NEW_SID>`

This directory must also contain the `sapbackup` directory.

## Note

With Windows, the `sapdata` directories can be distributed across several drives. When making a database copy, you can specify a target drive for each drive (see [ml-mode](#)). If you do not specify a target drive, all files are copied to the directory defined in the parameter.

See also:

[Structure-Retaining Database Copy](#)

[The SAP Tools with Windows](#)

## next\_exclude

This parameter defines the database objects (that is, tables, indexes, and tablespaces) to be excluded from the [BRCONNECT next function](#). You can use this parameter to exclude tables or indexes with exceptional space parameters from the [BRCONNECT next](#) function.

### Syntax

```
next_exclude = [<owner>.<table> | <owner>.<index> |  
[<owner>.] [<prefix>]* [<suffix>] | <tablespace> |  
(<object_list>) | all_part
```

End of the code.

Default value: no exclusion, all SAP tables and indexes are processed

Possible value:

`all_part`: excludes SAP partitions (such as in Business Information Warehouse and Advanced Planner and Optimizer) from the adapting of extents

## next\_limit\_count

This parameter defines the maximum number of next extents (`MAXEXTENTS`).

### Syntax

```
next_limit_count = <number>
```

End of the code.

Default value: 0 (that is, no limit)

If specified, the [BRCONNECT next function](#) sets the `MAXEXTENTS` storage parameter for all processed tables and indexes to this value.

## next\_max\_size

This parameter defines the upper limit for the next extent size, in KB, MB, or GB.

### Syntax

```
next_max_size = <size>
```

End of the code.

where `<size>: <n>K | <n>M | <n>G` is the upper limit of the next extent size

Default value: 2 GB - 5 \* <database block size>

0 means unlimited next extent size

The [BRCONNECT next function](#) does *not* change the NEXT\_EXTENT storage parameter of tables or indexes to a value greater than the value specified in this parameter.

Example

This command sets the upper limit for the next extent size to 1 MB:

```
next_max_size = 1M
```

## next\_owner

This parameter defines the database owner of tables and indexes to be processed by the [BRCONNECT next function](#). You can use this parameter to limit the next function to objects of selected SAP owners (that is, systems).

Syntax

```
next_owner = <owner>|(<owner_list>)
```

End of the code.

Default value: SAPR3/SAP<SID> in a standard SAP database or all SAP owners in a multi-schema database

## next\_special

This parameter defines the special sizes of next extent for exceptional tables and indexes, in KB, MB, or GB. You can use this parameter to specify your own sizes for next extents and maximum number of extents for selected tables and indexes.

Syntax

```
next_special =  [<owner>.]<table>:<size>[/<limit>], [<owner>.]  
<index>:<size>[/<limit>]| (<special_list>)|  
[ [<owner>.] [<prefix>]* [<suffix>]:<size>  
[/<limit>]| all_sel:<size>[/<limit>]
```

End of the code.

Default value: according to table category

Possible values:

- <size> <n>K|<n>M|<n>G: special NEXT\_EXTENT size
- <limit>: special MAX\_EXTENTS count
- all\_sel: sets NEXT\_EXTENT and MAX\_EXTENTS attributes to a certain value for *all* the database objects selected using the -t function option of [-f next](#) or the [next table](#) parameter. This option is provided for exceptional situations.

For more information, see:

- [Internal Rules for Determining Next Extent Size](#)



- [Methods of Adapting Next Extent Size](#)

## next\_table

This parameter defines the database objects to be processed by the [BRCONNECT next function](#).

Syntax

```
next_table = all|all_ind|special|[<owner>.]<table>|
[<owner>.]<index>|[<owner>.] [<prefix>]* [<suffix>]|
<tablespace>| (<object_list>)
```

End of the code.

Default value: all objects of selected owners

Possible values:

- all: all SAP tables and indexes
- all\_ind: all SAP indexes
- special: only tables and indexes defined in the [next\\_special](#) parameter

## orig\_db\_home

This parameter lets you rename the database file for the standby database and split mirror disk backup. The name of the `sapdata` home directory of the primary database is defined in `orig_db_home`. Normally this parameter is only set in the profile on the standby or backup server.

Syntax

```
orig_db_home = <dir>
```

End of the code.

Default value: none

Possible values:

- UNIX  
 <dir>: primary SAP database directory, that is, `/oracle/<SID>`
- Windows  
 <dir>: primary SAP database directory, that is, `<drive>:\oracle\<SID>`

See also:

[Standby Database](#)

[Split Mirror Backup](#)

## parallel\_instances

This parameter is only relevant if you are using Oracle Real Application Cluster (RAC).

The instances running in parallel to the dedicated database instance are defined by this parameter.

#### Syntax

```
parallel_instances = <inst_descr>|(<inst_descr_list>)
```

End of the code.

Default value: none.

Possible value:

```
<inst_descr>: <ORACLE_SID>:<ORACLE_HOME>@<conn_name>
```

where:

- <ORACLE\_SID>: Oracle System ID of the parallel instance
- <ORACLE\_HOME>: ORACLE\_HOME directory of the parallel instance
- <conn\_name>: Oracle connection name from TNSNAMES.ORA for connect to the parallel instance

#### Example

```
parallel_instances = (RAC001:/oracle/RAC/101_64@RAC001,  
RAC002:/oracle/RAC/102_64@RAC002, RAC003:/oracle/RAC/101_64@RAC003
```

#### Note

Specify all database instances in this parameter.

## pipe\_copy\_cmd

This parameter ensures that remote tape backups – [backup\\_dev\\_type](#) = pipe | pipe\_auto | pipe\_box – can use Secure Shell (ssh).

#### Syntax

```
pipe_copy_cmd = rsh | ssh
```

End of the code.

Default value: rsh

Possible values:

- rsh: uses remote shell
- ssh: uses secure shell

For more information on the secure copy command, see SAP Note [700733](#).

## post\_shut\_cmd

This parameter lets you execute external commands – that is, executable programs or scripts – directly *after* the database was restarted for an offline backup with BRBACKUP:

## Syntax

```
post_shut_cmd = "<post_shut_cmd>"
```

End of the code.

BRBACKUP only accepts that the external commands have been successfully executed when:

- Exit code 0 is returned
- No messages are returned or only messages starting with #INFO

## Example

You can use this parameter and [pre\\_shut\\_cmd](#) when performing offline backups in a cluster environment to deactivate and later reactivate the cluster resources on the database instance.

## post\_split\_cmd

This parameter lets you execute external commands – that is, executable programs or scripts – directly *after* a BRBACKUP disk split as follows.

- Offline split:
  - Split command scenario  
The external commands are executed after the database was restarted.
  - SPLITINT scenario  
The external commands are executed after the SPLITINT interface program was executed.
- Online split:
  - Split command scenario  
The external commands are executed after the tablespaces were taken out of backup status.
  - SPLITINT scenario  
The external commands are executed after the SPLITINT interface program was executed.

## Syntax

```
post_split_cmd = "<post_split_cmd>"
```

End of the code.

BRBACKUP only accepts that the external commands have been successfully executed when:

- Exit code 0 is returned
- No messages are returned or only messages starting with #INFO

## Example

You can use this parameter and [pre\\_split\\_cmd](#) when performing a [split-mirror backup](#) of a [standby database](#). In this case, you can use `pre_split_cmd` to stop the import of the offline redo log files with BRARCHIVE before the split and `post_split_cmd` to restart it again after the split.

## pre\_shut\_cmd

This parameter lets you execute external commands – that is, executable programs or scripts – directly *before* the database is stopped for an offline backup with BRBACKUP:

Syntax

```
pre_shut_cmd = "<pre_shut_cmd>"
```

End of the code.

BRBACKUP only accepts that the external commands have been successfully executed when:

- Exit code 0 is returned
- No messages are returned or only messages starting with `#INFO`

Example

You can use this parameter and [post\\_shut\\_cmd](#) when performing offline backups in a cluster environment to deactivate and later reactivate the cluster resources on the database instance.

## pre\_split\_cmd

This parameter lets you execute external commands – that is, executable programs or scripts – directly *before* a BRBACKUP disk split as follows:

- Offline split:
  - Split command scenario  
The external commands are executed before the database is stopped.
  - SPLITINT scenario  
The external commands are executed before the SPLITINT interface program is called.
- Online split:
  - Split command scenario  
The external commands are executed before the tablespaces are set to backup status.
  - SPLITINT scenario  
The external commands are executed before the SPLITINT interface program is called.

Syntax

```
pre_split_cmd = "<pre_split_cmd>"
```

End of the code.

BRBACKUP only accepts that the external commands have been successfully executed when:

- Exit code 0 is returned
- No messages are returned or only messages starting with #INFO

Example

You can use this parameter and [post\\_split\\_cmd](#) when performing a [split-mirror backup](#) of a [standby database](#). In this case, you can use `pre_split_cmd` to stop the import of the offline redo log files with BRARCHIVE before the split and `post_split_cmd` to restart it again after the split.

## primary\_db

This parameter is only relevant for [Standby Database](#) and [Split Mirror Backup](#).

This parameter defines the connect string to the primary database instance so that BRBACKUP can log on to the primary database.

Syntax

```
primary_db = <conn_name>
```

End of the code.

Default value: none

Possible value:

<conn\_name>: connection name from TNSNAMES.ORA for the connect from the standby or split host to the primary (that is, production) database

Example

```
primary_db = C11_PRIM
```

## rebuild\_index

This parameter specifies the database indexes for an index rebuild using BRSPACE.

Syntax

```
rebuild_index = [<owner>.]<index>|  
[<owner>.] [<prefix>]* [<suffix>] | (<index_list>)
```

End of the code.

Default value: none

Example

```
rebuild_index = (SDBAH~0, SAPSR3.SDBAD~0)
```

## recov\_copy\_dir

This parameter specifies the directory for file copies during a database recovery using BRRECOVER. The files are the control file, the online redo log file, and temporary files.

Syntax

```
recov_copy_dir = <directory name>
```

End of the code.

Default value: \$SAPDATA\_HOME/sapbackup

## recov\_degree

This parameter determines the degree of parallelism for applying the archive log files with SQLPLUS during a database recovery with BRRECOVER.

Syntax

```
recov_degree = 0
```

End of the code.

Default value: Oracle default

Possible values:

- 0: use the Oracle default recovery parallelism
- 1: serial recovery
- > 1: parallel recovery

This parameter corresponds to [BRRECOVER -e|-degree <number>](#).

## recov\_interval

This parameter specifies the interval in which BRRECOVER or BRESTORE searches for backups of database files and offline redo log files.

Syntax

```
recov_interval = <days>
```

End of the code.

Default value: 30

Possible values:

- 0: use all available backups
- 1: use backups performed today
- >1: use backups performed in the previous specified number of days

Example

```
recov_interval = 60
```

BRRECOVER or BRESTORE searches for backups performed in the last 60 days.

This parameter corresponds to [BRRECOVER -i|interval](#) and [BRRESTORE -i|interval](#).

## recov\_type

This parameter specifies the type of database recovery using BRRECOVER.

Syntax

```
recov_type = complete|dbpit|tspit|reset|restore|apply|disaster
```

End of the code.

Default value: complete

Possible values:

- complete: complete database recovery
- dbpit: database point-in-time (PIT) recovery
- tspit: tablespace point-in-time (PIT) recovery
- reset: whole database reset
- restore: restore of individual backup files
- apply: restore and application of archivelog (that is, offline redo log) files
- disaster: disaster recovery

This parameter corresponds to [BRRECOVER -t|type](#).

## remote\_host

This parameter specifies the name of the remote host for backing up to a remote tape or disk.

Note

The parameters `remote_host` and `remote_user` replace the parameter `read_fifo_cmd`, which is no longer used as of Release 4.5A.

Syntax

```
remote_host = <host>
```

End of the code.

Default value: none

See also:

[remote\\_user](#)

[Backup to a Remote Disk](#)

## remote\_user

This parameter specifies the user for a remote host for backing up to a remote tape or disk.

## Note

The parameters `remote_host` and `remote_user` replace the parameter `read_fifo_cmd`, which is no longer used as of Release 4.5A.

## Syntax

```
remote_user = <user>
```

End of the code.

Default value: none

If you use the `ftp` network to transfer data (see [stage\\_copy\\_cmd](#)) you can also specify the password which BRBACKUP uses to log on to the remote host:

```
remote_user = "<user> <password>"
```

If you do *not* specify a password, BRBACKUP uses the password of the database user. If this is the case, the two user passwords must be the same.

See also:

[remote\\_host](#)

[Backup to a Remote Disk](#)

## reorg\_table

This parameter specifies the database tables for a reorganization using BRSPACE.

### Syntax

```
reorg_table = [<owner>.]<table>|  
[<owner>.] [<prefix>]* [<suffix>] | (<table_list>)
```

End of the code.

Default value: none

### Example

```
reorg_table = (SDBAH, SAPSR3.SDBAD)
```

## restore\_mode

This parameter specifies the scope of restore activity for BRRESTORE.

### Syntax

```
restore_mode = all|all_data|full|incr|incr_only|incr_full|  
incr_all|<tablespace>|<file_ID>|  
<file_ID1>-<file_ID2>| (<object_list>)|partial|non_db
```

End of the code.

Default: value: all

Possible values:



- all  
Restores all tablespaces (without control files and redo log files)
- all\_data  
Restores all tablespaces that are not pure index tablespaces or empty
- full  
Restores a complete backup (that is, all backup files)
- incr  
Restores an incremental backup with RMAN.
- incr\_only  
Restores changes to all files that were in the database at the time of the last full backup. See *Restoring Incremental Backups with Structural Changes* in [Profile Parameters and BRBACKUP Command Options](#).
- incr\_full  
Restores files that have been added to database since the last full backup. See *Restoring Incremental Backups with Structural Changes* in [RMAN-Relevant Profile Parameters](#).
- incr\_all  
Restores an incremental backup with RMAN and also restore all redo log, control, and non-database files and directories, if any
- <tablespace>  
Files of the specified tablespaces are processed by the appropriate SAP utility
- <file\_ID>  
Restores the file with this file ID. For data files, this is the Oracle file ID. Control files can be addressed with the file ID 0. Online redo log files can be addressed using the file ID 0<n>, <n> is the redo log group number. Specify file ID 00 to restore all existing online redo log files. Temporary files are identified by negative numbers.
- <file\_ID1>-<file-ID2>  
Restores the files in this interval
- <object\_list>  
Restores the specified objects. You can enter an object list, as described in [Initialization Profile init<DBSID>.sap](#). This list can also include the keyword `all`. However, we recommend processing database files and non-database files separately.
- partial  
Restores all files from a partial backup without specifying them explicitly
- non\_db

Restores all non-database files and directories from a backup without specifying them explicitly

## resync\_cmd

This parameter defines the command for the synchronization of the mirror disks with the original disks under the control of BRBACKUP.

This parameter is only relevant when using the [split\\_mirror](#) configuration with the split command scenario.

### Syntax

```
resync_cmd = "<resync_cmd> [\$]"
```

End of the code.

Default value: none

This command has its own option:

<resync\_cmd>: program or shell script for synchronizing the mirror disks

The \$ character is optional. If it is set it replaces BRBACKUP in runtime with the name of a text file containing all file names to be backed up.

### Caution

The resync command must be created by the user in the form of a program, a shell script, or a batch file.

If the resync command has been performed successfully the exit code 0 is displayed. No other output is displayed. The only possible outputs not recognized as errors are messages beginning with the characters #INFO.

See also:

[split\\_cmd](#)

## rewind

This parameter defines the rewind command for your host operating system.

The SAP tools use this command to rewind a tape when performing a tape backup. This parameter is also used with `pipe` or `pipe_auto` to back up to a remote system or restore files from that system.

### Syntax

```
rewind = <cmd>
```

End of the code.

Default value: none

### Example

Rewind commands:

- HP-UX: "mt -f \$ rew"

- AIX: "tctl -f \$ rewind"
- Linux, SUN, Windows: "mt -f \$ rewind"

Make sure that you do not forget the double quotes when you enter parameters composed of several values.

The SAP tools replace the \$ character with the address of the device used.

## rewind\_offline

This parameter defines the rewind/set offline command for your host operating system.

BRBACKUP and BRARCHIVE use this parameter to rewind a tape after a backup and set it to offline. This means that the volume is automatically ejected from the tape device (even when you are backing up to a remote system using `pipe` or `pipe_auto`).

Syntax

```
rewind_offline = <cmd>
```

End of the code.

Default value: same as [rewind](#)

Example

Rewind commands:

- HP-UX: "mt -f \$ offl"
- AIX: "tctl -f \$ offline"
- Linux, SUN, Windows: "mt -f \$ offline"

Make sure that you do not forget the double quotes when you enter parameters composed of several values.

The SAP tools replace the \$ character with the address of the device used.

See also:

[Backup with Automatic Tape Changers](#)

## rman\_channels

This parameter defines the number of parallel channels used by RMAN with an external backup library, that is, `backup_dev_type = backup_dev_type = rman_util|rman_disk|rman_stage`

Syntax

```
rman_channels = <number>
```

End of the code.

Default value: 1

See also:

## [RMAN Backup with an External Backup Library](#)

Oracle documentation

### **rman\_compress**

This parameter controls the RMAN binary compression of save sets.

Syntax

```
rman_compress = no|yes
```

End of the code.

Possible value:

*yes*: activates the RMAN binary compression of save sets

Caution

RMAN compression is only supported for save sets and for incremental backups. It is *not* supported for 1:1 copies of database files that were created from disk backups with [disk\\_copy\\_cmd](#) = *rman*.

To use RMAN compression, save sets must be created by RMAN on disk with [disk\\_copy\\_cmd](#) = *rman\_set*.

### **rman\_copies**

This parameter defines the number of copies of each save set created by RMAN with an external backup library, that is, [backup\\_dev\\_type](#) = *rman\_util|rman\_disk|rman\_stage*.

Syntax

```
rman_copies = 0|1|2|3|4
```

End of the code.

Default value: 0 (one copy, same as 1)

Note

This parameter was previously called *rman\_duplex*.

See also:

## [RMAN Backup with an External Backup Library](#)

Oracle documentation

### **rman\_diskratio**

This parameter defines the number of disk drives used for reading datafiles by RMAN with an external backup library, that is, [backup\\_dev\\_type](#) = *rman\_util|rman\_disk|rman\_stage*.

Syntax

```
rman_diskratio = <number>
```

End of the code.

Default: 0 (uses RMAN default value)

This parameter is no longer valid in Oracle 10g.

See also:

[RMAN Backup with an External Backup Library](#)

Oracle documentation

## rman\_filesperset

This parameter defines the number of database files stored in a save set by RMAN with an external backup library, that is, [backup\\_dev\\_type](#) = `rman_util|rman_disk|rman_stage`.

Syntax

```
rman_filesperset = <number>
```

End of the code.

Default value: 0, which means one file per save set for non-incremental backups or all files in one save set for incremental backups

See also:

[RMAN Backup with an External Backup Library](#)

Oracle documentation

## rman\_maxcorrupt

This parameter specifies the acceptable number of corrupt data blocks during an RMAN backup, with respect to data files.

Syntax

```
rman_maxcorrupt = (<dbf_name>|<dbf_id>:<corr_cnt>, ...)
```

End of the code.

Possible values:

- `<dbf_name>`: data file name
- `<dbf_id>`: data file ID
- `<corr_cnt>`: maximum acceptable number of corrupt data blocks for the specified data file

## rman\_maxpiecesize

This parameter defines the maximum size in kilobytes of the backup pieces created by RMAN with an external backup library, that is, [backup\\_dev\\_type](#) = `rman_util|rman_disk|rman_stage`.

You can also use this parameter to limit the size of a save set created by an RMAN incremental backup to local or remote disk.

### Syntax

```
rman_maxpiecesize = <number>
```

End of the code.

Default value: 0 (no limit)

### Note

This parameter was previously called `rman_kbytes`.

See also:

[RMAN Backup with an External Backup Library](#)

Oracle documentation

## rman\_maxopenfiles

This parameter defines the maximum number of opened files by RMAN.

### Syntax

```
rman_maxopenfiles = <number>
```

End of the code.

Default value: 0 (no limit)

See also:

[RMAN Backup with an External Backup Library](#)

Oracle documentation

## rman\_parms

This parameter defines additional parameters that RMAN passes to an external backup library, that is, `backup_dev_type = rman_util|rman_disk|rman_stage`.

Ask the vendor of the external backup library if it requires you to set additional parameters.

### Syntax

```
rman_parms = "<string>"
```

End of the code.

Default value: none

### Example

```
rman_parms = "BLKSIZE=65536 ENV=(BACKUP_SERVER=HOSTNAME) "
```

See also:

[RMAN Backup with an External Backup Library](#)

Oracle documentation

## rman\_pool

This parameter defines the media pool in which the backups are stored by RMAN with an external backup library, that is, [backup\\_dev\\_type](#) = rman\_util|rman\_disk|rman\_stage.

Syntax

```
rman_pool = <number>
```

End of the code.

Default value: 0 (standard media pool)

See also:

[RMAN Backup with an External Backup Library](#)

Oracle documentation

## rman\_proxy

This parameter defines that the proxy copy functionality is to be used by RMAN with an external backup library, that is, [backup\\_dev\\_type](#) = rman\_util|rman\_disk|rman\_stage.

Syntax

```
rman_proxy = no|yes|only
```

End of the code.

Default value: no

See also:

[RMAN Backup with an External Backup Library](#)

Oracle documentation

## rman\_rate

This parameter defines the maximum number of KB per second to be read by RMAN for each datafile.

Syntax

```
rman_rate = <number>
```

End of the code.

Default value: 0 (no limit)

Note

This parameter was previously called `rman_readrate`.

See also:

[RMAN Backup with an External Backup Library](#)

Oracle documentation

## rman\_send

This parameter defines special commands to be sent to the external backup library, that is, [backup\\_dev\\_type](#): `backup_dev_type = rman_util|rman_disk|rman_stage`.

For more information on the Oracle SEND command, see the Oracle documentation. Many suppliers of backup libraries for RMAN use this command to pass additional control information to the backup library.

### Syntax

```
rmand_send = '<command>'
```

End of the code.

You can also send different commands to specified backup channels:

```
rman_send = ("channel sbt_1 '<command1>' parms='<parameters1>'",  
"channel sbt_2 '<command2>' parms='<parameters2>'")
```

## rman\_maxsetsize

This parameter defines the maximum size in KB of save sets created by RMAN with an external backup library, that is, [backup\\_dev\\_type](#) = `rman_util|rman_disk|rman_stage`.

### Syntax

```
rman_maxsetsize = <number>
```

End of the code.

Default value: 0 (no limit)

### Note

This parameter was previously called `rman_setsize`.

See also:

[RMAN Backup with an External Backup Library](#)

Oracle documentation

## saveset\_members

If you make a backup to tape with the Oracle Recovery Manager and the SAP backup library and want to use the file multiplexing option, then you can use this parameter to control the grouping of the savesets.

By setting the number of files to be saved in a single saveset, you can ensure that the speed of your tape device is optimal.

### Syntax

```
saveset_members = 1|2|3|4|tsp|all
```

End of the code.

Default value: 1 (no file multiplexing)

Possible values:



- 1, 2, 3, 4: number of files in a save set
- tsp: each save set contains all files of a tablespace
- all: creates only one save set with all database files

See also:

[RMAN Save-Set Grouping](#)

## scroll\_lines

This parameter specifies the number of lines for scrolling in list menus.

Syntax

```
scroll_lines = <lines>
```

End of the code.

Default value: 20

Possible values:

- 0: displays the whole list in one step
- >0: scrolls the list, showing the specified number of lines

This parameter corresponds to [BRRECOVER -s|-scroll](#), [BRSPACE -s|-scroll](#), and [BRTOOLS -s|-scroll](#).

## show\_period

This parameter specifies the time period for which BRTOOLS displays log files.

Syntax

```
show_period = <days>
```

End of the code.

Default value: 30

Possible values:

- 0: shows all available log files
- 1: shows log files created today
- >1: shows log files created in the previous specified number of days

This parameter corresponds to [BRTOOLS -w|-show <days>](#).

## space\_copy\_dir

This parameter specifies the directory for file copies during space management with BRSPACE.

Syntax

```
space_copy_dir = <directory name>
```

End of the code.

Default value: `$SAPDATA_HOME/sapreorg`

Example

```
space_copy_dir = $SAPDATA_HOME/sapreorg
```

## split\_cmd

This parameter is only relevant when using the [split\\_mirror](#) configuration with the split command scenario.

This parameter contains a program or shell script (with or without options) called by BRBACKUP to split the mirror disks.

Syntax

```
split_cmd = <cmd>
```

End of the code.

Default value: none

This command has its own options:

```
split_cmd = "<split_cmd> [$]"
```

<split\_cmd> is the program or shell script to split the mirror disks.

The \$ character is optional. If it is set it replaces BRBACKUP in runtime with the name of a text file containing all file names to be backed up.

Caution

The split command must be created by the user in the form of a program, a shell script, or a batch file.

If the split has finished successfully the exit code 0 is displayed. No other output is displayed. The only possible outputs not recognized as errors are messages beginning with the characters #INFO.

See also:

[resync\\_cmd](#)

## split\_options

This parameter is only relevant when you use the [split\\_mirror](#) configuration with the SPLITINT scenario.

This parameter specifies the additional options used when BRBACKUP calls SPLITINT during a split-mirror backup to split or resynchronize the mirrored disks. Your SPLITINT vendor can tell you how to set them.

## split\_resync

This parameter is only relevant when you use the [split-mirror](#) configuration with the SPLITINT scenario.

This parameter specifies whether the mirror disks are to be resynchronized immediately after the backup has finished.

## stage\_copy\_cmd

This parameter specifies which transfer program copies the data across the network to another host. You use it for backups to remote disks.

Syntax

```
stage_copy_cmd = rcp|ftp|scp
```

End of the code.

Default value: rcp

Possible values:

- rcp: uses the operating system remote copy command
- ftp: uses SAP implementation of file transfer protocol (sapftp)
- scp: uses the operating system secure copy command

An incremental backup to a remote disk always uses ftp. In this case, this parameter is ignored.

See also:

[Backup to a Remote Disk](#)

## stage\_db\_home

This parameter defines the name of the sapdata home directory for the database copy. You must set this parameter if you want to make a database copy on remote disk using BRBACKUP, that is, [backup\\_dev\\_type](#) = stage\_copy|stage\_standby.

Syntax

```
stage_db_home = <dir>
```

End of the code.

Default value: same as [new\\_db\\_home parameter](#):

Possible values:

- UNIX: <dir> is the new SAP database directory, /oracle/<NEW\_SID>
- Windows: <dir> is the new SAP database directory, <drive>:\oracle\<NEW\_SID>

Note

With Windows, the `sapdata` directories can be distributed across several drives. When making a database copy, you can specify a target drive for each drive by using [ml-mode](#). If you do not specify a target drive, all files are copied to the directory defined in the parameter.

See also:

[Structure-Retaining Database Copy](#)

[The SAP Tools with Windows](#)

## stage\_root\_dir

This parameter identifies the directories used by BRBACKUP to back up database backups to a remote disk. In exceptions (for example, no definition of the parameter `archive_stage_dir`), BRARCHIVE also uses the directories defined in this parameter if remote disk backup is required. If you enter more than one directory, you must enclose the names in parentheses and separate them by commas.

If necessary, you can make other directories on other remote disks available, and add their names to this parameter. BRBACKUP then uses these directories for database backup.

Syntax

```
stage_root_dir = <dir>|(<dir_list>)
```

End of the code.

Default value: directory defined in parameter [backup\\_root\\_dir](#)

Required value: directories to which the remote disk backups are to be written.

See also:

[Backup to a Remote Disk](#)

## standby\_db

This parameter is only relevant for [Standby Database](#) and [Split Mirror Backup](#).

The connect string to the standby database instance is defined with this parameter so that BRBACKUP can log on to the standby database.

Syntax

```
standby_db = <conn_name>
```

End of the code.

Default value: none

Possible value:

<conn\_name>: connection name from `TNSNAMES.ORA` for the connect from the backup host to the standby database

Example

```
standby_db = C11_STBY
```

This must permit connection with SYSDBA authorization. On UNIX, this requires the Oracle password file. Or you can use the environment variable BR\_RSC, as described in [SAP Note 914174](#).

## stats\_bucket\_count

This parameter defines the number of buckets for updating statistics with histograms for the [BRCONNECT update statistics function](#).

Syntax

```
stats_bucket_count = <count>|auto|skewonly|repeat
```

End of the code.

Default value: 75

Possible values:

- <count>: number of buckets
- auto: creates histograms based on data distribution and use
- skewonly: creates histograms on columns based on data distribution
- repeat: creates histograms only for columns that already have histograms

## stats\_change\_threshold

This parameter defines the threshold for the percentage of inserted or deleted rows causing an update in optimizer statistics for the [BRCONNECT update statistics function](#).

Syntax

```
stats_change_threshold = <number>
```

End of the code.

Default: 50

The value must be greater than zero.

Example

25% means that statistics are collected for a table if either of the following is true:

- Inserted row count is *greater than or equal to 25 %* of the old row count
- Updated row count is *greater than or equal to 25 %* of the old row count
- Deleted row count is *greater than or equal to 25 / (100\*25) %* of the old row count

## stats\_dbms\_stats

This parameter defines how the DBMS\_STATS package is used to update statistics with BRCONNECT. DBMS\_STATS is the default method used in Oracle 10g.

Syntax

```
stats_dbms_stats = all:R|B[<buckets>|A|S|R]:0|<degree>|A|D|
```

```

all_part:R|B[<buckets>|A|S|R]:0|<degree>|A|D|
info_cubes:R|B[<buckets>|A|S|R]:0|<degree>|A|D|
[<owner>.]<table>:R|B[<buckets>|A|S|R]:0|<degree>|A|D|
[<owner>.] [<prefix>]* [<suffix>]:R|B[<buckets>|A|S|R]:0|
<degree>|A|D| (<object_list>) |NO

```

End of the code.

Default value: ALL:R:0

Possible values:

- R|B[<buckets>|A|S|R] lets you specify the sampling type and number of buckets for histograms:
  - R: row sampling
  - B: block sampling
  - <buckets>: the number of buckets for histograms
  - A: auto
  - S: skew only
  - R: repeat
- <degree>|0|1|A|D lets you specify the degree of parallelism for DBMS\_STATS:
  - <degree>: degree of parallelism
  - 0: table default degree
  - 1: no parallelism
  - A: auto-degree option
  - D: default-degree option
- all\_part: only processes SAP partitioned tables
- info\_cubes: only processes InfoCube tables

The optional entry ALL:R|B:<degree> activates the DBMS\_STATS package for all selected tables and defines the standard sampling type and degree of parallelism.

The sampling type and degree of parallelism that you enter for the specified tables or table groups overrides the default sampling type and degree of parallelism (as defined for a table in Oracle dictionary).

The NO: ANALYZE statement is used to update statistics (not recommended in Oracle 10g or higher). This is the default method in Oracle 9i.

Example

- stats\_dbms\_stats = ALL:R:1

This parameter activates the DBMS\_STATS package with row sampling for all selected tables, without parallelism.

- `stats_dbms_stats = (ALL:R:2, ATAB:B:3, RFBLG:B:4)`

This parameter activates the DBMS\_STATS package for:

- ATAB table with block sampling and parallelism degree 3
- RFBLG table with block sampling and parallelism degree 4

All other selected tables with row sampling and parallelism degree 2

- `stats_dbms_stats = (NO, ATAB:R:3, RFBLG:R:4)`

This parameter activates the DBMS\_STATS package only for:

- ATAB table with row sampling and parallelism degree 3
- RFBLG table with row sampling and parallelism degree 4

All other selected tables are processed with the `ANALYZE` statement (not recommended in Oracle 10g).

- `stats_dbms_stats = (ALL:R:0, INFO_CUBES:R100:A)`

This parameter creates histograms with 100 buckets and parallelism degree “auto” for InfoCube tables. For other tables, statistics are created with default settings.

For more information on DBMS\_STATS, see [Update Statistics](#).

## stats\_exclude

This parameter defines the database objects (that is, tables, indexes, and tablespaces) to be excluded from the [BRCONNECT update statistics function](#). You can use this parameter to exclude tables or indexes with exceptional statistics handling from update statistics.

Syntax

```
stats_exclude = [<owner>.<table><[<owner>.<index>|
[<owner>.] [<prefix>]* [<suffix>]| <tablespace>|
([object_list])| info_cubes| all_part
```

End of the code.

Default value: no exclusion, all SAP tables and indexes are processed

Possible values:

- `info_cubes`: excludes all InfoCube tables from update statistics
- `all_part`: excludes all SAP partitioned tables from update statistics

## stats\_info\_cubes

This parameter defines which tables are treated as InfoCube tables by BRCONNECT. For more information, see [Update Statistics for InfoCube Tables](#).

Syntax

```
stats_info_cubes = null|default|rsnspace_tab|
```

```
[<owner>.<table>| [<owner>.] [<prefix>]* [<suffix>]| (<table_list>)
```

End of the code.

Default value: `rsnspace_tab`

Possible values:

- `null`: suppresses special handling of InfoCube tables
- `default`: includes the default table list of InfoCube tables specified in [Update Statistics for InfoCube Tables](#)
- `rsnspace_tab`: all tables with the name prefix specified in RSNSPACE control table

Example

```
stats_info_cubes = (default,ABC*, XYZ*)
```

In this example, table ABC and all tables starting with XYZ are treated as InfoCube tables, in addition to default tables.

Note

Normally you do *not* need to set this parameter.

## stats\_limit\_time

This parameter defines the processing time limit in minutes for updating statistics with the [BRCONNECT update statistics function](#).

You can use this parameter to terminate long-running update statistics jobs after a period of time. The processing terminates after statistics have been collected for the current table or index (this is the “soft limit”). However, if you set the option `-f limit` in the [update statistics function](#), processing terminates immediately (this is the “hard limit”).

Syntax

```
stats_limit_time = <minutes>
```

End of the code.

Default: 0, that is, no limit

## stats\_method

This parameter defines the method to be applied to tables that are not specified in the DBSTATC control table for the [BRCONNECT update statistics function](#).

Syntax

```
stats_method = E|EH|E|EI|EX|C|CH|CI|CX|  
A|AH|AI|AX|E|=C|=H|=I|=X|+H|+I
```

End of the code.

Default value: internal rules determine the update statistics method, as with [stats\\_sample\\_size](#)



Possible values:

- E: estimates
- EH: estimates with histograms
- EI: estimates with index validation
- EX: estimates with histograms and index validation
- C: computes
- CH: computes with histograms
- CI: computes with index validation
- CX: computes with histograms and index validation
- A: estimates with auto-sample size (DBMS.stats)
- AH: estimates with auto-sample size (DBMS.stats) and histograms
- AI: estimates with auto-sample size (DBMS.stats) and index validation
- AX: estimates with auto-sample size (DBMS.stats) and histograms and index validation
- E=: forces estimate for all tables, including tables in DBSTATC control table (option -f method must be set)
- C=: forces compute for all tables, including tables in DBSTATC control table (option -f method must be set)
- =H: forces creation of histograms for all tables, including tables in DBSTATC control table (option -f method must be set)
- =I: forces index validation for all tables, including tables in DBSTATC control table (option -f method must be set)
- =X: forces creation of histograms and index validation for all tables, including tables in DBSTATC control table (option -f method must be set)
- +H: forces creation of histograms for all tables, including tables in DBSTATC control table in addition in addition to index validation, if defined there (option -f method must be set)
- +I: forces estimate with index validation for all tables, including tables in DBSTATC control table in addition to creating histograms, if defined there (option -f method must be set)

## stats\_owner

This parameter defines the database owner of tables and indexes for the [BRCONNECT update statistics function](#). You can use this parameter to limit processing to objects of selected SAP owners (that is, systems).

Syntax

```
stats_owner = <owner>|(<owner_list>)
```

End of the code.

Default value: `SAPR3/SAP<SID>` in a standard SAP database or all SAP owners in a multi-schema database

## stats\_parallel\_degree

This parameter defines the number of parallel threads for updating statistics with the [BRCONNECT update statistics function](#). For example, you can set this parameter to the number of CPUs to speed up update statistics.

Syntax

```
stats_parallel_degree = <number>
```

End of the code.

Default value: 1

## stats\_sample\_size

This parameter defines the sample size with the [BRCONNECT update statistics function](#) using method E for tables that are *not* specified in the `DBSTATC` control table.

Syntax

```
stats_sample_size = P<p>|R<r>
```

End of the code.

Default value: [internal rules](#) determine the sample size, as with [stats\\_method](#)

Possible values:

- P<p>: percentage of rows
- R<r>: thousands of rows

Note

You can set the percentage of rows to less than 1, which improves performance on very large tables. For more information, see [Sample Sizes for Update Statistics](#).

For example, to set the sample size to 0.05%, enter `-s P.05`.

## stats\_special

This parameter lets you emulate the table `DBSTATC` if you want to run update statistics in a Java database, in which the table `DBSTATC` does not exist. You can use `stats_special` to specify tables that need special handling.

Syntax

```
stats_special = (<table>:[<owner>]:<active>:[<method>]:[<sample>],  
...)
```

End of the code.

Possible values:

- <table>: table name
- <owner>: table owner (optional)
- <active>: active indicator:
  - A: active (automatic check and possibly analysis) (default)
  - I: ignore (no automatic check and analysis)
  - N: negative (statistics are deleted)
  - U: forced (analysis is always performed, without check)
  - P: priority (same as "A", but execution occurs earlier)
- <method>: method of analysis:
  - E: estimates statistics (default)
  - EH: estimates statistics and generates histograms
  - EI: estimates statistics and structure validation of indices
  - EX: estimates statistics and histograms and structure validation
  - C: creates statistics exactly
  - CH: creates statistics exactly and generate histograms
  - CI: creates statistics exactly and structure validation of indices
  - CX: creates statistics exactly and histograms and structure validation
  - A: estimates statistics with auto sample size
  - AH: statistics with auto sample size and generate histograms
  - AI: statistics with auto sample and structure validation of indices
  - AX: statistics with auto sample and histograms and structure validation
- <sample>: sample size for analysis methods E, EH, EI, and EX
  - P<n>: <n> percentage of total number of table rows (from 1 to 100 and from .001 to .999)
  - P<n>: <n> thousand table rows

#### Example

```
stats_special = (SDBAD::A:E:P15, SDBAH::A:E:R2)
```

For more information, see the description of the DBSTATC table in the ABAP Dictionary (transaction SE11).

## stats\_system\_interval

This parameter defines the period in minutes for the collection of statistics using the DBMS\_STATS package with [BRCONNECT update statistics](#).

### Syntax

```
stats_system_interval = <minutes>
```

End of the code.

Default value: 10

## stats\_table

This parameter defines the database objects to be processed by the [BRCONNECT update statistics function](#). You can use this parameter to restrict update statistics to selected objects.

### Syntax

```
stats_table = all|all_ind|missing|dbstatc_tab|dbstatc_mon|  
dbstatc_mon| [<owner>.<table>| [<owner>.<index>|  
 [<owner>.<prefix>*<suffix>|<tablespace>|  
(<object_list>)|info_cubes|all_part|harmful|locked|  
system_stats|oradict_stats|oradict_tab
```

End of the code.

Default value: all objects of selected owners

Possible values

- all: all SAP tables and indexes
- all\_ind: all indexes only. For example,
  - you can use this to create space statistics for all indexes.
- missing: only tables and indexes
  - with missing statistics
- dbstatc\_tab: only tables specified
  - in the DBSTATC control table

- `dbstatc_mon`: only tables specified
- in the DBSTATC control table that are relevant for the application monitor
- `dbstatc_mona`: only application tables
- specified in the DBSTATC control table that are relevant for the application
- monitor
- `info_cubes`: only InfoCube tables
- `all_part`: only partitioned tables
- `harmful`: used in connection with
- option `-d` to delete damaging statistics
- `locked`: all tables with locked statistics
- `system_stats`: collects system (CPU,
- I/O) statistics using the DBMS\_STATS package. For more information, see SAP
- Note 601395.
- `oradict_stats`: collects statistics
- for Oracle dictionary objects using DBMS\_STATS package. For more information,
- see *SAP Note* [863811](#).
- `oradict_tab`: validates structure
- for Oracle dictionary objects. For more information, see *SAP Note* [863811](#).

## tape\_address

The SAP tools use this parameter to identify the device addresses to be used to write to a volume (tape). This parameter is also used when you back up to a remote system using `pipe`, `pipe_auto`, or `pipe_box`.

### Syntax

```
tape_address = <dev>|(<dev_list>)
```

End of the code.

Default value: none

Required values: addresses of backup devices (tape devices, no rewind) that are to be used for backing up or restoring backups. If you specify more than one device, you must separate the names with commas and enclose the list in parentheses. Pay special attention to the differences in the device address names between tape devices with rewind and those with no rewind. Often, the only difference is that no-rewind addresses have an additional “n” in their name.

### Example

Sample specification of backup device:

- HP-UX, Solaris: `/dev/rmt/0mn`
- AIX: `/dev/rmt0.1`
- Linux: `/dev/nst0`
- Windows: `/dev/nmt0`

When more than one address is specified, BRBACKUP performs [parallel backup](#). BRARCHIVE only uses the first of the specified device addresses (exception: the first two device addresses are used when the `-ss` or `-ssd` option is used). If parameters `tape_address_arch` and `tape_address_rew_arch` were defined, BRARCHIVE uses the devices defined there.

BRRESTORE can also use several backup devices in parallel. See [Restoring Files](#).

The number of device addresses in `tape_address` must agree with the number of device addresses in `tape_address_rew`.

## tape\_address\_arch

BRARCHIVE uses this parameter to identify device addresses for writing to a tape volume.

### Syntax

```
tape_address_arch = <dev>|(<dev_list>)
```

End of the code.

Default value: same as [tape\\_address](#)

Possible values:

Addresses of backup devices (tape devices, no rewind) to be used by BRARCHIVE. If this parameter is not set, BRARCHIVE uses the devices defined in parameter [tape\\_address](#).

## Note

The number of device addresses in `tape_address_arch` must agree with the number of device addresses in `tape_address_arch_rew`.

## tape\_address\_ctl

SAP tools use this parameter to define the control drivers for the mount or dismount commands. These commands cause the tapes to be mounted or dismounted when using a jukebox or autoloader, that is, when `backup_dev_type = tape_box|pipe_box`.

### Syntax

```
tape_address_ctl = <dev>|(<dev_list>)
```

End of the code.

Default value: same as [tape\\_address\\_rew](#)

<dev>: control driver address

## Note

The number of control driver addresses in `tape_address_ctl` must agree with the number of device addresses in `tape_address`.

See also:

[Backup with Automatic Tape Changers](#)

## tape\_address\_ctl\_arch

This parameter is used by BRARCHIVE to define the control drivers for the mount or dismount commands. These commands cause the tapes to be mounted or dismounted when using a jukebox or autoloader, that is, `backup_dev_type = tape_box|pipe_box`.

If this parameter is not set, BRARCHIVE uses the drivers specified in the parameter [tape\\_address\\_ctl](#).

### Syntax

```
tape_address_ctl_arch = <dev>|(<dev_list>]
```

End of the code.

Default value: same as [tape\\_address\\_rew\\_arch](#)

Possible value:

<dev>: control driver address

## Note

The number of `tape_address_ctl_arch` must agree with the number specified in [tape\\_address\\_arch](#).

See also:

## tape\_address\_rew

This parameter is used by the SAP utilities to identify the device addresses that will be used to write to a volume (tape). This parameter is also used when you back up to a remote system using `pipe`, `pipe_auto`, or `pipe_box`.

### Syntax

```
tape_address_rew = <dev>|(<dev_list>)
```

End of the code.

Default value: none

Required value: addresses of backup devices (tape devices, no rewind) that are to be used for backing up or restoring backups. If you specify more than one device, you must separate the names with commas and enclose the list in parentheses.

### Example

Sample specification of backup device:

- HP-UX, Solaris: `/dev/rmt/0m`
- AIX: `/dev/rmt0`
- Linux: `/dev/st0`
- Windows: `/dev/mt0`

Also see [tape\\_address](#).

### Note

The number of device addresses in `tape_address_rew` must agree with the number of device addresses in [tape\\_address](#).

## tape\_address\_rew\_arch

BRARCHIVE uses this parameter to identify device addresses for writing to a tape volume.

### Syntax

```
tape_address_rew_arch = <dev>|(<dev_list>)
```

End of the code.

Default value: same as [tape\\_address\\_rew](#)

Possible value: addresses of backup devices (tape devices, rewind) to be used by BRARCHIVE. If this parameter is not set, BRARCHIVE uses the devices defined in parameter [tape\\_address\\_rew](#).

### Note

The number of device addresses in `tape_address_rew_arch` must agree with the number of device addresses in parameter [tape\\_address\\_arch](#)



See also [tape\\_address\\_arch](#).

## tape\_copy\_cmd

This parameter defines the command used to copy database files (not to raw devices) and non-database files from disk to tape.

### Syntax

```
tape_copy_cmd = cpio|dd|rman|rman_dd|
cpio_gnu|dd_gnu| rman_gnu|rman_dd_gnu
```

End of the code.

Default value: `cpio`

Possible values:

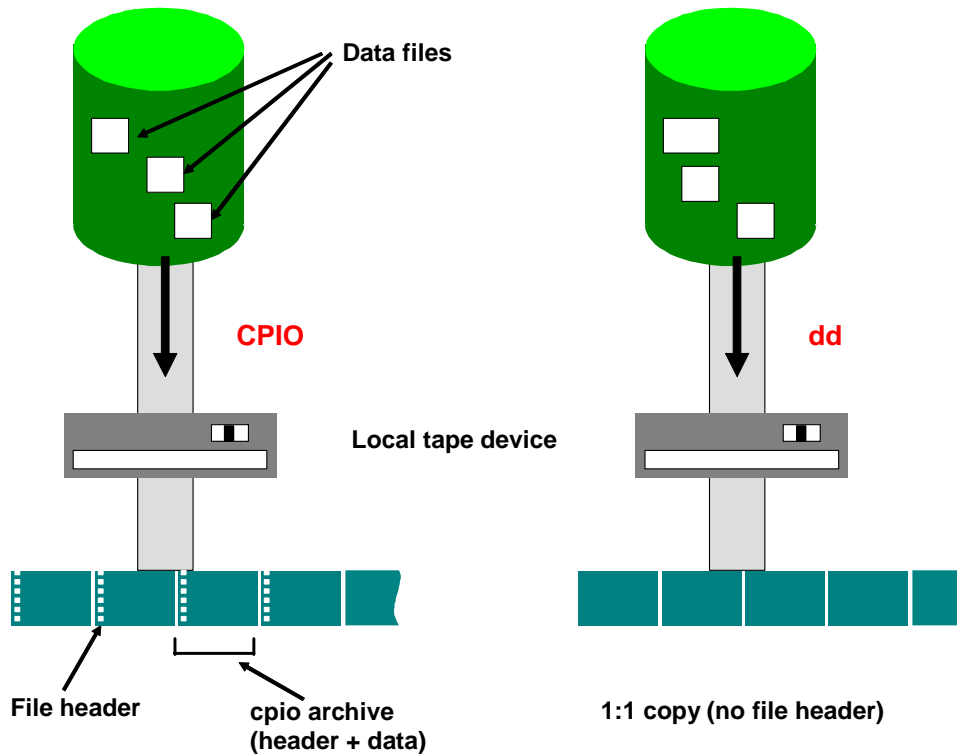
- `cpio`: copies data with the `cpio` command
- `dd`: copies data is copied with the `dd` command
- `rman`: performs backup with Oracle Recovery Manager. Data is copied by SAP backup library directly to tape.
- `rman_dd`: Backup with Oracle Recovery Manager. Data is copied by SAP backup library piped through `dd` command to tape.

### Note

The suffix `gnu` indicates that the GNU command syntax is used to copy files. On some platforms, GNU tools offer better performance.

For more information on `rman` and `rman_dd`, see [RMAN Backup with the SAP Backup Library](#).

This parameter does not affect either raw devices (they are always copied with `dd`) or directories (they are always copied with `cpio`). Tape header files (`tape file label`, `init_ora`, `init_sap`) and tape end files (`space_log`, `det_log`, `sum_log`) are always written with `cpio`.



The use of `dd` commands to back up the database can improve performance and so reduce the backup time.

**Caution**

The `dd` command reports an I/O error on UNIX platforms when it reaches the physical end of the tape. Do not confuse this message with the same message for hardware problems.

Check whether the end of the tape was really reached, taking the tape capacity into consideration. Reduce the value of parameter `tape_size` in this case.

However, if you suspect a hardware problem, start the same backup to the same tape with `tape_copy_cmd = cpio` to identify the situation.

**Note**

If you set the parameter `tape_copy_cmd = dd`, we recommend that you set parameters `dd_flags` and `dd_in_flags` as follows:

- UNIX
 

```
dd_flags = "obs=xk bs=xk"
dd_in_flags = "ibs=xk bs=xk"
```
- Windows
 

```
dd_flags = "bs=xk"
dd_in_flags = "bs=xk"
```

where `x >=16`

The `dd` options `obs` and `ibs` are *not* supported on Windows.

## tape\_pos\_cmd

This parameter defines the command for positioning the read head on the tape.

### Syntax

```
tape_pos_cmd = <cmd>
```

End of the code.

Default value: none

This parameter is used by `BRRESTORE` and by `BRBACKUP/BRARCHIVE` with option `-w|-verify`. For a file to be read from the volume, the tape must be fast-forwarded to the required position.

For example, you could make the following entry:

```
tape_pos_cmd = "mt -t $ fsf $"
```

### Example

Sample positioning commands:

- HP-UX: "mt -t \$ fsf \$"
- AIX: "tctl -f \$ fsf \$"
- Linux, Solaris, Windows: "mt -f \$ fsf \$"

Make sure that you do not forget the double quotes when you enter parameters composed of several values. The SAP tool replace the first `$` character with the address of the device you use. The second `$` character is replaced by the number of files that are to be skipped.

## tape\_size

This parameter defines the physical tape storage size (tape length \* write density) in gigabytes (G), megabytes (M) or kilobytes (K) for the tapes that are used for database backups and for backing up redo log files.

If you use software or hardware compression for files, this parameter specifies the total size of the files that can fit on the tape *after* compression. Make sure that the largest file for backup is no larger than the value specified in `tape_size` (after compression, when applicable).

### Syntax

```
tape_size = <size>K|<size>M|<size>G
```

End of the code.

Default value: 1200M

See also:

[Tape Volume Size](#)

## tape\_size\_arch

This parameter defines the tape size in gigabytes (G), megabytes (M) or kilobytes (K) used by BRARCHIVE. Use this parameter if different tape stations are addressed with parameters `tape_address_arch` and `tape_address_rew_arch` than with `tape_address` and `tape_address_rew`. You can then use different tape station types for BRBACKUP and BRARCHIVE without risking a tape overflow.

### Syntax

```
tape_size_arch = <size>K|<size>M|<size>G
```

End of the code.

Default value: same as [tape\\_size](#)

## tape\_use\_count

This parameter specifies the maximum number of times a volume can be written to. This is only a recommendation, and can be changed when the volume is still in a condition suitable for backup.

### Syntax

```
tape_use_count = <number>
```

End of the code.

Default value: 100

## uncompress\_cmd

This parameter defines the command to decompress files that were backed up with software compression.

### Syntax

```
uncompress_cmd = <cmd>
```

End of the code.

Default value: none

The command must contain two \$ characters, one for the file before decompression and one for after.

### Example

For example, you could enter the following command:

```
uncompress_cmd = "uncompress -c $ > $"
```

Be sure to follow the syntax rules. In particular, the double quotes are essential.

## util\_options

This parameter defines additional BACKINT options, which BR\*Tools places after the standard command-line options when calling the BACKINT program. The customer and backup partner are responsible for making sure that the values of these parameters do in fact correspond to the actual configuration of the disk storage and BACKINT functionality.

## Syntax

```
util_options = "<additional_backint_options>"
```

End of the code.

## util\_par\_file

This parameter defines the location of the parameter file with the specifications that might be required for a backup with an external backup program.

If you select the device type `backup_dev_type = util_file` or `util_file_online` (if supported), you initiate backup by a non-SAP backup program that is addressed through the BACKINT interface. Contact the supplier of these programs for the required specifications. To transfer additional parameters to the external backup program when you call one of the SAP utilities, you might have to create a parameter file with these values.

If possible, use parameter `util_file_online` for an online backup, since in this case the backup status of the individual tablespaces is set dynamically, which means that the volume of redo log entries is much smaller than if you use `util_file`. For more information, see [External Backup Programs](#).

## Syntax

```
util_par_file = <file>
```

End of the code.

Default value: none

Required value: directory and the name of the parameter file. If you have not entered a path, the system searches for the parameter file in the directory as follows:

- UNIX: `<ORACLE_HOME>/dbs`
- Windows: `<ORACLE_HOME>\database`

## util\_vol\_access

This parameter defines the type of access to the files backed up with [util\\_vol](#), in addition to the obligatory restore of the files to the original location (snap-back, clone-back) mainly for the purpose of a verification. The values `copy` and `both` also enable a restore to other directories.

## Syntax

```
util_vol_access = none|copy|mount|both
```

End of the code.

Possible values:

- `none`: specifies that there are no possibilities for additional access on the database server of the local computer. In this case such access on another computer is required for verification.
- `copy`: specifies that the backup tool can copy saved files individually to another target directory, which can be a temporary location for verification purposes or a new restore location
- `mount`: specifies that the saved files can be mounted locally by the backup tool with other paths, where they can be accessed for verification purposes

- `both`: specifies that both access types, `copy` and `mount`, are possible

## util\_vol\_nlist

This parameter defines the smallest unit to be considered as a disk volume in the corresponding configuration.

This parameter defines a list of non-database files or directories that are located on the database disk volumes but that need not appear in the list of files to back up in the input file. These files can be automatically included in the backup, but are never reported in the BACKINT interface messages.

### Caution

During a restore, these files might be overwritten without warning.

### Syntax

```
util_vol_nlist = (<nfile_name1>, <nfile_name2>, ...) | no_check
```

End of the code.

Possible value:

`no_check`: deactivates the BACKINT check of the backup volumes. This check makes sure that the backup volumes do not contain either non-database files or database files belonging to a database other than the one to be backed up.

### Caution

When `no_check` is set, you must make sure that the database volumes (directories `sapdata`, `origlog`, and `mirrlog`) only contain database files of the database to be backed up. Or, if the database volumes contain either non-database files or database files from a database other than the one to be backed up, be aware that such files can be overwritten without warning.

## util\_vol\_unit

This parameter defines the smallest unit to be considered as a disk volume in the corresponding configuration.

### Syntax

```
util_vol_unit = disk_vol | sap_data | all_data | all_dbf
```

End of the code.

Possible values:

- `disk_vol`: specifies that the smallest unit is a logical volume, file system, raw disk , or disk drive. This value is used when a snapshot or clone is created at logical-volume level, and the volumes are mounted on the subdirectories of the `sapdata` directories, as in the following example:

### Example

```
/oracle/SID/sapdata5/fact_1/fact.data1
```

is located on a separate file system, which is mounted on:

/oracle/SID/sapdata5/fact\_1

#### Caution

This is allowed but does not conform to SAP conventions.

- `sap_data`: specifies that the smallest unit is a `sapdata`, `sapraw`, `origlog`, or `mirrlog` directory. This value is used when a snapshot or clone is created at logical-volume or volume-group level, and the volumes are mounted on `sapdata`, `origlog`, or `mirrlog`. This means that the disk configuration is such that all the directories `sapdata`, `origlog`, or `mirrlog` are mount points that can be separately and independently split. If the split is done on volume-group level, only one logical volume can be created for each volume group.
- `all_data`: specifies that the smallest unit is all `sapdata`, `sapraw`, all `origlog`, or all `mirrlog` directories. This value is used when many or all `sapdata` directories are located on logical volumes belonging to the same volume group and the split can only be performed at volume group level. With this configuration, `origlog` and `mirrlog` directories must be on separate volume groups. Therefore, in this category there are at least three volume groups.
- `all_dbf`: specifies that the smallest unit is all database files. This value is used when all `sapdata` and all `origlog` and `mirrlog` directories are located on a single volume group and the volume group as a whole can be split. This means that there is only one split for all database files.

#### Caution

This is allowed but does not conform to SAP conventions because it contradicts the SAP recommendation on the separation of data files and redo log files.

## volume\_archive

BRARCHIVE uses this parameter to identify the volume set to be used for the backup of the offline redo log files.

#### Syntax

```
volume_archive = <vol>|(<vol_list>)
```

End of the code.

Default value: none

Required values: names of volumes (tapes) that are to be used for a backup run. The length of the volume name is limited to 10 characters.

If you specify more than one volume, you must separate the names with commas and enclose the list in parentheses. You can also enter `SCRATCH` to deactivate automatic volume management.

When BRARCHIVE starts, the automatic volume management checks all the volumes in the sequence of their names in `volume_archive` (only volumes whose expiration period has expired are suggested for backup). The volumes are suggested cyclically.

#### Example

A valid volume name was found based on parameter `volume_archive` (that is, the expiration period for this volume has expired). The program assumes that you have mounted this volume. Once several check mechanisms have run, backup starts using that volume.

See [Selecting Volumes Automatically](#).

The expiration period can be configured using parameter `expir_period`. The automatic volume management ensures that only free volumes are offered for backup, for example, those for which the expiration period has expired. As a result, you have to supply a quantity of volumes that is large enough for your selected expiration period.

#### Example

You perform backups once a day, one volume (tape) is required; parameter `expir_period` is set to 14 (each volume is locked for 14 days at a time).

In this case, you must specify at least 14 volumes to ensure that a volume is always available. Even better is to add a buffer of around 25% – for example, specifying 18 or more volume names in this case.

To deactivate the automatic volume management, use the command `volume_archive = SCRATCH`. In this case, you can mount any volume whose expiration period has expired. This name is then also recorded during the backup.

## volume\_backup

BRBACKUP uses this parameter to identify the volume set to be used for the backup of the database or non-database files.

#### Syntax

```
volume_backup = <vol>|<vol_list>
```

End of the code.

Default value: none

Required values: names of volumes that are to be used for database or other backups. The length of the volume name is limited to 10 characters.

If you specify more than one volume, you must separate the names with commas and enclose the list in parentheses. You can also enter `SCRATCH` to deactivate automatic volume management.

When BRBACKUP starts, the automatic volume management checks all the volumes in the sequence of their names in `volume_backup` (only volumes whose expiration period has expired are suggested for backup). The volumes are suggested cyclically.

#### Example

You have two backup devices available, and three valid volumes were found in parameter `volume_backup` (the expiration period for these volumes has expired). The program assumes that you have mounted two of the selected volumes in the devices. If a third volume is required to complete the backup, the program prompts you to mount the third selected volume at the appropriate time.

See [Selecting Volumes Automatically](#).



The expiration period can be configured using parameter `expir_period`. The automatic volume management ensures that only free volumes are offered for backup, for example, those whose expiration period has expired. As a result, you have to supply a quantity of volumes that is large enough for your selected expiration period.

#### Example

You perform a backup once a day, two volumes (tapes) are required; parameter `expir_period` is set to 14 (each volume is locked for 14 days at a time). In this case, you must specify at least 28 volumes to ensure that a volume is always available. Even better is to add a buffer of around 25%, for example, specifying 35 or more volumes in this case.

To deactivate the automatic volume management, use the command `volume_backup = SCRATCH`. In this case, you can mount any volume whose expiration period has expired. This name is then also recorded during the backup.

## Logs for BR\*Tools

Logs are created for BRBACKUP, BRARCHIVE, BRRESTORE, BRRECOVER, BRCONNECT, and BRSPACE. These supply information about database operations and are useful for problem analysis.

They are also a prerequisite for the smooth execution of the operations that you can perform using the SAP programs (for example, for the restore and recovery process).

#### Note

Be sure to include logs if you need to create a problem message, as this simplifies error analysis.

For more information, see:

- [Log Types](#)
- [BRBACKUP Logs](#)
- [BRARCHIVE Logs](#)
- [BRRESTORE Logs](#)
- [BRRECOVER Logs](#)
- [BRCONNECT Logs](#)
- [BRSPACE Logs](#)

## Log Types

The following types of logs are written:

- [File System Logs](#)
- [Database Logs](#) in tables SDBAH and SDBAD

# File System Logs

The BR\*Tools each write the following file system logs:

- A detail log generated each time a process is started
- A summary log appended after every run

You can [display the logs](#) with BR\*Tools.

For more information, see:

- [BRBACKUP Logs](#)
- [BRARCHIVE Logs](#)
- [BRRESTORE Logs](#)
- [BRRECOVER Logs](#)
- [BRSPACE Logs](#)
- [BRCONNECT Logs](#)

## Database Logs for BRBACKUP, BRARCHIVE, BRSPACE, and BRCONNECT

The database logs of BRBACKUP, BRARCHIVE, BRSPACE, and BRCONNECT are stored in tables SDBAH and SDBAD.

### Caution

Do not manually change tables SDBAH and SDBAD. Make sure that only the appropriate SAP tools and the SAP system appends or changes these tables.

This section does not describe these tables in detail, but instead gives an overview of which information BRBACKUP, BRARCHIVE, BRSPACE, and BRCONNECT stores in which tables.

### SDBAH

This table contains information that refers to the backup as a whole:

- Starting time of the backup, BRSPACE or BRCONNECT processing
- End time of the backup, BRSPACE or BRCONNECT processing
- BRBACKUP/BRARCHIVE/BRSPACE/BRCONNECT return code
- BRBACKUP/BRARCHIVE/ BRSPACE/BRCONNECT action ID (encoded timestamp of the file system log names)
- BRBACKUP/BRARCHIVE/ BRSPACE/BRCONNECT function ID (extension of the file system log names)

## SDBAD

This table contains information that refers to the backup of one file:

- File name
- Oracle file ID or log group number
- End time of the backup of the file
- Name of the volume where the file was saved
- Position of the file on the volume
- Backup ID of the external backup program
- Compression rate of the software compression

In addition, table SDBAD contains internal BRBACKUP information about compression rates and backup durations for the individual database files.

For BRSPACE and BRCONNECT, SDBAD tables contain information on the total number of objects processed by BRSPACE or BRCONNECT.

## Log Supplements

Using the option `-o dist|time[,time|dist]` or the option `-w|-verify` causes the detail log to be supplemented (the effects are similar for all the SAP tools). See [-o|output](#) or [-w|-verify](#) (for BRBACKUP), [-w|-verify](#) (for BRARCHIVE).

### Log Supplements with `-o dist`

Before the backup flow is logged, the SAP tool records information about the distribution of the files for backup among the volumes (tapes or disks) used. The following information is listed:

```
position size rate compressed duration speed name
```

- `position`: position of the file on the volume
- `size`: size of the file (in bytes)
- `rate`: compression rate when hardware or software compression is used. The compression rate is specified as: `<value>:1, value=<size (file size before compression)>/<compressed (file size after compression)>`.

If no compression rate is available yet for this file, the default value 3:1 is taken as the compression rate for most files (indicated by an asterisk "\*" after the compression rate). Exceptions: the files of tablespaces PSAPSOURCED, PSAPLOADD, PSAPPOOLD, PSAPDOCUD, PSAPCLUD and the online redo log files; the default value 1:1 is used for the compression rate.

- `compressed`: size of the file (in bytes) after their compression
- `duration`: indicates how long the last backup of this file took (`<minutes>:<seconds>`). This enables you to approximately estimate the duration of the backup.

- `speed`: throughput of the backup (calculation:  $\text{speed} = \frac{\text{size}}{\text{duration}}$  in MB/h)
- `name`: file name

This information is listed for each volume (tape or disk). The last line (`total`) shows the totals or averages of the individual columns. An asterisk '\*' in column `duration` indicates that the BRBACKUP optimized distribution of the files among the backup devices for time (only relevant for parallel backups). The goal of this is to balance the load of all the BRBACKUP backup devices. When optimization by time was not possible BRBACKUP attempts to equally distribute the data volume.

## Log Supplements with `-o time`

During the processing, SAP tools issue a timestamp after every important event as follows:

```
BR0284I <BRTOOL> time stamp YYYY-MM-DD hh.mm.ss, elapsed time:
<minutes>:<seconds>
```

You can use this information, for example, to track how much time was actually required for a backup (`<timestamp after operation complete>` minus `<timestamp at operation start>`). For a parallel backup, these two timestamps need not appear one after another.

After the log entries on the backup flow, a list appears with information about the saved files:

```
position duration size speed compressed rate name
```

This information was described above with the use of option `-o dist`. Of course, the values in column `duration` have now been updated. If you use software compression used, you can also determine the current compression rate.

## Log Supplements with `-w|-verify`

- Indicates that verification is active and that twice the backup time is required as a result
- Information about which volume was used for the verification run, which files are to be restored where, and the result of the check (for example, `BR0363I Verification of backup of <file name> successful`)

After successful backup to tape, the files are actually restored to disk, the check mechanisms are activated, and the files are deleted. This approximately doubles the backup time required.

# Messages and Return Codes for BR\*Tools

## Messages

BR\*Tools issues messages in the following form:

```
BR<NNNN><X> Message text
```

NNNN: sequential number of the message

X: type of message:

- I: information
- W: warning

- E: error message

The text of the message contains the following variables:

- %s: string
- %d: integer
- %f: real number in floating-point format

In most cases, the SAP tools issue a group of messages, which enables you to precisely analyze the error.

## Return Codes

The following return codes are possible:

| Cod<br>e | Meaning                                                                              |
|----------|--------------------------------------------------------------------------------------|
| 0        | Successful                                                                           |
| 1        | Warnings - all files were processed (for example, backed up or restored)             |
| 2        | Canceled during the initialization phase by a user or other signal                   |
| 3        | Errors occurred during the initialization phase, processing was not started          |
| 4        | Canceled by a user or other signal during processing                                 |
| 5        | Started, but not completed because errors occurred during processing                 |
| 6        | Internal termination                                                                 |
| 9        | Successful but not finished yet (can be seen in summary logs copied to backup media) |

## Common Features of BRBACKUP and BRARCHIVE

This section describes the *common* features of [BRBACKUP](#) and [BRARCHIVE](#). You can call these tools as follows:

- Directly from the UNIX command level, using the BR\*Tools menu – see [Backup and Database Copy with BR\\*Tools](#).
- From the SAP System, using the [Computing Center Management System for Oracle](#)

BRBACKUP and BRARCHIVE do not use a graphical user interface in a UNIX environment. You can run them in any UNIX window and under any shell.

Both programs use standard commands for backing up the relevant files on a volume:

- `cpio` or `dd` for backup to tape, `cp` or `dd` for backup to disk if you are working with file systems.

- dd for backup of raw devices on tape or disk

You must configure BRBACKUP and BRARCHIVE, using the parameters in the [initialization profile init<DBSID>.sap](#).

The default configurations of both programs often require online user interaction. You can also run the programs without interaction, as described in [Unattended Backup](#).

- BRBACKUP saves database objects as follows:
  - Data files of the database
  - Control file
  - Online redo log files

You can also use BRBACKUP to back up non-database files and directories. See [Backing Up Non-Database Files and Directories](#).

- BRARCHIVE normally backs up the offline redo logs (that is, the online redo logs that Oracle backs up in the archiving directory) to tape. It is also possible to back up the offline redo log files to disk for special purposes.
- BRBACKUP and BRARCHIVE also back up the following files to the volume (when database objects or offline redo log files are backed up) or to the directory named in `backup_root_dir` for a backup of database files to disk:
  - A copy of profile `init<DBSID>.ora`, `init<DBSID>.dba`
  - A copy of profile `init<DBSID>.sap`
  - The detail BRBACKUP and BRARCHIVE log
  - The summary BRBACKUP and BRARCHIVE log
  - The summary BRSPACE log `space<DBSID>.log`, database structure change log `struc<DBSID>.log`, and database parameter change log `param<DBSID>.log`.

The logs are saved on every volume (for example, tape) used for the backup. As a result, you can still determine the contents of the volume, even when the database and file system logs from BRBACKUP or BRARCHIVE have been lost.

- [BRCONNECT](#) monitors the database during a BRBACKUP process. [BRTOOLS](#) is called internally by BRBACKUP and BRARCHIVE. You cannot call BRTOOLS directly yourself for the internal functions.

## More Information

- [Supported Backup Media](#)
- [Backup with Automatic Tape Changers](#)

## Supported Backup Media

You can use [BRBACKUP](#) or [BRARCHIVE](#) for direct backup to the following media:

- Local tape devices: `backup_dev_type = tape`

- Remote tape devices: `backup_dev_type = pipe`
- Local disks: `backup_dev_type = disk`
- Remote disks: `backup_dev_type = stage`
- Tape devices with automatic tape changing (tape stacker, for example). These backup devices can be addressed:
  - Locally with `backup_dev_type = tape_auto`
  - Remotely with `backup_dev_type = pipe_auto`

For more information, see [Backup with Automatic Tape Changers](#).

- Jukebox and autoloader. These backup devices can be addressed:
  - Locally with `backup_dev_type = tape_box`
  - Remotely with `backup_dev_type = pipe_box`

For more information, see [Backup with Automatic Tape Changers](#).

BRBACKUP and BRARCHIVE support:

- Automatic volume management. See [Volume Management](#).
- An open interface to interface program BACKINT, to enable backups using external backup programs. See [External Backup Programs](#).

Using the parameter [tape\\_copy\\_cmd](#) you can choose whether files (not on raw devices) are copied from disk to tape with `cpio`, `dd`, or `RMAN`

## Effects of the Command Options

For more information about command options, see:

- [Command Options for BRBACKUP](#)
- [Command Options for BRARCHIVE](#)
- [Command Options for BRRESTORE](#)
- [Command Options for BRSPACE](#)
- [Command Options for BRCONNECT](#)

Several parameters of the [init<DBSID>.sap profile](#) can be overridden by calling BRBACKUP or BRARCHIVE with the appropriate command option.

The following table displays the corresponding profile parameters and the options that override them:

| Parameter                     | Command Option                                        | SAP tool  |
|-------------------------------|-------------------------------------------------------|-----------|
| <code>archive_function</code> | <code>-s -save -cs  -copy_save</code><br>(and others) | BRARCHIVE |

| Parameter       | Command Option | SAP tool                       |
|-----------------|----------------|--------------------------------|
| backup_mode     | -m -mode       | BRBACKUP                       |
| restore_mode    | -m -mode       | BRRESTORE                      |
| backup_type     | -t -type       | BRBACKUP                       |
| backup_dev_type | -d -device     | BRBACKUP, BRARCHIVE, BRRESTORE |
| compress        | -k -compress   | BRBACKUP, BRARCHIVE, BRRESTORE |
| exec_parallel   | -e -execute    | BRBACKUP, BRRESTORE            |
| volume_archive  | -v -volume     | BRARCHIVE                      |
| volume_backup   | -v -volume     | BRBACKUP                       |
| util_par_file   | -r -parfile    | BRBACKUP, BRARCHIVE, BRRESTORE |
| saveset_members | -s -saveset    | BRBACKUP                       |

## cpio Continuation Tape

If the volume size is defined too large, the `cpio` command might reach the physical end of the volume. In such cases, `cpio` issues a message displaying the end of the volume:

```
(end of tape..., end of medium..., end of volume...)
```

You can continue the backup on another volume or cancel the backup. Note that the `cpio` continuation mechanism is only possible during *serial* backups. If a parallel backup to several tape devices is involved, the backup is terminated for the relevant tape device when the end of tape is reached, but the backup is continued on the other tape devices. In such cases, you must make sure that all the files were saved.

Make sure that the continuation tape is not one of the volumes initialized for BRBACKUP or BRARCHIVE, since no label check takes place for the continuation volume.

The `cpio` continuation tape is not “visible” for BRBACKUP or BRARCHIVE, that is, it is regarded as one logical tape together with the first one. Therefore, in restore situations, BRRESTORE requests one tape. However, make sure that *both* tapes can be mounted.

### Caution

To avoid the situation described above, none of the database files should be larger than the size specified in `tape_size` (after compression, when applicable).

### Note



Do not confuse the `cpio` continuation mechanism with the management of tape continuation, which BRBACKUP operates when another tape is required to back up the files.

## cpio Error

`cpio` might report an I/O error:

```
I/O Error on output
```

If this error only occurs sporadically, check whether it might be related to the use of a specific volume. If so, stop using that volume for backups.

If the error occurs with different volumes, have your backup device inspected, as a hardware problem is probably to blame.

## Canceling a Backup

### Procedure

You can cancel the backup processes. You can use:

- The key combination `CTRL+C` and enter `stop` (entering `cont` to continue the backup is then only possible with restrictions)
- The UNIX command `kill` (but never `kill -9`)
- The command `brarchive -fill stop` to properly stop the archiving of offline redo log files started with `brarchive -f`. Do *not* use `CTRL+C` or a `kill` command to stop such BRARCHIVE runs, as this might terminate an active copy process.

## Other Tools for Oracle DBA

This section describes the tools supplied by SAP for database administration (DBA) with the Oracle database, except for the [BR\\*Tools](#).

For more information on how to work out an approach to Oracle DBA, see [Approach to Oracle DBA](#).

### Prerequisites

You have already [got started with Oracle and the SAP System](#).

### Features

- [Database Recovery with SQLPLUS](#)
- [Oracle Recovery Manager \(RMAN\)](#)
- [The SAP Tools with Windows](#)

# Database Recovery with SQLPLUS

Although [BR\\*Tools](#) can in most cases be used to recover a failed database, there are exceptions to this. Therefore, this section describes how to recover the Oracle database system using SQLPLUS functions.

The information here helps you with database recovery after an error in one of the following database components:

- Data files
- Online redo log files
- Control files

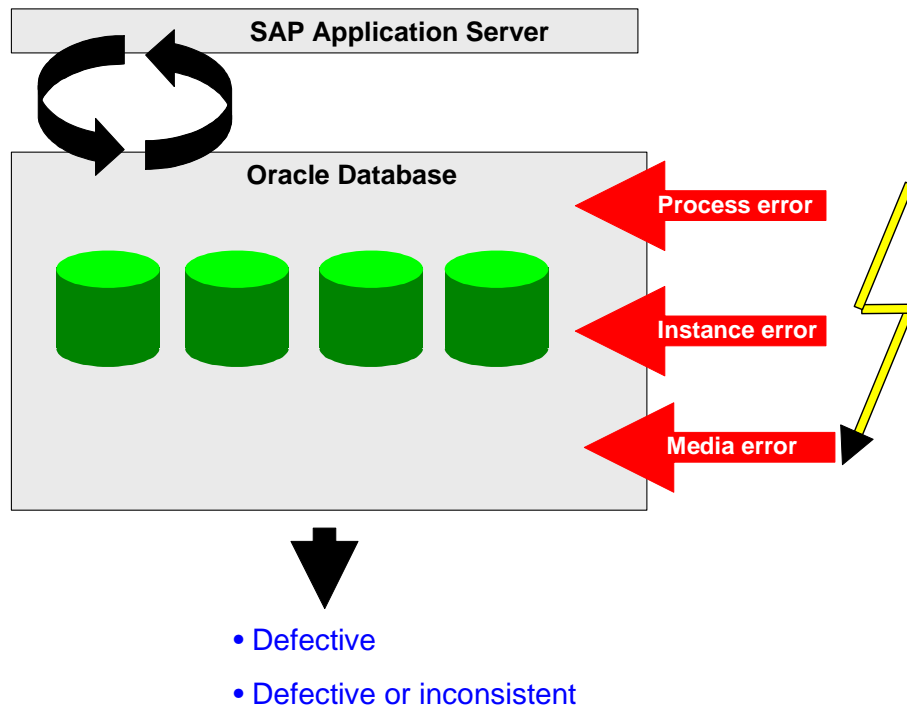
## Caution

An incorrectly performed recovery might lead to irreparable loss of data in your database. We recommend you to always discuss your recovery plan with an experienced database administrator, your SAP consultant, or an Oracle specialist before you start a recovery.

## Types of Database Errors

There are a number of errors that affect the operations of an Oracle database or can cause the database to crash. Depending on the type of error, a recovery can either be performed automatically or must be performed by the user or database administrator.

Therefore, you must find out the exact type of error so that you take the appropriate action to correct the error.



Errors can generally be divided into the following categories:

- User errors, see [Recovery after User Errors](#)
- Statement errors, see [Recovery after Statement Errors](#)
- Process errors, see [Recovery after Process Errors](#)
- Instance errors, see [Recovery after an Instance Error](#)
- Media errors, see [Recovery after Media Errors](#)

## More Information

[Error Analysis](#)

## Error Analysis

Information on all types of database problems can be found in the trace and ALERT files, where the actions in the database are recorded.

## Structure

- In the ALERT file, important system events and significant error messages are continually collected for the entire system. The ALERT file can inform you, for example, which of the online redo log groups is the current one.

ALERT file: <SAPDATA\_HOME>/saptrace/background/alert\_<DBSID>.log

- Trace files are created by the individual background and user processes of the database system. Trace files thus offer more detailed information for error analysis than the ALERT file.

Trace files of the background processes:

```
<SAPDATA_HOME>/saptrace/background/<process>_<number>.trc
```

Trace files of the user processes:

```
<SAPDATA_HOME>/saptrace/usertrace/<process>.trc
```

The directories where the ALERT and trace files are stored are defined by the `init.ora` parameters `background_dump_destination` and `user_dump_destination`.

The naming conventions for the trace files depend on the operating system. Under UNIX, the name of the trace file contains the operating process number and the Oracle process number.

A trace file can contain the following information:

- Date and timestamp
- Version number
- Process information
- Error messages
- Calling stack trace
- Dumps from the System Global Area (SGA) and the Program Global Area (PGA).

For more information, see the Oracle documentation.

## Recovery after User Errors

A user error occurs when a user deletes or falsifies data (for example, deletes a table or program that is required for further system operation), either by mistake or due to lack of knowledge.

You can generally correct such a user error if you can meet the following requirements:

- You exported the object using the SAP tool R3trans, thus backing it up. You can use this copy to restore the condition of the object at the time of the export, taking into account possible database inconsistencies.
- An object from the ABAP Dictionary or the ABAP Repository is involved. The ABAP Dictionary and the correction system both perform version backups of these objects within the SAP system. If you can continue working with that version of the object (ideally, the object has not been changed recently), then you can restore it.

In general, you cannot use the Oracle Export/Import tools to recover a lost SAP object. The reason for this is that the SAP database tables are often shared system-wide. A user cannot import the ATAB (central control table) to recover an individual SAP table, for example, as this risks overwriting the work of other users.

You cannot recover an object by recovering the database either. The recovery of a lost object requires an incomplete recovery up to the moment the user error occurred (point in time recovery). Any changes made to the database from that moment on are lost.

## Recovery after Statement Errors

A statement error occurs when an error in executing a database statement occurs in a running program.

If such an error occurs, the database system cancels the incorrect statement and issues an error message. The program containing the error is terminated. Furthermore, the transaction is completely canceled in the database (rollback), an ABAP dump is issued, and a message is usually written to the system log.

A statement error can also occur if an extensive operation entirely fills up the rollback segment. The reason for such an error is generally incorrect programming.

The database administrator does not have to intervene in order to execute a recovery after a statement error.

## Recovery after Process Errors

A process error occurs when a user process is canceled.

The Oracle instance is not normally affected by the termination. The Oracle process monitor (PMON) responds by canceling the database changes made by current transactions rollback and releasing the resources that were used by the process. Work with the database system can then continue normally.

The database administrator does not have to intervene to perform a recovery after a process error.

## Recovery after an Instance Error

An instance error occurs when the Oracle database instance (System Global Area (SGA) and the corresponding background processes) can no longer run.

An instance error can result from a hardware problem (such as a power failure) or a software error (for example, the crash of the operating system or of an Oracle background process).

An instance error generally results in an immediate abnormal termination of the entire instance. Even if the database system remains active, the data in the SGA is lost in any case, and the instance can no longer be shut down in the conventional way with `shutdown normal` or `shutdown immediate`.

Since only an abnormal termination is usually possible, using `shutdown abort`, the instance must be recovered. Only transactions completed normally with "COMMIT" can be processed; all others are rolled back. If you are working with a standard configuration, the system monitor (SMON) of the database system automatically carries performs the recovery of the instance when you restart the instance (instance recovery). It uses the entries in the appropriate redo log files to do so.

You do not need to intervene during the recovery, provided no database files were changed. At least one copy of the control file, all data files, and at least one online redo log file in each online redo log group must exist.

Before you restart the database system, find out which background process was canceled and why. Check the ALERT and trace files.

## Recovery Procedure

If you want to perform a recovery after an instance error, shut down the instance (if it is still active) with the command `shutdown abort`. Analyze and correct the problem. Restart the database with the command `startup`.

When you restart the database, the system monitor performs an automatic instance recovery, as described above.

## Recovery after Media Errors

A media error occurs when a read or write error takes place in a database file due to a physical defect on the disk drive where the file is located. The most frequent physical defect is a head crash, which usually renders all the files on a disk unusable.

In most cases, the database must be recovered after a media error. The recovery strategy depends on the type of damage in the database. You must therefore analyze the error and understand how it was caused before you can continue with the recovery.

### Caution

If you are not sure how to perform a recovery, be sure to contact your SAP consultant or an Oracle specialist. Do not attempt a recovery if you are not confident.

An incorrectly performed recovery can lead to irretrievable loss of data in your database. The damage you risk is much more costly than the time you spend on a carefully planned, properly executed recovery of your database.

In addition to the ALERT and trace files, SAP recommends analyzing the database using *Complete Recovery Check the Status of Database Files* in BRRECOVER. If defects in the data files of the database are involved, BRRECOVER can often perform the database recovery automatically.

See [Restore and Recovery with BR\\*Tools](#).

This section provides an overview of the recovery process using Oracle SQLPLUS. The sections below contain examples of the command line operations with SQLPLUS.

If you have followed the SAP recommendations regarding the installation and operation of the database system, you should generally succeed in carrying out a complete recovery after a media error. If, however, in an extreme case your backups of the database and your redo log backups have been damaged too, you might only be able to perform an incomplete recovery.

The procedures required for the following errors are listed in the following:

- Loss of one or all copies of the control file
- Loss of an online or offline redo log file or even an entire online redo log group
- Loss of a system, temporary, user, or index tablespace
- Error in the archiving process for online redo log files using ARCH

These errors are usually media-related (for example, if the archiving directory for the offline redo log files is lost or full).

## More Information

See also:

- [Recovering from One Control File Missing](#)
- [Recovering from Control Files Missing](#)
- [Recovering from Current Online Redo Log Missing](#)
- [Recovering from One Inactive Online Redo Log Missing](#)
- [Recovering from User Tablespace Missing](#)
- [Recovering from SYSTEM Tablespace Missing](#)
- [Recovering from Index Tablespace Missing](#)
- [Recovering from Errors During the Archiving of Online Redo Logs](#)
- [Performing an Incomplete Recovery](#)
- [Finishing an Incomplete Recovery](#)
- [Automating the Recovery](#)
- [Updating the Control File](#)

## Recovering from One Control File Missing

You use this procedure if a copy of the control file cannot be read or written to.

Database activities continue normally until the next update of the control file – that is, until the next checkpoint or redo log file switch – and the database then crashes.

In this case you do not need to perform an explicit recovery of the database. If a copy of the control file exists, the system can perform a recovery automatically.

### Procedure

You have the following main recovery options after one of the control files is lost:

- Replace the missing control file

Find out why the control file was lost. For example, you cannot replace the file when the disk is not available.

Check the ALERT and trace files of the database system to analyze the error. The error messages will indicate which control file is missing, and may also indicate how it was lost.

Replace the missing control file as follows:

1. Shut down the database and exit SQLPLUS.
2. Copy an existing control file to the desired storage location at operating system level (see parameter `control_files` in the `init.ora` profile).
3. Start SQLPLUS and start up the database.

- Modify the database system

Check the ALERT and trace files to find out which control file was lost.

1. Shut down the database and exit SQLPLUS.
2. Delete the missing file from parameter `control_files` in profile `init.ora` (default: `<ORACLE_HOME>/dbs/init<DBSID>.ora`).
3. Start SQLPLUS and start up the database. No other actions are required for recovery.

This solution is only acceptable if you have at least two copies of the control file. This guarantees that at least one mirror copy of the file still exists, even when one control file has been deleted. The database should always have at least two control files, original and mirror. In the SAP system, the control files are mirrored in three or more directories. For more information, see [Mirroring the Control File](#).

## Recovering from All Control Files Missing

You use this procedure if you have lost all copies of the control file.

The normal database activities continue until the next update of the control file. When this happens – during the next checkpoint or redo log file switch, at the latest – the database system crashes.

### Prerequisites

A complete recovery of the database is possible provided one of the following conditions is true:

- A current backup copy of the control file, that is, a copy with the current structure of the database, exists.
- A current log of the files in the database exists, enabling you to create the control file again.

If all the control files (even the backups) are lost, you must first reconstruct the control file before you can start the recovery process. This procedure is much more complicated and not always successful.

### Procedure

#### Recovery Using the Backed-Up Control File

This procedure assumes that you are able to restore the control file from your last database backup.

To update the database, the appropriate redo log files must exist.

It is helpful that the saved control file reflects the current structure of the database. The paths for the data and log files and the status of the log sequence numbers are not important, but the control file should have the exact information about the number of files and – indirectly – the number of tablespaces in the database.

Proceed as follows for recovery:

1. If the database system is still operating, shut down all instances with the following SQLPLUS command:



```
shutdown abort
```

ABORT is generally necessary because the control files are no longer available to include a checkpoint during the shutdown.

2. Use the ALERT and trace files to analyze the error.

Check whether other damage has occurred to the database: Find out whether all data files and redo log files are readable.

Back up the online redo log files of all instances (if they exist in readable form) so that you can repeat the recovery process if an error occurs.

3. Place the backup copies of the control file in the directories or on the raw devices specified in the `control_files` parameter in the `init.ora` profile.

If further files were damaged, restore the backup copies of these files. You do not need to restore undamaged files from the backup. If you have to restore data files, you also have to restore all the offline redo log files of all instances that have been backed up since the last backup (for SAP databases, offline redo log files are usually backed up by the BRARCHIVE program) in the local backup directory (default value: `<SAPDATA_HOME>/oraarch`). For more information on recovery after the loss of redo log or data files, see the relevant parts of this documentation and your Oracle documentation.

4. Enter the following SQLPLUS commands to mount the local instance:

```
connect / as sysdba
```

```
startup mount
```

5. If you were not able to load backed-up files to their original directories or had to change file name, update the control file, by changing path or group names:

```
alter database rename file '<file name>' to '<file name>;'
```

See [Updating the Control File](#).

6. If the data files of the database were set to status OFFLINE during the shutdown, change the status of the files in the control file to ONLINE.

To find OFFLINE files, search for "offline" in the ALERT file or check the `v$datafile` view:

```
select * from v$datafile
```

To change the status of a data file in the control file, use the following command:

```
alter database datafile '<file name>' online;
```

See [Updating the Control File](#).

7. Start recovery with the following SQLPLUS command:

```
recover database until cancel using backup controlfile;
```

8. If you are prompted to do so, enter the full path name for the redo log files required for recovery, including the active online redo log file.

9. When all redo log files are processed, end the recovery process with the command `cancel`.

10. After the message recovery canceled, you can reopen the database by using one of the following SQLPLUS commands:

```
alter database open resetlogs;
```

```
alter database open noresetlogs;
```

The `resetlogs` option initializes the existing online redo log files. Therefore, only use this option after an incomplete recovery. Do *not* use this option after a complete recovery.

The `noresetlogs` option causes the online redo log files to be used in their current form. A complete recovery is required to use this option.

The database system resumes operations with the log sequence number following the number of the last current online redo log file.

11. Perform a complete backup of the database.

The backup is necessary to back up the control file and to guarantee a recovery of the database if further database problems occur.

For more information, see [Finishing an Incomplete Recovery](#).

## Database Recovery Using a New Control File

If you do not have a valid copy of the control file, a full recovery is still possible by reconstructing the control file. To do this, you need a current log of all the database files, for example, the BRBACKUP log.

Proceed as follows during recovery:

1. If the database is still active, shut down all instances with the following SQLPLUS command:

```
shutdown abort
```

ABORT is generally necessary because the control files are no longer available to include a checkpoint during the shutdown.

2. Use the ALERT and trace files to analyze the error.

Make sure no further damage has occurred in the database, and find out whether all data files and online redo log files exist in readable form.

Back up the online redo log files of all instances (if they exist in readable form) so that you can repeat the recovery process if an error occurs.

3. If other files were damaged, restore the backup copies of these files. You do not need to restore undamaged files from the backup. If you have to recover data files, also restore all the offline redo log files of all instances that have been backed up since the backup of these data files in the backup directory.

4. Enter the following SQLPLUS commands to demount the database:

```
connect / as sysdba
```

```
startup nomount
```

5. Create the control file (for syntax information, refer to your Oracle documentation):

```
create controlfile database <name> logfile '<online redo log groups>' noresetlogs|resetlogs maxlogfiles 10 maxlogmembers
```

```
<your value> datafile '<names of all data files>' maxdatafiles  
1022 archivelog;
```

Path names: the path names of the online redo log files and data files can be found in the last detail log from BRBACKUP.

`noresetlogs/resetlogs`: only select `resetlogs` when an online redo log group was lost in addition to the control file. Otherwise always use `noresetlogs`.

6. Mount the database:

```
alter database mount;
```

7. Start the recovery:

```
recover database [until cancel] [using backup controlfile];
```

Note

A recovery is required whenever the control file was generated with `resetlogs` or when a data file was restored. Recovery is also recommended for security reasons in other cases.

You must select the option `using backup controlfile` when you used the `resetlogs` option to create the control file. If you select `until cancel`, you can interactively decide how many files of the existing redo log files you want to apply during the recovery. Enter all the redo log files of all instances, including the current ones.

8. Start up the database with this SQL command:

```
alter database open [noresetlogs/resetlogs];
```

- Use `alter database open` if you created the control file with `noresetlogs` and have performed no recovery or a complete recovery (without `until cancel`).
- Use `alter database open noresetlogs` if you created the control file with `noresetlogs` and performed a complete recovery despite the use of the `until cancel` option.
- Use `alter database open resetlogs` if you created the control file with `resetlogs` or when you performed an incomplete recovery.

9. After the recovery, perform a complete backup to save the newly created control file and to make sure that a recovery of the database in the event of failure is possible.

## Recovering from Current Online Redo Log Missing

A member of the group - or the entire group - of current online redo log files (that is, the redo log files in which the database changes are currently being recorded) is lost.

Use the entries in the ALERT file and in the LGWR trace file to analyze the error situation. You must check all sources of information for possible LGWR errors. Even if an error allows the instance to continue running (for example, at least one member of the current group can be written to, errors only in the other members), the error should be corrected as soon as possible.

If you have not been mirroring the online redo log files, as supported by Oracle (and have also not been using hardware-based mirroring), the risk of losing online redo log entries is significantly higher. To be able to perform a full recovery, only the entries from the current online redo log file are necessary. Use mirroring to guard against complete loss of the online redo log files. If the mirrored online redo log files are available, you can use these later to perform a complete recovery of the database. Otherwise you can only recover the database to the point of the missing redo log entries (that is, an incomplete recovery with loss of data).

For this reason, we strongly recommend once again that you make use of the Oracle options for mirroring the online redo log files.

## Prerequisites

You must meet the following requirements:

- You used the Oracle option for mirroring the online redo log files (or have hardware-based mirroring), and therefore have at least one copy of each online redo log file (SAP default: two copies of the online redo log files).
- Apart from the one member of the current online redo log group, no other files have been damaged.

If further files have been damaged, restore the missing files and the missing active redo log and follow the recovery procedure for the category of the missing file (control file or files of the system, user, or index tablespaces).

## Procedure

1. If the database system is still active, use the SQLPLUS command shutdown abort to shut it down.
2. Determine the reason why the current online redo log files were lost by examining the ALERT and trace files.
3. Replace the missing online redo log files with a mirrored copy.
4. Start the database with the SQLPLUS command startup.

The system automatically performs an instance recovery.

### Note

If all members of the current redo log group have been lost, you can only perform an incomplete partial recovery. For more information, see the documentation on your Oracle database system or [Performing Incomplete Recovery](#).

## Recovering from One Inactive Online Redo Log Missing

If only one member of an inactive online redo log group has been lost, you can use the recovery procedure described in [Recovering from Current Online Redo Log Missing](#). Experienced users can correct this error without shutting down the database. For more information, see the Oracle documentation.

The recovery procedure is different when the database pauses because a redo log switch to an online redo log file was unsuccessful. None of the members in this inactive online redo log group can be read or written to.

If the problem is temporary (for example, incorrect access rights), you only need to correct it, and you can then use the group again. If the files have been destroyed, however, the group can not be used again.

No data is lost, providing the missing redo log file was fully backed up and the backup can be read for media recovery.

## Procedure

1. Shut down the database with this SQLPLUS command:

```
shutdown abort
```

ABORT is needed because the database system cannot perform a proper shutdown, due to the damaged group.

2. Find out which file is missing, and check the ALERT and trace files for the reason why the redo log files were lost.
3. Mount the database with these SQLPLUS commands:

```
connect / as sysdba
```

```
startup mount
```

4. If you were running the database in ARCHIVELOG mode and archiving of the damaged online redo log group was not complete, you have to temporarily switch to NOARCHIVELOG mode before deleting the defective group, because otherwise the system does not let you delete the files:

```
alter database noarchivelog;
```

5. Delete the damaged online redo log files in one of the following ways:

- o As a group:

```
alter database drop logfile group <group number>;
```

- o As individual files:

```
alter database drop logfile member '<file name>' [,<file name>'];
```

6. Create the new online redo log files (to replace the damaged ones, which you just deleted):

```
alter database add logfile '<file name>' [,<file name>'] to group <group number>;
```

7. If the database was set to NOARCHIVELOG mode during these actions, change it back to ARCHIVELOG mode:

```
alter database archivelog;
```

8. If you were running the database in ARCHIVELOG mode, and the archiving of the online redo log group was not complete at the moment the problem occurred, it is essential that you now perform a backup of the entire database. If you do not, the offline redo log chain has a gap, which means that – in the event of another media error – only an incomplete recovery is possible.

# Recovering from User Tablespace Missing

You use this recovery procedure if:

- One or more data files are missing from a tablespace.
- A user tablespace does not contain data from the Oracle Data Dictionary, active rollback segments, or temporary segments.
- Oracle issues error messages when a user attempts to access the involved tablespace. Error information is also written to the database ALERT and trace files.

## Caution

If only one user tablespace is lost, you can perform a manual tablespace recovery without shutting down the database. However, note that user tablespaces are used intensively in the SAP system, and this procedure is therefore only recommended for experienced database administrators. The loss of a user tablespace often has similar consequences for the SAP system as the loss of the SYSTEM tablespace, because the effects of the loss of this one tablespace are very complex.

A complete recovery of the database is possible if you have a backup copy of the corresponding tablespace files and of all redo log files written since the backup.

## Prerequisites

- If you use an SAP database, you shut down the SAP system before starting the recovery procedure. Tables are used so intensively in the SAP system that it is generally impossible to set the affected tablespace to OFFLINE without terminating the activities of many users.
- This procedure describes the recovery procedure when the database is closed. For more information on recovery options with an open database, see the Oracle documentation.

## Procedure

1. If it is running, shut down the database system with this SQLPLUS command:

```
shutdown abort
```

You have to shut down the database with ABORT because the missing files cannot be closed.

2. Inspect the ALERT and trace files to determine the cause of the problem.

The problem is often that an entire disk has crashed, and you need to recover more than one tablespace.

3. Use the log files from the SAP tools BRBACKUP and BRARCHIVE to find the volumes that contain the following files:
  - Last backup of the lost tablespace files
  - Offline redo log files of all instances backed up since the last backup

It is important to identify the location of the lost files. This information appears at the start of the detail log of the BRBACKUP backup that you are using.

4. Restore only the damaged or lost files. You can minimize the time required for recovery by only restoring the missing or damaged files.

You also have to restore the backed-up redo logs of all instances that are required for the recovery. To do this, use [BRRESTORE](#).

5. Mount the database with these SQLPLUS commands:

```
connect / as sysdba  
  
startup mount
```

6. If you were not able to restore backed-up files to their original directories or if you had to change file names, update the control file.

Use the following command to change a path:

```
alter database rename file '<file name>' to '<file name>;'
```

See [Updating the Control File](#).

7. If the data files of the database were set to status OFFLINE when the error occurred, change the status of the files in the control file to ONLINE.

To find the relevant files, search for "offline" in the ALERT file or check the v\$datafile view:

```
select * from v$datafile
```

Change the status of a data file in the control file with this SQLPLUS command:

```
alter database datafile '<file name>' online;
```

See [Updating the Control File](#).

8. Start the recovery with this SQLPLUS command:

```
recover database;
```

When prompted to do so, enter the paths of the offline redo log files that you need to apply.

The system processes online redo logs automatically.

We do not discuss here the alternatives - `recover tablespace` and `recover datafile` - since SAP recommends shutting down the database when an error occurs. The `recover database` command only performs the actions necessary to recover the damaged files. Therefore, it does not take much longer than the `recover tablespace` and `recover datafile` commands.

For information on `recover tablespace` and `recover datafile`, see the Oracle documentation.

9. When the message `recovery complete` is displayed, start up the database system with this SQLPLUS command:

```
alter database open;
```

For more information, see the Oracle documentation.

Note

In most cases, you can use BRRECOVER to correct media errors affecting the data files of a user tablespace. For more information, see [Complete Database Recovery with BR\\*Tools](#).

## Recovering from SYSTEM Tablespace Missing

One or more files of the SYSTEM tablespace has been damaged or lost due to a media error.

Backups of the affected files exist. All offline redo log files that have been written since the last backup are available uninterrupted. The control files and all online redo log files are undamaged.

### Prerequisites

Since the SYSTEM tablespace is affected, the recovery must take place with the database closed. If a backup of the missing files and all redo log entries are available, a complete recovery is possible.

### Procedure

1. If the database system is still active, shut it down with this SQLPLUS command:

```
shutdown abort
```

ABORT is required because the loss of individual files from the tablespace means that the changes from SGA can no longer be recorded, and the database cannot be closed properly as a result.

2. Examine the ALERT and trace files to determine the cause of the problem.
3. Use the log files created by the SAP tools BRBACKUP and BRARCHIVE to find the volumes containing the following files:
  - Last backup of the SYSTEM tablespace
  - Offline redo log files of all instances backed up since the last backup

4. Restore the backups of the damaged or lost files and the backed up offline redo log files of all instances, using [BRRESTORE](#).

5. Mount the database with this SQLPLUS command:

```
connect / as sysdba
```

```
startup mount
```

6. If necessary, change the names and paths of the files in the control file. See [Updating the Control File](#).
7. If required, automate the recovery using the `autorecovery` option. See [Automating the Recovery](#) and read the corresponding Oracle documentation.

8. Start the recovery with the following SQLPLUS command:

```
recover database;
```

9. Depending on the recovery mode (`autorecovery on/off`), the required offline redo log files are either processed automatically or you have to enter their paths and names. The system applies the online redo log files automatically.



10. When the message `recovery complete` is displayed, open the database again with this SQLPLUS command :

```
alter database open;
```

#### Note

In most cases, BRRECOVER can be used to correct media errors affecting the data files of a user tablespace. For more information, see [Complete Database Recovery with BR\\*Tools](#).

## Recovering from Index Tablespace Missing

Generally, Oracle treats an index tablespace just like a user tablespace. Therefore, you can use [the recovery procedure for a user tablespace](#). The procedure below describes an additional recovery option for SAP databases.

### Procedure

1. Shut down the database with this SQLPLUS command:

```
shutdown immediate
```

If this fails, use:

```
shutdown abort
```

2. Find out which data file is affected by the media error, using the information in the ALERT and trace files.
3. Mount the database with these SQLPLUS commands:

```
connect / as sysdba
```

```
startup mount
```

4. Set the data files to OFFLINE:

```
alter database datafile '<complete file name>' offline;
```

5. Open the database:

```
alter database open;
```

6. Make sure that the index tablespaces do not contain any tables. You can check this using the Oracle tables DBA\_SEGMENTS and DBA\_TABLES.

7. Use the corresponding BRSPACE function to create the DDL statements for the affected indexes:

```
brspace -f tbreorg -s <tablespace_name> -d only_ind
```

8. Drop the affected tablespace, including contents:

```
brspace -f tsdrop -t <tablespace_name> -f
```

9. Recreate the affected tablespace:

```
brspace -f tscreate -t <tablespace_name> -d index
```

10. Recreate the indexes with script `ddl.sql` from subdirectory of `sapreorg` with this SQLPLUS command:

```
SQL> @ddl
```

The recovery of the index tablespace is complete.

#### Note

In most cases, you can use BRRECOVER to correct media errors affecting the data files of a user tablespace. For more information, see [Complete Database Recovery with BR\\*Tools](#).

## Recovering from Errors During the Archiving of Online Redo Logs

You use this procedure if there are errors during the archiving of the online redo logs.

For SAP systems with production Oracle databases, the database mode is set to ARCHIVELOG. This mode causes the database system to save an online redo log file – that is, the archive process ARCH is initiated and produces the corresponding offline redo log files in the archiving directory – before the database changes recorded in the redo log files are overwritten during a log file switch.

If the background process ARCH for archiving redo logs fails, the system shuts down until the error is corrected.

This problem is due to the following:

- The background process ARCH was not started
- The archiving directory is full (“archiver stuck”) or is not available

### Procedure

1. Make sure the ARCH archiving process was started by entering these SQLPLUS commands:

```
archive log list
```

If the ARCH process was not started or is not currently active – the line `Automatic archival DISABLED` is displayed – start it by entering these SQLPLUS commands:

```
connect / as sysdba
```

```
archive log start
```

Also check the parameters in the `init.ora` file that control the archiving process. Make sure the archiving process is started automatically during the next database startup. For more information, see [Setting Up Archiving](#).

2. If the disk or archive directory for the offline redo log files is full or is not available, choose one of the two following alternatives:
  - Enter a new target directory for the online redo log archiving.

You probably have to cancel the background process ARCH and restart it, specifying a new target directory. To do this, enter these SQLPLUS commands:

```
archive log stop  
  
archive log start '<new path>'
```

<new path> is the name of the “directory” that will be used for archiving. Note the following special naming conventions for this directory: Oracle can interpret the last part of the specified path as a file prefix. Therefore, it must not physically exist in the directory.

Find out about the default settings of the background process ARCH in the `init.ora` file. For more information, see [Setting Up Archiving](#). The database must remain active when you stop and restart the ARCH process.

Note that the path is automatically reset to the predefined value in profile `init.ora` the next time you start up the database.

Make sure that [BRARCHIVE](#) is able to back up the offline redo log files.

- Use BRARCHIVE to archive and delete the offline redo log files. This frees space in the archive directory.

## Performing an Incomplete Recovery

An incomplete recovery means that some data cannot be recovered after a database error. If one of the following problem constellations occurs, you cannot restore your database completely:

- All copies of the control file have been lost and you do not have any of the following files either:
  - Current backup copy of the control file
  - Log of the files in the database

In such cases, contact your SAP consultant or an Oracle specialist. It might be possible to perform a complete recovery anyway. However, this depends on the exact situation and cannot be explained in detail here.

- All members of the current online redo log group have been lost.
- One or more database files has been damaged and requires recovery, but one of the redo log files (offline redo log files or online redo log group) required for recovery is missing. You do not have a backup copy of the required redo log entries.

When you follow the SAP recommendations, this problem should not occur. The loss of all the members in an online redo log group is highly unlikely, since the mirrored copies should be stored on different disks. The offline redo logs should also be regularly backed up twice to tape using [BRARCHIVE](#).

- A tablespace has been lost and you have no backup copy of the tablespace. You should always have at least three generations of backups for every data file in the database, as well as the corresponding redo log files.

An incomplete recovery causes data to be irretrievably lost, because the database can only be recovered in an older version. You can significantly reduce this risk by using a continual

backup procedure for your database. Always follow the configuration and backup procedures that we recommend.

#### Caution

If you do not use the current control file for a recovery, but instead an older copy, it is essential that you indicate this by adding the following to your command:

```
using backup controlfile
```

The following section only describes the incomplete recovery after the loss of an offline redo log file.

For more information on incomplete recovery, see the Oracle documentation.

## Procedure

### Loss of an Offline Redo Log File

A media error has occurred in the data file area of the database, and one of the offline redo log files is no longer readable. For this reason, the recovery terminates with the last available redo log file in the sequence.

1. If the database system is still active, shut it down with these SQLPLUS commands:

```
connect / as sysdba  
  
shutdown abort
```

ABORT is required in most cases, because the loss of individual data files means that changes in the SGA can no longer be copied to disk.

2. Use the ALERT and trace files to analyze the error.
3. Restore all the available backups of all data files and the offline redo log files of all instances, using [BRRESTORE](#).
4. After an incomplete recovery, the structure of the database may no longer be identical to that fixed in the current control file. Therefore, use a copy of the control file that reflects the structure of the database at the end of the recovery if possible.
5. Mount the database with these SQLPLUS commands:

```
connect / as sysdba  
  
startup mount
```

6. If you were not able to load backed-up files to their original directories or had to change file name, update the control file, by changing path or group names:

```
alter database rename file '<file name>' to '<file name>';
```

See [Updating the Control File](#).

7. You might be able to [automate the recovery](#).
8. Start the recovery with this SQLPLUS command:

```
recover database until cancel;
```

The option `until cancel` means that the online redo logs are reset when opened or not, depending on whether parameter `resetlogs` or `noresetlogs` is used.

If you do *not* use the current control file, enter the command:

```
recover database until cancel using backup controlfile;
```

Depending on the recovery mode, the required offline redo log files are processed automatically (with `autorecovery on`) until the file for the next log sequence number cannot be found, or (with `autorecovery off`) the recovery is stopped with `cancel`.

9. Once the message `recovery complete` or `recovery canceled` is displayed, open the database again with this SQLPLUS command:

```
alter database open resetlogs;
```

Note

Decide which setting of `resetlogs` you want to use:

- o `resetlogs`: initializes the existing online redo log files and resets the current log sequence number to 1
- o `noresetlogs`: does not initialize the online redo log files. Only use this option when you did not use the option `using backup controlfile` and (unusually) all the online redo log files, including the current ones, were processed during the recovery.

The options `resetlogs` and `noresetlogs` are only possible after a `recover database until...` or after a recovery with the option `using backup controlfile`. See also [Finishing an Incomplete Recovery](#).

## Point-in-Time Recovery

You can also select a point-in-time recovery, which you can perform either manually or automatically. In contrast to the incomplete recovery with `until cancel`, this recovery is terminated at a specific time or specified system change number.

Use one of the following SQLPLUS commands:

- `recover database until time 'dd-mm-yyyy:hh:mm:ss';`
- `recover database until change <scm>;`

Depending on the recovery mode (manual or automatic), the required redo log files are processed automatically, or you have to enter their paths and names. When the specified point in time is reached, the recovery is terminated. See also [Finishing an Incomplete Recovery](#).

## Finishing an Incomplete Recovery

This section describes the measures you need to take after an incomplete recovery in the following cases:

- Case 1

Restore of a complete offline backup and subsequently opening the database, without performing a complete recovery of the database

- Case 2

Restore of a complete online or offline backup and subsequent point-in-time recovery of the database (with `ALTER DATABASE OPEN RESETLOGS`)

## Prerequisites

There are the following possible problem situations:

- Situation A

The information about the last backups and volumes used in database tables SDBAH and SDBAD has been lost, because neither item is current in the database. As a result, during the next backup, [BRBACKUP](#) prompts you to mount volumes (based on the automatic volume management) that are logically free, but are physically locked.

- Situation B

The current log sequence number was reset during a incomplete recovery.

- To a smaller value in case 1
- To the value 1 in case 2

[BRARCHIVE](#) does not find the newly written offline redo log files after the restore, because offline redo logs with these log sequence numbers have already been saved. The summary BRARCHIVE log `arch<DBSID>.log` still contains entries for successful backup runs of these offline redo log files and, as a result, BRARCHIVE does not detect the new offline redo log files as files requiring backup.

## Procedure

Depending on the situation and case, proceed as follows:

- Situation A

Using the detail BRBACKUP log, you can find out which volume was the last one used. Based on the information on the volume pool in initialization profile `init<DBSID>.sap` (parameter `backup_volumes`), you can determine which volume to use for the next backup. Explicitly name this volume when you start the next backup:

```
brbackup -v <volume name1>,<volume name2>,...
```

- Situation B

Make sure the old offline redo log files in the backup directory are renamed.

- Case 1

The current log sequence number can be seen in the detail BRBACKUP log of the backup you used to restore the data. Find the line `Current log sequence` (message BR0116I). Then change the log sequence number in the last line of the summary BRARCHIVE log `arch<DBSID>.log`, which starts with `#ARCHIVE`, to the value: `<(determined current log sequence number) - 1>`.

Example

```
#ARCHIVE. 86 /oracle/C11/saparch/C11arch_86 1995-04-18 15.55.55
```

Current log sequence number: 30

Change the entry to:

```
#ARCHIVE. 29 /oracle/C11/saparch/C11arch_86 1995-04-18 15.55.55
```

After the backup, reset the changes in this line. Note that new lines have been added.

- **Case 2**

Change the log sequence number in the last line of the summary BRARCHIVE log arch<DBSID>.log, which starts with #ARCHIVE, to zero (0).

Example

```
#ARCHIVE. 86 /oracle/C11/saparch/C11arch_86 1995-04-18 15.55.55
```

Change the entry to:

```
#ARCHIVE. 0 /oracle/C11/saparch/C11arch_86 1995-04-18 15.55.55
```

After the backup, reset the changes in this line. Note that new lines have been added.

BRARCHIVE automatically recognizes the resetting of log sequence numbers if the database is opened when BRARCHIVE is started. In this case, the actions described in situation B are not necessary.

Note

In general (as after all recovery operations), delete from the disk offline redo log files that were restored from tape to disk after the recovery. If you used it for recovery, BRRECOVER automatically performs this action .

## Automating the Recovery

You can control whether you want to perform a manual (autorecovery off) or an automatic recovery (autorecovery on).

### Procedure

Enter the appropriate SQLPLUS command before you enter the `recover` command.

- `set autorecovery on`

The required offline redo log files are automatically applied without requiring any user entry. The names and paths of the offline redo log files are derived from the `init.ora` parameters `log_archive_dest` and `log_archive_format`, which means that required offline redo log files must first be restored under the appropriate names. For more information, see [Setting Up Archiving](#).

If the files cannot be imported under `log_archive_dest`, you can override the source specified in `archive_log_dest` by entering the following command:

```
set logsource = <log source>
```

This means the files are now expected in the directory identified under `logsource`.

- `set autorecovery off`

The applying of the individual redo log files must be initiated by the user (default value).

In the process, Oracle automatically suggests a file derived from `log_archive_dest` (or `logsource`) and `log_archive_format`.

Choose `RETURN` to accept this value.

You can also explicitly enter the name of the redo log files by entering:

- `cancel` to interrupt or cancel the recovery
- `auto` to continue the recovery in automatic mode from this point
- `from <log source>` to change the log source

## Updating the Control File

In the following cases, you will have to update the control file before you restore data:

- The saved data files are to be restored on another hard disk, in a new directory or under new names.
- The status (`ONLINE` or `OFFLINE`) of one or several data files must be changed for the recovery to be continued.

The control file records the name or the path and the status of each data file in the database. You can update these specifications, which control the recovery process, with Oracle commands.

If a disk error has occurred, for example, it might be necessary to restore the tablespaces in question on another disk. After you have restored the tablespaces, you have to update the path of the affected files in the control file.

## Procedure

### Changing the Path Specifications

SAP recommends using the first of the following methods to rename files.

- Mount the database with these SQLPLUS commands:

```
connect / as sysdba  
  
startup mount
```

To update the path of data files in the control file, enter this SQLPLUS command:

```
alter database rename file '<file name>' to '<file name>;'
```

The target file must exist and the name of the source file must agree with that in the control file.



You can also specify a list of file names, to rename all the files at once. However, note that problems during renaming are easier to diagnose if you rename the files individually.

- You can also change the paths of the data files of a tablespace when the database is running. Set the corresponding tablespace to `OFFLINE` before renaming with these SQLPLUS commands:

```
alter tablespace <tablespace name> offline;
```

```
alter tablespace <tablespace name> rename datafile '<file name1>' [, '<file name2>',...] to '<file name1>' [, '<file name2>',...];
```

You have to enter this command separately for each tablespace in which you have to change the file information.

The target file must exist and the name of the corresponding source files must agree with those in the control file.

## Setting Files to ONLINE

If the data files of a tablespace are `OFFLINE`, when the database crashes or was stopped with `shutdown abort` and a recovery is necessary, you might have to reset the files that belong to this tablespace to `ONLINE` again.

To do this, enter these SQLPLUS commands:

```
connect / as sysdba
```

```
startup mount alter database datafile '<complete file name>' online;
```

## Oracle Recovery Manager

The Oracle Recovery Manager (RMAN) is an Oracle backup program. You can use it as a command line interface (CLI) or as a graphical user interface (GUI) in the Oracle Enterprise Manager (OEM). We support RMAN with the SAP backup tools [BRBACKUP](#) and [BRARCHIVE](#).

RMAN uses the System Backup to Tape (SBT) interface to back up to tape devices. We implement SBT using the SAP backup library. External backup tools can implement this interface as a dynamic link library (DLL).

## Integration

By integrating RMAN into BRBACKUP and BRARCHIVE, you can add security and flexibility to important functions in existing backup strategies and tools:

- The recovery catalog is not used. The backup information is stored in the control file. After the backup, the control file is also backed up. In a restore, the control file is restored first, followed by the data files.
- The integration of RMAN into BRBACKUP and BRARCHIVE also guarantees integration into the SAP [Computing Center Management System \(CCMS\)](#).
- BRBACKUP and BRARCHIVE tape management functions as previously (that is, as when using the SAP backup library).
- You can still use the BACKINT interface with external tools.

- All previous SAP backup strategies are supported while using RMAN. Nevertheless, RMAN is not supported for standby database backups and split-mirror backups.
- The following components are delivered with the standard Oracle8 installation:
  - RMAN with the Oracle SBT interface
  - Backup library and backup tool Networker from Legato

The SAP installation also delivers the SAP backup library with BRBACKUP and BRARCHIVE.

- You can integrate RMAN into your Oracle database with the SAP System using one of the following:
  - SAP implementation
 

You can use the SAP implementation of the Oracle interface SBT, using the SAP backup library. For more information on activating the SAP backup library, see SAP Note 142635.
  - External implementation
 

You can use the Legato implementation or other external backup libraries. These solutions differ in the backup media that they support and the characteristics of the backup programs used.

## Prerequisites

You are subject to the following restrictions when you use RMAN directly, that is, using the CLI or GUI in the OEM:

- RMAN places information about backups in a recovery catalog. For security reasons, this catalog is held in a separate database on a separate host. This means more administrative work.
- Restore or recovery is complicated in a disaster situation in which you lose your production database and recovery catalog. It is often only possible with the help of Oracle Support. Without the recovery catalog data, RMAN cannot recover the database automatically using previous backups.
- The database user making the backups currently needs the SYSDBA authorizations. SYSOPER authorizations are not sufficient.

## Features

- Incremental backups
 

This allows you to change your previous backup strategy and considerably reduce the amount of data to be backed up.
- Consistent backups
 

Logical database block errors are recognized automatically during the backup. This makes sure that each backup is consistent. This function can replace the weekly check with `DBVERIFY`.
- Reduction in amount of data backed up
 

Any database blocks that have never been used are not backed up.
- [Verify of backups](#)

You can verify backups to tape with the RMAN `VALIDATE` command.

- Reduction in redo log information

The command `BEGIN/END BACKUP` is not needed in online backups, since the blocks are checked to see if the data is consistent.

## RMAN Backup Strategies

The decision on whether to use the [Oracle Recovery Manager \(RMAN\)](#) and with which backup library, depends largely on the strategy that you use to back up your data. The best strategy depends on:

- The size of the database
- The amount of data added or changed each day
- The backup media you use
- Your security requirements

## Features

The following table summarizes the main features of backup with RMAN:

|              | Normal Backups                                                                                                                                                                                                                                                            | Full Backup (Level 0)                                                                                                                                                                                                                                       | Incremental Backup (Level 1)                                                                                                                                                                                                                                               |
|--------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| With RMAN    | <ul style="list-style-type: none"> <li>• To tape device with the <a href="#">SAP Backup Library</a></li> <li>• To local disk <a href="#">without backup library</a></li> <li>• To appropriate media with <a href="#">external backup library</a></li> </ul>               | <ul style="list-style-type: none"> <li>• To tape device with the <a href="#">SAP Backup Library</a></li> <li>• To local disk <a href="#">without backup library</a></li> <li>• To appropriate media with <a href="#">external backup library</a></li> </ul> | <ul style="list-style-type: none"> <li>• To tape device or remote disk with the <a href="#">SAP Backup Library</a></li> <li>• To local disk <a href="#">without backup library</a></li> <li>• To appropriate media with <a href="#">external backup library</a></li> </ul> |
| Without RMAN | <ul style="list-style-type: none"> <li>• To local or remote tape devices, local or remote disks with BRBACKUP</li> <li>• Backup with external tool and BACKINT</li> </ul> <p>You can not use RMAN to make a <a href="#">normal backup to a remote disk</a>, even with</p> | <ul style="list-style-type: none"> <li>• To local or remote tape devices, local or remote disks with BRBACKUP followed by cataloging</li> <li>• Backup with external tool and BACKINT followed by cataloging</li> </ul>                                     | Not possible                                                                                                                                                                                                                                                               |

|  | <b>Normal Backups</b> | <b>Full Backup (Level 0)</b> | <b>Incremental Backup (Level 1)</b> |
|--|-----------------------|------------------------------|-------------------------------------|
|  | a backup library.     |                              |                                     |

Note

For more information on the backup types referred to above, see:

- Normal and full backup – see [Complete Backup](#)
- [Incremental backup](#)

You can also [back up the offline redo log files with RMAN](#).

## Incremental Backup Strategies with RMAN

You can perform incremental backups of the database with RMAN and BRBACKUP. RMAN offers the following types of incremental backup:

- Several levels
- Cumulative
- Non-cumulative

The integration of RMAN into BRBACKUP is restricted to cumulative incremental backups at level 1. This means that, in the event of a recovery, only one incremental backup at the most has to be applied. In contrast to a full backup, an incremental backup only backs up the *changes* that have been made since the last full backup. This significantly reduces the amount of backup data. Therefore, we recommend this strategy particularly for large databases.

An incremental backup always requires an earlier full backup, that is, a level-0 backup. When RMAN backs up the database, it backs up all Oracle database blocks that have already been used. A subsequent incremental level-1 database backup backs up all Oracle database blocks that have changed since the last full backup. Changes to the whole database are taken into account.

Incremental backups at level 1 can only be made with RMAN. However, a full backup can also be made with BRBACKUP or with an external backup tool using BACKINT. In this case, BRBACKUP automatically catalogs the backup as a level-0 backup.

Note

You might not see a significant reduction in backup time by making an incremental backup. The reason is that it might take as much time to check whether a block has been used or changed as to simply back it up. This means that you only see a significant time reduction when the relatively low throughput of the tape devices is the reason for a long backup.

Note

Using the incremental backup strategy, we recommend you to set a lock period for the tapes of at least 28 days, so that several generations of full backups are available. The offline redo log files must cover the period up to the oldest full backup - that is, both sets of tapes must have the same lock period - and must also be backed up daily.

After structural changes to the database, you can make incremental backups with RMAN and BRBACKUP. This is not possible with RMAN on its own. For more information, see [RMAN Incremental Backups After Structural Changes](#).

## Activities

This is a possible incremental backup scenario:

- Sundays: full backup (level 0) of the database
- Monday to Saturday: incremental backup (level 1) of the database

The tools used to implement this backup strategy depend on your specific requirements for backup media, volume of data, and so on.

For more information on incremental backups, see the Oracle documentation.

## RMAN Incremental Backups After Structural Changes

By integrating the [Oracle Recovery Manager \(RMAN\)](#) into the SAP backup program BRBACKUP – with the [SAP backup library](#) or an [external backup library](#) – you can make incremental backups, even if the structure of the database has been changed. An example of a structure change is when a data file is added. This is not possible using RMAN on its own.

## Features

Save sets are created each time a System Backup to Tape (SBT) backup library is used. Save sets are units that are created on the backup medium. Each save set contains one or more files. During incremental backups, the changes are normally backed up in a single save set. If new files were added to the database since the last full backup, a second save set is created containing the backups of the new files. For more information, see [RMAN Save-Set Grouping](#). This means that you do not need to make a full backup of the entire database immediately after database extensions.

In subsequent incremental backups during the cycle, any changes that are made to files that existed when the last full backup was made are saved in one save set. Any new files are fully saved (that is, not just the changes) in the other save set. This method also has the advantage that you can be more specific about what you want to restore when [restoring an incremental backup](#).

## RMAN Restore of Incremental Backups

If you perform an Oracle database backup with the [Oracle Recovery Manager \(RMAN\)](#), you must generally also restore it with RMAN. However, this does not apply to backups made to disk. BRRECOVER supports the following:

- Restore followed by a recovery using incremental backups
- Database reset to an [incremental backup](#)

If required, you can continue to recover the database by importing the offline redo log files.

## Prerequisites

In contrast to the BRRESTORE restore process (without RMAN) where the database has to be closed, the database must be mounted for an RMAN restore.

## Activities

The procedure for resetting the entire database to an incremental backup, or to a point in time before the database failure (database reset, point-in-time recovery, disaster recovery) is as follows:

1. You close the database.
2. You restore the control file, and if needed, the online redo log files from the last incremental backup, using BRRESTORE (that is, not using RMAN).
3. You mount the database.
4. You perform a:
  1. Restore of the full backup (level 0) with RMAN
  2. Restore of the last incremental backup with RMAN
5. You restore offline redo log files with BRRESTORE.
6. You apply the offline redo log files to the database to reset the data to the required point in time.

Note

BRRECOVER automatically performs the procedure described above during recovery.

If only some of the files are corrupt, perhaps due to a media error, you only need to restore those files from the full backup and then restore the last incremental backup. RMAN then makes the changes to the files automatically, restoring them to the last incremental backup. The current database is recovered by importing the redo log files.

Caution

If the database has been restored since the last full backup (level 0), then you cannot use it as a reference for the next incremental backup. Be sure to start a full database backup immediately after the database has been successfully recovered.

See also:

[RMAN Incremental Backups After Structural Changes](#)

## RMAN Backup with the SAP Backup Library

This section describes how you can use the [Oracle Recovery Manager \(RMAN\)](#) with the SAP backup library.

### Integration

The SAP backup library is an implementation of the Oracle interface System Backup to Tape (SBT) in the form of a Dynamic Link Library (DLL). The Oracle server process calls this DLL to back up data, usually to tape.

Without this library, the Oracle Server process can only back up to a local disk. For tape management reasons, an SBT backup library is always required for a backup to tape.

## Features

### Backup to Tape Devices

BRBACKUP or BRARCHIVE calls RMAN for a backup to tape devices.

You can use the following profile parameters in the [initialization profile init<DBSID>.sap](#) for backup to:

- Local tape devices

```
backup_dev_type = tape|tape_auto|tape_box
```

```
tape_copy_cmd = rman|rman_dd
```

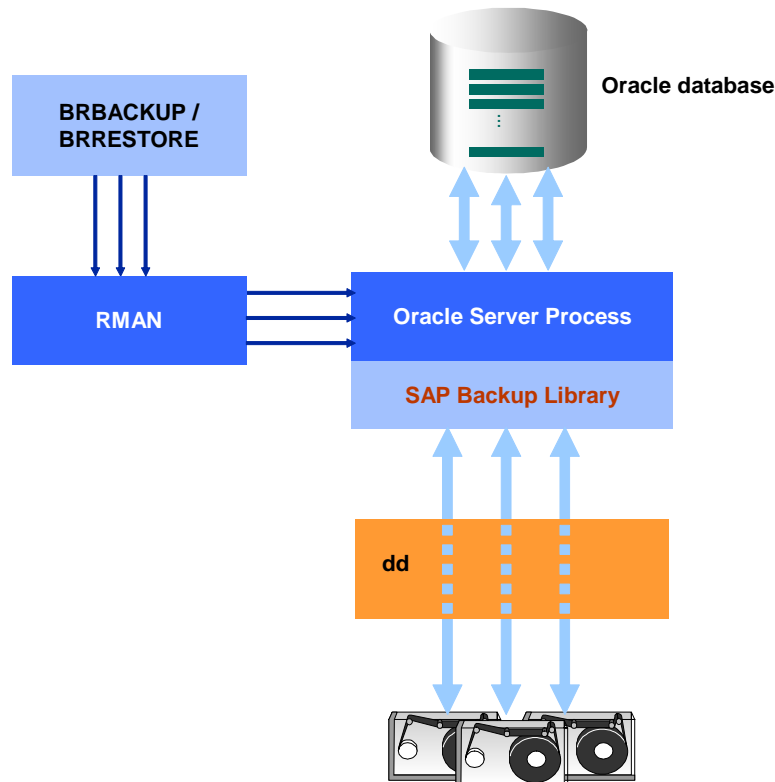
- Remote tape devices

```
backup_dev_type = pipe|pipe_auto|pipe_box
```

```
tape_copy_cmd = rman|rman_dd
```

For more information, see [backup\\_dev\\_type](#) and [tape\\_copy\\_cmd](#).

For a parallel backup to multiple tape devices, the required number of parallel processes are started. RMAN activates the Oracle server process to read the data from the database and transfer it to the SAP backup library. The library then backs up the data to local or remote tape devices, as shown in the following graphic:



Recommendation

For backups to remote tape devices, a remote shell is used to set up a network connection to the remote host. Since errors can occur when you transfer data over a network, we do not recommend you to back up a production system in this way.

You can use RMAN to back up to tape as follows:

| tape_copy_cmd \ backup_mode | cpio    | dd      | rman   rman_dd<br>+ SAP backup library |
|-----------------------------|---------|---------|----------------------------------------|
| Partial                     | *       | *       | X                                      |
| all                         | *       | *       | X                                      |
| full (level 0)              | X + cat | X + cat | X                                      |
| incr (level 1)              |         |         | X                                      |

**Key:**

- \* = not RMAN backup
- X = RMAN backup possible
- X + cat = backup followed by cataloging possible

## Backup to Remote Disks

You can make BRBACKUP backups to remote disks. You can use the following profile parameters in the [initialization profile init<DBSID>.sap](#):

```
backup_mode = incr
```

```
backup_dev_type = stage
```

```
remote_user = "<user_name> [<password>]" (precondition for SAPFTP)
```

```
remote_host = <host_name> (precondition for SAPFTP)
```

For more information, see [backup\\_mode](#), [backup\\_dev\\_type](#), [remote\\_user](#), [remote\\_host](#).

With RMAN you can make only incremental backups (level 1) to remote disks. The SAP-specific file transfer protocol SAPFTP is used alongside the SAP backup library. The full backup (level 0) needed for the incremental backup is made with BRBACKUP and cataloged automatically afterwards.

You can use RMAN to back up to remote disk as follows:



| stage_copy_cmd<br>backup_mode | rcp     | ftp     | RMAN<br>+ SAP backup library |
|-------------------------------|---------|---------|------------------------------|
| Partial                       | *       | *       |                              |
| all                           | *       | *       |                              |
| full (level 0)                | X + cat | X + cat |                              |
| incr (level 1)                |         |         | X + SAPFTP                   |

**Key:**

\* = not RMAN backup

X + cat = backup followed by cataloging possible

X + SAPFTP = backup with SAP File Transfer Protocol possible

Backups to remote disks are particularly useful for:

- [Standby databases](#), when you can use BRARCHIVE to copy offline redo log files to the backup host
- When you use virtual disks to access tape jukeboxes

For more information, see [RMAN-Relevant Profile Parameters](#).

## RMAN Backup with an External Backup Library

You can use external backup software together with the [Oracle Recovery Manager \(RMAN\)](#) to back up the database. To back up data to tape, the Oracle System Backup to Tape (SBT) interface is implemented as a dynamic function library. The type of backup media you can use depends on the external backup tool and the corresponding backup library.

Note

Before SAP Web Application Server (SAP Web AS) 6.20, you had to use BACKINT (that is, `backup_dev_type = rman_util`) when backing up with an external backup library. Starting from SAP Web AS Release 6.20, you do not have to use BACKINT.

RMAN saves the database files, but copies of profiles, logs, and a copy of the control files are made to a local or remote disk. Therefore, there are two new values for `backup_dev_type`:

- `backup_dev_type = rman_disk`
- `backup_dev_type = rman_stage`

When using a remote disk, you can form a common “staging” area for several systems, which simplifies the further backup of the copied profiles and logs.

## Activities

The database is backed up in the following phases:

1. BRBACKUP or BRARCHIVE calls RMAN and starts the Oracle server process. The Oracle server process reads the data to be backed up from the Oracle database, and transfers it to the external backup library. The external backup library acts as a backup client. The backup client passes the data to the backup server. The server backs up the data to the storage medium.
2. Normally, BRBACKUP uses the BACKINT interface to pass the control file, the initialization profile and the log files to the backup client. The backup client passes the data to the backup server. The server backs up the data to the storage medium. This control file contains information on the backup of the files from the first phase.
3. You can also let BRBACKUP copy control file, profile, and log files to local or remote disk instead of calling BACKINT. Therefore, you do not need BACKINT any longer when performing RMAN backups with an external backup library.

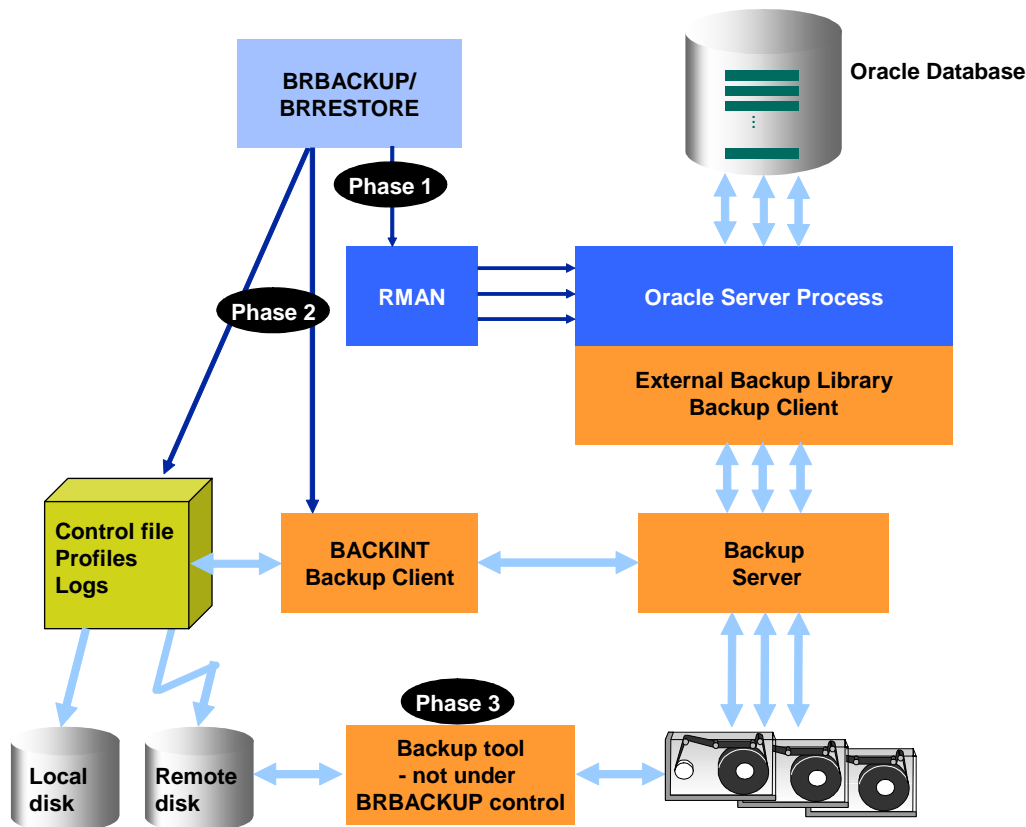
For more information, see the values `rman_disk` or `rman_stage` for the [backup\\_dev\\_type](#) parameter.

This feature is available starting with Release 6.20 of SAP Web AS.

### Caution

You must back up the copies of the control file, profile, and log files to storage medium on your own. You can normally do this directly with the external backup tool.

RMAN Backup with an External Backup Library



You can use the following backup variant for external software, with or without RMAN:

| backup_dev_type<br>backup_mode | util_file<br>without RMAN | rman_util<br>with RMAN and<br>external SBT<br>backup library | rman_disk<br>rman_stage<br>with RMAN and<br>external SBT<br>backup library |
|--------------------------------|---------------------------|--------------------------------------------------------------|----------------------------------------------------------------------------|
| Partial                        | *                         | X + BACKINT                                                  | X                                                                          |
| all                            | *                         | X + BACKINT                                                  | X                                                                          |
| full (level 0)                 | X + cat                   | X + BACKINT                                                  | X                                                                          |
| incr (level 1)                 |                           | X + BACKINT                                                  | X                                                                          |

**Key:**

- \* = non RMAN backup
- X = RMAN backup possible
- X + cat. = backup followed by cataloging possible
- X + BACKINT = backup followed by BACKINT backup of profiles, logs, and control files possible

As before, you can make normal database backups with the BACKINT interface and external backup tools, using the following entry in the [initialization profile init<DBSID>.sap](#):

```
backup_dev_type = util_file|util_file_online
```

For more information, see [backup\\_dev\\_type](#) and [backup\\_mode](#).

Subsequent cataloging means that a full backup can be used as a reference backup in an incremental backup strategy.

Since RMAN can back up only database files and not profiles or logs, the profile, log, and control files are backed up in a second phase with BRBACKUP and the BACKINT interface or to local or remote disk.

For more information, see [RMAN-Relevant Profile Parameters](#).

You can set additional parameters for the backup library using the [rman\\_parms](#) parameter in the initialization profile `init<DBSID>.sap`.

You can send commands to the backup library using the [rman\\_send](#) parameter in the initialization profile `init<DBSID>.sap`.

## RMAN Incremental Backups Without a Backup Library

If you do not have a backup library, you can develop a strategy involving an incremental backup of your Oracle database by performing:

1. A full backup at level 0, which you can perform in one of the following ways:
  - o To local disk with the [Oracle Recovery Manager \(RMAN\)](#), then copy it to tape using BRBACKUP

- To local or remote disk or tape with BRBACKUP or an external tool (that is, without RMAN), then use RMAN to catalog the backup
2. An incremental backup at level 1 to local or remote disk, for which you always have to use RMAN, then copy to tape using BRBACKUP

For more information, see [RMAN Backup Strategies](#). In all the above cases, BRBACKUP always starts RMAN (there is *no* native start from the command line).

## Features

### Full Backup at Level 0 to Local Disk with RMAN

You can make a full backup with RMAN and without a backup library only to *local* disk. You then use BRBACKUP to copy the disk backup to tape. This is a [two-phase backup](#), that is, first a backup to disk, then a copy to tape.

It has the following advantages:

- You can use RMAN functions, such as automatic block checks.
- There is less redo information in an online full backup, since RMAN checks the consistency of the data at block level, so removing the need for the `BEGIN/ END BACKUP` commands.

It has the disadvantage that you need to copy the full backup to tape, which is extra work.

### Full Backup at Level 0 with BRBACKUP or an External Tool, Without RMAN

You can make the full backup at level 0 *without* RMAN. After you have made the full backup - using BRBACKUP (with `cpio` or `dd`) or BACKINT and an external tool - BRBACKUP calls RMAN to catalog the backup as a level-0 backup.

This has the following advantages:

- There is additional security when you restore or recover the backup. You can recover the database without RMAN because no save sets are formed during the full backup. In this case, the import of the redo log files created since the last backup replaces the restore of the last incremental backup.
- Under certain conditions the backup can be quicker, since RMAN is not involved in the full backup at level 0. The data is backed up directly using `cpio`, `dd`, or `BACKINT`.

It has the following disadvantages:

- You cannot use RMAN features such as the database block check.
- The commands `BEGIN BACKUP` and `END BACKUP` are used in an online full backup, leading to extra redo log files. This data increases the load on the system and makes an “archiver stuck” error more likely. If you have an archiver stuck error, see *SAP Note* [316642](#).
- If you make a recovery without RMAN you might have to apply a lot of redo log files since the full backup is usually only made once a week.

### Incremental Backup at Level 1 to Disk with RMAN

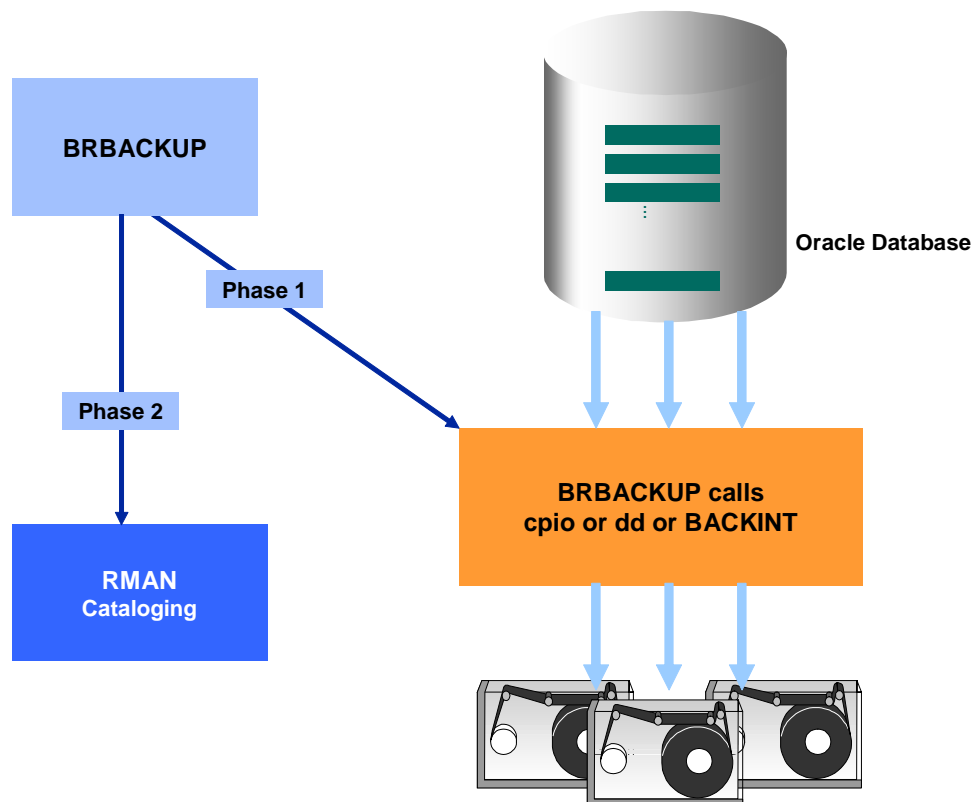
You can make incremental backups at level 1 with RMAN and without a backup library, but only to local disk. There is *no* one-to-one copy of the database files. Instead, save sets are

created, which means you can only recover the save sets of an incremental backup with RMAN. For more information, see [RMAN Save-Set Grouping](#). In a second phase, you copy the incremental backup to tape with BRBACKUP. This is a [two-phase backup](#), that is, first a disk backup, then a copy to tape.

## Example

Here is an example of how you can perform the two stages in an incremental backup without backup library: a full backup at level 0 and an incremental backup at level 1.

### Stage 1: Full Database Backup at Level 0 Without Backup Library



As shown in the graphic, the full backup consists of the following phases:

1. Phase 1

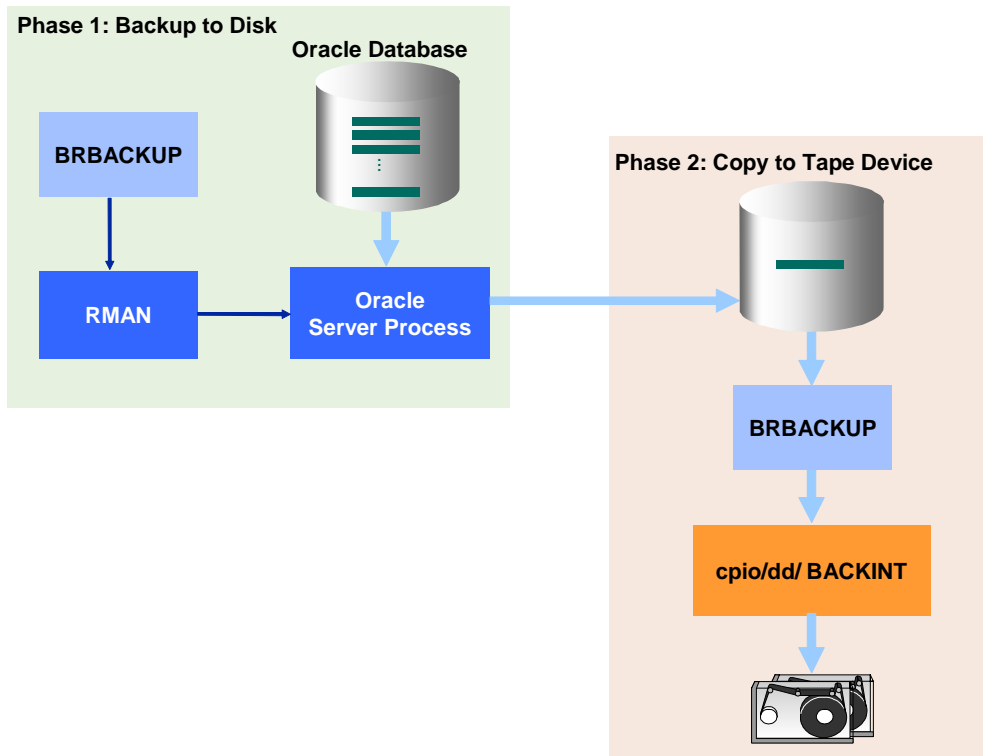
BRBACKUP calls `cpio`, `dd`, or `BACKINT` to back up the database. `cpio` or `dd` read the data from the disk and write it to tape. If you use the `BACKINT` interface, the database file names are transferred to external backup software that then backs up the files to the backup medium.

2. Phase 2

BRBACKUP calls RMAN to automatically catalog the backup as a full backup (level 0). This means it can be used as a reference for later incremental backups.

### Stage 2: Incremental Backup at Level 1 Without Backup Library

Since you cannot make an incremental backup directly to tape without RMAN, this stage is a [two-phase backup](#) (that is, a backup phase followed by a copy phase):



As shown in the graphic, the incremental backup consists of the following phases:

1. Phase 1

BRBACKUP calls RMAN, which internally activates an Oracle server process to read the data from the Oracle database and save it to disk in a save set.

2. Phase 2

BRBACKUP calls `cpio`, `dd`, or `BACKINT` to copy the incremental save set from disk to the backup medium.

For more information, see [RMAN Backup Strategies](#) and [RMAN-Relevant Profile Parameters](#).

## RMAN Backup of the Offline Redo Log File

You can use `BRARCHIVE` to back up the offline redo log files using RMAN, either with the SAP backup library or with an external backup library. This is available starting with SAP Web Application Server Release 6.10.

The advantage is that the data in the offline redo log files during the RMAN backup is checked for internal consistency. You can think of this as a replacement for the missing verification functionality for offline redo log files when using `DBVERIFY`.

## RMAN Tape Layout

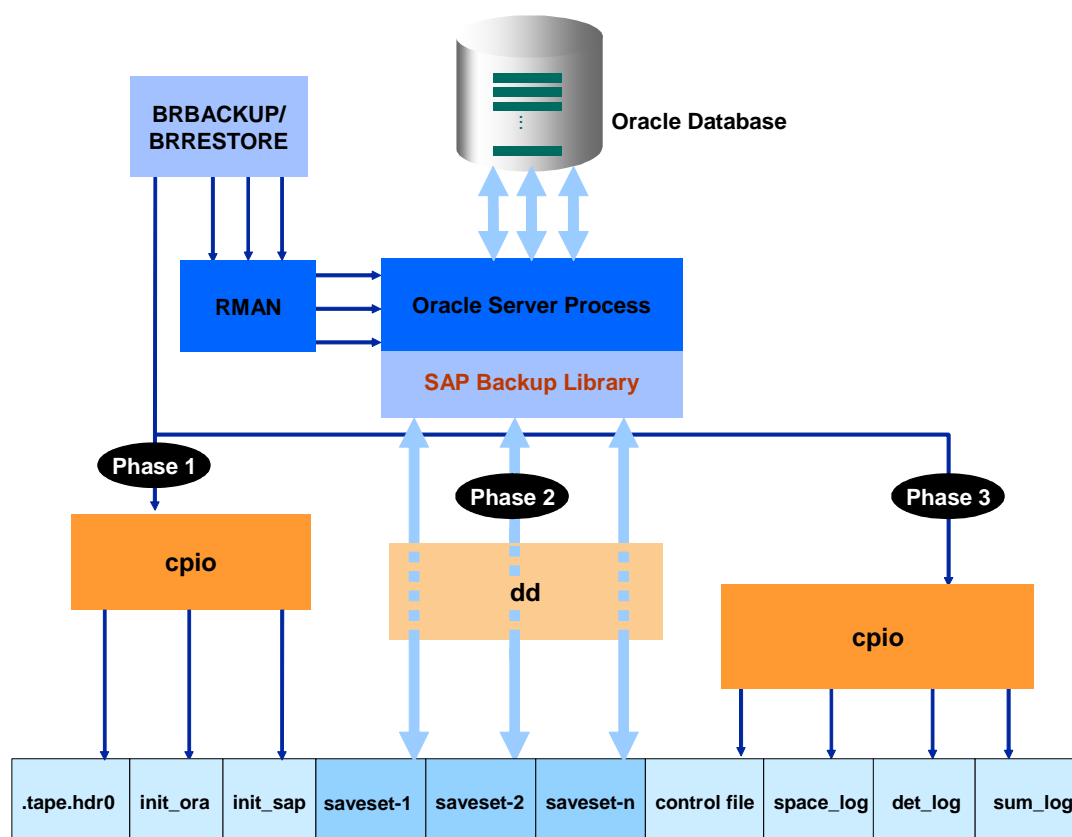
The tape layout with RMAN is basically the same as the backup methods without RMAN. This means that you do not have to reinitialize the tapes if you are using the SAP implementation of the System Backup to Tape (SBT) interface – that is, the SAP backup library.

BRBACKUP uses `cpio` to back up the tape administration information (that is, the tape label) and the initialization files `init<DBSID>.ora` and `init<DBSID>.sap` to tape, as follows:

1. The SAP backup library backs up the save sets to tape directly or using `dd`.
2. BRBACKUP uses `cpio` to back up the control file and the log files to tape.

The control file is backup with `dd` if `tape_copy_cmd = rman_dd` is set.

Tape Layout



## RMAN Backup Verify

The verification of backed-up data files or offline redo log files made with the [BRBACKUP](#) and [BRRESTORE](#) functions for verifying backups are also supported for backups with the [Oracle Recovery Manager \(RMAN\)](#).

### Features

- BRBACKUP can verify online or offline RMAN backups immediately.
- A delayed verify with BRRESTORE can only be performed on the database host, when the database is mounted or open.
- A verify with DBVERIFY is not supported for RMAN backups, since the Oracle block consistency is checked when the backups are made.

## Activities

RMAN verifies save sets with the `VALIDATE` command, as follows:

1. RMAN reads the save sets from the backup medium to check their readability.
2. RMAN checks the internal consistency of the data in the save sets using check sums. No binary comparison is made with the originals.

### Recommendation

We still recommend that you verify backups as often as before, that is, at least once in each backup cycle and every week if possible.

For more information, see [Backup Verify](#) and [RMAN-Relevant Profile Parameters](#).

## RMAN Save-Set Grouping

When you back up the Oracle database with the [Oracle Recovery Manager](#), the SAP backup library helps to optimize the utilization of fast tape drives by combining multiple data files in save sets. Multiple file access – also known as `file multiplexing` – maximizes the flow of data to keep tape drives in streaming mode.

### Prerequisites

You can define the number of files contained in each save set with the [saveset\\_members](#) parameter from the [initialization Profile init<DBSID>.sap](#) or the [BRBACKUP](#) command option [-sl-savesets](#).

For a preparation run, set the [backup\\_dev\\_type](#) parameter to `rman_prep`.

For an incremental backup with the SAP backup library, the [saveset\\_members](#) parameter is internally set to `all` so that only one "incremental save set" is created including all changed blocks.

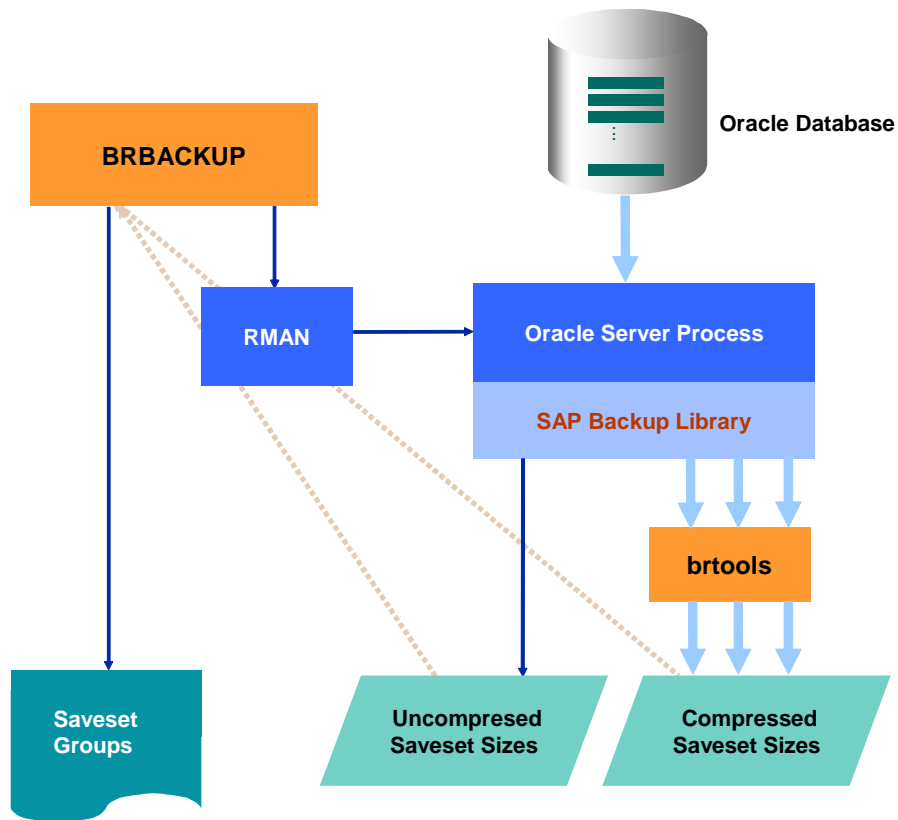
### Features

The SAP backup library optimizes the distribution of the data files across the save sets in a preparation run as follows:

1. It estimates the expected compression of the save sets by using BRTOOLS to determine the compressed save set sizes.
2. BRBACKUP forms save sets and stores the information on the composition of the save sets and the compression rates for future backups in the database.

The following graphic shows this:





A save set can contain from one to four individual data files, all files of a tablespace, or the complete data backup. The size of the save sets for the backup must be selected according to the tape device. A fast data flow with a minimum save set size is the optimum. Large save sets are not recommended, since in a restore the complete save set has to be read, even if only one data file is required.

## Activities

SAP recommends that you perform the preparation run once a month - using the BRBACKUP command [-dj-device](#) with the `rman_prep` parameter - and each time that major changes have been made to the database, for example after a reorganization, an SAP release change, database migration, or mass data import. After a tablespace extension, it is enough to execute the run for the newly created files only.

For more information, see [RMAN-Relevant Profile Parameters](#).

## RMAN-Relevant Profile Parameters

### Save-Set Grouping and Preparation Run

To define the size of the save sets by the number of files they contain, make the following entry in the [initialization profile](#) `init<DBSID>.sap`:

```
saveset_members = 1|2|3|4|tsp|all
```

Default: 1

The corresponding command option is: `brbackup -s|-savesets`

You can start the preparation run for determining the optimum distribution of files across the save sets as follows:

- `brbackup -d rman_prep`
- BRGUI or BRTOOLS menu: *Backup and database copy Additional functions Preparation of RMAN backups*

For more information, see [RMAN Save-Set Grouping](#).

## RMAN Backup with the SAP Backup Library

For backups with the SAP backup library you need to make the following special entries in the initialization file `init<DBSID>.sap` or set the corresponding command options for backup to:

- Local tape devices
  - `backup_dev_type = tape|tape_auto|tape_box`
  - `tape_copy_cmd = rman|rman_dd`
- Remote tape devices
  - `backup_dev_type = pipe|pipe_auto|pipe_box`
  - `tape_copy_cmd = rman|rman_dd`
- Remote disk (incremental)
  - `backup_mode = incr`
  - `backup_dev_type = stage`
  - `remote_user = "<user name> [<password>]"` (<password> is required for SAPFTP)
  - `remote_host = <host name>`

For more information, see [RMAN Backup with the SAP Backup Library](#).

## RMAN Backup with an External Backup Library

For backups with external backup libraries, you need to make the following entries in the initialization profile `init<DBSID>.sap` or the corresponding command options:

```
backup_dev_type = rman_util|rman_disk|rman_stage
```

You can set other parameters as well, which BRBACKUP sends directly to the Oracle RMAN (without the prefix `rman_`). These parameters affect how the save sets are formed and the parallel backup to different media.

- `rman_channels`  
Number of channels allocated to the parallel processes
- `rman_filesperset`  
Number of files in each save set  
Default is 0 (that is, one file in each save set for normal backups, or all files in one save set for incremental backups)
- `rman_maxpiecesize` (formerly `rman_kbytes`)

- `rman_rate`(formerly `rman_readrate`)
- `rman_maxopenfiles`
- `rman_maxsetsize`(formerly `rman_setsize`)
- `rman_diskratio` (no longer available in Oracle 10g)
- `rman_pool`
- `rman_copies` (formerly `rman_duplex`)
- `rman_proxy`
- `rman_parms`
- `rman_send`
- `rman_compress`
- `rman_maxcorrupt`

For more information on these parameters, see [Initialization Profile `init<DBSID>.sap`](#) and the Oracle backup and recovery documentation.

For more information, see [RMAN Backup with an External Backup Library](#).

## RMAN Backups Without Backup Library

For normal backups to disk with RMAN, you need to make the following entries in the initialization file `init<DBSID>.sap` or the corresponding command options:

- `backup_dev_type = disk`
- `disk_copy_cmd = rman|rman_set`

The individual steps of the incremental backup strategy without backup library have the following command options:

1. Full backup (level 0) without RMAN:
  - o `brbackup -m full -d tape|pipe|util_file`
  - o `tape_copy_cmd = cpio|dd`
2. Incremental backup (level 1) to disk with RMAN:
  - o `brbackup -m incr -d disk`
3. Backup of the incremental backup to tape without RMAN (two-phase backup):
  - o `brbackup -b last -m incr -d tape|pipe|util_file`

For more information, see [RMAN Incremental Backups Without a Backup Library](#).

## Backup of Offline Redo Log Files with RMAN

If you are using the SAP backup library, set the following entries in the initialization file `init<DBSID>.sap` or use the corresponding command options:

- `backup_dev_type = tape|pipe`

- `tape_copy_cmd = rman|rman_dd`

If you are using an external backup library, use the following setting:

```
backup_dev_type = rman_util
```

For backups to disk with RMAN but without a backup library, use the following settings:

- `backup_dev_type = disk`
- `disk_copy_cmd = rman|rman_set`

Then start BRARCHIVE, for example as follows:

```
brarchive -sd -c
```

For more information, see [RMAN Backup of the Offline Redo Log File](#).

## Restoring Incremental Backups

To make the database consistent again, for example after a media error, you can use the following BRRESTORE command (RMAN redoes the changes in the files that have been affected):

1. If necessary, restore of the control file and the online redo log files from the last incremental backup:

```
brrestore -b last -m 0[,00]
```

2. Restore affected files of a full backup (level 0):

```
brrestore -b <brb_log_name> -m all|<object list>|..
```

3. Restore of the last incremental backup (level 1):

```
brrestore -b last -m incr
```

4. Applying the offline redo log files with Oracle SQLPLUS.

For more information, see [RMAN Restore of Incremental Backups](#).

### Recommendation

We recommend you to use [BRRECOVER](#) instead of this manual procedure.

## Restoring Incremental Backups with Structural Changes

In an incremental backup with structural changes, the new files are backed up in full to a second save set in subsequent incremental backups. The following save sets are created if the SAP backup library is used:

- `<coded timestamp>.INCR` (changes to the "old" files)
- `<coded timestamp>.FULL` (newly added files)

The backup of the new files to a separate save set allows a precise specification of which files are to be restored.

Restore of changes to all files that were in the database at the time of the last full backup at level 0 (first save set of the last incremental backup):

```
brrestore -b last -m incr_only
```

Restore of the files that have been added since the last full backup at level 0 (second save set of the last incremental backup):

```
brrestore -b last -m incr_full
```

Restore of the whole incremental backup (both save sets, if they exist):

```
brrestore -b last -m incr
```

For more information, see [RMAN Incremental Backups After Structural Changes](#).

*See also:*

Oracle documentation

## The SAP Tools with Windows

This section describes special features of BRBACKUP, BRARCHIVE, and BRRESTORE with Microsoft Windows.

For more information, see:

- [SAP Conventions \(Windows\)](#)
- [Backup Strategy \(Windows\)](#)

Note

For information about Microsoft Windows operating system security, see:

[www.microsoft.com/security](http://www.microsoft.com/security)

## More Information

[Oracle Under Windows](#) for more information on operating system security for the Oracle database with your SAP system

[SAP System Security Under Windows](#) for more information on Windows operating system security for the SAP system

## SAP Conventions (Windows)

Due to the directory structure of Windows, the main differences to the documentation for the SAP tools are in the UNIX-specific file structures. We do not discuss the meanings and contents of file names, environment variables, profiles, and so on, because we assume knowledge of these.

- [Environment Variables \(Windows\)](#)
- [Directory Structure \(Windows\)](#)
- [Naming Conventions for Files \(Windows\)](#)

- [Executables](#)
- [Starting the SAP Utility Programs](#)
- [Database Analysis](#)

## Naming Conventions for Files (Windows)

The regular SAP naming conventions apply so that the data files are in directory `<drive>\oracle\<SID>\sapdata<n>\<tablespace name>_<file number>`. Therefore, the first file of tablespace PSAPPOOLD might be called:  
`F:\oracle\C11\sapdata2\pool_1\pool_1.data1.`

You can store the files of one tablespace on different disks. BR\*Tools require that only the name of the drive be changed, and the remainder of the path (`\oracle\C11` in the example above) remains the same, to avoid confusion with other databases.

BR\*Tools also require that the logical directory `sapdata<n>` is specified. A further subdirectory `<tablespace name>_<file number>` is created automatically.

For reasons of clarity, SAP recommends that you *not* spread a data pool (indicated by the subdirectory `sapdata<n>`) among different disks. This means that, for example, there should not be two `sapdata4` directories on different disk drives

For security reasons, BR\*Tools never create a subdirectory `sapdata<n>` itself. They only use an existing one created in the File Manager or using Windows commands.

## Executables

The programs that BR\*Tools call must be accessible from everywhere. Therefore, the paths of the following executables must be set in the `PATH` environment variable.

### SAP Programs

`BRBACKUP.EXE`, `BRARCHIVE.EXE`, `BRRESTORE.EXE`, `BRRECOVER.EXE`,  
`BRCONNECT.EXE`, `BRSPACE.EXE`, `BRTOOLS.EXE`

### Interface Program for External Backup Utility

`BACKINT.EXE`

### Operating System Programs

`CPIO.EXE` (MKS Tools), `DD.EXE` (MKS-Tools), `MT.EXE` (MKS Tools), `MKSZIP.EXE` (MKS Tools, compression), `UNCOMPRESS.EXE` (MKS Tools, decompression)

So far, `MKSZIP.EXE` and `UNCOMPRESS.EXE` are only used by `BRBACKUP`, `BRARCHIVE` and `BRRESTORE`.

The `mkzip` command is required if you want `BRBACKUP` or `BRARCHIVE` to save files with software compression (option `-k yes`).

This program is not used anymore to estimate the compression rate of tape devices with hardware compression (option `-k only`). Instead, `BRTOOLS` is called, which uses the SAP compression library.

## Oracle Programs

SQLPLUS, EXP, IMP, RMAN, DBV

See also:

[Starting the SAP Utility Programs](#)

## Starting the SAP Utility Programs

### Procedure

To avoid authorization problems when starting the BR\*Tools, bear in mind the following:

1. On the database server create local groups `ORA_<DBSID>_DBA` and `ORA_<DBSID>_OPER` (or `ORA_DB` and `ORA_OPER`). Include the SAP users `<SID>ADM` and `SAPSERVICE<SID>` in this group, if this has not been done during the installation.

When you log on to the database with `connect / as sysdba` using `SQLPLUS` you no longer have to enter a password. The user is authorized to start and stop the database.

You can now call `BRBACKUP` and `BRARCHIVE` as usual with the following command:

```
brbackup -u system/<password>
```

2. `BRBACKUP`

If you call `BRBACKUP` at operating system level, the following alternatives to the procedure described in 1. are available:

- Start `BRBACKUP` not with the standard ORACLE user `system`, but with the user `OP$` user, as in the following example:

```
brbackup -u /
```

The `OP$` user must have granted the `SAPDBA` role for backups to be successful

- Activate the full Oracle authorization check by performing the following steps:

- In the profile `init<SID>.ora`, enter the parameter `remote_login_passwordfile = exclusive`.
- Create an Oracle password file (if it does not already exist):  

```
orapwd file=%ORACLE_HOME%\DATABASE\PWD<DBSID>.ORA  
password=<internal_password> entries=10
```
- Restart the database.
- Call `SQLPLUS` as the user `SYS`. Give the following authorizations to the user `system`:

```
Connect / as sysdba  
  
grant sysdba to system;  
  
grant sysoper to system;
```

- Give a new password to the user `system` (optional):

```
alter user system identified by <password>;
```

Now you can call BRBACKUP as usual:

```
brbackup -u system/<password>
```

## Database Analysis

You can use the Scheduler from Microsoft to schedule the creation of a check or analysis log, or enter the following command in the command line (example for using `brconnect -f check`):

```
at \\<host name> <time> check.bat
```

Example

```
at \\ps0001 00.00 check.bat
```

The database must be active at the specified time. If the database is usually shut down, you can also schedule database startups and shutdowns, using the following commands:

```
brspace -u / -c force -f dbshut
```

```
brspace -u / -c force -f dbstartup
```

The `check.bat` file should have the following contents:

```
set PATH=%PATH%;<x:>\usr\sap\<SID>\SYS\exe\run set ORACLE_HOME=<path>
set ORACLE_SID=<DBSID> set SAPDATA_HOME=<path> brconnect -u / -c -f
check
```

Make sure the path for BR\*Tools is set correctly.

Example

For file `check.bat`

```
set PATH=%PATH%;D:\usr\sap\C11\SYS\exe\run set ORACLE_HOME=D:\orant
set ORACLE_SID=C11 set SAPDATA_HOME=E:\oracle\C11 brconnect -u / -c -
f check
```

To check any of the background statements that have been set, enter `at` in the command line.

## Backup Strategy (Windows)

The following backup programs are available:

- [NTBackup](#) (Microsoft)
- [BRBACKUP](#) (SAP)



- [BARCHIVE](#) (SAP)

This section describes the differences, advantages, and disadvantages of the individual programs. The SAP program BRRESTORE is available for restoring files which were backed up with BRBACKUP or BRARCHIVE.

## NTBackup

You can use NTBackup to perform offline backups of the SAP data files, the system or rollback data files, the control files, and the online and offline redo log files. Online backup is not possible.

Files saved with NTBackup can be restored using a corresponding graphic menu.

### Constraints

SAP only recommends this procedure for small databases or test databases for which an offline backup of the complete database can be performed regularly (for example, every night).

Files saved with NTBackup can be restored via a corresponding graphic menu.

If the backup was created using NTBackup, you *cannot* perform a [restore and recovery with BRRECOVER](#).

## BRBACKUP/BRARCHIVE

In contrast to NTBackup (file backup) [BRBACKUP](#) is a backup program (online and offline) that was specially designed for the Oracle database. SAP recommends BRBACKUP for large databases, because it also enables you to back up individual tablespaces online when the database is running, using ARCHIVELOG mode. For more information, see [Setting Up Archiving](#).

Offline redo log files can be backed up to tape using [BRARCHIVE](#). You can restore a complete backup and offline redo log files using [BRRESTORE](#).

### Prerequisites

BRBACKUP enables parallel backups to tape and disks. Some parameters in the initialization profile `init<DBSID>.sap` are specific to the operating system, as in the following example:

```
rewind_offline = "mt -f $ offline"
```

For more information about authorization checks, see [Executables](#).

### Activities

Make sure that the environment variables are set correctly. Create a file in which the correct environment is defined before you start BRBACKUP or BRARCHIVE. This file is scheduled in the `at job`.

Example

```
set PATH=%PATH%;d:\usr\sap\C11\SYS\exe\run set ORACLE_SID=C11 set
ORACLE_HOME=d:\orant set SAPDATA_HOME=d:\oracle\C11 brbackup -u / -c
force <other options>
```

See also:

[Database Analysis](#)

## Other Backup Programs

If you already use external backup programs for large systems, you can use an interface (BACKINT) to link them with BRBACKUP, BRARCHIVE, and BRRESTORE.

If you use a specific backup server with an operating system other than Windows, you can also use other external backup programs.

### Prerequisite

The BACKINT interface must support Windows as client.

### Activities

Contact the manufacturer of the external backup program to find out about the capabilities of the BACKINT interface.

For more information, see [External Backup Programs](#).

## Structure-Retaining Database Copy or Restore on Windows

If the database is on several drives on Windows, problems occur as follows:

- Database copy with retained structure, that is, `backup_dev_type = disk_copy | disk_standby | stage_copy | stage_standby`

BRBACKUP attempts to copy all databases onto the drive which is defined in the `init<DBSID>.sap parameter new_db_home`.

- Restore on another computer with changed database name:

BRRESTORE attempts to reload all databases to the drive which is defined in the environment variable `SAPDATA_HOME`.

In both cases the distribution of the database files to several drives is not maintained.

For more information about solving this problem, see *SAP Note* [122363](#).

## Offline Backup with Oracle Fail Safe for Cluster Systems

BRBACKUP supports offline backups with Oracle Fail Safe for cluster systems.

For more information about using Oracle Fail Safe to stop the database, see *SAP Note* [378648](#).

# Oracle Real Application Cluster

You can use the Oracle Real Application Cluster (RAC) option with your Oracle database to improve the availability of your SAP system. The RAC configuration replicates the database server.

If one of the RAC instances fails, a surviving instance performs the recovery for the failed instance. Transactions open at the time of the instance failure are rolled back and committed transactions are applied. The surviving database instance continues to provide the database service. Application servers connected to the failed database instance can reconnect to a surviving database instance and continue.

## Integration

The SAP tools for database administration - that is, BR\*Tools - support RAC. For more information, see:

- [RAC with BR\\*Tools](#)
- [Oracle Databases on Raw Devices](#)

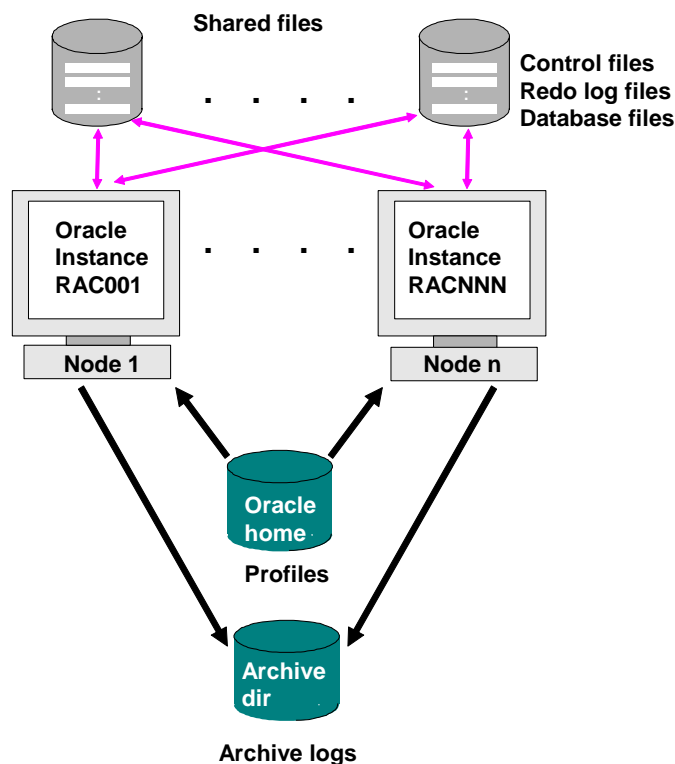
## Prerequisites

If you want to use RAC, your Oracle database *must* use a cluster file system or [raw devices](#).

See also *Activities* below for further prerequisites that you must meet to run RAC successfully.

## Features

The following diagram illustrates RAC:



For more information about RAC, see the Oracle documentation, but be aware of the following general features of RAC:

- All control files, online redo log files, and database files are stored on the cluster file system or shared raw devices, which can be accessed by all computers in the cluster.
- There are multiple Oracle instances, normally one per node, all of which can access the database.
- Each Oracle instance has a unique number and name, the Oracle system ID. When an Oracle instance connects to the database, the connection is specified by the environment variable `ORACLE_SID` (abbreviated to `DBSID`) or a connect string.
- Each Oracle instance writes its own groups of online redo log files, which normally include mirrored online redo log files, depending on your configuration. All the groups belonging to a single instance are called an online redo thread, identified by a unique number.
- Each Oracle instance writes its own offline redo log files, identified by the thread number and an ongoing sequence number. The offline redo log files are stored as normal files in a shared file system. You can access these files globally, that is, from each node running an instance.
- All Oracle instances have a common `spfile` (recommended), which specifies for each instance, among other things:
  - Instance name and number
  - Thread number, using `thread`
  - Archive directory, using `log_archive_dest`
  - Name format of the offline redo log files, using `log_archive_format`

The values for these parameters must not include `?` (`<ORACLE_HOME>`) and `@` (`<ORACLE_SID>`). To avoid problems, this is only valid as follows:

- `log_archive_dest = /oracle/<DBNAME>/oraarch/<DBNAME>arch`

In the standard installation, all instances ought to have the same shared `<ORACLE_HOME>`. For example:

```
log_archive_dest = /oracle/RAC/oraarch/RACarch
```

- `log_archive_format = %t_%s.dbf`

## Activities

You perform the following activities with RAC:

- One of the Oracle instances (normally the instance with thread number 1 in SAP installations) is the dedicated database instance (DDB instance). You use this instance to perform all database administration tasks, such as backup, tablespace extension, reorganization, and so on.
- Before working with your Oracle database, make sure you understand the details of your particular installation. The standard allocation of instance (`ORACLE_SID` or `DBSID`) and threads looks as follows in an SAP system called RAC:

| Instance<br>(DBSID) | Thread |
|---------------------|--------|
|                     |        |

| Instance (DBSID) | Thread           |
|------------------|------------------|
| RAC001           | 1 (DDB instance) |
| RAC002           | 2                |
| RAC003           | 3                |
| ...              | ...              |

- You must be able to administer all Oracle instances from the DDB instance, that is, you must be able to start and stop all instances from the DDB instance. You must be able to do this from a local SQLPLUS session. To enable this, you have to meet the following prerequisites:

- You create an Oracle password file by entering the following:

```
orapwd file=<ORACLE_HOME>/dbs/orapw
password=<sys_password> entries=10
```

- You make sure that the Oracle profile has the following entry:

```
remote_login_passwordfile = exclusive
```

If the parameters are only effective after a system restart, then you must restart all instances.

You make sure that, on the DDB instance, the `system` user has `SYSDBA` and `SYSOPER` authorization. To do this, start SQLPLUS as the `SYSDBA` user and execute the following Oracle command:

```
SQL> grant sysdba to system;
SQL> grant sysoper to system;
```

- If required, change the password for the `SYSTEM` user, as follows:

```
SQL> alter user system identified by <password>;
```

#### Note

For more general information about SAP support for Oracle RAC see SAP Notes [527843](#), [581320](#), [621293](#), and [830982](#).

## RAC with BR\*Tools

You can manage up to 16 parallel instances of the [Oracle Real Application Cluster \(RAC\)](#) with [BR\\*Tools](#). You normally start these tools only on the dedicated database (DDB) instance, as defined in [SAP Note 621293](#).

### Prerequisites

- Check the requirements in [Oracle Real Application Cluster \(RAC\)](#).

- The necessary BR\*Tools executables must exist on the DDB instance host in directory `/usr/sap/<SAPSID>/SYS/exe/run`.
- A common `spfile` must exist on the DDB instance host in directory `<ORACLE_HOME>/dbs` (UNIX) or `<ORACLE_HOME>\database` (Windows).  
`spfile.ora` contains all instance-independent and all instance-dependent parameters for all Oracle instances. For more information, see [SAP Note 830982](#).
- The backup devices (tape units or hard disks) are normally connected locally to the DDB instance host.
- The `init<DBSID>.sap` profile parameter [parallel\\_instances](#) contains all Oracle instances, including the local instance.
- The log directories for BR\*Tools `saparch`, `sapbackup`, `sapcheck`, and `sapreorg` are all on shared file systems.
- The Oracle archive directory, `oraarch`, is on a shared file system
- The following directories are on a shared file system:
  - Oracle profile directory:  
UNIX: `<ORACLE_HOME>/dbs`  
Windows: `<ORACLE_HOME>\database`
  - Trace directory: `saptrace`
  - Audit directory: `~/rdbms/audit`
  - The entire `<ORACLE_HOME>` directory, if possible
- The Oracle SQL\*Net configuration corresponds to the recommendations in [SAP Note 830982](#).
- UNIX only: BR\*Tools requires as standard an Oracle password file for the remote connection. For more information, see [SAP Note 131610](#).

## Activities

If you have set up the user `SYSTEM` correctly, it now has the authorization to start and stop the database instances remotely. Therefore, you can call `BRBACKUP` and `BRARCHIVE` as usual, for example:

```
brbackup -u system/<password>...
```

```
brarchive -u system/<password>...
```

## RAC with BRSPACE

This section describes how you can use [BRSPACE](#) with the [Oracle Real Application Cluster \(RAC\)](#).

## Prerequisites

- You should normally only start BRSPACE from the dedicated database (DDB) instance. If required – for example when looking for free space in local archiving directories – you can also start BRSPACE from another instance.
- Read the information about setting up the `SYSTEM` user. User `system` has the authorization to start and stop the database instances remotely. Therefore, you can call BRSPACE as usual with the `SYSTEM` user.

## Activities

The following functions in BRSPACE are specially designed to work with RAC databases:

- [Starting Up the Database with BR\\*Tools](#)
- [Shutting Down the Database with BR\\*Tools](#)
- [Altering the Database Instance with BR\\*Tools](#)
- [Showing Instance Status with BR\\*Tools](#)
- [Altering Database Parameters with BR\\*Tools](#)

## More Information

[RAC with the `init<DBSID>.sap` Profile](#)

## RAC with BRBACKUP

This section describes how to set up the [Oracle Real Application Cluster \(RAC\)](#) to run with [BRBACKUP](#).

### Prerequisites

You must set the [parallel\\_instances](#) and [db\\_services](#) parameter in the `init<DBSID>.sap` profile used by BRBACKUP so that BRBACKUP can access all the Oracle instances.

On *UNIX* systems, you can now perform BRBACKUP and BRARCHIVE backups of RAC databases with the `OPS$` user *without* entering the database user and password in the `-u` option. For more information, see *SAP Note 914174*.

On *Windows* systems, this is possible in the standard Oracle configuration.

### Features

BRBACKUP functionality is available for RAC databases, including the support of software and hardware compression and for external backup tools through the BACKINT interface. However, there is no continuation mechanism with the `dd` command. For more information, see [Raw Devices with BRBACKUP and BRRESTORE](#).

### Activities

Database backup with BRBACKUP for RAC runs as follows:

1. If the local Oracle instance is stopped, BRBACKUP starts it with the appropriate SQLPLUS commands.

2. BRBACKUP logs on to the local dedicated database (DDB) instance and performs the common actions in the database (determination of the database structure, the volumes used, and so on).
3. For an offline backup, database instances are stopped. For online backup, the instances continue running during the backup.
4. The required backup is started.
5. For an offline backup, database instances are restarted in parallel at the end of the backup.
6. BRBACKUP updates the SDBAH and SDBAD tables.
7. The status of all instances before the start of the backup with BRBACKUP is recovered. That is, the instances that were stopped before the backup are now stopped again.

During the entire backup, BRCONNECT monitors *all* database instances and checks them constantly to see that the required status of each instance remains unchanged. If the status of an instance changes during the backup, BRCONNECT interrupts the entire backup.

In the BRBACKUP log, both before and after the actual backup, a log is made of the parameters relevant to archiving for *all* database instances in question. The log consists of output of the `ARCHIVE LOG LIST` command.

If you run an offline backup of the entire database, a copy of *all* online redo log groups for the Oracle instances is saved in each case.

## RAC with BRARCHIVE

This section describes how to set up the [Oracle Real Application Cluster \(RAC\)](#) to run with [BRARCHIVE](#).

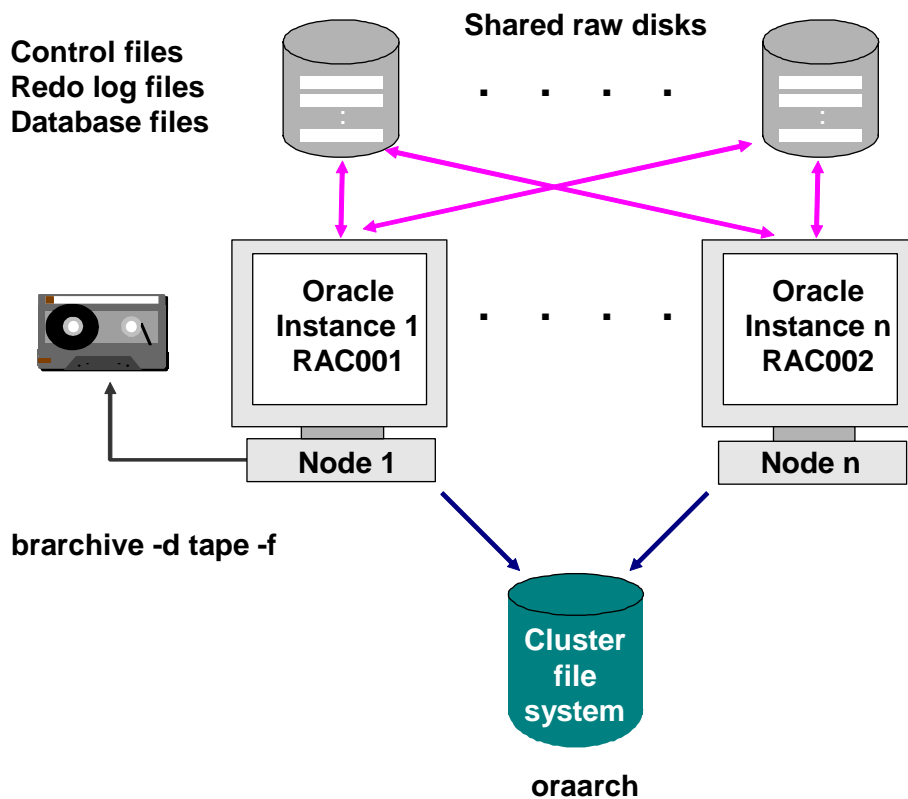
Caution

It is important to archive the offline redo log files, because these are essential if you have to [recover the database](#).

## Integration

The following graphic shows a typical archiving scenario with RAC:





## Prerequisites

Offline redo log files of different instances can be stored in the common archiving directory located on a shared file system of another instance. This can occur if the online redo log files of a particular instance have not been saved in the archiving directory for a long time.

Possible reasons are that the instance is closed or it is started but not active. Therefore, since a redo log file change is not forced, Oracle archiving is not activated. One of the “active” instances notices this status and makes sure that the current online redo log file of the affected instance is saved in the archiving directory.

Therefore, you must set the [parallel\\_instances](#) parameter in the `init<DBSID>.sap` profiles used by BRARCHIVE, so that BRARCHIVE can find and archive the offline redo log files of the other Oracle instances.

On *UNIX* systems, you can now perform BRBACKUP and BRARCHIVE backups of RAC databases with the `OPSS$` user *without* entering the database user and password in the `-u` option. For more information, see [SAP Note 914174](#).

On *Windows* systems, this is possible in the standard Oracle configuration.

## Features

- There is only one volume pool managed by BRARCHIVE.
- The entire BRARCHIVE functionality is available, even continuous archiving with the option `-f`.
- Only the disk space in the global archiving directory must be provided.

- Normally there is a global archiving directory created in the cluster file system. This directory can be accessed from every node of the RAC system. All the archiving processes save the online redo log files to the global directory in this way.

#### Example

Definition of the global archiving directory in `spfile`:

```
log_archive_dest = /oracle/RAC/oraarch/RACarch
```

```
log_archive_format = %t_%s.dbf
```

where:

- `%t`: thread number (for identifying the instance)
- `%s`: log sequence number

#### Caution

The Oracle placeholder `@` (for `ORACLE_SID`) must not appear in the name of an offline redo log file. This means that it must not appear in either `log_archive_dest` nor in `log_archive_format`.

## Activities

You start BRARCHIVE only on the host of one Oracle instance, generally on the dedicated database (DDB) instance. It archives all offline redo log files of all the Oracle instances.

BRARCHIVE creates a separate summary log, `arch<DBSID>.log`, for every Oracle instance. These logs are stored in the global `saparch` directory.

## RAC with BRRESTORE and BRRECOVER

This section describes how to set up the [Oracle Real Application Cluster \(RAC\)](#) with [BRRESTORE](#) and [BRRECOVER](#).

### Prerequisites

If the database uses [raw devices](#), BRRESTORE uses the `dd` command to restore files that were saved on a volume (tape, hard disk, and so on) with this command. Therefore, make sure that the `init<DBSID>.sap` parameters [dd flags](#) and [dd in flags](#) are set accordingly.

## Activities

### BRRESTORE for Restoring Database Files

There are no special BRRESTORE features to restore database files on RAC.

### BRRESTORE for Restoring the Archived Offline Redo Log Files

You can restore the required archived redo log files of all Oracle instances with one BRRESTORE call.

#### Caution

You must have the necessary archived offline redo log files in order to recover the database. For more information, see [RAC with BRARCHIVE](#).

The BRRESTORE options `-a` | `-a1` | `-a2` let you specify the instance name `<DBSID>`. For more information, see:

- [-a|-archive|a1|archive1](#)
- [RAC with BRARCHIVE](#)

Example

```
brrestore -a 110-120,RAC002,70-80,RAC003,85-95
```

The archived offline redo log files with log sequence numbers 110-120 of the local Oracle instance, those with log sequence numbers 70-80 of Oracle instance `RAC002` and those with log sequence numbers 85-95 of instance `RAC003` are restored.

## BRRECOVER for Recovering the Database

Caution

We recommend that only an experienced database administrator, familiar with RAC, performs recovery.

You normally perform restore and recovery from the dedicated database (DDB) instance. Other database instances are normally stopped at this time. BRRECOVER manages the restore and recovery of the RAC database by controlling the applying of the offline redo log files of all threads, that is, the files that have been generated by the different instances.

For more information, see [Restore and Recovery with BR\\*Tools](#).

## RAC with the `init<DBSID>.sap` Profile

This section contains information on the profile `init<DBSID>.sap` for the [Oracle Real Application Cluster \(RAC\)](#).

- [dd\\_flags](#) and [dd\\_in\\_flags](#)

You must install the RAC database on a cluster file system or raw devices. If installed on raw devices, the backup runs using the `dd` command. Therefore, make sure that you set the parameters [dd\\_flags](#) and [dd\\_in\\_flags](#), which specify the required `dd` command. See [Raw Devices with BRBACKUP and BRRESTORE](#).

Note

You do not need to set this parameter if you use external backup tools to back up your database, with the following setting for [backup\\_dev\\_type](#):

```
backup_dev_type = util_file |  
util_file_online|util_vol|util_vol_online.
```

- [parallel\\_instances](#)

This parameter defines the instances running in parallel to the DDB instance.

Caution

All database instances (including the local one, DDB) must be specified in the list of instance descriptions.

- [db\\_services](#)

This parameter activates the handling of database cluster services by BR\*Tools.