

Oracle9i

Recovery Manager Reference

Release 2 (9.2)

March 2002

Part No. A96565-01

Oracle9i Recovery Manager Reference, Release 2 (9.2)

Part No. A96565-01

Copyright © 1996, 2002, Oracle Corporation. All rights reserved.

Primary Author: Lance Ashdown

Contributors: Beldalker Anand, Tammy Bednar, Don Beusee, Senad Dizdar, Muthu Olagappan, Francisco Sanchez, Steve Wertheimer

Graphic Designer: Valarie Moore

The Programs (which include both the software and documentation) contain proprietary information of Oracle Corporation; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent and other intellectual and industrial property laws. Reverse engineering, disassembly or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. Oracle Corporation does not warrant that this document is error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of Oracle Corporation.

If the Programs are delivered to the U.S. Government or anyone licensing or using the programs on behalf of the U.S. Government, the following notice is applicable:

Restricted Rights Notice Programs delivered subject to the DOD FAR Supplement are "commercial computer software" and use, duplication, and disclosure of the Programs, including documentation, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement. Otherwise, Programs delivered subject to the Federal Acquisition Regulations are "restricted computer software" and use, duplication, and disclosure of the Programs shall be subject to the restrictions in FAR 52.227-19, Commercial Computer Software - Restricted Rights (June, 1987). Oracle Corporation, 500 Oracle Parkway, Redwood City, CA 94065.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and Oracle Corporation disclaims liability for any damages caused by such use of the Programs.

Oracle is a registered trademark, and Oracle Store, Oracle7, Oracle8, Oracle8i, Oracle9i, PL/SQL, and SQL*Plus are trademarks or registered trademarks of Oracle Corporation. Other names may be trademarks of their respective owners.

Contents

Send Us Your Comments	vii
Preface.....	ix
1 About RMAN Commands	
Conventions Used in this Reference.....	1-2
RMAN Command Entries.....	1-6
2 RMAN Commands	
Summary of RMAN Commands.....	2-2
@	2-6
@@	2-7
ALLOCATE CHANNEL	2-8
ALLOCATE CHANNEL FOR MAINTENANCE.....	2-12
allocOperandList.....	2-15
ALTER DATABASE.....	2-19
archiveLogRecordSpecifier	2-21
BACKUP	2-26
BLOCKRECOVER.....	2-48
CATALOG.....	2-52
CHANGE.....	2-55
cmdLine	2-59
completedTimeSpec	2-64
CONFIGURE.....	2-66

CONNECT	2-80
connectStringSpec	2-82
COPY	2-84
CREATE CATALOG	2-89
CREATE SCRIPT	2-91
CROSSCHECK	2-93
datafileSpec	2-96
DELETE	2-98
DELETE SCRIPT	2-103
deviceSpecifier	2-104
DROP CATALOG	2-105
DUPLICATE	2-106
EXECUTE SCRIPT	2-115
EXIT	2-117
HOST	2-118
keepOption	2-120
LIST	2-122
listObjList	2-140
maintQualifier	2-142
obsOperandList	2-144
PRINT SCRIPT	2-146
QUIT	2-148
recordSpec	2-149
RECOVER	2-151
REGISTER	2-157
RELEASE CHANNEL	2-159
releaseForMaint	2-160
REPLACE SCRIPT	2-161
REPORT	2-164
RESET DATABASE	2-172
RESTORE	2-175
RESYNC	2-187
RUN	2-190
SEND	2-193
SET	2-195

SHOW	2-202
SHUTDOWN.....	2-205
SPOOL	2-208
SQL	2-210
STARTUP.....	2-212
SWITCH	2-214
untilClause	2-216
UPGRADE CATALOG	2-219
VALIDATE	2-221

3 Recovery Catalog Views

Summary of RMAN Recovery Catalog Views	3-2
RC_ARCHIVED_LOG.....	3-4
RC_BACKUP_CONTROLFILE.....	3-5
RC_BACKUP_CORRUPTION.....	3-7
RC_BACKUP_DATAFILE	3-8
RC_BACKUP_PIECE	3-10
RC_BACKUP_REDOLOG	3-11
RC_BACKUP_SET.....	3-12
RC_BACKUP_SPFILE.....	3-14
RC_CHECKPOINT.....	3-15
RC_CONTROLFILE_COPY.....	3-15
RC_COPY_CORRUPTION.....	3-16
RC_DATABASE	3-17
RC_DATABASE_BLOCK_CORRUPTION	3-18
RC_DATABASE_INCARNATION.....	3-18
RC_DATAFILE	3-19
RC_DATAFILE_COPY	3-20
RC_LOG_HISTORY.....	3-22
RC_OFFLINE_RANGE.....	3-23
RC_PROXY_CONTROLFILE.....	3-24
RC_PROXY_DATAFILE	3-25
RC_REDO_LOG	3-27
RC_REDO_THREAD.....	3-27
RC_RESYNC.....	3-28

RC_RMAN_CONFIGURATION	3-29
RC_STORED_SCRIPT	3-29
RC_STORED_SCRIPT_LINE	3-30
RC_TABLESPACE	3-30

A Deprecated RMAN Commands

B RMAN Compatibility

About RMAN Compatibility	B-2
RMAN Compatibility Matrix	B-2
RMAN Compatibility: Scenario	B-4

Index

Send Us Your Comments

Oracle9i Recovery Manager Reference, Release 2 (9.2)

Part No. A96565-01

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, please indicate the document title and part number, and the chapter, section, and page number (if available). You can send comments to us in the following ways:

- Electronic mail: infodev_us@oracle.com
- FAX: (650) 506-7227 Attn: Server Technologies Documentation Manager
- Postal service:
Oracle Corporation
Server Technologies Documentation
500 Oracle Parkway, Mailstop 4op11
Redwood Shores, CA 94065
USA

If you would like a reply, please give your name, address, telephone number, and (optionally) electronic mail address.

If you have problems with the software, please contact your local Oracle Support Services.

Preface

This preface contains these topics:

- [Audience](#)
- [Organization](#)
- [Related Documentation](#)
- [Conventions](#)
- [Documentation Accessibility](#)

Audience

This manual is intended for database administrators who perform the following tasks:

- Back up, restore, and recover Oracle databases
- Perform maintenance on backups and copies of database files

To use this document, you need to know the following:

- Relational database concepts and basic database administration as described in *Oracle9i Database Concepts* and the *Oracle9i Database Administrator's Guide*
- Basic backup and recovery concepts and strategies as described in the *Oracle9i Backup and Recovery Concepts*
- Basic RMAN concepts and tasks as described in *Oracle9i Recovery Manager User's Guide*
- The operating system environment under which you are running Oracle

Organization

This document contains:

Chapter 1, "About RMAN Commands"

This chapter describes the basic conventions of RMAN syntax.

Chapter 2, "RMAN Commands"

This chapter displays the RMAN syntax diagrams, describes the elements of the syntax, and provides examples.

Chapter 3, "Recovery Catalog Views"

This chapter describes the recovery catalog views.

Appendix A, "Deprecated RMAN Commands"

This appendix describes RMAN syntax that is deprecated (that is, no longer supported) but still functional.

Appendix B, "RMAN Compatibility"

This appendix shows the compatible combinations of the RMAN executable, target database, recovery catalog database, and recovery catalog schema.

Related Documentation

For more information, see these Oracle resources:

- *Oracle9i Recovery Manager User's Guide*
- *Oracle9i Backup and Recovery Concepts*
- *Oracle9i User-Managed Backup and Recovery Guide*
- *Oracle9i Database Utilities*
- <http://www.oracle.com/database/recovery>

You can access information about the Backup Solutions Program at

<http://otn.oracle.com/deploy/availability>

Many of the examples in this book use the sample schemas of the seed database, which is installed by default when you install Oracle. Refer to *Oracle9i Sample Schemas* for information on how these schemas were created and how you can use them yourself.

In North America, printed documentation is available for sale in the Oracle Store at

<http://oraclystore.oracle.com/>

Customers in Europe, the Middle East, and Africa (EMEA) can purchase documentation from

<http://www.oraclebookshop.com/>

Other customers can contact their Oracle representative to purchase printed documentation.

To download free release notes, installation documentation, white papers, or other collateral, please visit the Oracle Technology Network (OTN). You must register online before using OTN; registration is free and can be done at

<http://otn.oracle.com/admin/account/membership.html>

If you already have a username and password for OTN, then you can go directly to the documentation section of the OTN Web site at

<http://otn.oracle.com/docs/index.htm>

To access the database documentation search engine directly, please visit

<http://tahiti.oracle.com>

Conventions

This section describes the conventions used in the text and code examples of this documentation set. It describes:

- [Conventions in Text](#)
- [Conventions in Code Examples](#)
- [Conventions for Windows Operating Systems](#)

Conventions in Text

We use various conventions in text to help you more quickly identify special terms. The following table describes those conventions and provides examples of their use.

Convention	Meaning	Example
Bold	Bold typeface indicates terms that are defined in the text or terms that appear in a glossary, or both.	When you specify this clause, you create an index-organized table .
<i>Italics</i>	Italic typeface indicates book titles or emphasis.	<i>Oracle9i Database Concepts</i> Ensure that the recovery catalog and target database do <i>not</i> reside on the same disk.
UPPERCASE monospace (fixed-width) font	Uppercase monospace typeface indicates elements supplied by the system. Such elements include parameters, privileges, datatypes, RMAN keywords, SQL keywords, SQL*Plus or utility commands, packages and methods, as well as system-supplied column names, database objects and structures, usernames, and roles.	You can specify this clause only for a NUMBER column. You can back up the database by using the BACKUP command. Query the TABLE_NAME column in the USER_TABLES data dictionary view. Use the DBMS_STATS.GENERATE_STATS procedure.

Convention	Meaning	Example
lowercase monospace (fixed-width) font	Lowercase monospace typeface indicates executables, filenames, directory names, and sample user-supplied elements. Such elements include computer and database names, net service names, and connect identifiers, as well as user-supplied database objects and structures, column names, packages and classes, usernames and roles, program units, and parameter values. Note: Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.	Enter <code>sqlplus</code> to open SQL*Plus. The password is specified in the <code>orapwd</code> file. Back up the datafiles and control files in the <code>/disk1/oracle/dbs</code> directory. The <code>department_id</code> , <code>department_name</code> , and <code>location_id</code> columns are in the <code>hr.departments</code> table. Set the <code>QUERY_REWRITE_ENABLED</code> initialization parameter to <code>true</code> . Connect as <code>oe</code> user. The <code>JRepUtil</code> class implements these methods.
lowercase italic monospace (fixed-width) font	Lowercase italic monospace font represents placeholders or variables.	You can specify the <i>parallel_clause</i> . Run <code>Uold_release.SQL</code> where <i>old_release</i> refers to the release you installed prior to upgrading.

Conventions in Code Examples

Code examples illustrate SQL, PL/SQL, SQL*Plus, or other command-line statements. They are displayed in a monospace (fixed-width) font and separated from normal text as shown in this example:

```
SELECT username FROM dba_users WHERE username = 'MIGRATE';
```

The following table describes typographic conventions used in code examples and provides examples of their use.

Convention	Meaning	Example
[]	Brackets enclose one or more optional items. Do not enter the brackets.	DECIMAL (<i>digits</i> [, <i>precision</i>])
{ }	Braces enclose two or more items, one of which is required. Do not enter the braces.	{ENABLE DISABLE}
	A vertical bar represents a choice of two or more options within brackets or braces. Enter one of the options. Do not enter the vertical bar.	{ENABLE DISABLE} [COMPRESS NOCOMPRESS]

Convention	Meaning	Example
...	Horizontal ellipsis points indicate either: <ul style="list-style-type: none"> That we have omitted parts of the code that are not directly related to the example That you can repeat a portion of the code 	<pre>CREATE TABLE ... AS subquery; SELECT col1, col2, ... , coln FROM employees;</pre>
. . .	Vertical ellipsis points indicate that we have omitted several lines of code not directly related to the example.	<pre>SQL> SELECT NAME FROM V\$DATAFILE; NAME ----- /fs1/dbs/tbs_01.dbf /fs1/dbs/tbs_02.dbf . . . /fs1/dbs/tbs_09.dbf 9 rows selected.</pre>
Other notation	You must enter symbols other than brackets, braces, vertical bars, and ellipsis points as shown.	<pre>acctbal NUMBER(11,2); acct CONSTANT NUMBER(4) := 3;</pre>
<i>Italics</i>	Italicized text indicates placeholders or variables for which you must supply particular values.	<pre>CONNECT SYSTEM/system_password DB_NAME = database_name</pre>
UPPERCASE	Uppercase typeface indicates elements supplied by the system. We show these terms in uppercase in order to distinguish them from terms you define. Unless terms appear in brackets, enter them in the order and with the spelling shown. However, because these terms are not case sensitive, you can enter them in lowercase.	<pre>SELECT last_name, employee_id FROM employees; SELECT * FROM USER_TABLES; DROP TABLE hr.employees;</pre>
lowercase	Lowercase typeface indicates programmatic elements that you supply. For example, lowercase indicates names of tables, columns, or files. Note: Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.	<pre>SELECT last_name, employee_id FROM employees; sqlplus hr/hr CREATE USER mjones IDENTIFIED BY ty3MU9;</pre>

Conventions for Windows Operating Systems

The following table describes conventions for Windows operating systems and provides examples of their use.

Convention	Meaning	Example
Choose Start >	How to start a program.	To start the Database Configuration Assistant, choose Start > Programs > Oracle - <i>HOME_NAME</i> > Configuration and Migration Tools > Database Configuration Assistant.
File and directory names	File and directory names are not case sensitive. The following special characters are not allowed: left angle bracket (<), right angle bracket (>), colon (:), double quotation marks ("), slash (/), pipe (), and dash (-). The special character backslash (\) is treated as an element separator, even when it appears in quotes. If the file name begins with \\, then Windows assumes it uses the Universal Naming Convention.	<code>c:\winnt\ "system32</code> is the same as <code>C:\WINNT\SYSTEM32</code>
<code>C:\></code>	Represents the Windows command prompt of the current hard disk drive. The escape character in a command prompt is the caret (^). Your prompt reflects the subdirectory in which you are working. Referred to as the <i>command prompt</i> in this manual.	<code>C:\oracle\oradata></code>
Special characters	The backslash (\) special character is sometimes required as an escape character for the double quotation mark (") special character at the Windows command prompt. Parentheses and the single quotation mark (') do not require an escape character. Refer to your Windows operating system documentation for more information on escape and special characters.	<code>C:\>exp scott/tiger TABLES=emp QUERY=\ "WHERE job='SALESMAN' and sal<1600\ C:\>imp SYSTEM/password FROMUSER=scott TABLES=(emp, dept)</code>
<i>HOME_NAME</i>	Represents the Oracle home name. The home name can be up to 16 alphanumeric characters. The only special character allowed in the home name is the underscore.	<code>C:\> net start Oracle<i>HOME_NAME</i>INSListener</code>

Convention	Meaning	Example
<i>ORACLE_HOME</i> and <i>ORACLE_BASE</i>	<p>In releases prior to Oracle8i release 8.1.3, when you installed Oracle components, all subdirectories were located under a top level <i>ORACLE_HOME</i> directory that by default used one of the following names:</p> <ul style="list-style-type: none"> ■ C:\orant for Windows NT ■ C:\orawin98 for Windows 98 <p>This release complies with Optimal Flexible Architecture (OFA) guidelines. All subdirectories are not under a top level <i>ORACLE_HOME</i> directory. There is a top level directory called <i>ORACLE_BASE</i> that by default is C:\oracle. If you install the latest Oracle release on a computer with no other Oracle software installed, then the default setting for the first Oracle home directory is C:\oracle\orann, where <i>nn</i> is the latest release number. The Oracle home directory is located directly under <i>ORACLE_BASE</i>.</p> <p>All directory path examples in this guide follow OFA conventions.</p> <p>Refer to <i>Oracle9i Database Getting Started for Windows</i> for additional information about OFA compliances and for information about installing Oracle products in non-OFA compliant directories.</p>	Go to the <i>ORACLE_BASE\ORACLE_HOME\rdms\admin</i> directory.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Standards will continue to evolve over time, and Oracle Corporation is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For additional information, visit the Oracle Accessibility Program Web site at

<http://www.oracle.com/accessibility/>

Accessibility of Code Examples in Documentation JAWS, a Windows screen reader, may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, JAWS may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation This documentation may contain links to Web sites of other companies or organizations that Oracle Corporation does not own or control. Oracle Corporation neither evaluates nor makes any representations regarding the accessibility of these Web sites.

About RMAN Commands

This chapter describes the basic elements of RMAN syntax. It includes the following sections:

- [Conventions Used in this Reference](#)
- [RMAN Command Entries](#)

Conventions Used in this Reference

This section explains the conventions used in this chapter including:

- [RMAN Text Conventions](#)
- [RMAN Syntax Diagrams and Notation](#)
- [RMAN Code Examples](#)

RMAN Text Conventions

The text in this reference adheres to the following conventions:

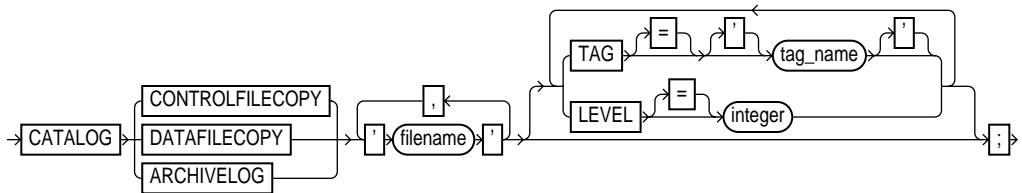
<code>UPPERCASE monospace</code>	Uppercase monospace text calls attention to RMAN keywords, SQL keywords, column headings in tables and views, and initialization parameters.
<code>lowercase monospace</code>	Lowercase monospace text calls attention to variable text in RMAN examples.
<i>italics</i>	Italicized monospace text calls attention to RMAN or SQL placeholders, that is, text that should not be entered as-is but represents a value to be entered by the user.

RMAN Syntax Diagrams and Notation

This section describes the conventions for RMAN command syntax.

Syntax Diagrams

This reference uses syntax diagrams to show Recovery Manager commands. These syntax diagrams use lines and arrows to show syntactic structure, as shown in [Figure 1-1](#).

Figure 1–1 CATALOG Command

This section describes the components of syntax diagrams and gives examples of how to write RMAN commands. Syntax diagrams are made up of these items:

- **Keywords**
- **Placeholders**

Keywords Keywords have special meanings in Recovery Manager syntax. In the syntax diagrams, keywords appear in rectangular boxes and an uppercase font, like the word `CATALOG` in [Figure 1–1](#). When used in text and code examples, RMAN keywords appear in uppercase, monospace font, for example, `CATALOG DATAFILECOPY`. You must use keywords in RMAN statements exactly as they appear in the syntax diagram, except that they can be either uppercase or lowercase.

The RMAN language is free-form. Keywords must be separated by at least one white space character, but otherwise there are no restrictions. A command can span multiple lines.

Placeholders Placeholders in syntax diagrams indicate non-keywords. In the syntax diagrams, they appear in ovals, as in the word *integer* in [Figure 1–1](#). When described in text, RMAN placeholders appear in lowercase italic, for example, *'filename'*. Placeholders are usually:

- Names of database objects (*tablespace_name*)
- Oracle datatype names (*date_string*)
- Subclauses (*datafileSpec*)

When you see a placeholder in a syntax diagram, substitute an object or expression of the appropriate type in the RMAN statement. For example, to write a `DUPLICATE TARGET DATABASE TO 'database_name'` command, use the name of the duplicate database you want to create, such as `dupdb`, in place of the *database_name* placeholder in the diagram.

Some placeholder values are enclosed in required or optional quotes. The syntax diagrams show single quotes, though in all cases double quotes are also legal in RMAN syntax. For example, you specify either *'filename'* or *"filename"*. For the SQL command, it is recommended that you use double quotes because the SQL statement itself may also contain a quote, and the most common type of quote in a SQL statement is a single quote. Single and double quotes do not mean the same in SQL as they do in RMAN.

The only system-independent, legal environment variables in RMAN quoted strings are *?* for the Oracle home and *@* for the SID. However, you can use operating system specific environment variables on the target system within quoted strings. The environment variables are interpreted by the database server and not the RMAN client.

The following table shows placeholders that appear in the syntax diagrams and provides examples of the values you might substitute for them in your statements.

Placeholder	Description	Examples
Quoted strings such as <i>'filename'</i> , <i>'tablespace_name'</i> , <i>'channel_name'</i> , <i>'channel_parms'</i>	A string of characters contained in either single or double quotes. A quoted string may contain white space, punctuation, and RMAN and SQL keywords.	" <i>?</i> /dbs/cf.f" <i>'dev1'</i>
Nonquoted strings such as <i>channel_id</i> , <i>tag_name</i> , <i>date_string</i>	A sequence of characters containing no white space and no punctuation characters and starting with an alphabetic character.	ch1
<i>integer</i>	Any sequence of only number characters.	67843

Reserved Words

This section describes the RMAN reserved words. If you use one of these words by itself without surrounding it in quotes, then RMAN generates an error. These are examples of correct and incorrect entries:

```

ALLOCATE CHANNEL backup DEVICE TYPE DISK;           # incorrect
ALLOCATE CHANNEL 'backup' DEVICE TYPE DISK;         # correct
BACKUP DATABASE TAG full;                             # incorrect
BACKUP DATABASE TAG 'full';                           # correct
    
```

ABORT	CONSISTENT	FORMAT	MAXPIECESIZE	PLSQL	SETLIMIT
AFFINITY	CONTROLFILE	FROM	MAXSEQ	PLUS	SETSIZE
AFTER	CONTROLFILECOPY	FULL	MAXSETSIZE	POLICY	SHOW
ALL	COPIES	G	MAXSIZE	POOL	SHUTDOWN
ALLOCATE	COPY	GET	MISC	PRINT	SINCE
ALTER	CORRUPTION	GROUP	MOUNT	PROXY	SIZE
AND	CREATE	HIGH	MSGLOG	PUT	SKIP
APPEND	CROSSCHECK	HOST	MSGNO	QUIT	SLAXDEBUG
ARCHIVELOG	CUMULATIVE	ID	NAME	RATE	SNAPSHOT
AT	CURRENT	IDENTIFIER	NEED	RCVCAT	SPFILE
ATALL	DATABASE	IMMEDIATE	NEW	RCVMAN	SPOOL
AUTOBACKUP	DATAFILE	INACCESSIBLE	NEW-LINE	READONLY	SQL
AUTOLOCATE	DATAFILECOPY	INCARNATION	NEWNAME	READRATE	STANDBY
AUXILIARY	DAYS	INCLUDE	NOCATALOG	RECOVER	STARTUP
AUXNAME	DBA	INCREMENTAL	NOCFAU	RECOVERABLE	STEP
AVAILABLE	DBID	INPUT	NOCHECKSUM	RECOVERY	SUMMARY
BACKED	DEBUG	IO	NOEXCLUDE	REDUNDANCY	SWITCH
BACKUP	DEFAULT	JOB	NOFILENAMECHECK	REGISTER	TABLESPACE
BACKUPPIECE	DEFINE	K	NOFILEUPDATE	RELEASE	TAG
BACKUPSET	DELETE	KBYTES	NOKEEP	RELOAD	TARGET
BEFORE	DESTINATION	KEEP	NOLOGS	REMOVE	TEST
BETWEEN	DEVICE	LEVEL	NOMOUNT	RENORMALIZE	THREAD
BLOCK	DISK	LIBNAME	NONE	REPLACE	TIME
BLOCKRECOVER	DISKRATIO	LIBPARM	NOPROMPT	REPLICATE	TIMEOUT
BLOCKS	DISPLAY	LIBRARY	NOREDO	REPORT	TIMES
BY	DORECOVER	LIBTEXT	NORMAL	RESET	TO
CANCEL	DROP	LIKE	NOT	RESETLOGS	TRACE
CATALOG	DUMP	LIMIT	NULL	RESTART	TRANSACTIONAL
CHANGE	DUPLEX	LIST	OBSOLETE	RESTORE	TXT
CHANNEL	DUPLICATE	LOG	OF	RESYNC	TYPE
CHARSET	ECHO	LOGFILE	OFF	RETENTION	UNAVAILABLE
CHECK	EXCLUDE	LOGICAL	OFFLINE	REUSE	UNCATALOG

CLEAR	EXECUTE	LOGS	ON	RPC	UNLIMITED
CLONE	EXIT	LOGSCN	ONLY	RPCTEST	UNRECOVERABLE
CLONENAME	EXPIRED	LOGSEQ	OPEN	RUN	UNTIL
CLONE_CF	FILE	LOW	OPTIMIZATION	SAVE	UNUSED
CMDFILE	FILES	M	ORPHAN	SCHEMA	UP
COMMAND	FILESPERSET	MAINTENANCE	PACKAGES	SCN	UPGRADE
COMPATIBLE	FINAL	MASK	PARALLELISM	SCRIPT	VALIDATE
COMPLETED	FOR	MAXCORRUPT	PARMS	SEND	VERBOSE
CONFIGURE	FORCE	MAXDAYS	PFILE	SEQUENCE	WINDOW
CONNECT	FOREVER	MAXOPENFILES	PIPE	SET	

RMAN Code Examples

This reference contains many examples of RMAN commands. These examples show you how to use elements of RMAN. This example shows the use of a BACKUP command:

```

RUN
{
  ALLOCATE CHANNEL ch1 DEVICE TYPE sbt;
  BACKUP DATABASE;
}

```

Note that examples are set off from the text and appear in a monospace font.

RMAN Command Entries

The description of each command or subclause contains the following sections:

Syntax	Shows the keywords and parameters that make up the statement.
	Note: Not all keywords and parameters are valid in all circumstances. Be sure to refer to the "Keywords and Parameters" section of each statement to learn about any restrictions on the syntax.
Purpose	Describes the basic uses of the statement.
Restrictions and Usage Notes	Lists requirements, restrictions, and guidelines for proper use of the command.

Keywords and Parameters

Describes the purpose of each keyword and parameter. Restrictions and usage notes also appear in this section.

Examples

Shows how to use various clauses and options of the statement.

Note: Optional sections following the examples provide more information on how and when to use the statement.

RMAN Commands

This chapter describes, in alphabetical order, Recovery Manager commands and subclauses. For a summary of the RMAN commands and command-line options, refer to "[Summary of RMAN Commands](#)" on page 2-2.

Summary of RMAN Commands

Table 2–1 provides a functional summary of RMAN commands that you can execute at the RMAN prompt, within a `RUN` command, or both. All commands from previous RMAN releases work with the current release.

For RMAN options that you can specify on the operating system command line, refer to "[cmdLine](#)" on page 2-59.

Table 2–1 Recovery Manager Commands

Command	Purpose
"@" on page 2-6	Run a command file.
"@@" on page 2-7	Run a command file in the same directory as another command file that is currently running. The @@ command differs from the @ command only when run from within a command file.
" ALLOCATE CHANNEL " on page 2-8	Establish a channel, which is a connection between RMAN and a database instance.
" ALLOCATE CHANNEL FOR MAINTENANCE " on page 2-12	Allocate a channel in preparation for issuing maintenance commands such as DELETE .
" allocOperandList " on page 2-15	A subclause that specifies channel control options such as <code>PARMS</code> , <code>FORMAT</code> , and <code>MAXOPENFILES</code> .
" ALTER DATABASE " on page 2-19	Mount or open a database.
" archivelogRecordSpecifier " on page 2-21	Specify a range of archived redo logs files.
" BACKUP " on page 2-26	Back up a database, tablespace, datafile, archived log, or backup set.
" BLOCKRECOVER " on page 2-48	Recover an individual data block or set of data blocks within one or more datafiles.
" CATALOG " on page 2-52	Add information about a datafile copy, archived redo log, or control file copy to the repository.
" CHANGE " on page 2-55	Mark a backup piece, image copy, or archived redo log as having the status <code>UNAVAILABLE</code> or <code>AVAILABLE</code> ; remove the repository record for a backup or copy; override the retention policy for a backup or copy.
" completedTimeSpec " on page 2-64	Specify a time range during which the backup or copy completed.
" CONFIGURE " on page 2-66	Configure persistent RMAN settings. These settings apply to all RMAN sessions until explicitly changed or disabled.

Table 2-1 Recovery Manager Commands

Command	Purpose
"CONNECT" on page 2-80	Establish a connection between RMAN and a target, auxiliary, or recovery catalog database.
"connectStringSpec" on page 2-82	Specify the username, password, and net service name for connecting to a target, recovery catalog, or auxiliary database. The connection is necessary to authenticate the user and identify the database.
"COPY" on page 2-84	Create an image copy of a datafile, control file, or archived redo log.
"CREATE CATALOG" on page 2-89	Create the schema for the recovery catalog.
"CREATE SCRIPT" on page 2-91	Create a stored script and store it in the recovery catalog.
"CROSSCHECK" on page 2-93	Determine whether files managed by RMAN, such as archived logs, datafile copies, and backup pieces, still exist on disk or tape.
"datafileSpec" on page 2-96	Specify a datafile by filename or absolute file number.
"DELETE" on page 2-98	Delete backups and copies, remove references to them from the recovery catalog, and update their control file records to status DELETED.
"DELETE SCRIPT" on page 2-103	Delete a stored script from the recovery catalog.
"deviceSpecifier" on page 2-104	Specify the type of storage device for a backup or copy.
"DROP CATALOG" on page 2-105	Remove the schema from the recovery catalog.
"DUPLICATE" on page 2-106	Use backups of the target database to create a duplicate database that you can use for testing purposes or to create a standby database.
"EXECUTE SCRIPT" on page 2-115	Run an RMAN stored script.
"EXIT" on page 2-117	Quit the RMAN executable.
"HOST" on page 2-118	Invoke an operating system command-line subshell from within RMAN or run a specific operating system command.
"keepOption" on page 2-120	Specify that a backup or copy should or should not be exempt from the current retention policy.
"LIST" on page 2-122	Produce a detailed listing of backup sets or copies.
"listObjList" on page 2-140	A subclause used to specify which items will be displayed by the LIST command.
"maintQualifier" on page 2-142	A subclause used to specify additional options for maintenance commands such as DELETE and CHANGE.

Table 2–1 Recovery Manager Commands

Command	Purpose
"obsOperandList" on page 2-144	A subclause used to determine which backups and copies are obsolete.
"PRINT SCRIPT" on page 2-146	Display a stored script.
"QUIT" on page 2-148	Exit the RMAN executable.
"recordSpec" on page 2-149	A subclause used to specify which objects the maintenance commands should operate on.
"RECOVER" on page 2-151	Apply redo logs or incremental backups to a restored backup set or copy in order to update it to a specified time.
"REGISTER" on page 2-157	Register the target database in the recovery catalog.
"RELEASE CHANNEL" on page 2-159	Release a channel that was allocated with an <code>ALLOCATE CHANNEL</code> command.
"releaseForMaint" on page 2-160	Release a channel allocated with an <code>ALLOCATE CHANNEL FOR MAINTENANCE</code> command.
"REPLACE SCRIPT" on page 2-161	Replace an existing script stored in the recovery catalog. If the script does not exist, then <code>REPLACE SCRIPT</code> creates it.
"REPORT" on page 2-164	Perform detailed analyses of the content of the recovery catalog.
"RESET DATABASE" on page 2-172	Inform RMAN that the SQL statement <code>ALTER DATABASE OPEN RESETLOGS</code> has been executed and that a new incarnation of the target database has been created, or reset the target database to a prior incarnation.
"RESTORE" on page 2-175	Restore files from backup sets or from disk copies to the default or a new location.
"RESYNC" on page 2-187	Perform a full resynchronization, which creates a snapshot control file and then copies any new or changed information from that snapshot control file to the recovery catalog.
"RUN" on page 2-190	Execute a sequence of one or more RMAN commands, which are one or more statements executed within the braces of <code>RUN</code> .
"SEND" on page 2-193	Send a vendor-specific quoted string to one or more specific channels.

Table 2–1 Recovery Manager Commands

Command	Purpose
"SET" on page 2-195	Make the following session-level settings: <ul style="list-style-type: none"> ■ Control whether RMAN commands are displayed in the message log ■ Set the DBID when restoring a control file or server parameter file ■ Specify new filenames for restored datafiles ■ Specify a limit for the number of permissible block corruptions ■ Override default archived redo log destinations ■ Specify the number of copies of each backup piece ■ Determine which server session corresponds to which channel ■ Control where RMAN searches for backups when using an Oracle Real Application Clusters configuration ■ Override the default format of the control file autobackup
"SHOW" on page 2-202	Displays the current CONFIGURE settings.
"SHUTDOWN" on page 2-205	Shut down the target database. This command is equivalent to the SQL*Plus SHUTDOWN command.
"SPOOL" on page 2-208	Write RMAN output to a log file.
"SQL" on page 2-210	Execute a SQL statement from within Recovery Manager.
"STARTUP" on page 2-212	Start up the target database. This command is equivalent to the SQL*Plus STARTUP command.
"SWITCH" on page 2-214	Specify that a datafile copy is now the current datafile , that is, the datafile pointed to by the control file. This command is equivalent to the SQL statement ALTER DATABASE RENAME FILE as it applies to datafiles.
"untilClause" on page 2-216	A subclause specifying an upper limit by time, SCN, or log sequence number. This clause is usually used to specify the desired point in time for an incomplete recovery.
"UPGRADE CATALOG" on page 2-219	Upgrade the recovery catalog schema from an older version to the version required by the RMAN executable.
"VALIDATE" on page 2-221	Examine a backup set and report whether its data is intact. RMAN scans all of the backup pieces in the specified backup sets and looks at the checksums to verify that the contents can be successfully restored.

@

@

Syntax

→ [@] → (filename) →

Purpose

To execute a series of RMAN commands stored in an operating system file with the specified full path name, for example, `@/oracle/dbs/cmd/cmd1.rman`. If you do not specify the full path name, then the current working directory is assumed, for example, `@cmd1.rman`. Do not use quotes around the string or leave whitespace between the `@` and filename. RMAN processes the specified file as if its contents had appeared in place of the `@` command.

Note: The file must contain complete RMAN commands; partial commands generate syntax errors.

Restrictions and Usage Notes

Execute at the operating system command line, at the RMAN prompt, or within the braces of a [RUN](#) command.

Example

Running a Command File from the Command Line: Example This example creates a command file and then runs it from the operating system command line:

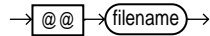
```
echo "BACKUP DATABASE;" > backup_db.rman
rman TARGET / @backup_db.rman
```

Running a Command File Within RMAN: Example This example runs a command file from the RMAN prompt and from within a `RUN` command:

```
@backup_db.rman
RUN { @backup_db.rman }
```

@@

Syntax



Purpose

To execute a series of RMAN commands stored in an operating system file with the specified filename, for example, `@@cmd2.rman`. The `@@` command is identical to the `@` command unless it is used within a script. If contained in a script, then `@@filename` directs RMAN to look for the specified filename in the same directory as the command file from which it was called.

For example, assume that you invoke RMAN as follows:

```
% rman @$ORACLE_HOME/rdbms/admin/dba/scripts/cmd1.rman
```

Assume that the command `@@cmd2.rman` appears inside the `cmd1.rman` script. In this case, the `@@` command directs RMAN to look for the file `cmd2.rman` in the directory `$ORACLE_HOME/rdbms/admin/dba/scripts/`.

Note: The file must contain complete RMAN commands.

Restrictions and Usage Notes

Execute at the operating system command line, at the RMAN prompt, or within the braces of a [RUN](#) command.

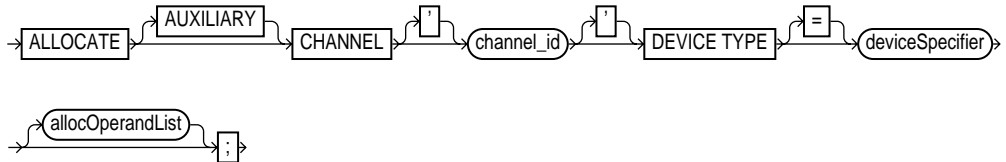
Example

Calling a Script Within a Script: Example Assume that you create command files called `backup_logs.rman` and `backup_db.rman` as in the following example. Then, you execute `bkup_db.rman` from the command line, which specifies that RMAN should look for the `bkup_logs.rman` script in the Oracle home directory:

```
echo "BACKUP ARCHIVELOG ALL;" > $ORACLE_HOME/bkup_logs.rman
echo "BACKUP DATABASE;" > $ORACLE_HOME/bkup_db.rman
echo "@@bkup_logs.rman" >> $ORACLE_HOME/bkup_db.rman
rman TARGET / @$ORACLE_HOME/bkup_db.rman
```

ALLOCATE CHANNEL

Syntax



Purpose

To manually allocate a **channel**, which is a connection between RMAN and a database instance. Each connection initiates an Oracle server session on the target or auxiliary instance: this server session performs the work of backing up, restoring, or recovering backup sets and copies.

Manually allocated channels (allocated by using `ALLOCATE`) are mutually exclusive with automatically allocated channels (specified by using `CONFIGURE`). Manually allocated channels apply only to the `RUN` job in which you issue the command. Automatic channels apply to any RMAN job in which you do *not* manually allocate channels. You can always override automatic channel configurations by manually allocating channels within a `RUN` command.

Each channel operates on one backup set at a time (for `BACKUP`, `RESTORE`, or `RECOVER`) or one image copy at a time (for `COPY`). RMAN automatically releases the channel at the end of the job.

You can control the degree of parallelism within a job by the number of channels. You can allocate multiple channels simultaneously, thus allowing a single job to read or write multiple backup sets or copies in parallel. If you establish multiple connections, then each connection operates on a separate backup set or file copy.

Whether `ALLOCATE CHANNEL` causes operating system resources to be allocated depends on the operating system. On some platforms, operating system resources are allocated at the time the command is issued. On other platforms, operating system resources are not allocated until you open a file for reading or writing.

Note: When you specify `DEVICE TYPE DISK`, no operating system resources are allocated other than for the creation of the server session.

Restrictions and Usage Notes

- Execute `ALLOCATE` only within the braces of a `RUN` command.
- The target instance must be started.
- You cannot make a connection to a shared server session.
- You must allocate either a manually allocated or automatic channel when executing a `BACKUP`, `DUPLICATE`, `COPY`, `RESTORE`, `RECOVER`, or `VALIDATE` command.
- You cannot specify `BACKUP DEVICE TYPE` or `RESTORE DEVICE TYPE` after running `ALLOCATE CHANNEL`.
- You must use a recovery catalog when backing up a standby database.
- You cannot prefix `ORA_` to a channel name. RMAN reserves channel names beginning with the `ORA_` prefix for its own use.

Keywords and Parameters

Syntax Element	Description
<code>AUXILIARY</code>	<p>Specifies a connection between RMAN and an auxiliary database instance. An auxiliary instance is used when executing the <code>DUPLICATE</code> command or performing TSPITR. An auxiliary database can reside on the same host as its parent or on a different host. When specifying this option, the auxiliary database must be mounted but not open.</p> <p>See Also: "<code>DUPLICATE</code>" on page 2-106 to learn how to duplicate a database, and "<code>CONNECT</code>" on page 2-80 to learn how to connect to a duplicate database</p>
<code>CHANNEL 'channel_id'</code>	<p>Specifies a connection between RMAN and the target database instance. Each connection initiates an Oracle server session on the database instance: this server session performs the work of backing up, restoring, and recovering backups and copies.</p> <p>Specify a channel id, which is the case-sensitive name of the channel, after the <code>CHANNEL</code> keyword. Oracle uses the to report I/O errors.</p>

Syntax Element	Description
<p>DEVICE TYPE = <i>deviceSpecifier</i></p>	<p>Specifies the type of storage device.</p> <p>See Also: "deviceSpecifier" on page 2-104</p> <p>Note: If you do not specify the DEVICE TYPE parameter, then you must specify the NAME parameter to identify a particular sequential I/O device. Query the V\$BACKUP_DEVICE view for information about available device types and names.</p>
<p><i>allocOperandList</i></p>	<p>Specifies control options for the allocated channel.</p> <p>See Also: "allocOperandList" on page 2-15</p>

Examples

Allocating a Single Channel for a Backup: Example This command allocates a tape channel for a whole database and archived redo log backup:

```

RUN
{
  ALLOCATE CHANNEL c1 DEVICE TYPE sbt;
  BACKUP DATABASE PLUS ARCHIVELOG;
}

```

Spreading a Backup Set Across Multiple Disks: Example When backing up to disk, you can spread the backup across several disk drives. Allocate one DEVICE TYPE DISK channel for each disk drive and specify the format string so that the filenames are on different disks:

```

RUN
{
  ALLOCATE CHANNEL disk1 DEVICE TYPE DISK FORMAT '/disk1/backups/%U';
  ALLOCATE CHANNEL disk2 DEVICE TYPE DISK FORMAT '/disk2/backups/%U';
  BACKUP DATABASE PLUS ARCHIVELOG;
}

```

Creating Multiple Copies of a Backup Set: Example When creating multiple copies of a backup set, you can specify the SET BACKUP COPIES command. The following example generates a single backup of the database, and then creates two identical backups of datafile 1 to two different file systems:

```

RUN
{
  ALLOCATE CHANNEL c1 DEVICE TYPE DISK MAXPIECESIZE 5M;
  BACKUP DATABASE PLUS ARCHIVELOG;
}

```

```
SET BACKUP COPIES = 2;
BACKUP DATAFILE 1 FORMAT '/disk1/backups/%U', '/disk2/backups/%U';
}
```

Allocating an Auxiliary Channel for Database Duplication: Example When creating a duplicate database, allocate a channel by using the `AUXILIARY` option:

```
RUN
{
  ALLOCATE AUXILIARY CHANNEL c1 DEVICE TYPE sbt;
  ALLOCATE AUXILIARY CHANNEL c2 DEVICE TYPE sbt;
  DUPLICATE TARGET DATABASE TO ndbnewh
  LOGFILE
    '?/oradata/auxl/redo01.log' SIZE 200K,
    '?/oradata/auxl/redo02.log' SIZE 200K
    '?/oradata/auxl/redo03.log' SIZE 200K
  SKIP READONLY
  NOFILENAMECHECK;
}
```

ALLOCATE CHANNEL FOR MAINTENANCE

Syntax



Purpose

To manually allocate a channel in preparation for issuing a [CHANGE](#), [DELETE](#), or [CROSSCHECK](#) command. Note that if you use [CONFIGURE](#) to set up automatic channels, then RMAN can use these automatic channels for maintenance operations; you do not have to manually allocate them.

If RMAN allocates the an automatic maintenance channel, then it uses the same naming convention as any other automatically allocated channel. If you manually run `ALLOCATE CHANNEL FOR MAINTENANCE`, then RMAN uses the following convention for channel naming: `ORA_MAINT_devicetype_n`, where *devicetype* refers to `DISK` or `sbt` and *n* refers to the channel number. For example, RMAN uses these names for two manually allocated disk channels:

```

ORA_MAINT_DISK_1
ORA_MAINT_DISK_2
  
```

You can allocate multiple maintenance channels for a single job, but you should only use this feature in these scenarios:

- To allow crosschecking or deletion of all backup pieces or proxy copies, both on disk and tape, with a single command
- To make crosschecking and deleting work correctly in an Oracle Real Application Clusters configuration in which each backup piece or proxy copy exists only on one node

See Also: *Oracle9i Recovery Manager User's Guide* to learn how to crosscheck and delete on multiple channels

Restrictions and Usage Notes

- Execute this command only at the RMAN prompt.
- The target instance must be started.

- Do not specify a channel ID.
- You cannot allocate a maintenance channel to a shared session.
- You cannot prefix `ORA_` to a channel name. RMAN reserves channel names beginning with the `ORA_` prefix for its own use.

Keywords and Parameters

Syntax Element	Description
<code>DEVICE TYPE =</code> <i>deviceSpecifier</i>	Specifies the type of storage device. See Also: " <i>deviceSpecifier</i> " on page 2-104 Note: If you do not specify the <code>DEVICE TYPE</code> parameter, then you must specify the <code>NAME</code> parameter to identify a particular sequential I/O device. Query the <code>V\$BACKUP_DEVICE</code> view for information about available device types and names.
<i>allocOperandList</i>	Specifies control options for the allocated channel. See Also: " <i>allocOperandList</i> " on page 2-15

Examples

Deleting a Backup Set: Example This example deletes backup sets from tape created more than a week ago:

```
ALLOCATE CHANNEL FOR MAINTENANCE DEVICE TYPE sbt;
DELETE NOPROMPT BACKUP OF DATABASE COMPLETED BEFORE 'SYSDATE-7';
```

Crosschecking Archived Logs: Example This example crosschecks all archived logs on disk (by using the preconfigured disk channel) and tape. If the logs are not found, then RMAN marks them as `EXPIRED` in the repository:

```
ALLOCATE CHANNEL FOR MAINTENANCE DEVICE TYPE sbt;
CROSSCHECK ARCHIVELOG ALL;
```

Crosschecking on Multiple Nodes of an Oracle Real Application Clusters Configuration: Example In this example, you perform a crosscheck of backups on two nodes of an Oracle Real Application Clusters configuration:

```
RUN
{
  ALLOCATE CHANNEL FOR MAINTENANCE DEVICE TYPE DISK CONNECT 'SYS/change_on_install@inst1';
  ALLOCATE CHANNEL FOR MAINTENANCE DEVICE TYPE DISK CONNECT 'SYS/change_on_install@inst2';
  CROSSCHECK BACKUP;
}
```

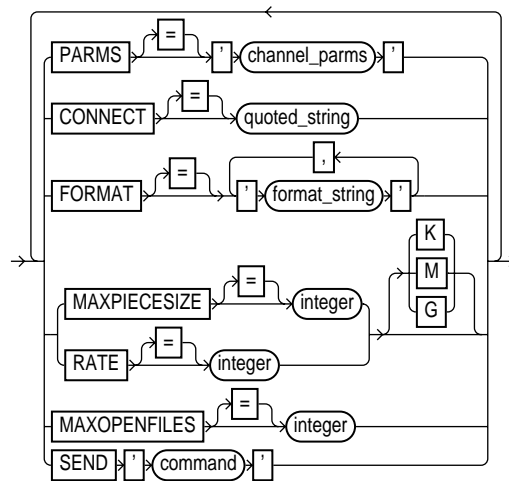
Deleting on Disk and sbt Channels with One Command: Example In this example, you delete a backup from both disk and tape:

```
# back up datafile to disk and tape
BACKUP DEVICE TYPE DISK DATAFILE 1 TAG "weekly_bkup";
BACKUP DEVICE TYPE sbt DATAFILE 1 TAG "weekly_bkup";

# manually allocate sbt channel (disk channel automatically allocated)
ALLOCATE CHANNEL FOR MAINTENANCE DEVICE TYPE sbt;
DELETE BACKUPSET TAG "weekly_bkup";
```

allocOperandList

Syntax



Purpose

A subclause specifying control options on a **channel**, which is a connection between RMAN and a database instance. Specify this clause on the following commands:

- `ALLOCATE CHANNEL`
- `ALLOCATE CHANNEL FOR MAINTENANCE`
- `CONFIGURE`

Keywords and Parameters

Syntax Element	Description
<pre>PARMS = 'channel_parms'</pre>	<p>Specifies parameters for the device to allocate. Do not use this port-specific string if you have specified <code>DEVICE TYPE DISK</code>.</p> <p>If you use <code>PARMS</code> in conjunction with <code>DEVICE TYPE sbt</code>, then you can specify the following environment variables within a quoted string. The maximum length of the quoted string containing <code>channel_parms</code> is 1000 bytes. For example, you can specify:</p> <pre>PARMS="BLKSIZE=16384,ENV=(NSR_SERVER=tape_server, NSR_CLIENT=oracleclnt,NSR_GROUP=oracle_tapes)"</pre> <p>See Also: <i>Oracle9i Recovery Manager User's Guide</i> to learn how Oracle links to media management libraries</p>
<pre>'ENV=(var1=val1, var2=val2,...)'</pre>	<p>Specifies one or more environment variables required by the media management vendor in the Oracle server session corresponding to this RMAN client. Because RMAN is a client program, the <code>ENV</code> parameter can be used to set server session specific variables that perform backup and restore operations on behalf of the RMAN client. For example:</p> <pre>PARMS="ENV=(NSR_SERVER=svr1)"</pre>
<pre>'BLKSIZE=integer'</pre>	<p>Sets the level of granularity for I/O on this channel. The value should be a multiple of the default device block factor (in the Solaris Operating Environment, the value is 16 K). For example:</p> <pre>PARMS="BLKSIZE=16384"</pre>
<pre>'SBT_LIBRARY= lib_name'</pre>	<p>Specifies which media library should be used on this <code>sbt</code> channel. The default library is operating system specific (for example, <code>libobk.so</code> in the Solaris Operating Environment and <code>ORASBT.DLL</code> on Windows NT). For example:</p> <pre>PARMS="SBT_LIBRARY=/oracle/lib/mmiv.so"</pre>
<pre>CONNECT = connectStringSpec</pre>	<p>Specifies a connect string to the database instance where RMAN should conduct the backup or restore operations. Use this parameter to spread the work of backup or restore operations across different instances in an Oracle Real Application Clusters configuration.</p> <p>If you do not specify this parameter, and if you did not specify the <code>AUXILIARY</code> option, then RMAN conducts all operations on the target database instance specified by the command-line parameter or the instance connected to when you issued the <code>CONNECT</code> command. Typically, you should not use the <code>CONNECT</code> parameter in conjunction with the <code>AUXILIARY</code> option.</p> <p>See Also: "connectStringSpec" on page 2-82 and "cmdLine" on page 2-59</p>

Syntax Element	Description
<code>FORMAT = 'format_string'</code>	<p>Specifies the format to use for the names of backup pieces that are created on this channel. If you do not specify a format, RMAN uses %U by default, which guarantees a unique identifier.</p> <p>Because the channels correspond to server sessions on the target database, the <code>FORMAT</code> string must use the conventions of the target host, not the client host. For example, if the RMAN client runs on a Windows machine and the target database runs on a UNIX machine, then the <code>FORMAT</code> string must adhere to the naming conventions of a UNIX file system or raw device.</p> <p>You can specify up to four <code>FORMAT</code> strings. RMAN uses the second, third, and fourth values only when <code>BACKUP COPIES</code>, <code>SET BACKUP COPIES</code>, or <code>CONFIGURE . . . BACKUP COPIES</code> is in effect. When choosing which format to use for each backup piece, RMAN uses the first format value for copy 1, the second format value for copy 2, and so forth. If the number of format values exceeds the number of copies, then the extra formats are not used. If the number of format values is less than the number of copies, then RMAN reuses the format values, starting with the first one.</p> <p>This parameter is useful if you allocate multiple disk channels and want each channel to write to a different directory. If you specify the <code>FORMAT</code> parameter in the <code>BACKUP</code> command, then it overrides the <code>FORMAT</code> parameter specified in <code>CONFIGURE CHANNEL</code> or <code>ALLOCATE CHANNEL</code>.</p> <p>See Also: "BACKUP" on page 2-26 for available <code>FORMAT</code> parameters</p>
<code>MAXPIECESIZE = integer</code>	<p>Specifies the maximum size of each backup piece created on this channel. Specify the size in bytes, kilobytes (K), megabytes (M), or gigabytes (G). The default setting is in bytes and is rounded down into kilobytes. For example, if you set <code>MAXPIECESIZE</code> to 5000, RMAN sets the maximum piece size at 4 kilobytes, which is the lower kilobyte boundary of 5000 bytes.</p>
<code>RATE = integer</code>	<p>Sets the maximum number of bytes (default), kilobytes (K), megabytes (M), or gigabytes (G) that RMAN reads on this channel. This parameter sets an upper limit for bytes read so that RMAN does not consume too much disk bandwidth and degrade performance. Note that decimals are not permitted in the <i>integer</i> value.</p>
<code>MAXOPENFILES = integer</code>	<p>Controls the maximum number of input files that a <code>BACKUP</code> command can have open at any given time (the default is 8). Use this parameter to prevent "Too many open files" error messages when backing up a large number of files into a single backup set.</p>
<code>SEND 'command'</code>	<p>Sends a vendor-specific command string to all allocated channels.</p> <p>See Also: Your media manager documentation to see whether this feature is supported</p>

Examples

Configuring an Automatic Channel: Example This example configures a persistent disk channel:

```
CONFIGURE CHANNEL DEVICE TYPE DISK RATE = 1500K FORMAT = '?/oradata/bkup_%U';
```

Configuring a Channel for a Backup: Example This example manually allocates an sbt channel and then runs a whole database backup:

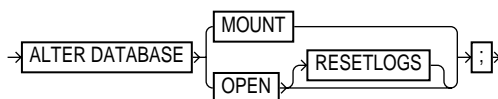
```
RUN
{
  ALLOCATE CHANNEL c1 DEVICE TYPE sbt MAXOPENFILES 20 RATE 2M MAXPIECESIZE 800M;
  BACKUP DATABASE;
}
```

Allocating a Channel for a Backup: Example This example configures a default media management library, then makes a database backup by using this library. Then, the example backs up the database again using a different library, then finally makes a third backup using the default library:

```
CONFIGURE DEFAULT DEVICE TYPE TO sbt;
CONFIGURE CHANNEL DEVICE TYPE sbt PARMS="SBT_LIBRARY=?/lib/mm_lib1.so";
BACKUP DATABASE;
RUN
{
  ALLOCATE CHANNEL c1 DEVICE TYPE sbt PARMS="SBT_LIBRARY=?/lib/mm_lib2.so";
  BACKUP DATABASE;
}
BACKUP ARCHIVELOG ALL;
```

ALTER DATABASE

Syntax



Purpose

To mount or open a database.

See Also: *Oracle9i SQL Reference* for ALTER DATABASE syntax

Restrictions and Usage Notes

- Execute this command either within the braces of a [RUN](#) command or at the RMAN prompt.
- The target instance must be started.

Keywords and Parameters

Syntax Element	Description
MOUNT	Mounts the database without opening it. This option is equivalent to the SQL statement ALTER DATABASE MOUNT.
OPEN	Opens the database.
RESETLOGS	Resets the online redo logs to log sequence 1. The RMAN RESETLOGS option is equivalent to the SQL statement ALTER DATABASE OPEN RESETLOGS. If you use a recovery catalog, then RMAN performs an implicit RESET DATABASE after the database is opened to make this new incarnation the current one in the catalog. If you execute the SQL statement ALTER DATABASE OPEN RESETLOGS (not the RMAN command of the same name), then you must manually run the RESET DATABASE command.

Examples

Opening the Database After a Backup: Example This example mounts the database, takes a whole database backup, then opens the database. At the RMAN prompt enter:

```
STARTUP MOUNT;
BACKUP DATABASE;
# now that the backup is complete, open the database.
ALTER DATABASE OPEN;
```

Mounting the Database After Restoring the Control File: Example To restore the control file to its default location when connected to a recovery catalog, enter the following:

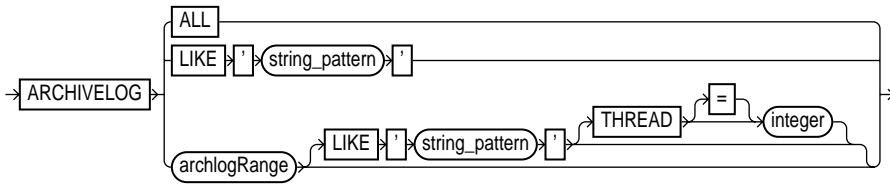
```
STARTUP NOMOUNT;
RESTORE CONTROLFILE;
ALTER DATABASE MOUNT;
# you must run the RECOVER command after restoring a control file even if no datafiles
# require recovery
RECOVER DATABASE;
ALTER DATABASE OPEN RESETLOGS;
# if the database uses locally-managed temporary tablespaces, then you must add tempfiles
# to these tablespaces after restoring a backup control file
SQL " ALTER TABLESPACE temp ADD TEMPFILE '"/oradata/trgt/temp01.dbf' REUSE";
```

Performing RESETLOGS After Incomplete Recovery: Example This example uses a manually allocated channel to perform incomplete recovery and then resets the online redo logs:

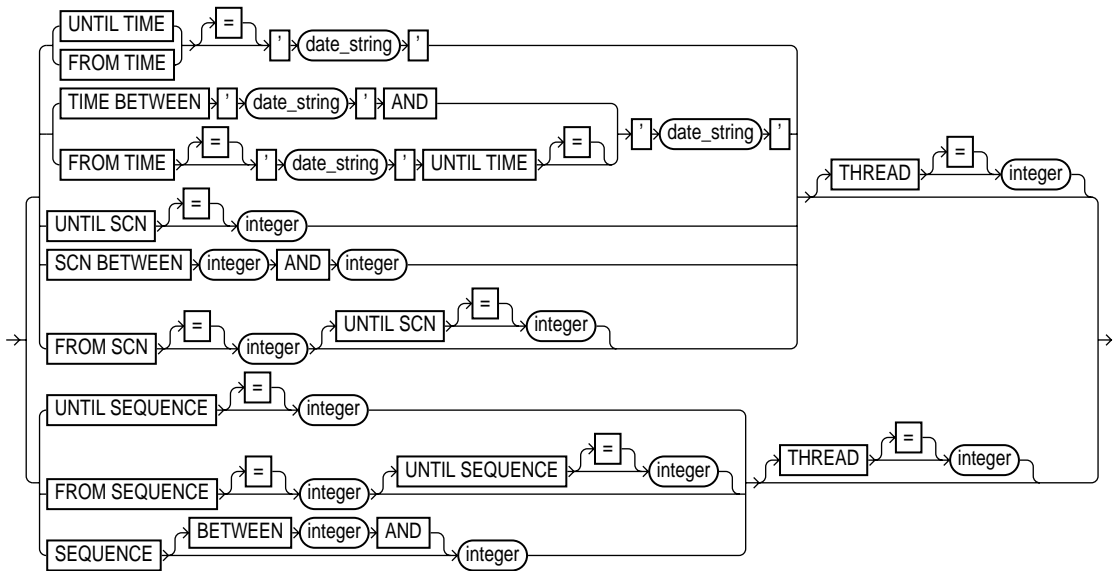
```
RUN
{
  ALLOCATE CHANNEL ch1 DEVICE TYPE sbt;
  SET UNTIL SCN 1024;
  RESTORE DATABASE;
  RECOVER DATABASE;
  ALTER DATABASE OPEN RESETLOGS;
}
```

archivelogRecordSpecifier

Syntax



archlogRange ::=



Purpose

A subclause used to specify an archived log or range of archived redo logs files for use in backup, restore, recovery, and maintenance operations.

When backing up archived redo logs, RMAN can perform archived log failover automatically. RMAN backs up the log when at least one archived log corresponding to a given log sequence number and thread is available. Also, if the copy that RMAN is backing up contains corrupt blocks, then it searches for good copies of these blocks in other copies of the same archived logs.

Specifying a range of archived redo logs does not guarantee that RMAN includes all redo data in the range: for example, the last available archived log file may end before the end of the range, or an archived log file in the range may be missing from all archiving destinations. RMAN includes the archived redo logs it finds and does not issue a warning for missing files.

Note: Query the `V$ARCHIVED_LOG` view or `RC_ARCHIVED_LOG` recovery catalog view to determine the time stamps, SCNs, and log sequence numbers for an archived log. For information on how to use the `NLS_LANG` and `NLS_DATE_FORMAT` environment variables to specify the format for the time, see *Oracle9i Database Reference*.

Keywords and Parameters

Syntax Element	Description
ALL	Specifies all available archived logs.
LIKE ' <i>string_pattern</i> '	Specifies all archived logs that match the specified <i>string_pattern</i> . The same pattern matching characters that are valid in the LIKE operator in the SQL language can be used to match multiple files. See Also: <i>Oracle9i Recovery Manager User's Guide</i> to learn how to make archived log backups in an Oracle Real Application Clusters configuration, and <i>Oracle9i Real Application Clusters Administration</i> for more information about the Oracle Real Application Clusters configuration

Syntax Element	Description
UNTIL TIME = 'date_string'	<p>Specifies the end date for a sequence of archived redo log files. The clause specifies those logs that could be used to recover to the indicated time.</p> <p>The time specified in the string must be formatted according to the Globalization Technology date format specification currently in effect, but can also be any SQL expression with the DATE datatype, such as SYSDATE. The TO_DATE function can be used to specify hard-coded dates that work regardless of the current Globalization Technology settings.</p> <p>If you do not specify the FROM TIME parameter, then the beginning time for the sequence will be the earliest available archived redo log.</p> <p>Note: The UNTIL TIME clause is not the same as the COMPLETED BEFORE clause. UNTIL TIME specifies logs that could be used to <i>recover</i> to the indicated time, whereas COMPLETED BEFORE specifies logs that were <i>created</i> before the indicated time (refer to "completedTimeSpec" on page 2-64).</p> <p>See Also: <i>Oracle9i Database Reference</i> for information on how to use the NLS_LANG and NLS_DATE_FORMAT environment variables to specify the format for the time</p>
FROM TIME = 'date_string'	<p>Specifies the beginning date for a sequence of archived redo log files. The clause specifies those logs that could be used in a recovery starting at the indicated time.</p> <p>The time specified in the string must be formatted according to the Globalization Technology date format specification currently in effect, but can also be any SQL expression with the DATE datatype, such as SYSDATE. The TO_DATE function can be used to specify hard-coded dates that work regardless of the current Globalization Technology settings.</p> <p>If you do not specify the UNTIL TIME parameter, RMAN will include all available log files beginning with the date specified in the FROM TIME parameter. Use the V\$ARCHIVED_LOG data dictionary view to determine the time stamps for the first and last entries in a log file.</p> <p>Note: The FROM TIME clause is not the same as the COMPLETED AFTER clause. FROM TIME specifies logs that could be used to <i>recover</i> starting at the indicated time, whereas COMPLETED AFTER specifies logs that were <i>created</i> after the indicated time (refer to "completedTimeSpec" on page 2-64).</p> <p>See Also: <i>Oracle9i Database Reference</i> for information on how to use the NLS_LANG and NLS_DATE_FORMAT environment variables to specify the format for the time</p>
TIME BETWEEN 'date_string' AND 'date_string'	<p>Specifies a range of times. Note that TIME BETWEEN 'date_string' AND 'date_string' is exactly equivalent to FROM TIME 'date_string' UNTIL TIME 'date_string'.</p>

Syntax Element	Description
UNTIL SCN = <i>integer</i>	Specifies the ending SCN for a sequence of archived redo log files. If you do not specify the FROM SCN parameter, then RMAN will start with the earliest available archived log.
SCN BETWEEN <i>integer</i> AND <i>integer</i>	Specifies a range of SCNs. SCN BETWEEN <i>integer1</i> AND <i>integer2</i> is exactly equivalent to FROM SCN <i>integer1</i> UNTIL SCN <i>integer2</i> .
FROM SCN = <i>integer</i>	Specifies the beginning SCN for a sequence of archived redo log files. If you do not specify the UNTIL SCN parameter, RMAN will include all available log files beginning with SCN specified in the from SCN parameter.
UNTIL SEQUENCE = <i>integer</i>	Specifies the terminating log sequence number for a sequence of archived redo log files. If you do not specify the FROM SEQUENCE parameter, RMAN uses the lowest available log sequence number to begin the sequence.
FROM SEQUENCE <i>integer</i>	Specifies the beginning log sequence number for a sequence of archived redo log files. If you do not specify the UNTIL SEQUENCE parameter, RMAN will include all available log files beginning with log sequence number specified in the FROM SEQUENCE parameter. Note: You can specify all log sequence numbers in a thread by using the following syntax, where <i>thread_number</i> is an integer referring to the thread: ... ARCHIVELOG FROM SEQUENCE 0 THREAD <i>thread_number</i>
SEQUENCE <i>integer</i>	Specifies either a single log sequence number or a range of sequence numbers. Specifies a single log sequence number.
BETWEEN <i>integer</i> AND <i>integer</i>	Specifies a range of log sequence numbers. SEQUENCE BETWEEN <i>integer1</i> AND <i>integer2</i> is exactly equivalent to FROM SEQUENCE <i>integer1</i> UNTIL SEQUENCE <i>integer2</i> .
THREAD = <i>integer</i>	Specifies the thread containing the archived redo log files you wish to include. You only need to specify this parameter when running the database in an Oracle Real Application Clusters configuration. THREAD is only valid when SEQUENCE is also specified. Note also that although the SEQUENCE parameter does not require that THREAD be specified, a given log sequence always implies a thread. The thread defaults to 1 if not specified. Query V\$ARCHIVED_LOG to check the thread number for a log.

Examples

Specifying Records by Time: Example This example backs up all logs that could be used to recover to a point one week ago, and then deletes all archived redo logs that were created more than two weeks ago:

```
BACKUP ARCHIVELOG UNTIL TIME 'SYSDATE-7';
DELETE COPY OF ARCHIVELOG ALL COMPLETED BEFORE 'SYSDATE-14';
```

Specifying Records by SCN: Example This example restores backup archived redo log files from tape that fall within a range of SCNs:

```
RESTORE ARCHIVELOG SCN BETWEEN 94097 AND 106245;
```

Specifying a Single Log Sequence Number: Example This example backs up only archived log 30 of thread 1 and then deletes it.

```
BACKUP ARCHIVELOG SEQUENCE 30 DELETE INPUT;
```

Specifying a Range of Records by Log Sequence Number: Example This example backs up all archived logs from sequence 431 to sequence 440 on thread 1 and deletes the archived logs after the backup is complete. If the backup fails, the logs are not deleted.

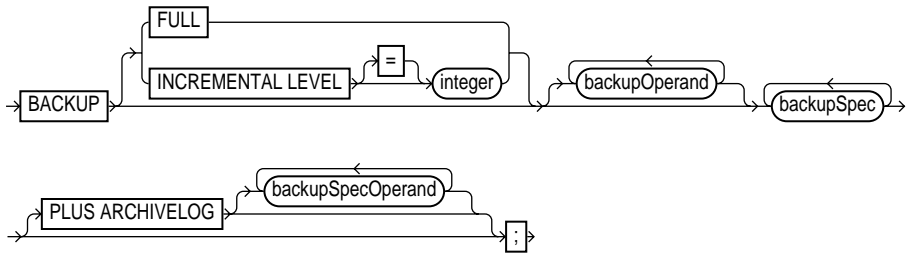
```
RUN
{
  ALLOCATE CHANNEL dev1 DEVICE TYPE sbt;
  BACKUP ARCHIVELOG
    SEQUENCE BETWEEN 31 AND 40 THREAD 1
    # delete original archived redo logs after backup completes
    DELETE INPUT;
}
```

Specifying All Log Sequence Numbers in a Thread This example crosschecks all archived redo logs in thread 1:

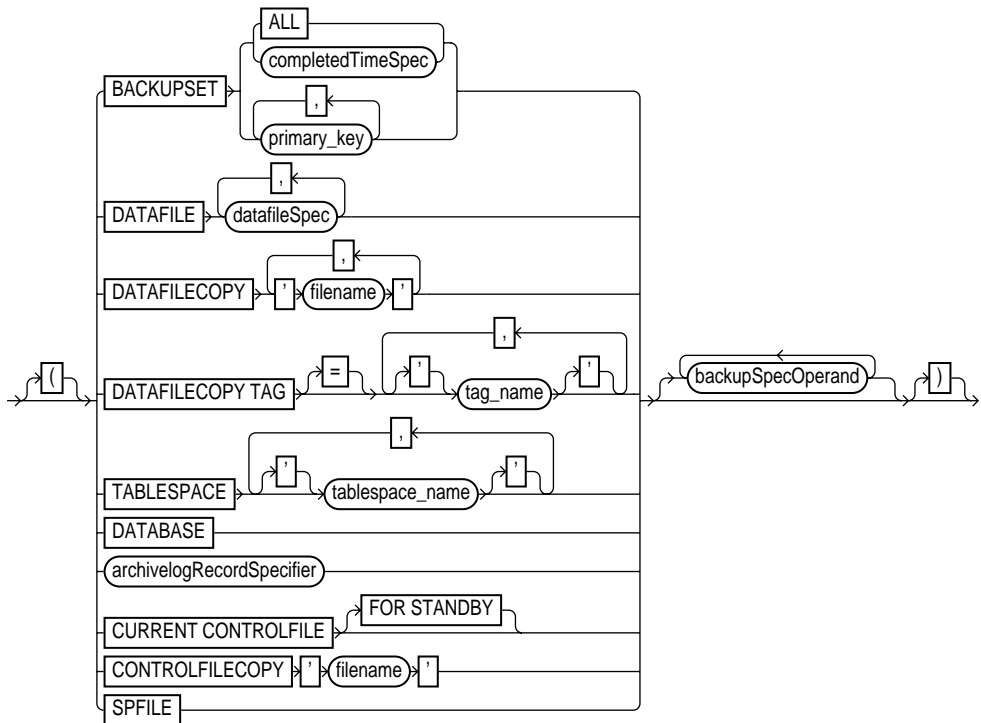
```
CROSSCHECK ARCHIVELOG FROM SEQUENCE 0 THREAD 1;
```

BACKUP

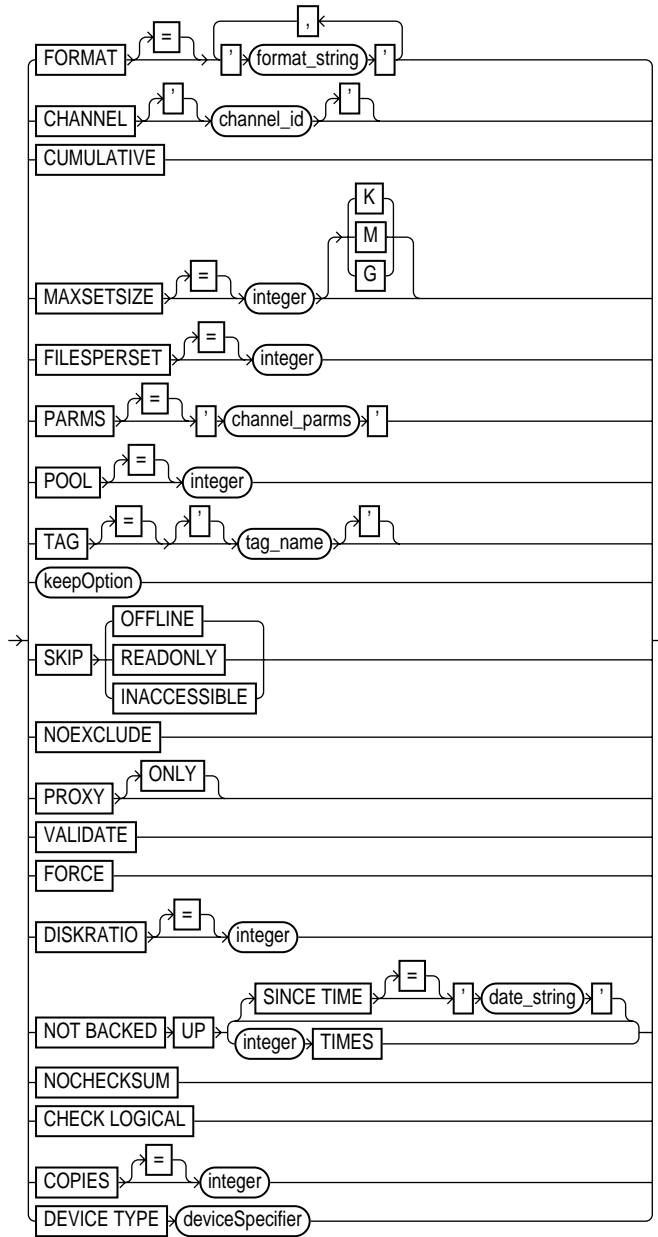
Syntax



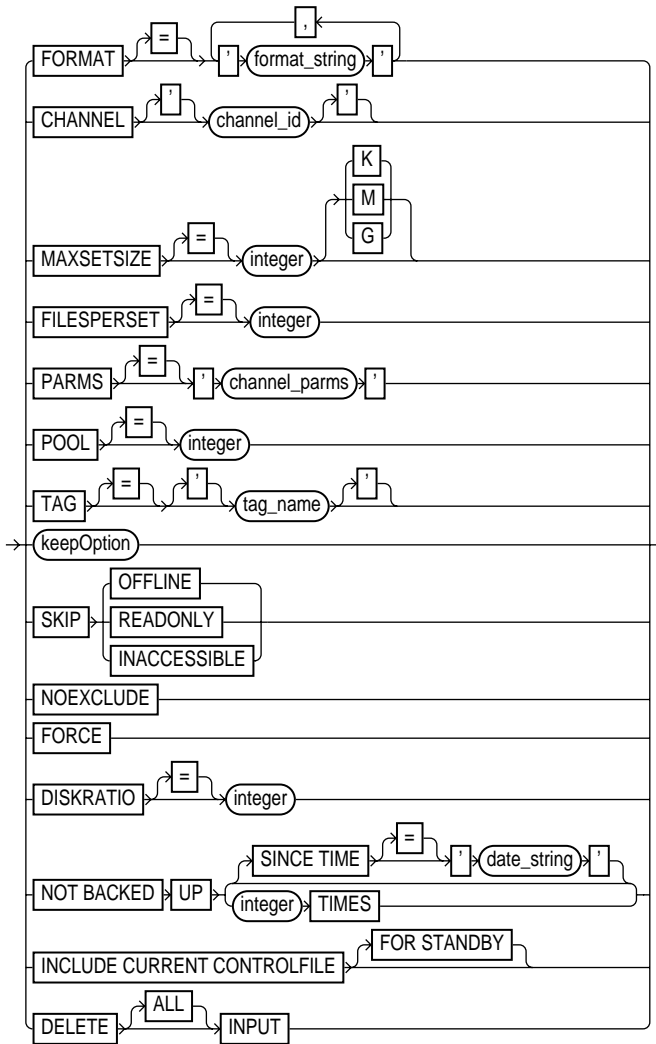
backupSpec ::=



backupOperand ::=



backupSpecOperand ::=



Purpose

To back up a database, tablespace, datafile (current or copy), control file (current or copy), archived log, or backup set. You can back up a target or standby database.

When performing a backup, specify the files that you want to back up. RMAN puts the input files into one or more **backup sets**, which are RMAN-specific logical structures. The backup set is the smallest unit of a backup. RMAN only records backup sets in the repository that complete successfully.

Each backup set contains at least one **backup piece**, which is a physical file containing the backed up data. You can also use the `BACKUP` command to generate a **proxy copy**, which is a backup to a third-party medium in which the entire data transfer is conducted by a media manager.

If `CONFIGURE CONTROLFILE AUTOBACKUP` is set to `ON`, then RMAN automatically backs up the control file after `BACKUP` commands. "`CONFIGURE`" on page 2-66 describes the complete set of circumstances in which autobackups occur.

The `BACKUP` command optimizes backups, that is, does not back up files that are identical to files that are already backed up, when the following conditions are met:

- The `CONFIGURE BACKUP OPTIMIZATION ON` command has been run.
- You run `BACKUP DATABASE`, `BACKUP ARCHIVELOG` with `ALL` or `LIKE` options, or `BACKUP BACKUPSET ALL`.
- You specify a channel of only one device type, that is, you do not mix channels that use different device types.

In a Real Application Clusters configuration, RMAN automatically uses the appropriate channels to back up datafile and control file copies, archived logs, and backup sets. RMAN only performs the backups on channels that can read the files on the file system. For example, if channel 1 connected to instance 1 can read log 1000 on its local file system, but a channel 2 connected to instance 2 cannot read the log on its local file system, then channel 1 backs up the log. Autolocation is automatically enabled when the channels meet any of the following criteria:

- Different `PARMS` settings
- Different `CONNECT` strings

See Also: *Oracle9i Recovery Manager User's Guide* for a conceptual overview of RMAN backup sets, and *Oracle9i Recovery Manager User's Guide* to learn how to back up files

Restrictions and Usage Notes

When using the `BACKUP` command you must:

- Mount or open the target database. RMAN can make an **inconsistent backup** when the database is in `ARCHIVELOG` mode, but you must apply redo logs to make the backups consistent for use in restore operations.
- Use a current control file.
- Manually allocate a channel for each execution of the `BACKUP` command if no automatic channel is configured for the specified device type. If no manual channel is allocated, then RMAN uses the default channels (refer to description of `CONFIGURE` command).
- Note that RMAN comes with a preconfigured `DISK` channel.

Note: Backups that use the disk test API are not supported for production backups (refer to *Oracle9i Recovery Manager User's Guide*). Instead, use the preconfigured `DISK` channel or manually allocate a `DISK` channel.

- Give each backup piece a unique name.
- Back up files onto valid media. If you specify `DEVICE TYPE DISK`, then RMAN will back up to random access disks. You can make a backup on any device that can store an Oracle datafile: in other words, if the statement `CREATE TABLESPACE tablespace_name DATAFILE 'filename'` works, then `'filename'` is a valid backup path name. If you specify `DEVICE TYPE sbt`, then you can back up to any media supported by the media manager.
- Set the `BACKUP_TAPE_IO_SLAVES` initialization parameter to `TRUE` when duplexing backups (this parameter is not required for non-duplexed backups). Otherwise, Oracle signals an error. RMAN configures as many spawned processes as needed for the number of duplexed backup copies you request.

When using the RMAN `BACKUP` command, you *cannot* perform any of the following actions:

- Make a backup (either normal or incremental) in `NOARCHIVELOG` mode when the database is open or is closed after an instance failure or `SHUTDOWN ABORT`. You can only make a `NOARCHIVELOG` backup when the database after a consistent shutdown.
- Stripe a single backup set across multiple channels.

- Stripe a single input file across multiple backup sets.
- Combine archived redo log files and datafiles into a single backup.
- Back up files with different block sizes into the same backup set. RMAN can back up tablespaces with different block sizes, but puts each differently sized datafile into its own backup set.
- Back up locally-managed temporary tablespaces (although you can back up dictionary-managed tablespaces)
- Back up transportable tablespaces that were not made read/write after being transported.
- Use the `DELETE INPUT` option when backing up objects other than datafile copies, archived redo logs, or backup sets.
- Specify the number of backup pieces that should go in a backup set.
- Back up a backup set on tape to disk or on tape to tape.
- Specify the `PLUS ARCHIVELOG` clause on the `BACKUP ARCHIVELOG` command.
- Open a `NOARCHIVELOG` mode database while it is being backed up. If you do, and some data blocks in the files being backed up are modified before being read by the backup session, then the backup is not usable after being restored because it requires recovery.
- Make the length of a backup piece filename longer than the port-specific length limit. If you use a media manager, then the limit is partially governed by the version of the media management API. Vendors that use:
 - SBT 1.1 must support filenames up to 14 characters, but may support longer filenames.
 - SBT 2.0 must support filenames up to 512 characters, but may support longer filenames.
- Specify the `DEVICE TYPE` option for a device other than `DISK` if you have not already run `CONFIGURE DEVICE TYPE` for this device.
- Manually allocate channels and run `BACKUP` with the `DEVICE TYPE` option.
- Validate the backup of backup sets.

Keywords and Parameters

Syntax Element	Description
FULL	copies all blocks into the backup set, skipping only datafile blocks that have never been used. RMAN makes full backups by default if neither FULL nor INCREMENTAL is specified. The server session does not skip blocks when backing up archived redo logs or control files. A full backup has no effect on subsequent incremental backups, so it is not considered a part of the incremental backup strategy.
INCREMENTAL LEVEL = <i>integer</i>	<p>Copies only those data blocks that have changed since the last incremental <i>integer</i> backup, where <i>integer</i> is any integer from 0 to 4. For example, in a level 2 backup RMAN backs up all blocks used since the most recent level 2, level 1, or level 0 backup. This type of incremental backup is also called a differential backup to distinguish it from a cumulative backup.</p> <p>A level 0 backup must exist as the base backup for an incremental strategy. An incremental backup at level 0 is identical in content to a full backup, but unlike a full backup the level 0 backup is considered a part of the incremental strategy. If no level 0 backup exists when you run a level 1 or higher backup, then RMAN makes a level 0 backup automatically.</p> <p>Oracle performs checks when attempting to create an incremental backup at a level greater than 0. These checks ensure that the incremental backup is usable by a subsequent RECOVER command. Among the checks performed are:</p> <ul style="list-style-type: none"> ■ A level 0 backup set must exist, or level 0 datafile copies must exist for each datafile in the BACKUP command. These backup sets must not be marked UNAVAILABLE. If no level 0 backup exists, then RMAN automatically generates one. ■ Sufficient incremental backups taken since the level 0 must exist and be available such that the incremental backup to be created is usable. <p>If you specify INCREMENTAL, then in the <i>backupSpec</i> clause you must set one of the following parameters: DATAFILE, DATAFILECOPY, TABLESPACE, or DATABASE. RMAN does not support incremental backups of control files, archived redo logs, or backup sets.</p> <p>Note: You cannot make inconsistent incremental backups when the database is in NOARCHIVELOG mode. Hence, you cannot generate incremental backups when a NOARCHIVELOG database is open and in use.</p> <p>See Also: "CHANGE" on page 2-55</p>

Syntax Element	Description
PLUS ARCHIVELOG	<p>When you specify PLUS ARCHIVELOG, RMAN performs these steps:</p> <ol style="list-style-type: none"> 1. Runs an ALTER SYSTEM ARCHIVE LOG CURRENT statement. 2. Runs the BACKUP ARCHIVELOG ALL command. Note that if backup optimization is enabled, then RMAN only backs up logs that have not yet been backed up. 3. Backs up the files specified in the BACKUP command. 4. Runs an ALTER SYSTEM ARCHIVE LOG CURRENT statement. 5. Backs up any remaining archived redo logs. <p>The <i>backupSpecPlus</i> clause includes the same options as the <i>backupSpec</i> clause.</p>
<i>backupSpec</i>	<p>A BACKUP specification list contains a list of one or more <i>backupSpec</i> clauses. A <i>backupSpec</i> clause minimally contains a list of one or more objects to be backed up.</p> <p>Each <i>backupSpec</i> clause generates one or more backup sets. A <i>backupSpec</i> clause generates multiple backup sets if the number of datafiles specified in or implied by its list of objects exceeds the FILESPERSET limit.</p>
BACKUPSET	<p>Backs up either ALL backup sets or backup sets specified by <i>primary_key</i> or completion time. Use this parameter in conjunction with the DEVICE TYPE <i>sbt</i> clause to back up all backups on disk to tape. You cannot back up from tape to tape or from tape to disk: only from disk to disk or disk to tape.</p> <p>Note if you specify the DELETE INPUT option, then RMAN deletes all copies of the backup set that exist on disk. For example, if you duplexed a backup to 4 locations, then RMAN deletes all 4 backup sets. The ALL option is redundant, that is, it does not add any functionality.</p> <p>RMAN performs backup set failover when backing up backup sets. RMAN searches for all available backup copies when the copy that it is trying to back up is corrupted or missing. This behavior is similar to RMAN's behavior when backing up archived logs that exist in multiple archiving destinations.</p> <p>Note: You can duplex backups of backup sets by using BACKUP COPIES and SET BACKUP COPIES.</p> <p>See Also: "completedTimeSpec" on page 2-64</p>
DATAFILE <i>datafileSpec</i>	<p>Specifies a list of one or more datafiles. Refer to description of BACKUP DATABASE for RMAN behavior when datafile 1 is backed up.</p> <p>See Also: "datafileSpec" on page 2-96</p>
DATAFILECOPY ' <i>filename</i> '	<p>Specifies the filenames of one or more datafile image copies.</p>

Syntax Element	Description
DATAFILECOPY TAG = <i>tag_name</i>	Specifies a list of one or more datafile copies, identified by tag. If multiple datafile copies with this tag exist, then Oracle backs up only the most current datafile copy of any particular datafile. Tags are not case sensitive.
TABLESPACE <i>tablespace_name</i>	Specifies the names of one or more tablespaces. RMAN backs up all datafiles that are currently part of the tablespaces. This keyword is provided merely as a convenience; Oracle translates the tablespace name internally into a list of datafiles.
DATABASE	Creates a backup set containing all datafiles in the database. RMAN can include only datafiles and control files in the backup set; it cannot include archived redo logs. If a backup includes datafile 1, and if <code>CONFIGURE CONTROLFILE AUTOBACKUP</code> is <code>OFF</code> , then RMAN automatically includes the control file in the backup set. If the instance is started with a server parameter file, then RMAN also includes this parameter file in the backup. If a backup includes datafile 1, and if <code>CONTROLFILE AUTOBACKUP</code> is <code>ON</code> , then RMAN does <i>not</i> automatically include the control file in the datafile backup set. Instead, RMAN generates a separate control file autobackup piece. If the instance is started with a server parameter file, then RMAN includes this parameter file in the autobackup piece. Note: To force RMAN to include the current control file in the backup set when <code>CONTROLFILE AUTOBACKUP</code> is <code>ON</code> , specify the <code>INCLUDE CURRENT CONTROLFILE</code> clause.
<i>archivelogRecordSpecifier</i> clause	Specifies a range of archived redo logs to be backed up. RMAN does not signal an error if the command finds no logs to back up, because this situation probably exists because no new logs were generated after the previous <code>BACKUP ARCHIVELOG ALL DELETE INPUT</code> command. If you specify <code>BACKUP ARCHIVELOG ALL</code> , then RMAN backs up exactly one copy of each distinct log sequence number. For example, if you archive logs to multiple destinations, RMAN backs up <i>one</i> copy of each log sequence number—not each archived copy of each log sequence number. For other commands, such as <code>DELETE, ALL</code> does refer to every log, even duplicate log sequences. See Also: " archivelogRecordSpecifier " on page 2-21 for syntax, and <i>Oracle9i Recovery Manager User's Guide</i> for explanations of backup failover for logs and automatic log switching
CURRENT CONTROLFILE	Specifies the current control file. If you specify <code>FOR STANDBY</code> , then RMAN generates a backup of the control file that is usable during creation of a standby database. The backup contains only the standby control file. Note: You cannot assign a tag to a backup of the current control file.

Syntax Element	Description
CONTROLFILECOPY 'filename'	<p>Specifies the filename of a control file copy. The control file copy can be:</p> <ul style="list-style-type: none"> ■ A copy of a normal control file (that is, not a standby control file) created with the COPY CURRENT CONTROLFILE command or the SQL statement ALTER DATABASE BACKUP CONTROLFILE TO '...' ■ A standby control file copy created with the COPY STANDBY CONTROLFILE command or the SQL statement ALTER DATABASE CREATE STANDBY CONTROLFILE <p>RMAN inspects the header of the control file copy to determine whether it is a standby or nonstandby control file.</p>
SPFILE	<p>Backs up the server parameter file currently used by the database. RMAN cannot back up other copies of the server parameter file, and cannot back up the server parameter file when the instance was started with an initialization parameter file. RMAN cannot make incremental backups of the SPFILE.</p>
backupOperand	<p>Specifies various options for the BACKUP command. Most of the options for this clause are shared with "backupSpecOperand" on page 2-42.</p>
FORMAT = 'format_string'	<p>Specifies a filename to use for the backup piece. Any name that is legal as a sequential filename on the platform is allowed, if each backup piece has a unique name. If backing up to disk, then any legal disk filename is allowed, provided it is unique. If you do not specify the FORMAT parameter, RMAN stores the backup pieces in a port-specific directory (?/dbs on UNIX).</p> <p>You can specify up to four FORMAT strings. RMAN uses the second, third, and fourth values only when BACKUP COPIES, SET BACKUP COPIES, or CONFIGURE ... BACKUP COPIES is in effect. When choosing the format for each backup piece, RMAN uses the first format value for copy 1, the second format value for copy 2, and so on. If the number of format values exceeds the number of copies, then the extra formats are not used. If the number of format values is less than the number of copies, then RMAN reuses the format values, starting with the first one.</p> <p>Specify FORMAT in any of these places, listed in order of precedence:</p> <ol style="list-style-type: none"> 1. The backupSpec clause 2. The BACKUP command 3. The ALLOCATE CHANNEL command 4. The CONFIGURE CHANNEL command <p>If specified in more than one of these places, then RMAN searches for the FORMAT parameter in the order shown.</p> <p>Note: The entire FORMAT string is processed in a port-specific manner by the target instance to derive the final backup piece name. The following substitution variables are available in FORMAT strings to aid in generating unique filenames. The formatting of this information varies by platform.</p>

Syntax Element	Description
%c	Specifies the copy number of the backup piece within a set of duplexed backup pieces. If you did not duplex a backup, then this variable is 1 for backup sets and 0 for proxy copies. If one of these commands is enabled, then the variable shows the copy number. The maximum value for %c is 256.
%d	Specifies the name of the database.
%D	Specifies the current day of the month from the Gregorian calendar in format DD.
%F	Combines the DBID, day, month, year, and sequence into a unique and repeatable generated name.
%M	Specifies the month in the Gregorian calendar in format MM.
%n	Specifies the name of the database, padded on the right with x characters to a total length of eight characters. For example, if the prod1 is the database name, then the padded name is prod1xxx.
%p	Specifies the piece number within the backup set. This value starts at 1 for each backup set and is incremented by 1 as each backup piece is created. Note: If you specify PROXY, then the %p variable must be included in the FORMAT string either explicitly or implicitly within %U.
%s	Specifies the backup set number. This number is a counter in the control file that is incremented for each backup set. The counter value starts at 1 and is unique for the lifetime of the control file. If you restore a backup control file, then duplicate values can result. Also, CREATE CONTROLFILE initializes the counter back to 1.
%t	Specifies the backup set time stamp, which is a 4-byte value derived as the number of seconds elapsed since a fixed reference time. The combination of %s and %t can be used to form a unique name for the backup set.
%T	Specifies the year, month, and day in the Gregorian calendar in this format: YYYYMMDD.
%u	Specifies an 8-character name constituted by compressed representations of the backup set number and the time the backup set was created.
%U	Specifies a convenient shorthand for %u_%p_%c that guarantees uniqueness in generated backup filenames. If you do not specify a format, then RMAN uses %U by default.
%Y	Specifies the year in this format: YYYY.
%%	Specifies the '%' character. For example, %%Y translates to the string %Y.

Syntax Element	Description
<code>CHANNEL <i>channel_id</i></code>	<p>Specifies the case-sensitive name of a channel to use when creating backup sets. Use any name that is meaningful, for example <i>ch1</i> or <i>dev1</i>. Oracle uses the channel ID to report I/O errors. If you do not set this parameter, then RMAN dynamically assigns the backup sets to any available channels during execution.</p> <p>Note: You can also specify this parameter in the <i>backupSpec</i> clause.</p>
<code>CUMULATIVE</code>	<p>Copies the data blocks used since the most recent backup at level <i>n</i>-1 or lower, where <i>n</i> is an integer from 1 to 4. For example, in a cumulative level 2 backup RMAN backs up all blocks used since the most recent level 1 or level 0 backup.</p>
<code>MAXSETSIZE = <i>integer</i></code>	<p>Specifies a maximum size for a backup set in bytes (default), kilobytes (K), megabytes (M), and gigabytes (G). Thus, to limit a backup set to 3 MB, specify <code>MAXSETSIZE = 3M</code>. The default size is in bytes, rounded down from kilobytes. For example, <code>MAXSETSIZE = 3000</code> is rounded down to 2 KB (2048 bytes). The minimum value must be greater than or equal to the database block size.</p> <p>RMAN limits all backup sets to this size. Use <code>MAXSETSIZE</code> to configure backup sets so that each fits on one tape rather than spans multiple tapes. Otherwise, if one tape of a multivolume backup set fails, then you lose the data on all the tapes rather than just one.</p> <p>When files are located on one disk and do not create an I/O distribution problem, the <code>MAXSETSIZE</code> parameter is easier to use than <code>FILESERSET</code>. The <code>FILESERSET</code> parameter is more useful for managing I/O distribution for backups of files on multiple disks.</p> <p>Note: Because <code>FILESERSET</code> has a default, both <code>MAXSETSIZE</code> and <code>FILESERSET</code> take effect when <code>MAXSETSIZE</code> is set. RMAN attempts to limit the size in bytes of the backup sets according to the <code>MAXSETSIZE</code> parameter, treating <code>FILESERSET</code> as an upper limit for the number of files to include in each set.</p>
<code>FILESERSET = <i>integer</i></code>	<p>Specifies the maximum number of input files in each backup set. If you set <code>FILESERSET = n</code>, then RMAN never includes more than <i>n</i> files in a backup set. The default for <code>FILESERSET</code> is the lesser of these two values: 64, number of input files divided by the number of channels. For example, if you back up 100 datafiles by using two channels, RMAN sets <code>FILESERSET</code> to 50.</p> <p>RMAN always attempts to create enough backup sets so that all allocated channels have work to do. An exception to the rule occurs when there are more channels than files to back up. For example, if RMAN backs up two datafiles when three channels are allocated and <code>FILESERSET = 1</code>, then one channel is necessarily idle.</p> <p>See Also: The <code>MAXSETSIZE</code> parameter, which limits backup sets by total bytes rather than number of files included</p>

Syntax Element	Description
<code>PARMS = 'channel_parms'</code>	Specifies a quoted string containing operating system-specific information. RMAN passes the string to the operating system-dependent layer each time a backup piece is created. Currently, no <code>PARMS</code> settings are available when specified in the <code>BACKUP</code> command, although you can specify <code>PARMS</code> in the <code>ALLOCATE CHANNEL</code> or <code>CONFIGURE CHANNEL</code> command.
<code>POOL = integer</code>	Specifies the media pool in which the backup should be stored. Consult your media management documentation to see whether the <code>POOL</code> option is supported.
<code>TAG tag_name</code>	<p>Creates a user-specified tag name for a backup set. The tag name is not case sensitive.</p> <p>If you do not specify a tag name, then by default RMAN creates a tag for backups and copies (except for control file autobackups) in the format <code>TAGYYYYMMDDTHHMMSS</code>, where <code>YYYY</code> is the year, <code>MM</code> is the month, <code>DD</code> is the day, <code>HH</code> is the hour (in 24-hour format), <code>MM</code> is the minutes, and <code>SS</code> is the seconds. For example, a backup of datafile 1 may receive the tag <code>TAG20020208T133437</code>. The date and time refer to when RMAN started the backup. If multiple backup sets are created by one <code>BACKUP</code> command each backup piece is assigned the same default tag.</p> <p>A tag applies to each backup piece in a given copy of a backup set. For example, if you run <code>BACKUP COPIES 1 DATABASE TAG TUE_PM</code>, then only one copy of the backup set exists and each piece in the backup set has tag <code>TUE_PM</code>. Assume that this backup set has primary key 1234. If you then run <code>BACKUP BACKUPSET 1234 TAG WED_PM</code>, then the first copy of the backup set has tag <code>TUE_PM</code> and the second copy of the backup set has tag <code>WED_PM</code>.</p> <p>Typically, a tag is a meaningful name such as <code>MONDAY_EVENING_BACKUP</code> or <code>WEEKLY_FULL_BACKUP</code>. Tags must be 30 characters or less. Tags are reusable, so that backup set 100 can have the tag <code>MONDAY_EVENING_BACKUP</code> one week while backup set 105 has the same tag the next week.</p> <p>You can also specify the tag at the <i>backupSpec</i> level. If you specify the tag at:</p> <ul style="list-style-type: none"> ■ The command level, then all backup sets created by the command have the tag. ■ The <i>backupSpec</i> level, then backup sets created as a result of different backup specifications can have different tags. ■ Both levels, then the tag in the <i>backupSpec</i> takes precedence.
<code>keepOption</code>	<p>Overrides any configured retention policy for this backup so that the backup is not considered obsolete. You can use <code>CHANGE</code> to alter the keep status. Note that you must be connected to a recovery catalog when you specify <code>KEEP FOREVER</code>.</p> <p>See Also: "<code>keepOption</code>" on page 2-120</p>

Syntax Element	Description
SKIP	<p>Excludes datafiles or archived redo logs from the backup set according to the criteria specified by the following keywords.</p> <p>Note: You can also specify this option in the <i>backupSpec</i> clause.</p>
OFFLINE	Specifies that offline datafiles should be excluded from the backup set.
READONLY	Specifies that read-only datafiles should be excluded from the backup set.
INACCESSIBLE	<p>Specifies that datafiles or archived redo logs that cannot be read due to I/O errors should be excluded from the backup set.</p> <p>A datafile is only considered inaccessible if it cannot be read. Some offline datafiles can still be read because they still exist on disk. Others have been deleted or moved and so cannot be read, making them inaccessible.</p>
NOEXCLUDE	<p>When specified on <code>BACKUP DATABASE</code> command, RMAN backs up all tablespaces, including any for which a <code>CONFIGURE EXCLUDE</code> command has been entered. This option does not override <code>SKIP OFFLINE</code> or <code>SKIP READONLY</code>.</p>
PROXY	<p>Backs up the specified files by means of the proxy copy functionality, which gives the media management software control over the data transfer between storage devices and the Oracle datafiles on disk. The media manager—not RMAN—decides how and when to move data.</p> <p>When you run <code>BACKUP</code> with the <code>PROXY</code> option, RMAN performs these steps:</p> <ol style="list-style-type: none"> 1. Searches for a channel of the specified device type that is proxy-capable. If no such channel is found, then RMAN issues a warning. RMAN attempts a conventional (that is, nonproxy) backup of the specified files unless the <code>ONLY</code> option is specified, in which case it does not attempt a conventional backup. 2. If RMAN locates a proxy-capable channel, it calls the media manager to check whether it can proxy copy the file. If the media manager cannot proxy copy the file, then RMAN uses conventional backup sets to back up the file. <p>Note: If you specify <code>PROXY</code>, then the <code>%P</code> variable must be included in the <code>FORMAT</code> string either explicitly or implicitly within <code>%U</code>.</p>
ONLY	Causes Oracle to issue an error message when it cannot proxy copy rather than creating conventional backup sets.
VALIDATE	<p>Causes RMAN to scan the specified files and verify their contents. RMAN creates no output files. Use this command periodically to check for physical and logical errors in database files.</p> <p>Note: You cannot validate backups of backup sets.</p>

Syntax Element	Description
FORCE	<p>Causes RMAN to ignore backup optimization. In other words, even if <code>CONFIGURE BACKUP OPTIMIZATION</code> is set to <code>ON</code>, RMAN backs up all specified files.</p> <p>Note: You can also specify this option in the <i>backupSpecOperand</i> clause.</p>
DISKRATIO = <i>integer</i>	<p>Directs RMAN to populate each backup set with datafiles from at least <i>integer</i> disks. This parameter is only enabled when you are backing up datafiles or control files, and when the operating system can give RMAN disk contention and node affinity information. To manually disable this feature, set <code>DISKRATIO=0</code>.</p> <p>For example, assume that datafiles are distributed across 10 disks. If the disks supply data at 10 bytes/second, and if the tape drive requires 50 bytes/second to keep streaming, then set <code>DISKRATIO=5</code> to direct RMAN to include datafiles from at least 5 disks in each backup set.</p> <p>If you set <code>FILESPERSET</code> but not <code>DISKRATIO</code>, then <code>DISKRATIO</code> defaults to the same value as <code>FILESPERSET</code>. If you specify neither parameter, then <code>DISKRATIO</code> defaults to 4. RMAN compares the <code>DISKRATIO</code> value to the actual number of devices involved in the backup and uses the lowest value. For example, if <code>DISKRATIO=4</code> and the datafiles are located on three disks, then RMAN attempts to include datafiles from three disks in each backup set.</p> <p>The <code>DISKRATIO</code> parameter is easier for datafile backups when the datafiles are striped or reside on separate disk spindles and you either:</p> <ul style="list-style-type: none"> ■ Use a high-bandwidth tape drive that requires several datafiles to be multiplexed in order to keep the tape drive streaming ■ Make backups while the database is open and you need to spread the I/O load across several disk spindles to leave bandwidth for online operations <p>Note: Do not spread I/O over more than the minimum number of disks to keep the tape streaming. Otherwise, you increase restore time for a file without increasing performance.</p>
NOT BACKED UP	<p>Backs up only those files (of the files specified on the command) that RMAN has never backed up. This option is a convenient way to back up new files after adding them to the database.</p>

Syntax Element	Description
<p>SINCE TIME = '<i>date_string</i>'</p>	<p>Specifies the date after which RMAN should back up files that have no backups. The <i>date_string</i> is either a date in the current NLS_DATE_FORMAT, or a SQL date expression such as 'SYSDATE-1'. When calculating the number of backups for a file, RMAN only considers backups created on the same device type as the current backup.</p> <p>This option is a convenient way to back up files that were not backed up during a previous failed backup. For example, you back up the database, but the instance fails halfway through. You can restart the backup with the NOT BACKUP UP SINCE TIME clause and avoid backing up those files that you already backed up. This feature is only useful if RMAN generates multiple backup sets during the backup. Thus, set the FILESPERSET parameter to a low value so that RMAN generates multiple backup sets.</p> <p>When determining whether a file has been backed up, the SINCE date is compared with the completion time of the most recent backup. The completion time for a file in a backup set is the completion time of the entire backup set. In other words, all files in the same backup set have the same completion time.</p>
<p><i>integer</i> TIMES</p>	<p>Backs up only those archived logs that have not been backed up at least <i>integer</i> times. When calculating the number of backups for a file, RMAN only considers backups created on the same device type as the current backup.</p> <p>This option is a convenient way to back up archived logs on a specified media (for example, you want to keep at least three copies of each log on tape).</p>
<p>NOCHECKSUM</p>	<p>Suppresses block checksums. A checksum is a number that is computed from the contents of a data block. If the DB_BLOCK_CHECKSUM initialization parameter is true, then Oracle computes a checksum for each block and stores it in the block before writing the block to disk. When Oracle reads the block from disk later, it makes sure that the block generates the same checksum. If it does not, then the block is damaged.</p> <p>Unless you specify the NOCHECKSUM option, Oracle computes a checksum for each block and stores it in the backup. The checksum is verified when restoring from the backup and written to the datafile when restored. If the database is already maintaining block checksums, then this flag has no effect. The checksum is always verified and stored in the backup in this case.</p> <p>See Also: <i>Oracle9i Database Reference</i> for more information about the DB_BLOCK_CHECKSUM initialization parameter</p>

Syntax Element	Description
CHECK LOGICAL	<p>Tests data and index blocks that pass physical corruption checks for logical corruption, for example, corruption of a row piece or index entry. If RMAN finds logical corruption, then it logs the block in the <code>alert.log</code> and server session trace file.</p> <p>If the sum of physical and logical corruptions detected for a file remain below its <code>MAXCORRUPT</code> setting, then the <code>RMAN</code> command completes and Oracle populates <code>V\$DATABASE_BLOCK_CORRUPTION</code> with corrupt block ranges. If <code>MAXCORRUPT</code> is exceeded, then the command terminates without populating the views.</p> <p>Note: For <code>COPY</code> and <code>BACKUP</code> the <code>MAXCORRUPT</code> setting represents the total number of physical and logical corruptions permitted on a file.</p>
<code>COPIES = integer</code>	<p>Sets the number of identical backups (1 - 4) that RMAN should create. The default value is 1. You can specify duplexing on more than one command. The order of precedence is as follows, with settings higher on the list overriding lower settings:</p> <ol style="list-style-type: none"> 1. <code>BACKUP COPIES</code> 2. <code>SET BACKUP COPIES</code> 3. <code>CONFIGURE . . . BACKUP COPIES</code>
DEVICE TYPE <i>deviceSpecifier</i>	<p>Allocates automatic channels for the specified device type only. This option is valid only if you have configured channels and have not manually allocated channels. For example, if you configure disk and tape channels, then configure <code>sbt</code> as the default device type, this command allocates disk channels only:</p> <pre>BACKUP DEVICE TYPE DISK DATABASE;</pre> <p>See Also: "<i>deviceSpecifier</i>" on page 2-104</p>
<i>backupSpecOperand</i>	<p>Specifies a variety of options and parameters that affect the <i>backupSpec</i> clause. Most of the <i>backupSpecOperand</i> options are the same as the <i>backupOperand</i> options. Only the options unique to this clause are included in the following parameter descriptions.</p>
INCLUDE CURRENT CONTROLFILE	<p>creates a snapshot of the current control file and places it into each backup set produced by this clause.</p> <p>If you specify <code>FOR STANDBY</code>, then RMAN creates a backup of the control file usable for creation of a standby database. The backup set includes the standby control file and the object backed up.</p>

Syntax Element	Description
DELETE INPUT	<p>deletes the input files upon successful creation of the backup set. Specify this option only when backing up archived logs, datafile copies, or backup sets. It is equivalent to issuing DELETE for the input files.</p> <p>The ALL option applies only to archived logs. If you run DELETE ALL INPUT, then the command deletes all copies of corresponding archived redo logs or datafile copies that match the selection criteria. For example, if you specify the SEQUENCE <i>n</i> clause, then RMAN deletes all archive logs with same sequence number <i>n</i>, including duplicate archived logs (that is, logs with same log sequence number and thread).</p> <p>Note: The BACKUP ARCHIVELOG command only backs up one copy of each distinct log sequence number, so if the DELETE INPUT option is used <i>without</i> the ALL keyword, RMAN only deletes the copy of the file that it backs up.</p> <p>See Also: "CONNECT" on page 2-80 for information on the effect of recovery catalog compatibility on this command</p>

Examples

Backing Up a Database: Example This example assumes that CONFIGURE CONTROLFILE AUTOBACKUP is OFF. The command backs up all datafiles to tape, as well as the current control file, the server parameter file, and archived logs:

```
BACKUP DATABASE PLUS ARCHIVELOG;
```

Backing Up Tablespaces and Datafiles: Example This command uses two *backupSpec* clauses to back up tablespaces and datafiles and lets RMAN perform automatic parallelization of the backup:

```
RUN
{
  ALLOCATE CHANNEL dev1 DEVICE TYPE DISK FORMAT '/disk1/%U';
  ALLOCATE CHANNEL dev2 DEVICE TYPE DISK FORMAT '/disk2/%U';
  BACKUP
    (TABLESPACE SYSTEM, tools, users, undotbs FILESPERSET 20)
    (DATAFILE 2,4,5);
}
```

Backing Up Multiple Copies of Archived Redo Logs: Example This example backs up the archived redo logs in `%/arch1` to one set of tapes and the logs from

%/arch2 to another set of tapes. This scenario assumes that you have two tape drives available.

```
# channel configuration
CONFIGURE DEVICE TYPE sbt PARALLELISM 2;
CONFIGURE CHANNEL 1 DEVICE TYPE sbt PARMS="ENV=(NSR_SERVER=tape_server_1)";
CONFIGURE CHANNEL 2 DEVICE TYPE sbt PARMS="ENV=(NSR_SERVER=tape_server_2)";

# backup
BACKUP
  FILESPERSET=20
  FORMAT='AL_%d_%t_%s_%p'
  (ARCHIVELOG LIKE '%/arch1/%' CHANNEL ORA_SBT_TAPE_1)
  (ARCHIVELOG LIKE '%/arch2/%' CHANNEL ORA_SBT_TAPE_2);
```

Backing Up Multiple Copies of Archived Logs and Deleting the Input: Example

This example assumes that you have two archive destinations set: %/arch1 and %/arch2. The command backs up one log for each unique sequence number and then deletes all logs from both archiving directories.

```
BACKUP ARCHIVELOG LIKE '%/arch/%' DELETE ALL INPUT;
```

Backing Up Backup Sets to Tape: Example In this example, you want to keep recent backup sets on disk and older backup sets on tape. You do not want backup sets to exist on disk and tape simultaneously. Hence, you execute this command to back up older backups created more than two weeks ago to tape and then delete the input backup pieces:

```
BACKUP DEVICE TYPE sbt BACKUPSET COMPLETED BEFORE 'SYSDATE-14'
  DELETE INPUT;
```

Specifying DEVICE TYPE on the BACKUP Command: Example This example configures DISK as the default device type, then backs up the server parameter file and all archived logs to tape:

```
CONFIGURE DEFAULT DEVICE TYPE TO DISK;
BACKUP DEVICE TYPE sbt SPFILE ARCHIVELOG ALL;
```

Performing a Cumulative Incremental Backup: Example This example backs up all blocks changed in the database since the most recent level 0 or level 1 backup:

```
BACKUP INCREMENTAL LEVEL 2 CUMULATIVE SKIP INACCESSIBLE DATABASE;
```

Duplexing a Backup Set: Example This example duplexes a backup of datafile 1 (which includes the current control file and server parameter file) to separate disks:

```
BACKUP DEVICE TYPE DISK
  COPIES 2 DATAFILE 1
  FORMAT '/disk1/df1_%U', '/disk2/df1_%U';
```

Specifying How Channels Divide Workload: Example This example parallelizes a backup operation by specifying which channels should back up which files and to which location:

```
RUN
{
  ALLOCATE CHANNEL ch1 DEVICE TYPE sbt PARMS="ENV=(NSR_SERVER=tape_server_1)";
  ALLOCATE CHANNEL ch2 DEVICE TYPE DISK RATE 1500K;
  ALLOCATE CHANNEL ch3 DEVICE TYPE sbt PARMS="ENV=(NSR_SERVER=tape_server_2)";
  BACKUP
    (DATAFILE 1,2,3,4          # channel ch1 backs up datafiles to tape drive #1
     CHANNEL ch1)
    (CONTROLFILECOPY '/oracle/copy/cf.f'
     CHANNEL ch2)             # channel ch2 backs up control file copy to disk
    (ARCHIVELOG FROM TIME 'SYSDATE-14'
     CHANNEL ch3);           # channel ch3 backs up archived redo logs to tape drive #2
}
```

Performing an Oracle Real Application Clusters Backup: Example The following script distributes datafile and archived log backups across two nodes in an Oracle Real Application Clusters environment:

```
RUN
{
  ALLOCATE CHANNEL node_1 DEVICE TYPE sbt
    CONNECT 'SYS/sys_pwd@node_1'
    PARMS 'ENV=(DSMO_NODE=KDFDWD01_ORACLE)'
    FORMAT '%d_set%s_%t_piece%p';
  SET LIMIT CHANNEL node_1 KBYTES=1000000;
  SET COMMAND ID TO 'node_1';
  ALLOCATE CHANNEL node_2 DEVICE TYPE sbt
    CONNECT 'SYS/sys_pwd@node_2'
    PARMS 'ENV=(DSMO_NODE=KDFDWD01_ORACLE)'
    FORMAT '%d_set%s_%t_piece%p';
  SET LIMIT CHANNEL node_2 KBYTES=1000000;
  SET COMMAND ID TO 'node_2';
  BACKUP FILESPERSET 1
    (TABLESPACE SYSTEM, rbs, data1, data2
     CHANNEL node_1)
    (TABLESPACE temp, reccat, data3, data4
     CHANNEL node_2);
  BACKUP FILESPERSET 20
```

```
(ARCHIVELOG UNTIL TIME 'SYSDATE' LIKE '%/node1/arch/%'
DELETE ALL INPUT # deletes all logs that match LIKE criteria
CHANNEL node_1);
(ARCHIVELOG UNTIL TIME 'SYSDATE' LIKE '%/node2/arch/%'
DELETE ALL INPUT # deletes all logs that match LIKE criteria
CHANNEL node_2);
}
```

Creating a Control File for a Standby Database: Example This example creates a backup of the current control file that can be used to create a standby database:

```
BACKUP CURRENT CONTROLFILE FOR STANDBY;
```

Checking for Corruption: Example This example backs up datafile 3 and specifies that no more than two blocks with corruption should be tolerated:

```
RUN
{
  SET MAXCORRUPT FOR DATAFILE 3 TO 2;
  BACKUP CHECK LOGICAL
    DATAFILE 3;
}
```

Creating a Long-Term Backup: Example This example creates a consistent backup of the database and server parameter file that is exempt from the retention policy. The command instructs RMAN to keep the backup for the next year, but not to keep the archived logs necessary to recover it:

```
SHUTDOWN IMMEDIATE;
STARTUP MOUNT;
BACKUP DATABASE
  KEEP UNTIL TIME 'SYSDATE+365'
  NOLOGS;
ALTER DATABASE OPEN;
```

Backing Up Files That Need Backups: Example This example backs up all datafiles that have not been backed up to tape in the last month, and then backs up all archived logs that do not have at least two backups on tape:

```
BACKUP DEVICE TYPE sbt DATABASE NOT BACKED UP SINCE TIME 'SYSDATE-31';
BACKUP DEVICE TYPE sbt ARCHIVELOG ALL NOT BACKED UP 2 TIMES;
```

Backing Up a Noncurrent Server Parameter File: Example The following UNIX shell script backs up an older of the server parameter file. RMAN can only back up the copy of the server parameter file currently in use by the target database:

```
#!/usr/bin/tcsh
# create temporary initialization parameter file that points to old SPFILE
echo "SPFILE=/tmp/old_spfile.ora" > /tmp/initTEMP.ora
rman TARGET / <<EOF
# start database with old SPFILE
STARTUP FORCE PFILE=/tmp/initTEMP.ora
# backup old SPFILE
BACKUP SPFILE TAG "old_spfile.bak";
# restart database with current SPFILE
STARTUP FORCE;
EXIT
EOF
```


Typically, block corruption is reported in Oracle error messages in trace files. Block-level data loss usually results from:

- I/O errors causing minor data loss
- Memory corruptions that get flushed to disk

You need to specify the datafile number and block number or the tablespace and data block address (DBA) when executing the `BLOCKRECOVER` command, or use the `CORRUPTION LIST` keyword to recover all blocks reported in the `V$DATABASE_BLOCK_CORRUPTION` view.

See Also: *Oracle9i Recovery Manager User's Guide* for concepts and *Oracle9i Recovery Manager User's Guide* for procedures

Restrictions and Usage Notes

- This command is available only in the Enterprise Edition.
- The target database must be mounted or open. You do *not* have to take a datafile offline if you are performing block media recovery on it.
- You can only perform complete media recovery of individual blocks. Point-in-time recovery of individual data blocks is not supported.
- You can only perform block media recovery on corrupt blocks.
- Blocks marked media corrupt are not accessible until recovery completes.
- You cannot perform block media recovery when using a backup control file.
- You cannot use proxy backups to perform block media recovery. If the only backups that you have are proxy backups, then you can restore them to a nondefault location on disk, which causes RMAN to view the restored files as datafile copies. You can then use the datafile copies for block media recovery.
- You must have a full backup of the file containing the corrupt blocks: block media recovery cannot use incremental backups.
- Block media recovery cannot survive a missing or inaccessible archived log, although it can sometimes survive missing or inaccessible records (refer to *Oracle9i Recovery Manager User's Guide*).
- The datafile header block (block 1) cannot be recovered.
- You cannot perform block media recovery in `NOARCHIVELOG` mode.

Keywords and Parameters

Syntax Element	Description
DEVICE TYPE <i>deviceSpecifier</i>	Specifies the device type for the backup used in the block recovery. See Also: "deviceSpecifier" on page 2-104
<i>bmrBlockSpec</i>	Specifies the data blocks that require recovery.
DATAFILE <i>datafileSpec</i>	Specifies a list of one or more datafiles that contain blocks requiring recovery. See Also: "datafileSpec" on page 2-96
BLOCK <i>integer</i>	Specifies the block number of the block requiring media recovery. Typically, the block number is obtained from error message output.
TABLESPACE <i>tablespace_name</i>	Specifies the tablespace name or number containing the corrupt blocks.
DBA <i>integer</i>	Specifies the data block address (DBA) of the corrupt block.
CORRUPTION LIST	Recovers all blocks listed in the V\$DATABASE_BLOCK_CORRUPTION view. This view displays blocks marked corrupt by the most recent BACKUP (with or without the VALIDATE option), VALIDATE, or COPY command. The following types of corruption result in rows added to this view: <ul style="list-style-type: none"> ■ Physical corruption (sometimes called media corruption). Oracle does not recognize the block at all: the checksum is invalid, the block contains all zeros, or the header and footer of the block do not match. Physical corruption checking is on by default, and can be turned off with the NOCHECKSUM option. ■ Logical corruption. The block has a valid checksum, the header and footer match, and so forth, but the contents are logically inconsistent. Logical corruption checking is off by default, and can be turned on with the CHECK LOGICAL option.
<i>bmrOption</i>	Specifies various restore options relating to the block recovery.
FROM BACKUPSET	indicates that only backup set should be restored.
FROM DATAFILECOPY	indicates that only datafile image copies should be restored.
FROM TAG = 'tag_name'	indicates that only the copy of the backup set with the specified tag should be restored. Tag names are not case sensitive. See Also: "BACKUP" on page 2-26 to learn how a tag is applied to a copy of a backup set
RESTORE <i>untilClause</i>	Specifies that only backups and copies created before the specified time, SCN, or log sequence number should be restored. See Also: "untilClause" on page 2-216

Examples

Recovering a Group of Corrupt Blocks: Example This example recovers corrupt blocks in three datafiles:

```
BLOCKRECOVER DATAFILE 2 BLOCK 12, 13 DATAFILE 3 BLOCK 5, 98, 99 DATAFILE 4 BLOCK 19;
```

Limiting Block Media Recovery by Type of Restore: Example The following example recovers a series of blocks and restores only from datafile copies:

```
RUN
{
  BLOCKRECOVER DATAFILE 3 BLOCK 2,3,4,5 TABLESPACE sales DBA 4194405, 4194409, 4194412
  FROM DATAFILECOPY;
}
```

Limiting Block Media Recovery by Backup Tag: Example This example recovers blocks and restores only from the backup set with the tag `weekly_backup`:

```
BLOCKRECOVER TABLESPACE SYSTEM DBA 4194404, 4194405 FROM TAG "weekly_backup";
```

Limiting Block Media Recovery by Time: Example The following example recovers two blocks in the `SYSTEM` tablespace. It restores only from backups that could be used to recover the database to a point two days ago:

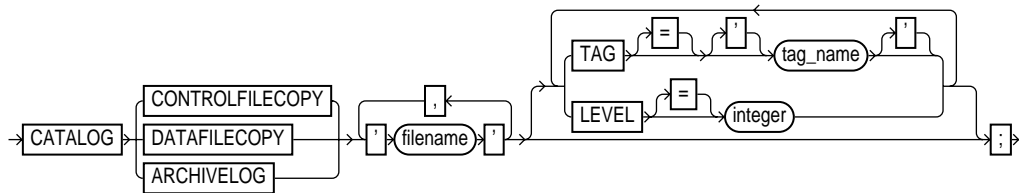
```
BLOCKRECOVER TABLESPACE SYSTEM DBA 4194404, 4194405 RESTORE UNTIL TIME 'SYSDATE-2';
```

Repairing All Block Corruption in the Database: Example The following example runs a backup validation to populate `V$DATABASE_BLOCK_CORRUPTION`, then repairs any corrupt blocks recorded in the view:

```
BACKUP VALIDATE DATABASE;
BLOCKRECOVER CORRUPTION LIST;
```

CATALOG

Syntax



Purpose

Use the `CATALOG` command to:

- Add metadata about a user-managed datafile, control file, or archived log copy to the recovery catalog and control file.
- Record a datafile copy as a level 0 backup in the RMAN repository, which enables you to use it as part of an incremental backup strategy.
- Record the existence of user-managed copies of Oracle release 8.0 or later databases created before RMAN was installed.
- Record the existence of the last user-managed datafile copies made after the final shutdown in Oracle version 7 and before running the migration utility.

See Also: *Oracle9i Recovery Manager User's Guide* to learn how to manage target database records stored in the catalog

Restrictions and Usage Notes

- Execute the `CATALOG` command within the braces of a `RUN` command or at the RMAN prompt.
- You must be connected to the target database, and the database must be mounted or open. If you use a recovery catalog, then the catalog must be open.
- For a user-managed backup to be cataloged, it must be:
 - Accessible on disk.
 - A complete image copy of a single file.

- A datafile, control file, or archived log backup. If the datafile backup is inconsistent, then it must have been created with the `BEGIN BACKUP/END BACKUP` statements.

RMAN treats all user-managed backups as image copies. Note that during cataloging, RMAN does not check whether the file was correctly copied by the operating system utility: it just checks the header.

You *cannot* use `CATALOG` to perform the following operations:

- Catalog any datafile copies that were created in Oracle7 unless they were made after the final consistent shutdown in Oracle7 and before running the migration utility, or were made of a tablespace that was offline normal or read-only at the time of the migration. In other words, no Oracle7 redo must need to be applied to the backups to roll them forward.
- Recatalog backup pieces or backup sets. If you run `CHANGE . . . UNCATALOG` against an RMAN backup, then it is permanently unusable by RMAN.
- Rename a backup piece or move a backup piece from one place to another.

Keywords and Parameters

Syntax Element	Description
<code>CONTROLFILECOPY 'filename'</code>	Specifies the filename of a control file copy to be added to or updated in the recovery catalog and control file. The control file copy can be: <ul style="list-style-type: none"> ■ A copy of a normal control file (that is, not a standby control file) created with the RMAN command <code>COPY CURRENT CONTROLFILE</code> command or the SQL statement <code>ALTER DATABASE CREATE CONTROLFILE</code> ■ A standby control file copy created with the RMAN command <code>COPY STANDBY CONTROLFILE</code> or the SQL statement <code>ALTER DATABASE CREATE STANDBY CONTROLFILE</code>
<code>DATAFILECOPY 'filename'</code>	Specifies the filename of a datafile copy to be added to or updated in the recovery catalog and control file.
<code>ARCHIVELOG 'filename'</code>	Specifies the filename of an archived log to be added to or updated in the recovery catalog and control file.
<code>TAG = tag_name</code>	Specifies the tag that will be assigned to this copy in the control file and recovery catalog, for example, <code>SUNDAY_PM_BACKUP</code> . Note that tag names are not case sensitive.
<code>LEVEL = integer</code>	Indicates that the file copy should be recorded as an incremental backup at the specified level. Only level 0 is supported. You can perform incremental backups by using a datafile copy as the base level 0 backup.

Examples

Cataloging an Archived Redo Log: Example This example assumes that you made operating system copies of archived logs or transferred them from another location, and then added them to the RMAN repository:

```
CATALOG ARCHIVELOG '?/oradata/archive1_30.dbf', '?/oradata/archive1_31.dbf',  
                '?/oradata/archive1_32.dbf';
```

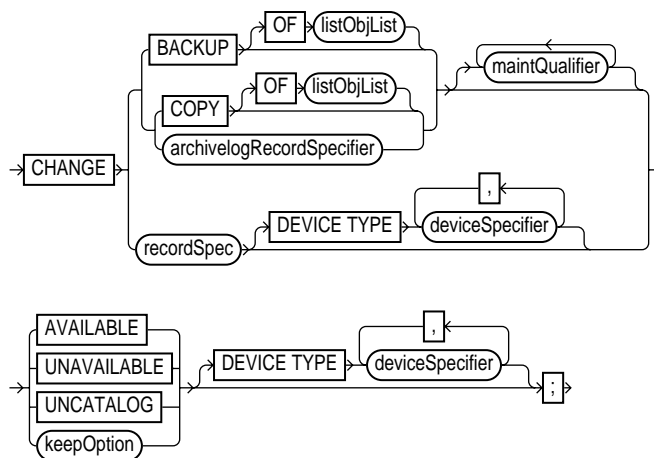
Cataloging a File Copy as an Incremental Backup: Example The following example catalogs datafile copy users01.bak as an incremental level 0 backup:

```
CATALOG DATAFILECOPY '?/oradata/users01.bak' LEVEL 0;
```

Note that you can create datafile copies both with the RMAN COPY command and by using operating system utilities in conjunction with ALTER TABLESPACE BEGIN/END BACKUP.

CHANGE

Syntax



Purpose

To make the following changes:

- Change the status of backups, copies, and archived logs in the repository to `AVAILABLE` or `UNAVAILABLE`. This feature is useful when a previously unavailable file is made available again, or you do not want a specific backup or copy to be eligible to be restored but also do not want to delete it.
- Remove catalog records for backups and copies, and update the corresponding records in the target control file to status `DELETED`. This feature is useful when you remove a file by using an operating system command rather than the RMAN `CHANGE` command, and want to remove its repository record as well.
- Specify that a backup or copy should either abide by the currently configured retention policy or be exempt from it.

See Also: *Oracle9i Recovery Manager User's Guide* to change the availability status of a backup or copy, and *Oracle9i Recovery Manager User's Guide* to learn how to make a backup or copy exempt from the retention policy

Restrictions and Usage Notes

- Execute the `CHANGE` command within the braces of a `RUN` command or at the `RMAN` prompt.
- The target instance must be started.
- Use `CHANGE` only on files that are recorded in the `RMAN` repository and belong to the current database incarnation.
- The `KEEP FOREVER` clause requires use of a recovery catalog.
- The only `CHANGE` command that requires either a manual or automatic maintenance channel is the `CHANGE . . . AVAILABLE` command. However, a maintenance channel is not required when `CHANGE . . . AVAILABLE` is used with a file that is disk only (that is, an `ARCHIVELOG`, `DATAFILECOPY`, or `CONTROLFILECOPY`).

If you use `CHANGE . . . AVAILABLE` on files that are not disk-only, and have objects created on device types that are not configured for automatic channels, then issue manual maintenance commands on these channels. For example, if you created a backup on an `sbt` channel, but have only a `DISK` channel automatically configured, then you must manually allocate an `sbt` channel before `CHANGE . . . AVAILABLE` can operate on the backup.

Keywords and Parameters

To obtain the primary keys of the records whose status you want to change, run a `LIST` command or query the recovery catalog views.

Syntax Element	Description
BACKUP	Operates on the specified backups: backup sets, backup pieces, and proxy copies. If you do not specify an option for <code>BACKUP</code> , then <code>CHANGE BACKUP</code> operates on all backups recorded in the repository. Note: Use the <code>KEY</code> column of the <code>LIST</code> output to obtain the primary key usable in the <code>CHANGE</code> command.
COPY	Operates on datafile copies, archived redo logs, and image copies of archived redo logs. If you do not specify an option for <code>COPY</code> , then <code>CHANGE COPY</code> operates on all copies recorded in the repository.
OF <i>listObjList</i>	restricts the list of objects operated on to the object type specified in the <i>listObjList</i> clause. If you do not specify an object, then <code>CHANGE</code> defaults to all copies. Note that <code>CHANGE COPY OF DATABASE</code> includes datafiles but not control files. See Also: " <code>listObjList</code> " on page 2-140

Syntax Element	Description
<i>maintQualifier</i>	restricts the command based on the specified options. See Also: "maintQualifier" on page 2-142
<i>archivelogRecordSpecifier</i>	Specifies archived logs. See "archivelogRecordSpecifier" on page 2-21.
<i>recordSpec</i>	Specifies the object whose availability status you are changing. See "recordSpec" on page 2-149.
AVAILABLE	Changes the status of a backup or copy to AVAILABLE in the repository. View the status in the LIST output or recovery catalog views.
UNAVAILABLE	Changes the status of a backup or copy to UNAVAILABLE in the repository. View the status in the LIST output or recovery catalog views. This option is provided for cases when the file cannot be found or has migrated offsite. RMAN does not use a file that is marked UNAVAILABLE in a RESTORE or RECOVER command. If the file is later found or returns to the main site, then use the AVAILABLE option to update its status.
UNCATALOG	Removes references to a datafile copy or archived redo log (but not a backup piece or backup set) from the recovery catalog, and updates records in the target control file to status DELETED. The CHANGE . . . UNCATALOG command does not touch physical backups and copies. Use this command to notify RMAN when a file is deleted by some means other than a DELETE command. Caution: If you resynchronize from a backup control file, or upgrade the recovery catalog, then uncataloged records can sometimes reappear in the catalog metadata.
DEVICE TYPE <i>deviceSpecifier</i>	Executes the CHANGE for the specified device type only (see "deviceSpecifier" on page 2-104). This option is valid only if you have configured automatic channels and have not manually allocated channels. For example, if you run CHANGE UNCATALOG . . . DEVICE TYPE DISK, then RMAN only uncatalogs files on disk.
<i>keepOption</i>	Changes the exemption status of a backup or copy in relation to the configured retention policy. For example, specify CHANGE . . . NOKEEP to make a backup that is currently exempt from the retention policy eligible for OBSOLETE status. Note: You can also specify KEEP in the <i>backupSpec</i> clause. See Also: "keepOption" on page 2-120

Examples

Updating Backups to Status UNAVAILABLE: Example This example changes the status of backup set 100 as well as all backups of server parameter files created more than one day ago to UNAVAILABLE. You do not need to allocate a maintenance channel:

```
CHANGE BACKUPSET 100 UNAVAILABLE;  
CHANGE BACKUP OF SPFILE COMPLETED BEFORE 'SYSDATE-1' UNAVAILABLE;
```

Uncataloging and Cataloging Archived Logs: Example In this example, you first move all archived logs to a temporary directory and then uncatalog them:

```
HOST 'mv $ORACLE_HOME/oradata/trgt/arch/* /tmp/arch';  
CHANGE COPY OF ARCHIVELOG ALL UNCATALOG;  
EXIT;
```

From the UNIX command shell you use Perl to create a command file that contains a CATALOG command for each archived log:

```
% perl5 -e '@files=</tmp/arch/*>; foreach (@files) \  
? {print "CATALOG ARCHIVELOG '$_'\n";}' > $HOME/rman.cmd
```

Finally, you start RMAN and run the command file to catalog the relocated logs:

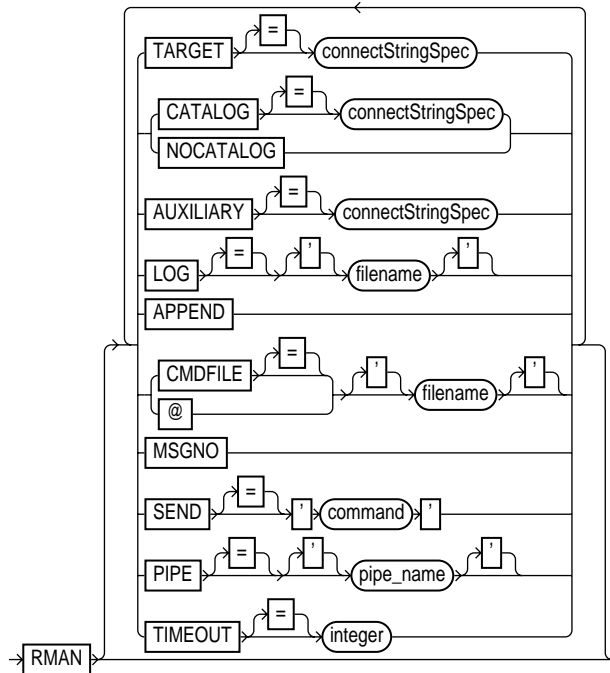
```
% rman TARGET / @$HOME/rman.cmd
```

Changing the Retention Status of a Backup Set: Example This example, which requires a recovery catalog, changes an ordinary backup into a long-term backup:

```
CHANGE BACKUP TAG 'consistent_db_bkup'  
KEEP FOREVER NOLOGS;
```

cmdLine

Syntax



Purpose

To start RMAN from the operating system command line. Use these arguments to:

- Connect to the target, recovery catalog, or auxiliary database.

Note: On some platforms, you may not want to connect at the operating system command line because the password is visible to other users on the system. The [CONNECT](#) command is an alternative method that avoids this problem.

- Specify that you are using RMAN without a recovery catalog.

- Run a command file, which is a user-defined file containing RMAN commands.
- Specify the file in which RMAN records the results of processed commands.
- Append output to the existing RMAN log file.
- Send a command to the media manager.
- Cause RMAN to print message numbers in the RMAN output.

If you start RMAN without specifying either `CATALOG` or `NOCATALOG` on the command line, then RMAN makes no connection to a repository. If you run a command that requires the repository, and if no `CONNECT CATALOG` command has been issued yet, then RMAN automatically connects in the default `NOCATALOG` mode. After that point, the `CONNECT CATALOG` command is not valid in the session.

See Also: *Oracle9i Recovery Manager User's Guide* to learn how to connect RMAN to database instances

Restrictions and Usage Notes

Use these arguments at the operating system command line rather than at the RMAN prompt.

Keywords and Parameters

Syntax Element	Description
TARGET = <i>connectStringSpec</i>	Specifies a connect string to the target database, for example, TARGET SYS/change_on_install@inst1. See Also: " <i>connectStringSpec</i> " on page 2-82
CATALOG = <i>connectStringSpec</i>	Specifies a connect string to the database containing the recovery catalog, for example, CATALOG rman/rman@inst2. See Also: " <i>connectStringSpec</i> " on page 2-82
NOCATALOG	Indicates that you are using RMAN without a recovery catalog. Note: If you do not specify either <code>CATALOG</code> or <code>NOCATALOG</code> on the command line, then RMAN defaults to <code>NOCATALOG</code> mode when it requires a repository connection (assuming that you have not issued <code>CONNECT CATALOG</code>).
AUXILIARY = <i>connectStringSpec</i>	Specifies a connect string to an auxiliary database, for example, AUXILIARY SYS/change_on_install@dupdb. See Also: " <i>connectStringSpec</i> " on page 2-82

Syntax Element	Description
LOG = 'filename'	<p>Specifies the file where Recovery Manager will record RMAN output, that is, the commands that were processed and their results. If you do not specify this argument, then RMAN writes its message log file to standard output.</p> <p>The LOG parameter does not cause RMAN to terminate if the specified file cannot be opened. Instead, the output goes to standard output.</p>
APPEND	<p>Causes new output to be appended to the end of the message log file. If you do not specify this parameter, and if a file with the same name as the message log file already exists, then RMAN overwrites it.</p>
CMDFILE = 'filename'	<p>Parses and compiles all RMAN commands in a file and then sequentially executes each command in the file. RMAN exits if it encounters a syntax error during the parse phase or if it encounters a runtime error during the execution phase. If no errors are found, then RMAN exits after the job completes.</p> <p>If the first character of the filename is alphabetic, then you can omit the quotes around the filename. The contents of the command file should be identical to commands entered at the RMAN prompt. For example, the following file contents connect to a target database and recovery catalog <code>r_cat</code>, then back up the target:</p> <pre>CONNECT TARGET; CONNECT CATALOG rman/rman@r_cat; BACKUP DATABASE PLUS ARCHIVELOG;</pre> <p>Note: If you run a command file at the RMAN prompt rather than as an option on the operating system command line, then RMAN does <i>not</i> run the file as a single job. RMAN reads each line sequentially and executes it, only exiting when it reaches the last line of the script.</p>
@filename	<p>Equivalent to CMDFILE.</p>
MSGNO	<p>Causes RMAN to print message numbers, that is, RMAN-xxxxx, for the output of all commands. By default, RMAN does not print the RMAN-xxxxx prefix.</p>
SEND = 'command'	<p>Sends a vendor-specific command string to all allocated channels.</p> <p>See Also: Your media management documentation to determine whether this feature is supported, and "SEND" on page 2-193</p>

Syntax Element	Description
<code>PIPE = 'pipe_name'</code>	<p>Invokes the RMAN pipe interface. RMAN uses two public pipes: one for receiving commands and the other for sending output. The names of the pipes are derived from the value of the <code>PIPE</code> parameter. For example, you can invoke the RMAN pipe interface with the following options: <code>PIPE rpi TARGET SYS/pwd@tdb</code>.</p> <p>RMAN opens the following pipes in the target database:</p> <ul style="list-style-type: none"> ■ <code>ORA\$RMAN_RPI_IN</code>, which RMAN uses to receive user commands ■ <code>ORA\$RMAN_RPI_OUT</code>, which RMAN uses to send all output <p>All messages on both the input and output pipes are of type <code>VARCHAR2</code>.</p> <p>See Also: <i>Oracle9i Recovery Manager User's Guide</i> to learn how to pass commands to RMAN through a pipe</p>
<code>TIMEOUT = integer</code>	<p>Causes RMAN to exit automatically if it does not receive input from an input pipe within <i>integer</i> seconds. The <code>PIPE</code> parameter must be specified when using <code>TIMEOUT</code>.</p> <p>See Also: <i>Oracle9i Recovery Manager User's Guide</i> to learn how to pass commands to RMAN through a pipe</p>

Examples

Connecting Without a Recovery Catalog: Example This example connects to the target database `prod1` without a recovery catalog:

```
% rman TARGET SYS/oracle@inst1 NOCATALOG
```

Connecting in Default NOCATALOG mode: Example This example connects to the target database `prod1` without specifying catalog options. Because `CONNECT CATALOG` is not run at the RMAN prompt, RMAN connects in default `NOCATALOG` mode when the first command requiring a repository connection is run:

```
% rman
RMAN> CONNECT TARGET
RMAN> BACKUP DATABASE;
```

Connecting to an Auxiliary Instance: Example This example connects to target database `prod1`, recovery catalog database `rcat`, and auxiliary instance `aux1`:

```
% rman TARGET SYS/sys_pwd@prod1 CATALOG rman/rman@rcat AUXILIARY sys/aux_pwd@aux1
```

Specifying a Command File: Example This example connects to the target database `prod1` and the recovery catalog database `rcat`, and then runs the command file `b_whole_10.rcv`:

```
% rman TARGET SYS/sys_pwd@prod1 CATALOG rman/rman@rcat @'/oracle/dbs/b_whole_10.rcv'
```

Specifying a Message Log in Append Mode: Example This example connects to the target database `prod1` without a recovery catalog and then specifies that RMAN should append messages to the message log:

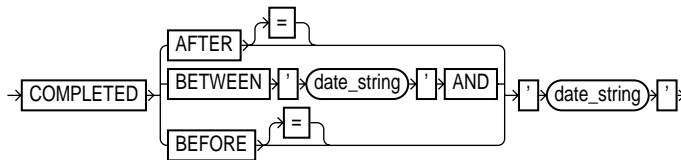
```
% rman TARGET / NOCATALOG LOG = $ORACLE_HOME/dbs/log/msglog.f APPEND
```

Invoking the RMAN Pipe Interface: Example This example invokes the RMAN pipe `newpipe` with a 90 second timeout option:

```
% rman PIPE newpipe TARGET SYS/oracle@inst1 TIMEOUT = 90
```

completedTimeSpec

Syntax



Purpose

A subclause that specifies when a backup or copy completed.

Restrictions and Usage Notes

All date strings must be either:

- Formatted according to the Global Technology date format specification currently in effect.
- Created by a SQL expression that returns a DATE value, as in the following examples:
 - 'SYSDATE-30'
 - TO_DATE('09/30/2000 08:00:00', 'MM/DD/YY HH24:MI:SS').

The TO_DATE technique specifies dates independently of the current Global Technology environment variable settings.

Note: In Oracle8i, the FROM/UNTIL . . . TIME syntax in the LIST, CROSSCHECK, and DELETE commands was replaced with *completedTimeSpec*. If you are adapting an RMAN script from before Oracle8i for use in the current release, then you must update these commands for the script to work correctly.

Keywords and Parameters

Syntax Element	Description
AFTER ' <i>date_string</i> '	Specifies the time after which the backup was completed.
BETWEEN ' <i>date_string</i> ' AND ' <i>date_string</i> '	Specifies a time range during which the backup was completed. Note that BETWEEN ' <i>date1</i> ' AND ' <i>date2</i> ' is exactly equivalent to AFTER ' <i>date1</i> ' BEFORE ' <i>date2</i> '.
BEFORE ' <i>date_string</i> '	Specifies the time before which the backup was completed.

Examples

Crosschecking Backups Within a Time Range: Example This example crosschecks the backup sets of the database made last month:

```
CROSSCHECK BACKUP OF DATABASE COMPLETED BETWEEN 'SYSDATE-62' AND 'SYSDATE-31';
```

Deleting Expired Backups: Example This example deletes expired backup sets of datafile 1 made in the last two weeks:

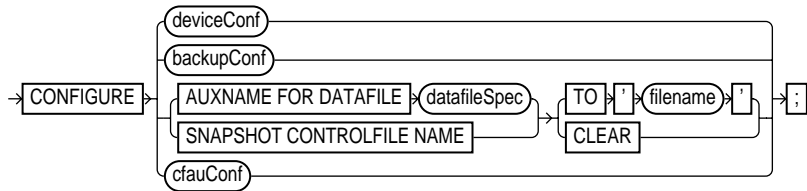
```
DELETE EXPIRED BACKUP OF DATAFILE 1 COMPLETED AFTER 'SYSDATE-14';
```

Listing Copies: Example This example lists image copies of datafile `*/oradata/trgt/users01.dbf` made before September 27, 2001:

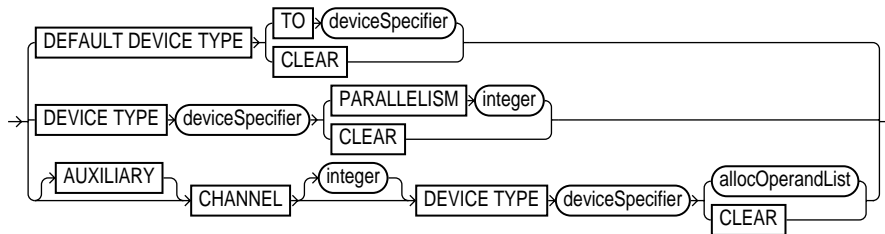
```
LIST COPY OF DATAFILE '*/oradata/trgt/users01.dbf'
  COMPLETED BEFORE '27-SEP-01';
```

CONFIGURE

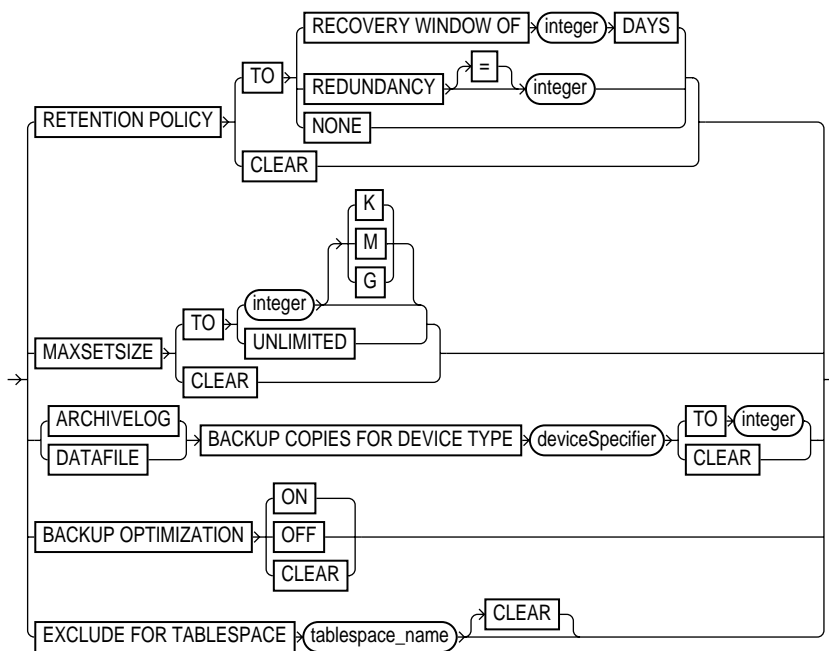
Syntax



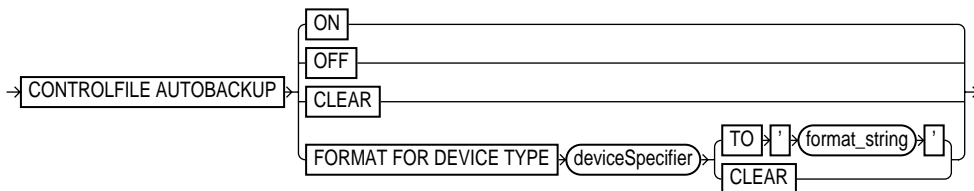
deviceConf ::=



backupConf ::=



cfauConf ::=



Purpose

To configure persistent settings affecting RMAN backup, restore, duplication, and maintenance jobs. These configurations are in effect for any RMAN session until the configuration is cleared or changed.

Use CONFIGURE to set the following:

- An ongoing retention policy that automatically determines which backups and copies are eligible for deletion because they are no longer needed
- The device type (for example, `DISK` or `sbt`) for RMAN jobs
- The default number of channels of each device type that RMAN should allocate for automated backup and restore jobs
- The settings for automatic channels for a specified device type
- The maximum size of backup pieces and sets created on automatic channels
- Backup optimization either `ON` or `OFF`
- The exclusion policy for tablespaces in whole database backups
- The filename of the snapshot control file
- Filenames for files in an auxiliary database
- The control file autobackup feature to `ON` or `OFF`
- The default format for the control file autobackup output files

RMAN uses default settings for `CONFIGURE` options. You can return to the default value for any `CONFIGURE` command by running the same command with the `CLEAR` option.

See Also: *Oracle9i Recovery Manager User's Guide* to learn how to configure the RMAN environment

Restrictions and Usage Notes

- Execute this command at the RMAN prompt.
- The target database must be mounted or open.
- Channels allocated with `ALLOCATE CHANNEL` override any configured automatic channels.
- RMAN does not simultaneously allocate automatic channels for multiple device types in `BACKUP` and `COPY` jobs.
- To direct backups or restores to specific channels, use the RMAN-generated channel names. If you specify channel numbers in the `CONFIGURE CHANNEL` command, then RMAN uses the same numbers in the system-generated channel names.
- If you configure channels by using the nondefault `CONNECT` or `PARMS` options to create backups or copies, then you must either use the same configured

channels or manually allocate channels with the same options to restore or crosscheck these backups.

- You cannot exclude the `SYSTEM` tablespace from whole database backups.
- The `REDUNDANCY` and `RECOVERY WINDOW` options are mutually exclusive. Only one type of retention policy can be in effect at any time.
- You cannot clear individual parameters when running `CONFIGURE . . . CLEAR`. For example, you can run `CONFIGURE CHANNEL DEVICE TYPE sbt CLEAR` but not `CONFIGURE CHANNEL DEVICE TYPE sbt RATE 1000 CLEAR`.
- The channel number in a manually numbered channel must be less than 255.
- You must specify at least one channel option when running `CONFIGURE CHANNEL`. In other words, you cannot issue a command such as `CONFIGURE CHANNEL 2 DEVICE TYPE DISK`, but you can issue a command such as `CONFIGURE CHANNEL 2 DEVICE TYPE DISK MAXPIECESIZE 2500K`.
- The `CONFIGURE CONTROLFILE AUTOBACKUP FORMAT` format string must include the `%F` substitution variable.

Keywords and Parameters

Syntax Element	Description
<code>DEFAULT DEVICE TYPE TO</code> <i>deviceSpecifier</i>	<p>Specifies the default device type for automatic channels. By default, <code>DISK</code> is the default device type. <code>CLEAR</code> returns the default device type to <code>DISK</code>.</p> <p>By default, the <code>BACKUP</code> and <code>COPY</code> commands only allocate channels of the default device type. For example, if you configure automatic channels for <code>DISK</code> and <code>sbt</code> and set the default device type to <code>DISK</code>, then <code>RMAN</code> only allocates disk channels when you run the <code>BACKUP DATABASE</code> command. You can override this behavior either by manually allocating channels in a <code>RUN</code> command, or by specifying <code>DEVICE TYPE</code> on the <code>BACKUP</code> command itself.</p> <p>The <code>RESTORE</code> command allocates automatic channels of all configured device types, regardless of the default device type. The <code>RESTORE</code> command obeys the <code>PARALLELISM</code> setting for each configured device type.</p>

Syntax Element	Description
<code>DEVICE TYPE</code> <i>deviceSpecifier</i> <code>PARALLELISM integer</code>	<p data-bbox="449 251 1288 338">Configures the device types that are eligible for use in jobs that use automatic channels and sets the degree of channel parallelism. The <code>DISK</code> device type is the default.</p> <p data-bbox="449 347 1288 460">The <code>PARALLELISM</code> parameter specifies the number of automatic channels of the specified device type allocated for RMAN jobs. RMAN always allocates the number of channels specified by <code>PARALLELISM</code>, although it may actually use only a subset of these channels.</p> <p data-bbox="449 468 1288 607">By default, <code>PARALLELISM = 1</code>. Specifying <code>CLEAR</code> for a device type resets its settings to the default. For example, you can set <code>PARALLELISM</code> for disk backups to 3. If you configure automatic channels of type disk and tape, and set the default device type as disk, then RMAN allocates three disk channels when you run <code>BACKUP DATABASE</code> at the RMAN prompt.</p> <p data-bbox="449 616 1288 703">To change the parallelism for a device type to <i>n</i>, run a new <code>CONFIGURE DEVICE TYPE . . . PARALLELISM n</code> command. For example, you can change <code>PARALLELISM</code> to 3 for <code>sbt</code> and then change it to 2 as follows:</p> <pre data-bbox="449 711 899 772">CONFIGURE DEVICE TYPE sbt PARALLELISM 3; CONFIGURE DEVICE TYPE sbt PARALLELISM 2;</pre> <p data-bbox="449 772 1288 859">Note: If you configure <i>n</i> manually numbered channels, the <code>PARALLELISM</code> setting can be greater than or less than <i>n</i>. For example, you can manually number 10 automatic channels and configure <code>PARALLELISM</code> to 2 or 12.</p>

Syntax Element	Description
CHANNEL <i>integer</i> DEVICE TYPE <i>deviceSpecifier</i>	<p>Specifies the standard or AUXILIARY channel that you are configuring or clearing, as well as the device type (DISK or sbt) of the channel. You can either configure a generic channel or specify a channel by number, where <i>integer</i> is less than 255.</p> <p>If you configure a generic channel (that is, if you do not specify a channel number), then RMAN uses the generic settings for every parallelized channel <i>except</i> any channel number that you have explicitly configured. In other words, a generic channel setting specifies options for all channels not configured explicitly.</p> <p>For generic channels of a specified device type, a new command erases previous settings for this device type. Assume that you run these commands:</p> <pre>CONFIGURE CHANNEL DEVICE TYPE sbt MAXPIECESIZE 1G; CONFIGURE CHANNEL DEVICE TYPE sbt RATE 1700K;</pre> <p>The second command erases the MAXPIECESIZE setting of the first command.</p> <p>If AUXILIARY is specified, then this configuration is used only for channels allocated at the auxiliary instance. If no auxiliary device configuration is specified, and if RMAN needs to automatically allocate auxiliary channels, then RMAN uses the target database device configuration. It is not necessary to specify configuration information for auxiliary channels unless they require different parameters from the target channels.</p> <p>See Also: <i>Oracle9i Recovery Manager User's Guide</i> to learn how configure automatic channels specified by channel number</p>
<i>allocOperandList</i>	<p>Specifies control options for the allocated channel.</p> <p>See Also: "<i>allocOperandList</i>" on page 2-15</p>
CLEAR	<p>Clears the specified channel. For example, CONFIGURE CHANNEL 1 DEVICE TYPE DISK CLEAR returns only channel 1 to its default, whereas CONFIGURE CHANNEL DEVICE TYPE DISK CLEAR returns the generic disk channel to its default. Note that you cannot specify any other channel options (for example, PARMS) when you specify CLEAR.</p>
RETENTION POLICY TO	<p>Specifies a persistent, ongoing policy for datafile and control file backups and copies that RMAN marks as obsolete, that is, not needed and eligible for deletion. As time passes, RMAN marks backups and copies as obsolete according to the criteria you specify in the retention policy. RMAN does not automatically delete any backups or copies: manually run the DELETE OBSOLETE command to remove obsolete files. By default, RETENTION POLICY is configured to REDUNDANCY 1.</p> <p>For backups, the basic unit of the retention policy is a backup set, not a backup piece. For example, BACKUP COPIES 4 TABLESPACE users generates a single backup set that is duplexed into four identical backup pieces. The retention policy considers this as one backup, not four separate backups.</p>

Syntax Element	Description
RECOVERYWINDOWOF <i>integer</i> DAYS	Specifies a time window in which RMAN should be able to recover the database. The window stretches from the current time (SYSDATE) to the point of recoverability , which is the earliest date to which you want to recover. The point of recoverability is SYSDATE - <i>integer</i> days in the past.
REDUNDANCY <i>integer</i>	Specifies that RMAN should retain <i>integer</i> backups or copies of each datafile and control file. If more than <i>integer</i> backups or copies exist, RMAN marks these extra files as obsolete. Then, RMAN determines the oldest of the retained backups and copies, and marks all archived logs and log backups older than this backup or copy as obsolete. The DELETE OBSOLETE command removes obsolete backups and copies as well as archived log backups and copies.
NONE	Disables the retention policy feature. RMAN does not consider any backups or copies as obsolete.
CLEAR	Resets the retention policy to its default (REDUNDANCY = 1).
MAXSETSIZE	specifies the maximum size of each backup set created on a channel. By default MAXSETSIZE is set to UNLIMITED, meaning that it is disabled.
TO <i>integer</i>	Specifies the maximum set size in bytes, kilobytes (K), megabytes (M), or gigabytes (G). The default setting is in bytes and is rounded down to kilobytes. For example, if you set MAXSETSIZE to 5000, RMAN sets the maximum set size at 4 kilobytes (that is, 4096 bytes), which is the lower kilobyte boundary of 5000. The minimum value must be greater than or equal to the database block size.
TO UNLIMITED	Specifies that there is no size limit for backup sets.
CLEAR	Resets the maximum set size to its default value (UNLIMITED).
{ARCHIVELOG DATAFILE} BACKUP COPIES FOR DEVICE TYPE <i>deviceSpecifier</i> TO <i>integer</i>	Specifies the number of copies of each backup set for DATAFILE (both datafiles and control files) or ARCHIVELOG files on the specified device type, from 1 (default) to 4. If duplexing is specified in the BACKUP command or in a SET BACKUP COPIES command, then the CONFIGURE setting is overridden. Note: Control file autobackups on disk are a special case and are never duplexed. RMAN always writes one and only copy.

Syntax Element	Description
BACKUP OPTIMIZATION	<p>Toggles backup optimization ON or OFF (default). Specify CLEAR to return optimization to its default value of OFF.</p> <p>Optimization does not back up a file to a device type if the identical file is already backed up on the device type. For two files to be identical, their content must be exactly the same. You can override backup optimization by using the FORCE option of the BACKUP command.</p> <p>RMAN does not signal an error if optimization causes all files to be skipped during a backup. Note that BACKUP . . . DELETE INPUT deletes all specified files whether or not optimization would skip these files during a backup.</p> <p>Backup optimization is enabled when all of the following conditions are met:</p> <ul style="list-style-type: none"> ■ The CONFIGURE BACKUP OPTIMIZATION ON command has been run. ■ You run BACKUP DATABASE, BACKUP ARCHIVELOG with ALL or LIKE options, or BACKUP BACKUPSET ALL. ■ The RMAN job uses a channel of only one device type. <p>The retention policy has an effect on which files backup optimization skips.</p> <p>See Also: <i>Oracle9i Recovery Manager User's Guide</i> for a description of how RMAN determines that it can skip the backup of a file</p>
EXCLUDE FOR TABLESPACE <i>tablespace_name</i>	<p>Excludes the specified tablespace from BACKUP DATABASE commands. Note that you cannot exclude the SYSTEM tablespace. By default, each tablespace is not excluded, that is, the exclude functionality is disabled. The exclusion is stored as an attribute of the tablespace, not the individual datafiles, so the exclusion applies to any files that are added to this tablespace in the future. If you run CONFIGURE . . . CLEAR on a tablespace after excluding it, then it returns to the default configuration of "not excluded."</p> <p>You can still back up the configured tablespace by explicitly specifying it in a BACKUP command or by specifying the NOEXCLUDE option on a BACKUP DATABASE command.</p> <p>See Also: <i>Oracle9i Recovery Manager User's Guide</i> to learn more about snapshot control files</p>

Syntax Element	Description
AUXNAME FOR DATAFILE <i>datafileSpec</i> TO ' <i>filename</i> '	<p>Configures the auxiliary filename for the specified target datafile to '<i>filename</i>'. For example, you can set the auxiliary name for datafile 2 to <i>/df2.f</i>, and then unspecify this auxiliary name by running <code>CONFIGURE AUXNAME FOR DATAFILE 2 NULL</code>.</p> <p>If you are performing TSPITR or running the <code>DUPLICATE</code> command, then by setting <code>AUXNAME</code> you can preconfigure the filenames for use on the auxiliary database without manually specifying the auxiliary filenames during the procedure.</p> <p>For example, use this command during TSPITR if the datafiles are on raw disk and you need to restore auxiliary datafiles to raw disk for performance reasons. Typically, you set the <code>AUXNAME</code> parameter in TSPITR for the datafiles of the <code>SYSTEM</code> tablespace and the tablespaces containing rollback segments. Do not overlay files which are in use by the production database and can be discarded after TSPITR completes. In essence, the <code>AUXNAME</code> of a datafile is the location where TSPITR can create a temporary copy of it.</p> <p>When renaming files with the <code>DUPLICATE</code> command, <code>CONFIGURE AUXNAME</code> is an alternative to <code>SET NEWNAME</code>. The difference is that after you set the <code>AUXNAME</code> the first time, you do not need to reset the filename when you issue another <code>DUPLICATE</code> command: the <code>AUXNAME</code> setting remains in effect until you issue <code>CONFIGURE AUXNAME . . . CLEAR</code>. In contrast, you must reissue the <code>SET NEWNAME</code> command every time you rename files.</p> <p>See Also: <i>Oracle9i Recovery Manager User's Guide</i> to learn how to perform RMAN TSPITR, and <i>Oracle9i Recovery Manager User's Guide</i> to learn how to duplicate a database</p>
SNAPSHOT CONTROLFILE NAME TO ' <i>filename</i> '	<p>Configures the snapshot control file filename to '<i>filename</i>'. If you run <code>CONFIGURE SNAPSHOT CONTROLFILE NAME CLEAR</code>, then RMAN sets the snapshot control file name to its default.</p> <p>The default value for the snapshot control file name is platform-specific and dependent on the Oracle home. For example, the default on some UNIX system is <i>/?/dfs/snapcf_@.f</i>. If you clear the control file name, and you change the Oracle home, then the default location of the snapshot control file changes as well.</p> <p>See Also: <i>Oracle9i Recovery Manager User's Guide</i> for more information about snapshot control files</p>
CONTROLFILE AUTOBACKUP	<p>Controls the control file autobackup feature. By default, this feature is not enabled.</p>

Syntax Element	Description
ON	<p>If <code>CONFIGURE CONTROLFILE AUTOBACKUP</code> is ON (by default it is OFF), then RMAN performs a control file autobackup in the following circumstances:</p> <ul style="list-style-type: none"> ■ After every <code>BACKUP</code> or <code>COPY</code> command issued at the RMAN prompt. ■ Whenever a <code>BACKUP</code> or <code>COPY</code> command within a <code>RUN</code> block is followed by a command that is neither <code>BACKUP</code> nor <code>COPY</code>. ■ At the end of every <code>RUN</code> block if the last command in the block was either <code>BACKUP</code> or <code>COPY</code>. ■ After database structural changes such as adding a new tablespace, altering the state of a tablespace or datafile (for example, bringing it online), adding a new online redo log, renaming a file, adding a new redo thread, and so forth. This type of autobackup, unlike autobackups that occur in the preceding circumstances, goes only to disk. You can run <code>CONFIGURE CONTROLFILE AUTOBACKUP FOR DEVICE TYPE DISK</code> to set a nondefault disk location. <p>The first channel allocated during the backup or copy job creates the autobackup and places it into its own backup set; for post-structural autobackups, the default disk channel makes the backup. RMAN writes the control file and the server parameter file to the same backup piece. After the control file autobackup completes, Oracle writes a message containing the complete path of the backup piece and the device type to the alert log.</p> <p>RMAN automatically backs up the current control file using the default format of <code>%F</code> (see entry for <code>CONFIGURE CONTROLFILE AUTOBACKUP FORMAT</code> for an explanation of this substitution variable). You can change this format using the <code>CONFIGURE CONTROLFILE AUTOBACKUP FORMAT</code> and <code>SET CONTROLFILE AUTOBACKUP FORMAT</code> commands.</p>
OFF	<p>Disables the autobackup feature (default). Note that when this command is OFF, any <code>BACKUP</code> command that includes datafile 1 (including <code>BACKUP DATABASE</code>) automatically includes the current control file and server parameter file in the backup set. Otherwise, RMAN does not include these files.</p>
CLEAR	<p>Returns the feature to its default setting of OFF.</p>

Syntax Element	Description
CONTROLFILE AUTOBACKUP FORMAT FOR DEVICE TYPE <i>deviceSpecifier</i> TO 'format_string'	<p>Configures the default filename format for the control file autobackup on the specified device type. By default, the initial format is %F for all devices. Any default format string specified with CONFIGURE must include the %F substitution variable (see BACKUP). This variable translates into <i>c-IIIIIIIIII-YYYYMMDD-QQ</i>, where:</p> <ul style="list-style-type: none"> ■ <i>IIIIIIIIII</i> stands for the DBID. The DBID is printed in decimal so that it can be easily associated with the target database. ■ <i>YYYYMMDD</i> is a time stamp in the Gregorian calendar of the day the backup is generated ■ <i>QQ</i> is the sequence in hexadecimal number that starts with 00 and has a maximum of 'FF' (256) <p>Specify CLEAR to return the format to the default %F.</p>
CLEAR	<p>Clears all user settings and returns the specified CONFIGURE command to its default value. For example, CONFIGURE RETENTION POLICY CLEAR returns the retention policy configuration to its default value of REDUNDANCY = 1. CLEAR affects only the CONFIGURE command on which it is an option. For example, the second command does not clear the configuration for the first command, whereas the last command does clear the first command:</p> <pre>CONFIGURE DEVICE TYPE sbt PARALLELISM 3; CONFIGURE CHANNEL DEVICE TYPE sbt CLEAR; CONFIGURE DEVICE TYPE sbt CLEAR;</pre>

Examples

Configuring Backup Optimization: Example This example configures RMAN so that the BACKUP command does not back up files to a device type if the identical file has already been backed up to the device type:

```
CONFIGURE BACKUP OPTIMIZATION ON;
```

Configuring a Retention Policy: Example This example configures a retention policy with a recovery window of 2 weeks, and then resets the retention policy to its default value of REDUNDANCY = 1:

```
CONFIGURE RETENTION POLICY TO RECOVERY WINDOW OF 14 DAYS;
CONFIGURE RETENTION POLICY CLEAR;
```

Configuring Automatic Disk and Tape Channels: Example This example configures generic DISK and sbt channels, sets the default device type to sbt, and sets PARALLELISM to 3:

```
CONFIGURE CHANNEL DEVICE TYPE DISK RATE 5M;
CONFIGURE CHANNEL DEVICE TYPE sbt PARMS 'ENV=(NSR_SERVER=bksrv1)';
CONFIGURE DEFAULT DEVICE TYPE TO sbt;
CONFIGURE DEVICE TYPE sbt PARALLELISM 3;
```

Overriding the Default Device Type: Example This example configures the default device type to sbt, backs up the archived logs on the default sbt channel, and then backs up the database to disk on the default disk channel:

```
CONFIGURE CHANNEL DEVICE TYPE sbt PARMS 'ENV=(NSR_SERVER=bksrv1)';
CONFIGURE DEFAULT DEVICE TYPE TO sbt;
BACKUP ARCHIVELOG ALL;
BACKUP DEVICE TYPE DISK DATABASE;
```

Configuring Automatic Channels Across File Systems: Example This example configures automatic disk channels across three file systems:

```
CONFIGURE DEVICE TYPE DISK PARALLELISM 3;
CONFIGURE CHANNEL 1 DEVICE TYPE DISK FORMAT '/disk1/backup/%U';
CONFIGURE CHANNEL 2 DEVICE TYPE DISK FORMAT '/disk2/backup/%U';
CONFIGURE CHANNEL 3 DEVICE TYPE DISK FORMAT '/disk3/backup/%U';
BACKUP DEVICE TYPE DISK DATABASE PLUS ARCHIVELOG;
```

Configuring Automatic Channels in an Oracle Real Application Clusters Configuration: Example This example allocates automatic sbt channels for two nodes of an Oracle Real Application Clusters database:

```
CONFIGURE DEVICE TYPE sbt PARALLELISM 2;
CONFIGURE DEFAULT DEVICE TYPE TO sbt;
CONFIGURE CHANNEL 1 DEVICE TYPE sbt CONNECT 'SYS/change_on_install@node1'
  PARMS 'ENV=(NSR_SERVER=bkserv1)';
CONFIGURE CHANNEL 2 DEVICE TYPE sbt CONNECT 'SYS/change_on_install@node2'
  PARMS ENV=(NSR_SERVER=bkserv2)';
```

Clearing Automatic Channels: Example This example clears manually numbered DISK channels 2 and 3 and the generic sbt channel:

```
CONFIGURE CHANNEL 2 DEVICE TYPE DISK CLEAR;
CONFIGURE CHANNEL 3 DEVICE TYPE DISK CLEAR;
CONFIGURE CHANNEL DEVICE TYPE sbt CLEAR;
```

Configuring and Clearing Parallelism: Example This example sets `DISK` parallelism to 2, then changes it to 3, then returns it to the default parallelism of 1:

```
CONFIGURE DEVICE TYPE DISK PARALLELISM 2;
CONFIGURE DEVICE TYPE DISK PARALLELISM 3;
CONFIGURE DEVICE TYPE DISK CLEAR;
```

Configuring Backup Copies: Example This example configures duplexing to 3 for `DISK` backups of datafiles and control files (control file autobackups on disk are a special case and are never duplexed) and then runs a database backup, specifying three different file systems for the copies:

```
CONFIGURE DATAFILE BACKUP COPIES FOR DEVICE TYPE DISK TO 3;
BACKUP DEVICE TYPE DISK DATABASE
  FORMAT '/disk1/backup/%U', '/disk2/backup/%U', '/disk3/backup/%U';
```

Configuring the Snapshot Control File Location: Example This example configures a new location for the snapshot control file and then resynchronizes the recovery catalog.

```
CONFIGURE SNAPSHOT CONTROLFILE NAME TO '?/oradata/snap.cf';
```

Excluding a Tablespace from a Whole Database Backup: Example This example excludes the `example` tablespace from whole database backups, then returns the tablespace to its default value of "not excluded":

```
CONFIGURE EXCLUDE FOR TABLESPACE example;
CONFIGURE EXCLUDE CLEAR;
```

Specifying Auxiliary Filenames: Example This example duplicates a database to a remote host with a different directory structure, by using `CONFIGURE AUXNAME` to specify new filenames for the datafiles:

```
# set auxiliary names for the datafiles
CONFIGURE AUXNAME FOR DATAFILE 1 TO '/oracle/auxfiles/aux_1.f';
CONFIGURE AUXNAME FOR DATAFILE 2 TO '/oracle/auxfiles/aux_2.f';
CONFIGURE AUXNAME FOR DATAFILE 3 TO '/oracle/auxfiles/aux_3.f';
CONFIGURE AUXNAME FOR DATAFILE 4 TO '/oracle/auxfiles/aux_4.f';
```

```
RUN
{
  ALLOCATE AUXILIARY CHANNEL dupdb1 TYPE DISK;
  DUPLICATE TARGET DATABASE TO dupdb
```

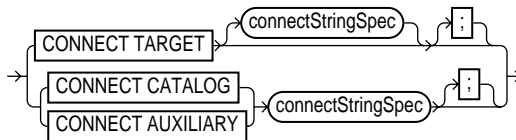
```
LOGFILE
  GROUP 1 ('?/dbs/dupdb_log_1_1.f',
          '?/dbs/dupdb_log_1_2.f') SIZE 200K,
  GROUP 2 ('?/dbs/dupdb_log_2_1.f',
          '?/dbs/dupdb_log_2_2.f') SIZE 200K REUSE;
}
# Un-specify the auxiliary names for the datafiles so that they are not overwritten
# by mistake:
CONFIGURE AUXNAME FOR DATAFILE 1 CLEAR;
CONFIGURE AUXNAME FOR DATAFILE 2 CLEAR;
CONFIGURE AUXNAME FOR DATAFILE 3 CLEAR;
CONFIGURE AUXNAME FOR DATAFILE 4 CLEAR;
```

Specifying the Default Format for the Control File Autobackup: Example This example turns on the autobackup feature, then changes the default format for the DISK and sbt devices, then clears the autobackup setting:

```
CONFIGURE CONTROLFILE AUTOBACKUP ON;
CONFIGURE CONTROLFILE AUTOBACKUP FORMAT FOR DEVICE TYPE DISK TO '?/oradata/%F';
CONFIGURE CONTROLFILE AUTOBACKUP FORMAT FOR DEVICE TYPE sbt TO 'cf_auto_%F';
CONFIGURE CONTROLFILE AUTOBACKUP CLEAR; # returns to default setting of OFF
```

CONNECT

Syntax



Purpose

To establish a connection between RMAN and a target, auxiliary, or recovery catalog database.

Note: When connecting from the command line, the password may be visible to other users on the system. The [CONNECT](#) command avoids this problem.

See Also: ["cmdLine"](#) on page 2-59 for command line connection options

Restrictions and Usage Notes

- You can only run the `CONNECT TARGET`, `CONNECT CATALOG`, and `CONNECT AUXILIARY` commands if you are at the RMAN prompt and if you are not already connected to the specified databases.
- If you need to connect to a different target, catalog, or auxiliary database, then you should start a new RMAN session.
- You cannot run `CONNECT CATALOG` command when RMAN is in the default `NOCATALOG` mode, that is, when these conditions are met:
 - You started RMAN at the command without specifying either `CATALOG` or `NOCATALOG`.
 - You have already run a command such as `BACKUP` that requires a repository connection.

Keywords and Parameters

Syntax Element	Description
CONNECT TARGET <i>connectStringSpec</i>	Establishes a connection between RMAN and the target database.
CONNECT CATALOG <i>connectStringSpec</i>	Establishes a connection between RMAN and the recovery catalog database. You must run this command <i>before</i> running any command that requires a repository. Otherwise, RMAN defaults to NOCATALOG mode and invalidates the use of CONNECT CATALOG in the session.
CONNECT AUXILIARY <i>connectStringSpec</i>	Establishes a connection between RMAN and an auxiliary instance. You can use an auxiliary instance with the DUPLICATE command or during TSPITR.

Examples

Connecting Without a Recovery Catalog: Example This example starts RMAN and then connects to the target database with an Oracle Net service name `prod1`:

```
% rman NOCATALOG
RMAN> CONNECT TARGET sys/change_on_install@prod1;
```

Connecting in the Default NOCATALOG Mode: Example This example starts RMAN and then connects to the target through Oracle Net. Because BACKUP is run and no CONNECT CATALOG has been run, RMAN defaults to NOCATALOG mode:

```
% rman
RMAN> CONNECT TARGET sys/change_on_install@prod1;
RMAN> BACKUP DATAFILE 7;
# You cannot run CONNECT CATALOG after this point because RMAN has defaulted to NOCATALOG
```

Connecting with a Recovery Catalog: Example This example starts RMAN and then connects to the target database `prod1` by using operating system authentication and the recovery catalog database `rcat` by using a password file:

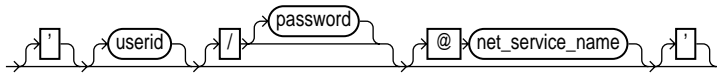
```
% rman
RMAN> CONNECT TARGET /
RMAN> CONNECT CATALOG rman/rman@rcat
```

Connecting to Target, Recovery Catalog, and Duplicate Databases: Example This example connects to three databases specifying a username and password for each:

```
% rman
RMAN> CONNECT TARGET SYS/sysdba@prod1
RMAN> CONNECT CATALOG rman/rman@rcat
RMAN> CONNECT AUXILIARY SYS/sysdba@dupdb
```

connectStringSpec

Syntax



Purpose

A subclause specifying the username, password, and net service name for connecting to a target, recovery catalog, or auxiliary database. The connection is necessary to authenticate the user and identify the database.

Restrictions and Usage Notes

- You must have `SYSDBA` privileges on the target and auxiliary databases.
- Do not connect to the recovery catalog database as user `SYS`.

Keywords and Parameters

Syntax Element	Description
<code>/</code>	<p>If you do not specify a user ID or password when connecting to the target database, then a forward slash establishes a connection as user <code>SYS</code> by using operating system authentication. For example, enter the following to connect to the target database:</p> <pre>% rman TARGET /</pre> <p>Note: The forward slash depends on platform-specific environment variables.</p>
<code>userid</code>	<p>Establishes a connection to the database for the specified user. If you do not specify a password, RMAN obtains the password interactively by displaying a prompt. The characters will not be echoed to the terminal.</p> <p>You must have <code>SYSDBA</code> authority when connecting to the target or auxiliary database, but must <i>not</i> connect as <code>SYS</code> to the recovery catalog database.</p> <p>Note: The connect string must not contain any white space, but it can contain punctuation characters such as a forward slash (<code>/</code>) and an at sign (<code>@</code>).</p>
<code>/password</code>	<p>Establishes a connection for the specified user by using a password. If the target database is not open, then a password file must exist.</p>

Syntax Element	Description
<code>@net_service_name</code>	Establishes a connection to the database through an optional Oracle Net net service name.

Examples

Connecting Without a Recovery Catalog: Example This example connects to the target database by using a password and the Oracle Net service name `prod1` in the default `NOCATALOG` mode:

```
% rman TARGET SYS/change_on_install@prod1
```

Entering the Password Interactively: Example This example connects to the target database as user `SYS` but without specifying a password at the command line:

```
% rman TARGET SYS
```

```
Recovery Manager: Release 9.0.1.0.0
```

```
target database Password:
```

Connecting with Operating System Authentication: Example This example starts RMAN and then connects to the target database `prod1` by using operating system authentication and the recovery catalog database `rcat` using a net service name:

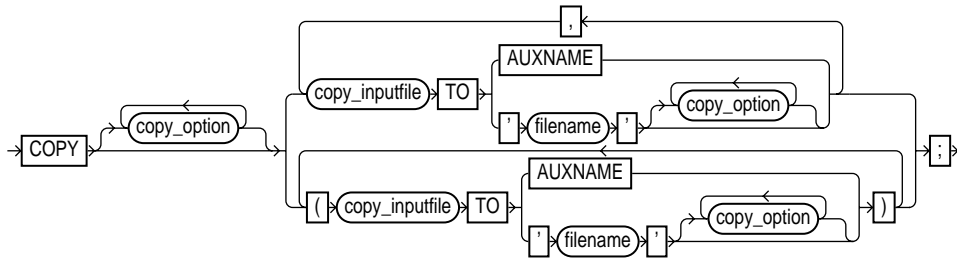
```
% rman
RMAN> CONNECT TARGET /
RMAN> CONNECT CATALOG rman/rman@rcat
```

Connecting to a Target Database, Recovery Catalog, and Auxiliary Instance: Example This example connects to three different databases from the command line, specifying a username, password, and net service name for each:

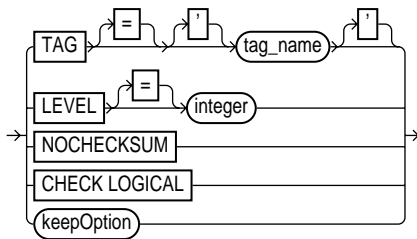
```
% rman TARGET SYS/pwd1@prod1 CATALOG rman/rman@rcat AUXILIARY SYS/pwd2@dupdb
```

COPY

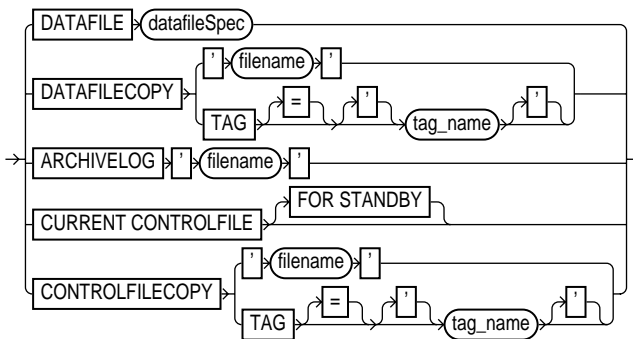
Syntax



copy_option ::=



copy_inputfile ::=



Purpose

Create an image copy of a file. The output file is always written to disk. You can copy the following types of files:

- Datafiles (current or copies)
- Archived redo logs
- Control files (current or copies)

In many cases, copying datafiles is more beneficial than backing them up, since the output is suitable for use without any additional processing. In contrast, you must process a backup set with a [RESTORE](#) command before it is usable. So, you can perform media recovery on a datafile copy, but not directly on a backup set, even if it backs up only one datafile and contains a single backup piece.

If `CONFIGURE CONTROLFILE AUTOBACKUP` is set to `ON`, then RMAN automatically backs up the control file after `COPY` commands. "[BACKUP](#)" on page 2-26 describes the complete set of circumstances in which autobackups occur.

See Also: *Oracle9i Recovery Manager User's Guide* to learn how to copy files with RMAN

Restrictions and Usage Notes

- The target database must be mounted or open.
- You cannot make incremental copies.

Keywords and Parameters

Syntax Element	Description
<i>copy_option</i>	Specifies optional parameters affecting either the input or output files or both.
<code>TAG = 'tag_name'</code>	Specifies the tag of the input file or output file copy. Tag names are not case sensitive. RMAN gives each copy a default TAG. Refer to the TAG description in the BACKUP command for the default filename format. Note: You cannot apply a tag to a copy of an archived log, current control file, or standby control file.
<code>LEVEL = integer</code>	Includes the input file or output file copy in the incremental backup strategy by making it serve as a basis for subsequent incremental backup sets. Typically, you specify <code>LEVEL 0</code> . If you do not use the <code>LEVEL</code> option, then the datafile copy has no impact on the incremental backup strategy.

Syntax Element	Description
NOCHECKSUM	Suppresses block checksums. Unless you specify this option, Oracle computes a checksum for each block. RMAN verifies the checksum when restoring the copy. If the database is already maintaining block checksums, then this flag has no effect.
CHECK LOGICAL	<p>Tests data and index blocks that pass physical corruption checks for logical corruption, for example, corruption of a row piece or index entry. If RMAN finds logical corruption, then it logs the block in the <code>alert.log</code> and server session trace file.</p> <p>Provided the sum of physical and logical corruptions detected for a file remain below its <code>MAXCORRUPT</code> setting, the RMAN command completes and Oracle populates <code>V\$DATABASE_BLOCK_CORRUPTION</code> with corrupt block ranges. If <code>MAXCORRUPT</code> is exceeded, then the command terminates without populating the views.</p> <p>Note: For <code>COPY</code> and <code>BACKUP</code>, the <code>MAXCORRUPT</code> setting represents the total number of physical and logical corruptions permitted on a file.</p>
<i>keepOption</i>	<p>Overrides any configured retention policy for this copy so that the copy is not considered obsolete.</p> <p>See Also: "<i>keepOption</i>" on page 2-120</p>
<i>copy_inputfile</i>	specifies the type of input file, that is, the file that you want to copy.
DATAFILE <i>datafileSpec</i>	<p>Specifies a list of one or more datafiles as input.</p> <p>See Also: "<i>datafileSpec</i>" on page 2-96</p> <p>Note: If you specify a filename, then it must be the name of a current datafile as listed in the control file.</p>
DATAFILECOPY 'filename'	Specifies a list of one or more datafile copies as input. Specify the datafile copies by ' <i>filename</i> ' or <code>TAG = 'tag_name'</code> . The filename must <i>not</i> be the name of a current datafile listed in the control file. The existing copy may have been created by either a previous <code>COPY</code> command or by an external operating system utility. Note that tag names are not case sensitive.
ARCHIVELOG 'filename'	Specifies the filename of an input archived redo log. The archived log may have been created by the Oracle archiving session or by a previous <code>copy</code> command. Specify the archived redo log by filename.
CURRENT CONTROLFILE	<p>Specifies the current control file.</p> <p>If you specify the <code>FOR STANDBY</code> option, RMAN makes a control file that can be used for creation of a standby database.</p>

Syntax Element	Description
CONTROLFILECOPY 'filename'	<p>Specifies the filename of a control file copy. You can also set TAG = 'tag_name' to specify a list of one or more control file copies.</p> <p>This command copies a control file copy. The copy can be:</p> <ul style="list-style-type: none"> ■ A copy of a normal control file (that is, not a standby control file) ■ A standby control file copy created by using the COPY STANDBY CONTROLFILE command or the SQL statement ALTER DATABASE CREATE STANDBY CONTROLFILE <p>RMAN inspects the header of the control file copy to determine whether it is a standby or nonstandby control file.</p> <p>Note: The control file copy is marked as a backup control file, so media recovery will be necessary if you mount the control file copy. This command is equivalent to the ALTER DATABASE BACKUP CONTROLFILE TO '...' statement.</p>
TO AUXNAME	Specifies that Oracle should copy the input datafile to the filename specified in the CONFIGURE AUXNAME command for the input datafile.
TO 'filename'	Specifies the filename of the output file copy.

Examples

Copying a Datafile: Example This example copies the datafile `tools01.dbf` with the `NOCHECKSUM` option to the output file `tools01.bak`, specifying it as a level 0 backup:

```
COPY
  NOCHECKSUM
  DATAFILE '?/oradata/trgt/tools01.dbf' TO '?/oradata/tools01.bak'
  LEVEL 0;
```

Copying the Control File: Example This example copies the current control file:

```
COPY CURRENT CONTROLFILE TO '?/oradata/cf1.bak';
```

Creating a Standby Control File: Example This example uses the preconfigured DISK channel to create a copy of the current target control file that can be used as a standby control file:

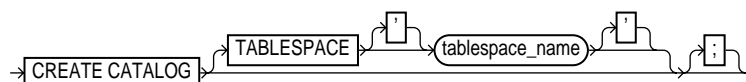
```
COPY CURRENT CONTROLFILE FOR STANDBY TO '?/oradata/cf_standby.cpy';
```

Creating a Long-Term Database Copy: Example The following shell script copies the control file and two datafiles and exempts them from the retention policy:

```
#!/usr/bin/tcsh
# a connection to the catalog is required for KEEP FOREVER
rman TARGET / CATALOG rman/rman@rcat <<EOF
SHUTDOWN IMMEDIATE;
STARTUP MOUNT;
COPY
  KEEP FOREVER NOLOGS
  CURRENT CONTROLFILE TO '?/oradata/cf_longterm.cpy',
  DATAFILE 1 TO '?/oradata/df1_longterm.cpy',
  DATAFILE 2 TO '?/oradata/df2_longterm.cpy';
ALTER DATABASE OPEN;
EOF
```

CREATE CATALOG

Syntax



Purpose

To create a schema for the recovery catalog. Typically, you create this schema in a separate recovery catalog database. The catalog is created in the default tablespace of the recovery catalog owner.

Note: In releases prior to 8.1.5, you created the recovery catalog schema by connecting to the recovery catalog database and executing the `catrman.sql` script.

See Also: *Oracle9i Recovery Manager User's Guide* to learn how to create the recovery catalog

Restrictions and Usage Notes

- Execute this command only at the RMAN prompt.
- RMAN must be connected to the recovery catalog either through the `CATALOG` command-line option or the `CONNECT CATALOG` command, and the catalog database must be open. A connection to the target database is not required.
- The recovery catalog owner must be granted the `RECOVERY_CATALOG_OWNER` role, and also be granted space privileges in the tablespace where the recovery catalog tables will reside.
- Do not create the recovery catalog in the `SYS` schema.

See Also:

- *Oracle9i Database Administrator's Guide* for more information about the RECOVERY_CATALOG_OWNER role
- ["cmdLine"](#) on page 2-59 for information about RMAN command-line options

Keywords and Parameters

Syntax Element	Description
TABLESPACE ' <i>tablespace_name</i> '	Specifies the tablespace in which to store the recovery catalog schema. The catalog owner must be granted quota privileges. If you do not specify a tablespace, then RMAN stores the recovery catalog in the default tablespace of the recovery catalog owner.

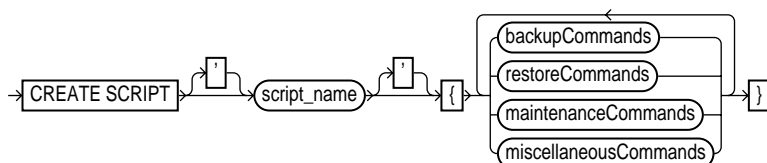
Example

Creating a Catalog Schema: Example The following UNIX shell script creates a user `rman`, grants `rman` the `RECOVERY_CATALOG_OWNER` role, then creates the recovery catalog in the schema `rman.cattbs` of the database `rcat`:

```
#!/usr/bin/tcsh
# create user rman in recovery catalog database as catalog owner
sqlplus 'SYS/change_on_install@rcat AS SYSDBA' <<EOF
CREATE USER rman IDENTIFIED BY rman
    DEFAULT TABLESPACE cattbs
    QUOTA UNLIMITED ON cattbs;
GRANT recovery_catalog_owner TO rman;
EXIT
EOF
# connect to database as catalog owner and create catalog
rman CATALOG rman/rman@rcat <<EOF
CREATE CATALOG;
EOF
```

CREATE SCRIPT

Syntax



See Also: The syntax diagrams for **"RUN"** on page 2-190 to determine which commands are included in the *backupCommands*, *restoreCommands*, *maintenanceCommands*, and *miscellaneousCommands* clauses

Purpose

To create a script and store it in the recovery catalog. Stored scripts provide a common repository for frequently executed collections of RMAN commands. Any command that is legal within a **RUN** command is permitted in the stored script. The script is not executed immediately; run the script with **EXECUTE SCRIPT**.

See Also: *Oracle9i Recovery Manager User's Guide* to learn how to store scripts in the recovery catalog

Restrictions and Usage Notes

Note the following restrictions:

- Execute **CREATE SCRIPT** only at the RMAN prompt.
- RMAN must be connected to a target database.
- RMAN must be connected to the recovery catalog either through the **CATALOG** command-line option or the **CONNECT CATALOG** command, and the catalog database must be open.
- You cannot run **CREATE SCRIPT** once and then use this same script on multiple target databases. You must connect to a specific target database and then run **CREATE SCRIPT** to create a script for this specific target. Each script is associated with one and only one target database.

- You cannot execute a [RUN](#) command within a stored script.
- The `@` and `@@` commands do not work within `CREATE SCRIPT`.
- Quotes must be used around the script name when the name contains either spaces or reserved words.

Keywords and Parameters

For descriptions of the individual commands that you can use in a stored script, refer to the appropriate entry, for example, "[BACKUP](#)" on page 2-26.

Syntax Element	Description
<code>'script_name'</code>	<p>Creates a stored script with the specified name. The statements allowable within the brackets of the <code>CREATE SCRIPT 'script_name' { ... }</code> command are the same allowable within the <code>RUN</code> command. The statements within the braces constitute the <code>job_command_list</code>.</p> <p>Note: To run the stored script, specify <code>EXECUTE SCRIPT</code> within the braces of the <code>RUN</code> command.</p>

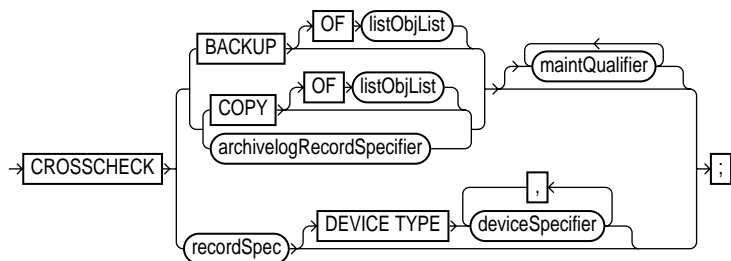
Example

Creating a Script: Example The following UNIX shell script creates a recovery catalog script called `b_whole_10` that backs up the database and archived redo logs, then executes `b_whole_10`:

```
#!/usr/bin/tcsh
rman TARGET / CATALOG rman/rman@catdb <<EOF
# creates recovery catalog script to back up database and archived logs
CREATE SCRIPT b_whole_10
{
    BACKUP INCREMENTAL LEVEL 0 TAG b_whole_10
    DATABASE PLUS ARCHIVELOG;
}
# runs the recovery catalog script b_whole_10
RUN { EXECUTE SCRIPT b_whole_10; }
EOF
```

CROSSCHECK

Syntax



Purpose

To verify the existence of backups (backup sets or media-managed proxy copies) and copies (both archived logs and image copies) stored on disk or tape. The `CROSSCHECK` command only processes files created on the same device type as the channels running the crosscheck.

The `CROSSCHECK` command checks only objects marked `AVAILABLE` or `EXPIRED` by examining the files on disk for `DISK` channels or by querying the media manager for `sbt` channels. Table 2-2 describes the meaning of each status:

Table 2-2 *Meaning of Crosscheck Status*

Status	Description
EXPIRED	Object is not found either in file system (for <code>DISK</code>) or in the media manager (for <code>sbt</code>). For a backup set to be <code>EXPIRED</code> , all backup pieces in the set must be <code>EXPIRED</code> . Note: <code>EXPIRED</code> does not mean the same as <code>OBSOLETE</code> .
AVAILABLE	Object is available for use by RMAN. For a backup set to be <code>AVAILABLE</code> , all backup pieces in the set must have the status <code>AVAILABLE</code> .
UNAVAILABLE	Object is not available for use by RMAN. For a backup set to be <code>UNAVAILABLE</code> , all backup pieces in the set must have the status <code>UNAVAILABLE</code> .

The `CROSSCHECK` command does not delete any files that it is unable to find, but updates their repository records to `EXPIRED`. Then, you can run `DELETE EXPIRED`

to remove the repository records for all expired files as well as any existing physical files whose records show the status `EXPIRED`.

If some backup pieces or copies were erroneously marked as `EXPIRED`, for example, because the media manager was misconfigured, then after ensuring that the files really do exist in the media manager, run the `CROSSCHECK BACKUP` command again to restore those files to `AVAILABLE` status.

See Also: *Oracle9i Recovery Manager User's Guide* to learn how to manage target database records in the catalog

Restrictions and Usage Notes

- The target instance must be started.
- A maintenance channel is not required when `CROSSCHECK` is used with a file that is on disk. If you use `CROSSCHECK` on nondisk files, and have objects created on device types that are not configured for automatic channels, then you must manually allocate maintenance channels for these objects. For example, if you created a backup on an `sbt` channel, but have only a `DISK` channel automatically configured, then you must manually allocate an `sbt` channel before the `CROSSCHECK` command can check the backup.

Keywords and Parameters

Syntax Element	Description
<code>BACKUP</code>	<p>Crosschecks backup sets, backup pieces, and proxy copies that have status <code>AVAILABLE</code> or <code>EXPIRED</code>. Note that <code>RMAN</code> crosschecks all backups not marked <code>UNAVAILABLE</code>, even if they cannot be restored or if they are incremental backups whose parent full backup or copy no longer exists. By default, <code>RMAN</code> crosschecks backups of the whole database.</p> <p>See Also: "LIST Output" on page 2-125. Use the <code>KEY</code> column of the output to obtain the primary key usable in the <code>CHANGE</code> and <code>DELETE</code> commands</p>
<code>COPY</code>	<p>Crosschecks datafile copies, control file copies, archived redo logs, and image copies of archived redo logs. By default, <code>CROSSCHECK</code> checks copies of all files in the database with status <code>AVAILABLE</code> or <code>EXPIRED</code>.</p>
<code>OF listObjList</code>	<p>Restricts the list of objects operated on to the object type specified in the <code>listObjList</code> clause. If you do not specify an object, <code>CROSSCHECK</code> defaults to all copies.</p> <p>See Also: "listObjList" on page 2-140</p>

Syntax Element	Description
<i>archivelogRecordSpecifier</i>	Crosschecks the specified archived redo logs. See Also: "archivelogRecordSpecifier" on page 2-21
<i>maintQualifier</i>	Restricts the command based on the specified options. See Also: "maintQualifier" on page 2-142
<i>recordSpec</i>	Specifies the object whose availability status you are changing. See "recordSpec" on page 2-149.
DEVICE TYPE <i>deviceSpecifier</i>	Allocates automatic channels for the specified device type only. This option is valid only if you have configured automatic channels and have not manually allocated channels. For example, if you configure automatic disk and tape channels and run CROSSCHECK . . . DEVICE TYPE DISK, then RMAN allocates only disk channels. See Also: "deviceSpecifier" on page 2-104

Examples

Crosschecking All Backups and Copies: Example The following example, which assumes that the default configured channel is DEVICE TYPE sbt, queries the status of all backups and copies on tape and disk. Because RMAN preconfigures a disk channel, you do not need to manually allocate a disk channel:

```
CROSSCHECK BACKUP;
CROSSCHECK COPY;
```

Crosschecking Within a Range of Dates: Example The following example queries the media manager for the status of the backup sets in a given six month range. Note that RMAN uses the date format specified in the NLS_DATE_FORMAT parameter, which is 'DD-MON-YY' in this example:

```
# if you manually allocate an sbt channel, then RMAN does not crosscheck disk
ALLOCATE CHANNEL FOR MAINTENANCE DEVICE TYPE sbt;
CROSSCHECK BACKUP
  COMPLETED BETWEEN '01-JAN-01' AND '01-SEP-01';
RELEASE CHANNEL;

# to crosscheck only disk, specify CROSSCHECK DEVICE TYPE DISK
CROSSCHECK BACKUP DEVICE TYPE DISK
  COMPLETED BETWEEN '01-JAN-01' AND '01-SEP-01';

# assuming that the default channel is sbt, you can crosscheck on both disk
# and sbt backups by simply running CROSSCHECK with the default channels
CROSSCHECK BACKUP COMPLETED BETWEEN '01-JAN-01' AND '01-SEP-01';
```

datafileSpec

Syntax



Purpose

A subclause that specifies a datafile by filename or absolute file number.

Restrictions and Usage Notes

- You can specify the relative or absolute path name.
- Double and single quotes are both legal (although only single quotes are shown in the diagram). Double quotes are recommended in the SQL command.
- Use ? to represent the Oracle home and @ for the Oracle SID.

See Also: ["Placeholders"](#) on page 1-3 to learn about the difference between single and double quotes, as well as the behavior of environment variables in RMAN quoted strings

Keywords and Parameters

Syntax Element	Description
'filename'	Specifies the datafile by using either the full path or a relative filename. If you specify a relative filename, the filename is qualified in a port-specific manner by the target database.
integer	Specifies the datafile by using its absolute file number. Obtain the file number from the V\$DATAFILE, V\$DATAFILE_COPY, or V\$DATAFILE_HEADER views or REPORT SCHEMA command output.

Examples

Specifying a Datafile by Filename: Example This example copies datafile `*/oradata/trgt/users01.dbf` to disk, specifying it by filename:

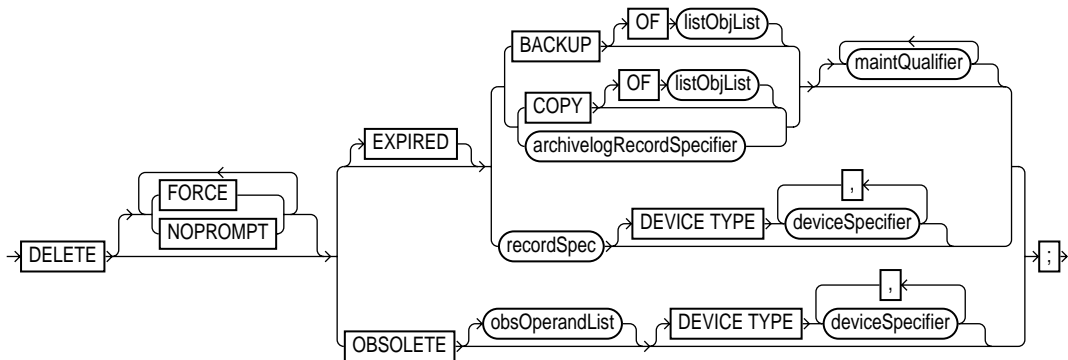
```
COPY DATAFILE '*/oradata/trgt/users01.dbf' TO '*/oradata/users01.cpy';
```

Specifying a Datafile by Absolute File Number: Example This example copies datafiles 3 and 4 to disk, specifying them by file number:

```
COPY DATAFILE 3 TO '?/oradata/df3.cpy',  
DATAFILE 4 TO '?/oradata/df4.cpy';
```

DELETE

Syntax



Purpose

To delete physical backups and copies as well as do the following:

- Update their records in the target control file to status DELETED
- Remove their records from the recovery catalog (if you use a catalog)

By default, `DELETE` displays a list of the files and prompts you for confirmation before deleting any file in the list, unless you are running a command file.

If you specify the `EXPIRED` option, then `DELETE` only removes files marked `EXPIRED`, that is, "not found," by the `CROSSCHECK` command. Use the `LIST` command or query the recovery catalog views to determine which backups or copies are expired.

Caution: If for some reason a backup or copy marked `EXPIRED` exists when you run the `DELETE EXPIRED` command, then `RMAN` deletes the physical files.

If you specify the `OBSOLETE` option, then `DELETE` removes files considered `OBSOLETE`, that is, "not needed," by the retention policy or because it is orphaned. Specify a retention policy by using `CONFIGURE RETENTION POLICY` or the `REDUNDANCY` and `RECOVERY WINDOW` options on the `DELETE` command.

The repository record for an object can sometimes fail to reflect the physical status of the object. For example, you back up a log to disk and then use an operating system utility to delete the object. If you do not run the `CROSSCHECK` command to update the repository, and if you then run `DELETE` against the object, then the repository shows that the object is `AVAILABLE` while the object is in fact missing. The following table indicates the behavior of `DELETE` in such situations.

Repository Status	Physical Status	Behavior of DELETE Command
<code>AVAILABLE</code>	Not found on media	Does not delete the object and reports the list of mismatched objects at the end of the job. <code>RMAN</code> does not update the repository status.
<code>EXPIRED</code>	Found on media	Does not delete the object and reports the list of mismatched objects at the end of the job. <code>RMAN</code> does not update the repository status.
<code>UNAVAILABLE</code>	Any	Removes repository record and deletes object if it exists. All I/O errors are ignored.
<code>FORCE</code>	Any	Deletes file if it exists, ignoring any I/O errors, and removes repository records. <code>RMAN</code> displays the number of objects deleted at the end of the job.

See Also: ["BACKUP"](#) on page 2-26 to learn about the `BACKUP` . . . `DELETE INPUT` command

Restrictions and Usage Notes

- The target instance must be started.
- A maintenance channel is not required when `DELETE` is used with a file that is disk-only (that is, an `ARCHIVELOG`, `DATAFILECOPY`, `CONTROLFILECOPY`). Otherwise, you must use a manual or automatic maintenance channel.

If you use `DELETE` on files that are not disk-only, and if you have objects created on device types that are not configured for automatic channels, then run manual maintenance commands on these channels. For example, if you created a backup using an `sbt` channel, but have only a `DISK` channel automatically configured, you must manually allocate an `sbt` channel for `DELETE`.

Keywords and Parameters

Syntax Element	Description
FORCE	Deletes specified files (whether or not they exist on the media) and removes repository records. RMAN ignores any I/O errors for the deleted objects. RMAN displays the number of deleted objects at the end of the job.
NOPROMPT	Deletes specified files without first listing the files or prompting for confirmation. The <code>DELETE NOPROMPT</code> command still displays each item as it is deleted. By default, <code>DELETE</code> displays files and then prompts for confirmation. If the user confirms, then RMAN shows each item as it is deleted. If you are running commands from a command file, then <code>NOPROMPT</code> is the default.
EXPIRED	Removes only files whose status in the repository is <code>EXPIRED</code> . RMAN marks backups and copies as expired when you run a <code>CROSSCHECK</code> command and the files are absent or inaccessible. To determine which files are expired, run a <code>LIST EXPIRED</code> command. Note: Beginning in Oracle9i, RMAN's default behavior is to prompt for confirmation when you run <code>DELETE EXPIRED</code> . In prior releases, RMAN did not prompt.
BACKUP	Deletes backup sets, backup pieces, and proxy copies. By default, RMAN deletes backups of the whole database. Specify the <code>EXPIRED</code> option to remove only backups that are marked <code>EXPIRED</code> in the repository. The <code>KEY</code> column of the <code>LIST</code> output indicates the primary key usable in the <code>CHANGE</code> and <code>DELETE</code> commands.
COPY	Deletes datafile copies, archived redo logs, and image copies of archived redo logs. By default, <code>DELETE . . . COPY</code> removes copies of all files in the database. Specify the <code>EXPIRED</code> option to remove only copies that are marked <code>EXPIRED</code> in the repository.
OF <i>listObjList</i>	restricts the list of objects operated on to the object type specified in the <i>listObjList</i> clause. See " listObjList " on page 2-140. If you do not specify an object, <code>CHANGE</code> defaults to all copies.
<i>maintQualifier</i>	restricts the command based on the specified options (see " maintQualifier " on page 2-142). Note: <i>listObjList</i> and <i>maintQualifier</i> are valid options on both the <code>BACKUP</code> and <code>COPY</code> commands.
<i>archivelogRecordSpecifier</i>	Deletes the specified archived redo logs. See Also: " archivelogRecordSpecifier " on page 2-21
<i>recordSpec</i>	Specifies the object that you are deleting. See Also: " recordSpec " on page 2-149

Syntax Element	Description
DEVICE TYPE <i>deviceSpecifier</i>	allocates automatic channels for the specified device type only (see " <i>deviceSpecifier</i> " on page 2-104). This option is valid only if you have configured channels and have not manually allocated channels. For example, if you configure disk and tape channels, and run <code>DELETE . . . DEVICE TYPE DISK</code> , then RMAN allocates only disk channels.
OBSOLETE	Deletes backups and datafile copies recorded in the RMAN repository that are obsolete, that is, no longer needed. In addition to obsolete datafile backups, RMAN deletes obsolete archived logs and archived log backups. RMAN determines which backups and copies of datafiles are no longer needed, which in turn determines when logs (and backups of logs) are no longer needed. RMAN considers the creation of a datafile as a backup when deciding which logs to keep. RMAN first uses the options that you specify with <i>obsOperandList</i> to determine what is obsolete. If you do not specify options in <i>obsOperandList</i> , then RMAN uses the options specified in <code>CONFIGURE RETENTION POLICY</code> .
<i>obsOperandList</i>	Specifies the criteria for determining which backups and copies are obsolete. See Also: " <i>obsOperandList</i> " on page 2-144
DEVICE TYPE <i>deviceSpecifier</i>	Restricts the deletion to obsolete backups and copies created on the specified device type only. See Also: " <i>deviceSpecifier</i> " on page 2-104

Examples

Deleting Expired Backups: Example The following example uses a configured `sbt` channel to check the media manager for expired backups of the tablespace `users` that are more than one month old and removes their catalog records:

```
CROSSCHECK BACKUP OF TABLESPACE users
  DEVICE TYPE sbt COMPLETED BEFORE 'SYSDATE-31';
DELETE NOPROMPT EXPIRED BACKUP OF TABLESPACE users
  DEVICE TYPE sbt COMPLETED BEFORE 'SYSDATE-31';
```

Deleting Obsolete Backups: Example The following example deletes backups and copies that are not needed to recover the database to a random point within the last week. RMAN also deletes archived redo logs that are no longer needed:

```
DELETE NOPROMPT OBSOLETE RECOVERY WINDOW OF 7 DAYS;
```

Deleting Files That Have Already Been Backed Up: Example The following example deletes backups and copies (including archived redo logs) that have already been backed up at least twice to tape:

```
DELETE NOPROMPT BACKUP BACKED UP 2 TIMES TO DEVICE TYPE sbt;  
DELETE NOPROMPT COPY BACKED UP 2 TIMES TO DEVICE TYPE sbt;
```

Forcing the Deletion of a Backup Set: Example The following example attempts to delete the backup set copy with tag `weekly_bkup`:

```
DELETE NOPROMPT BACKUPSET TAG weekly_bkup;
```

However, RMAN displays a warning because the repository shows the backup set as available, but the object is not actually available on the media:

```
RMAN-06207: WARNING: 1 objects could not be deleted for SBT_TAPE channel(s) due  
RMAN-06208:           to mismatched status. Use CROSSCHECK command to fix status  
List of Mismatched objects  
=====
```

Object	Type	Filename/Handle
Backup Piece		0id270ud_1_1

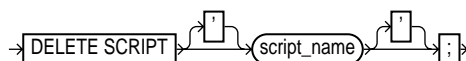
```
-----
```

The following command forces RMAN to delete the backup:

```
DELETE FORCE NOPROMPT BACKUPSET TAG weekly_bkup;
```

DELETE SCRIPT

Syntax



Purpose

To delete a stored script from the recovery catalog.

Restrictions and Usage Notes

- Execute `DELETE SCRIPT` only at the RMAN prompt.
- You must be connected to the target database that you connected to when you ran `CREATE SCRIPT` or `REPLACE SCRIPT`.
- RMAN must be connected to the recovery catalog either through the `CATALOG` command-line option or the `CONNECT CATALOG` command, and the catalog database must be open.
- Quotes must be used around the script name when the name contains either spaces or reserved words.

Keywords and Parameters

Syntax Element	Description
<code>'script_name'</code>	Deletes the specified script. The script name must be a name used in a previous <code>CREATE SCRIPT</code> or <code>REPLACE SCRIPT</code> command. See Also: " CREATE SCRIPT " on page 2-91

Example

Deleting a Script: Example The following shell script deletes script `b_whole_10`:

```

#!/usr/bin/sh
rman TARGET / CATALOG rman/cat@catdb<<EOF
DELETE SCRIPT b_whole_10;
EOF
  
```

deviceSpecifier

Syntax



Purpose

A subclause specifying the type of storage for a backup or copy.

Keywords and Parameters

Syntax Element	Description
DISK	Specifies disk storage device.
'media_device'	Specifies a sequential I/O device or access method for storage. The syntax and semantics of sequential I/O device types are platform-specific. Example values are <i>sbt</i> and <i>sbt_tape</i> (with or without quotes). These values are synonymous. The <i>media_device</i> variable specifies a media manager. Media device names are case insensitive. The <i>sbt</i> variable is legal as input, but RMAN output always displays its synonym <i>sbt_tape</i> . It is stored in the catalog as <i>sbt_tape</i> for backward compatibility.

Examples

Allocating a Tape Channel: Example This example allocates a maintenance channel for a media management device:

```
ALLOCATE CHANNEL FOR MAINTENANCE DEVICE TYPE sbt;
```

Backing Up to Disk: Example This example backs up the database to disk:

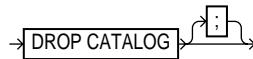
```
BACKUP DEVICE TYPE DISK DATABASE;
```

Restoring from Tape: Example This example restores archived logs from tape:

```
RESTORE DEVICE TYPE sbt ARCHIVELOG ALL;
```

DROP CATALOG

Syntax



Purpose

To remove the schema from the recovery catalog.

Caution: This command deletes all metadata from the recovery catalog. If you have no backups of the catalog, then all backups of all databases managed by this recovery catalog become unusable.

See Also: *Oracle9i Recovery Manager User's Guide* to learn how to drop the recovery catalog schema

Restrictions and Usage Notes

- Execute this command only at the RMAN prompt.
- You must be connected to the recovery catalog database through the `CATALOG` command-line option or the `CONNECT CATALOG` command. The catalog database must be open. You do not have to be connected to the target database.
- Enter the command twice to confirm that you want to drop the schema.

Example

Deleting the Catalog: Example This example drops the schema from the recovery catalog (you must enter the command twice to confirm):

```
RMAN> DROP CATALOG
```

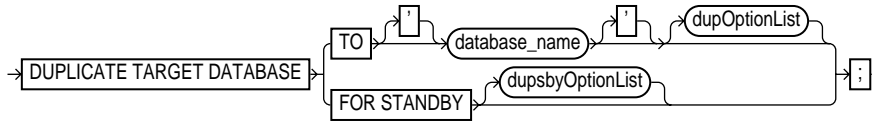
```
recovery catalog owner is RMAN
```

```
enter DROP CATALOG command again to confirm catalog removal
```

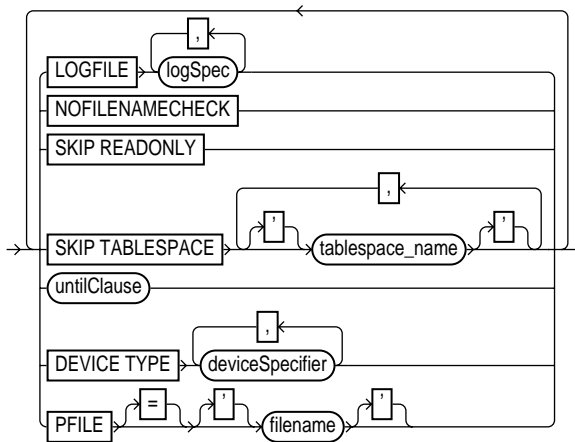
```
RMAN> DROP CATALOG
```

DUPLICATE

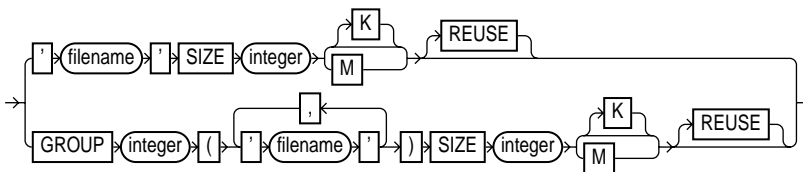
Syntax



dupOptionList ::=



logSpec ::=



`dupsbyOptionList::=`



Purpose

To use backups of the target database to create either of the following:

- A **duplicate database**, which is a copy of the target database (or a subset of the target database) with a unique DBID. Because a duplicate database has a unique DBID, it is entirely independent of the primary database and can be registered in the same recovery catalog as the primary database. Typically, duplicate databases are used for testing.
- A **standby database**, which is a special copy of the primary database that is updated by applying archived redo logs from the primary database. A standby database does *not* get a new DBID.

To create a standby database with the `DUPLICATE` command you must specify the `FOR STANDBY` option. The `DUPLICATE . . . FOR STANDBY` command creates the standby database by restoring a standby control file, mounting the standby control file, and then restoring and recovering backups of the target datafiles. The standby database is left mounted after duplication is complete. Note that backups of the standby database are interchangeable with backups of the primary database.

When duplicating a database that is currently in `NOARCHIVELOG` mode, recovery occurs with the `NOREDO` option. Hence, if incremental backups exist, `RMAN` applies only these backups to the restored files during recovery. For databases in `ARCHIVELOG` mode, `DUPLICATE` recovers by default up to the last archived redo log generated at the time the command was executed—unless the `SET UNTIL` clause is specified, in which case recovery is bounded by the parameter setting.

See Also:

Oracle9i Recovery Manager User's Guide to learn how to create a duplicate database with the `DUPLICATE` command

Oracle9i Recovery Manager User's Guide to learn how to create a standby database with the `DUPLICATE . . . FOR STANDBY` command

Oracle9i Data Guard Concepts and Administration to learn how to create and manage standby database

Restrictions and Usage Notes

These restrictions apply to all uses of the `DUPLICATE` command (both for creation of a standby database and creation of a nonstandby duplicate database):

- Issue one or more `ALLOCATE AUXILIARY CHANNEL` commands before executing the `DUPLICATE` command, or `CONFIGURE` automatic auxiliary channels. RMAN uses the automatic target channel configuration for auxiliary channels in the following circumstances:
 - You have not manually allocated auxiliary channels.
 - You have not configured automatic auxiliary channels.
 - The automatic target channels do not have `CONNECT` strings.

Note that the `DUPLICATE` command does not require non-AUXILIARY channels (that is, normal target database channels).

- You must be connected to both the target database and auxiliary instance. The auxiliary instance must be started with the `NOMOUNT` option, and the target database must be mounted or open.
- You cannot duplicate a database when some backups of the target database do not exist unless you specify `SKIP TABLESPACE`. If you do not specify `SKIP TABLESPACE`, then RMAN attempts to duplicate the following:
 - All datafiles in online tablespaces, whether or not the datafiles are online.
 - All tablespaces taken offline with an option *other than* `NORMAL`. For example, RMAN attempts to duplicate tablespaces taken offline with the `IMMEDIATE` option. You cannot duplicate `OFFLINE NORMAL` tablespaces, although you can add these tablespaces manually after duplication.

If no valid backups exist of any tablespace or datafile, then the `DUPLICATE` command fails.

- You can skip all tablespaces in the target database except the `SYSTEM` tablespace, undo tablespaces, and tablespaces containing rollback segments. RMAN does not check for completeness. For example, you can duplicate a data tablespace but not the tablespace containing the index for the data, or duplicate a tablespace that contains only one partition of a partitioned table.
- If the target and duplicate databases reside on the same host, set the `CONTROL_FILES` parameter appropriately so that the `DUPLICATE` command does not generate an error because the target control file is in use.

- If the target and duplicate databases share the same host, set all *_PATH and *_DEST initialization parameters appropriately so that the target database files are not overwritten by the duplicate database files.
- You cannot set the DB_NAME parameter in the duplicate parameter file to a value different from the database name specified in the DUPLICATE command.
- You cannot use the same database name for the target and duplicate databases when the duplicate database resides in the same Oracle home as the target. Note that if the duplicate database resides in a different Oracle home from the target, then its database name just has to differ from other database names in that same Oracle home.
- If the target and duplicate databases reside on different hosts, then you must do one of the following for duplication to be successful:
 - Move backups and disk copies from the target host to the duplicate host and CATALOG the image copies.
 - Make sure that all backups and copies (disk or sbt) on the target host are remotely accessible from the duplicate host. Make sure that the archived redo logs are available in the expected location in the new host.
- If you can make an operating system copy of a database file on one platform and then restore it to another platform, then it is also possible to duplicate a database from one platform to another. Otherwise, duplication is not possible.
- You cannot recover the duplicate database to the current point in time, that is, the most recent SCN. RMAN recovers the duplicate database up to or before the most recent available archived redo log; it cannot recover into the online logs.
- Specify new filenames or convert target filenames for the datafiles and online redo logs when the duplicate filenames must be different from the target filenames (as when duplicating to the same host as the primary). If you do not specify filenames for duplicate online redo logs and datafiles, then RMAN reuses the target datafile names.
- If you want the duplicate filenames to be the same as the target filenames, and if the databases are in different hosts, then you must specify NOFILENAMECHECK.
- If duplicating a database on the same host as the target database, do not specify the NOFILENAMECHECK option. Otherwise, RMAN may signal this error:

```
RMAN-10035: exception raised in RPC: ORA-19504: failed to create file
           "/oracle/dbs/tbs_01.f"
ORA-27086: skgfglk: unable to lock file - already in use
```

```
SVR4 Error: 11: Resource temporarily unavailable
Additional information: 8
RMAN-10031: ORA-19624 occurred during call to
DBMS_BACKUP_RESTORE.RESTOREBACKUPPIECE
```

The following restrictions apply when you use the `DUPLICATE` command with the `FOR STANDBY` option:

- All backups and copies located on disk must be available at the standby host with the same path names as in the target host.
- Backups on tape must be accessible from the standby host.
- If archived logs have not been backed up, then archived logs must be available at the standby host with the same path names as in the target host.
- If RMAN recovers the standby database, then the checkpoint SCN of the control file must be included in an archived redo log that is either available at the standby site or included in an RMAN backup. For example, assume that you create the standby control file and then immediately afterward archive the current log, which has a sequence of 100. You must recover the standby database up to at least log sequence 100, or Oracle signals an `ORA-1152` error message because the standby control file backup or copy was taken after the point in time.
- You cannot use `SET NEWNAME` or `CONFIGURE AUXNAME` to transform the filenames for the online redo logs on the standby database.
- You cannot use the `DUPLICATE` command to activate a standby database.
- You cannot connect to the standby database and then `DUPLICATE . . . FOR STANDBY` to create an additional standby database. To create additional standby databases, connect to the original *primary* database and run `DUPLICATE . . . FOR STANDBY`.
- Do not attempt to register the standby database in the primary database repository.

Keywords and Parameters

Syntax Element	Description
<code>TO 'database_name'</code>	Specifies the name of the duplicate database. The name should match the name in the initialization parameter file of the duplicate database or Oracle signals an error when creating the control file.
<code>dupOptionList</code>	Specifies options that apply when creating a nonstandby duplicate database.

Syntax Element	Description
LOGFILE <i>logSpec</i>	<p>Specifies the online redo logs when creating a nonstandby duplicate database. The syntax is the same used in the LOGFILE option of the CREATE DATABASE statement.</p> <p>Refer to the description of <i>logSpec</i> for the legal options.</p>
NOFILENAMECHECK	<p>Prevents RMAN from checking whether target datafiles sharing the same names as the duplicated files are in use. The user is responsible for determining that the duplicate operation will not overwrite useful data.</p> <p>This option is necessary when you are creating a duplicate database in a different host that has the same disk configuration, directory structure, and filenames as the host of the target database. For example, assume that you have a small database located in the /dbs directory of <i>host1</i>:</p> <pre> /oracle/dbs/system_prod1.dbf /oracle/dbs/users_prod1.dbf /oracle/dbs/tools_prod1.dbf /oracle/dbs/rbs_prod1.dbf </pre> <p>Assume that you want to duplicate the database in machine <i>host2</i>, which has the same file system <code>/oracle/dbs/*</code>, and you want to use the same filenames in the duplicate database as in the primary. In this case, specify the NOFILENAMECHECK option to avoid an error message. Because RMAN is not aware of the different hosts, RMAN cannot determine automatically that it should not check the filenames.</p>
SKIP READONLY	<p>Excludes datafiles in read-only tablespaces from the duplicate database.</p> <p>Note: A record for the skipped read-only tablespace still appears in DBA_TABLESPACES. By using this feature, you can activate the read-only tablespace later. For example, you can store the read-only tablespace data on a CD-ROM, then mount the CD-ROM later and view the data.</p>
SKIP TABLESPACE 'tablespace_name'	<p>Excludes the specified tablespace from the duplicate database. Note that you cannot exclude the SYSTEM tablespace, undo tablespaces, and tablespaces with rollback segments.</p>
<i>untilClause</i>	<p>Sets the end point for incomplete recovery of the duplicate database. You can achieve the same result by running SET UNTIL before the DUPLICATE command.</p> <p>See Also: "<i>untilClause</i>" on page 2-216</p>
DEVICE TYPE <i>deviceSpecifier</i>	<p>Allocates automatic channels for the specific <i>deviceSpecifier</i> only (for example, DISK or sbt). This option is valid only if you have configured automatic channels and have <i>not</i> manually allocated channels. For example, if you CONFIGURE automatic disk and tape channels, and if you run DUPLICATE . . . DEVICE TYPE DISK, then RMAN allocates only disk channels.</p> <p>See Also: "<i>deviceSpecifier</i>" on page 2-104</p>

DUPLICATE

Syntax Element	Description
<code>PFILE = 'filename'</code>	Specifies a client-side initialization parameter used by the auxiliary instance. RMAN automatically shuts down and restarts the auxiliary instance during duplication. If the auxiliary does not use a server-side parameter file in the default location, you must specify the client-side parameter file that RMAN should use when starting the auxiliary instance. Otherwise, you do not need to specify <code>PFILE</code> .
<code>logSpec</code>	Specifies the online redo logs when creating a nonstandby duplicate database. If you do not specify <code>LOGFILE</code> , then RMAN uses <code>LOG_FILE_NAME_CONVERT</code> if it is set. If neither <code>LOGFILE</code> nor <code>LOG_FILE_NAME_CONVERT</code> is set, then RMAN uses the original target log filenames for the duplicate files. You must specify the <code>NOFILENAMECHECK</code> option in this case. See Also: <i>Oracle9i SQL Reference</i> for <code>CREATE DATABASE</code> syntax
<code>'filename' SIZE integer</code>	Specifies the filename of the online redo log member and the size of the file in kilobytes (K) or megabytes (M). The default is in bytes.
<code>REUSE</code>	Allows Oracle to reuse an existing file. If the file already exists, then Oracle verifies that its size matches the value of the <code>SIZE</code> parameter. If the file does not exist, then Oracle creates it.
<code>GROUP integer</code>	Specifies the group containing the online redo log members.
<code>FOR STANDBY</code>	Specifies that database being duplicated is to be used as a standby database. RMAN restores the most recent files (unless <code>SET UNTIL</code> is specified). If <code>DORECOVER</code> is specified, then RMAN also recovers database. RMAN always leaves standby database in mounted state after executing <code>DUPLICATE</code> command.
<code>dupsbyOptionList</code>	Specifies options that only apply when creating a standby database.
<code>DORECOVER</code>	Specifies that RMAN should recover the database after creating it. If you specify an <i>untilClause</i> , then RMAN recovers to the specified point and leaves the database mounted.
<code>NOFILENAMECHECK</code>	Prevents RMAN from checking whether target datafiles sharing the same names as the duplicated files are in use. Note that the <code>NOFILENAMECHECK</code> option is required when the standby and primary datafiles and logs have identical filenames. See Also: The description in <i>dupOptionList</i>

Examples

Setting New Filenames Manually: Example This example assumes that the target database is on `host1` and you wish to duplicate the database to `newdb` on `host2` with the file structure `?/oradata/newdb/*`. Because the filenames in `host1` are

irregularly named and located in various subdirectories, you use SET NEWNAME commands to rename the files consistently. The DUPLICATE command uses backup sets stored on tape to duplicate the target database to database newdb, and recovers it to a point 24 hours ago:

```

RUN
{
  ALLOCATE AUXILIARY CHANNEL newdb1 DEVICE TYPE sbt;
  SET NEWNAME FOR DATAFILE 1 TO '?/oradata/newdb/system01.dbf';
  SET NEWNAME FOR DATAFILE 2 TO '?/oradata/newdb/undotbs01.dbf';
  SET NEWNAME FOR DATAFILE 6 TO '?/oradata/newdb/indx01.dbf';
  SET NEWNAME FOR DATAFILE 7 TO '?/oradata/newdb/tools01.dbf';
  SET NEWNAME FOR DATAFILE 8 TO '?/oradata/newdb/users01.dbf';
  DUPLICATE TARGET DATABASE TO newdb
  UNTIL TIME 'SYSDATE-1' # specifies incomplete recovery
  SKIP TABLESPACE cmlite, drsys, example # skip desired tablespaces
  PFILE = $ORACLE_HOME/dbs/initNEWDB.ora
  LOGFILE
    GROUP 1 ('?/oradata/newdb/redo01_1.f',
             '?/oradata/newdb/redo01_2.f') SIZE 200K,
    GROUP 2 ('?/oradata/newdb/redo02_1.f',
             '?/oradata/newdb/redo02_2.f') SIZE 200K
    GROUP 3 ('?/oradata/newdb/redo03_1.f',
             '?/oradata/newdb/redo03_2.f') SIZE 200K REUSE;
}

```

Reusing the Target Filenames: Example This example assumes the following:

- You are restoring to a new host without a catalog.
- You have configured automatic channels.
- The target host and duplicate host have the same file structure.
- You wish to name the duplicate files exactly like the target database files.
- You do not want to duplicate read-only tablespaces.
- You want to prevent RMAN from checking whether files on the target database that have the same names as the duplicated files are in use.

```

CONNECT TARGET
CONNECT AUXILIARY SYS/aux_pwd@newdb
DUPLICATE TARGET DATABASE TO ndbnewh
LOGFILE
  '?/dbs/log_1.f' SIZE 200K,
  '?/dbs/log_2.f' SIZE 200K
SKIP READONLY
NOFILENAMECHECK;

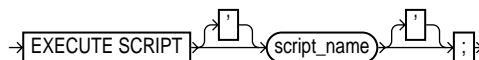
```

Creating a Standby Database: Example This example creates a standby database on a remote host with the same directory structure as the primary host. In this example, the `NOFILENAMECHECK` option is specified because the standby and primary datafiles and logs have the same names. Note that an automatic auxiliary channel is already configured, so you do not need to manually allocate a channel:

```
DUPLICATE TARGET DATABASE FOR STANDBY
NOFILENAMECHECK;
```

EXECUTE SCRIPT

Syntax



Purpose

To run an RMAN script stored in the recovery catalog. Use the [CREATE SCRIPT](#) command to generate stored scripts.

When you run an `EXECUTE SCRIPT` command within a `RUN` command, RMAN places the contents of the script between the braces of `RUN`. For this reason, you should not allocate a channel at the `RUN` command level if you already allocated it in the script.

See Also: *Oracle9i Recovery Manager User's Guide*, and "[CREATE SCRIPT](#)" on page 2-91

Restrictions and Usage Notes

- Execute this command only within the braces of a [RUN](#) command.
- You must be connected to the target database that you connected to when you created or replaced the script.
- RMAN must be connected to the catalog with the `CATALOG` command-line option or the [CONNECT CATALOG](#) command, and the catalog must be open.

Keywords and Parameters

Syntax Element	Description
<code>'script_name'</code>	<p>Runs the specified stored script. To obtain a listing of all stored scripts, use SQL*Plus to connect to the recovery catalog database as the catalog owner and run the following query:</p> <pre>SQL> SELECT * FROM RC_STORED_SCRIPT;</pre> <p>See Also: "RC_STORED_SCRIPT" on page 3-29 for more information about <code>RC_STORED_SCRIPT</code>, and "CREATE SCRIPT" on page 2-91 for information about creating scripts</p>

Example

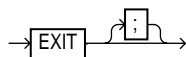
Executing a Script: Example This example creates and then runs a stored script:

```
REPLACE SCRIPT b_whole_10
{
  BACKUP DATABASE PLUS ARCHIVELOG;
}

RUN { EXECUTE script b_whole_10; }
```

EXIT

Syntax



Purpose

To shut down the Recovery Manager utility.

Restrictions and Usage Notes

Execute only at the RMAN prompt.

Example

Exiting RMAN: Example This example starts RMAN and then shuts it down:

```
% rman  
RMAN> EXIT
```

HOST

Syntax



Purpose

To invoke an operating system command-line sub-shell from within RMAN.

Restrictions and Usage Notes

Execute this command at the RMAN prompt or within the braces of a [RUN](#) command.

Keywords and Parameters

Syntax Element	Description
HOST	<p>Enables you to execute an operating system command. Use this parameter:</p> <ul style="list-style-type: none"> With a ' <i>command</i> ', in which case RMAN runs the command in the specified string and then continues. Without a ' <i>command</i> ', in which case RMAN displays a command prompt and resumes after you exit the subshell.

Examples

Executing an Operating System Copy Within RMAN: Example This example shuts down the database, makes a backup of datafile `system01.dbf`, then executes the UNIX `ls` command to display all backed up datafiles:

```

SHUTDOWN IMMEDIATE;
STARTUP MOUNT;
BACKUP DATAFILE '?/oradata/trgt/system01.dbf'
  FORMAT '/tmp/system01.dbf';
HOST 'ls -l /tmp/*dbf';
ALTER DATABASE OPEN;
  
```

Hosting to the Operating System Within a Copy Job: Example This example makes an image copy of `datafile 3`, hosts out to the UNIX prompt to check that the copy is in the directory (the UNIX session output is indented and displayed in bold), then resumes the RMAN session:

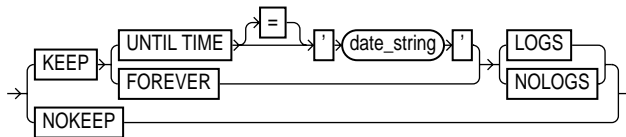
```
RMAN> COPY DATAFILE 3 TO '?/oradata/df3.cpy';
RMAN> HOST;
  % ls $ORACLE_HOME/oradata/df3.cpy
  /net/oracle/oradata/df3.cpy
  % exit
RMAN> LIST COPY;
```

Issuing a SQL*Plus Query Through the Host Command The following example uses `HOST` to start the UNIX shell and then runs a SQL*Plus data dictionary query:

```
HOST " echo 'SELECT * FROM DBA_TABLESPACES;' | sqlplus '/ AS SYSDBA' ";
```

keepOption

Syntax



Purpose

A subclause specifying the status of a backup or copy in relation to a retention policy. The `KEEP` option marks the backup or copy as exempt from the retention policy (that is, not obsolete), and the `NOKEEP` option undoes any existing exemptions.

Keywords and Parameters

Syntax Element	Description
<code>KEEP</code>	Overrides any configured retention policy for this backup or copy so that the backup is not obsolete. The <code>BACKUP . . . KEEP</code> command or <code>COPY . . . KEEP</code> specifies a new retention time for this backup or copy. Use this option to create a long-term backup , that is, a backup that you want to archive.
<code>UNTIL TIME = 'date_string'</code>	Specifies the date until which the backup or copy must be kept. You can either specify a specific time by using the current <code>NLS_DATE_FORMAT</code> , or a SQL date expression, such as <code>'SYSDATE+365'</code> .
<code>FOREVER</code>	Specifies that the backup or copy never expires. You must use a recovery catalog when <code>FOREVER</code> is specified, because the backup records eventually age out of the control file.
<code>LOGS</code>	Specifies that all of the archived logs required to recover this backup or copy must remain available as long as this backup or copy is available.
<code>NOLOGS</code>	Specifies that this backup or copy cannot be recovered because the archived logs needed to recover this backup will not be kept. The only use for this backup or copy is to restore the database to the point in time that the backup or copy was taken. This is the only valid recoverability option when the database operates in <code>NOARCHIVELOG</code> mode. This option is not valid if the backup or copy is inconsistent.

Syntax Element	Description
NOKEEP	Specifies that the backup or copy expires according to the user's retention policy. This is the default behavior if no KEEP option is specified.

Examples

Making a Long-Term Backup: Example This example makes a long-term backup of the database and specifies that it should never become obsolete and that the logs required to recover it should not be retained:

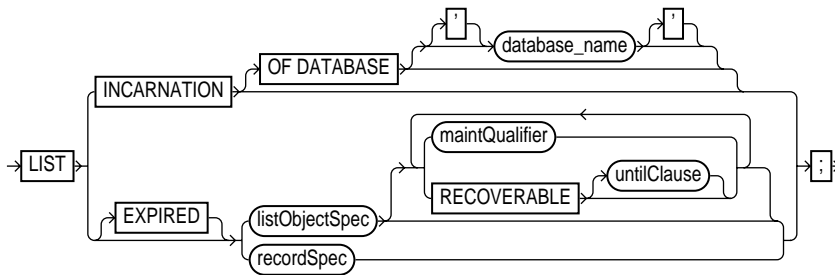
```
SHUTDOWN IMMEDIATE;  
STARTUP MOUNT;  
BACKUP DATABASE  
    KEEP FOREVER NOLOGS;  
ALTER DATABASE OPEN;
```

Changing the Status of a Copy: Example This example specifies that any long-term copies of datafiles and control files should lose their exempt status and so become eligible to be obsolete according to the existing retention policy:

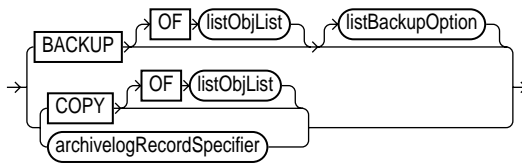
```
CHANGE COPY OF DATABASE CONTROLFILE NOKEEP;
```

LIST

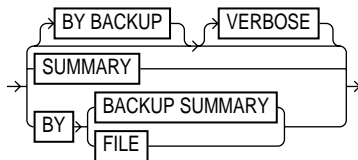
Syntax



listObjectSpec ::=



listBackupOption ::=



Purpose

To display information about backup sets, proxy copies, and image copies recorded in the repository. The `LIST` command displays the files against which you can run `CROSSCHECK` and `DELETE` commands. Use this command to list:

- Backups and copies that do not have the status `AVAILABLE` in the RMAN repository

- Backups and copies of datafiles that are available and can possibly be used in a restore operation
- Specified archived logs, backup sets, backup pieces, control file copies, datafile copies, and proxy copies
- Backups and copies restricted by tag, completion time, recoverability, or device
- Incarnations of a specified database or of all databases known to the catalog

RMAN records the output to either standard output or the message log, but not to both at the same time. You can control how the output is organized (`BY BACKUP` or `BY FILE`) as well as the level of detail in the output (`VERBOSE` or `SUMMARY`).

See Also: *Oracle9i Recovery Manager User's Guide* to learn how to make lists and reports, and "[cmdLine](#)" on page 2-59

Restrictions and Usage Notes

- Execute `LIST` only at the RMAN prompt.
- RMAN must be connected to the target database. If RMAN is connected in `NOCATALOG` mode, then the database must be mounted. If RMAN is connected to a recovery catalog, then the target instance must be started but the target database does not need to be mounted.

Keywords and Parameters

Syntax Element	Description
<code>INCARNATION</code>	<p>Displays information about the incarnations of a database. Whenever you open a database with the <code>RESETLOGS</code> option, then you create a new incarnation of the database. So, if <code>LIST INCARNATION</code> displays <i>n</i> incarnations of a specified database, then you have reset the online logs for this database <i>n-1</i> times.</p> <p>The <code>LIST</code> output includes the primary keys of all database incarnation records for the specified database name. Use the key in a <code>RESET DATABASE</code> command to change the incarnation that RMAN considers to be current to a previous incarnation.</p> <p>See Also: Table 2-18 for an explanation of the column headings of the <code>LIST INCARNATION</code> output table</p>
<code>OF DATABASE</code> <code>'database_name'</code>	<p>Specifies the name of the database. If you do not specify the <code>OF DATABASE</code> option, then the command lists all databases registered in the recovery catalog.</p>

Syntax Element	Description
EXPIRED	<p>Displays backup sets, proxy copies, and image copies marked in the repository as EXPIRED, that is, "not found."</p> <p>To ensure that LIST EXPIRED shows up-to-date output, issue a CROSSCHECK command periodically. When you issue a CROSSCHECK command, RMAN searches on disk and tape for the backups and copies recorded in the repository. If it does not find them, then it updates their repository records to status EXPIRED.</p>
<i>listObjectSpec</i>	<p>Specifies the type of object or objects that you are listing.</p> <p>See Also: "recordSpec" on page 2-149</p>
BACKUP	<p>Displays information about backups: backup sets, backup pieces, and proxy copies. The output displays a unique key for each. The LIST BACKUP command defaults to BY BACKUP.</p> <p>Unless you specify the RECOVERABLE option, RMAN lists both usable and unusable backups, even those that cannot be restored, are expired or unavailable, or are incrementals that cannot be restored because their parent full backup or copy no longer exists.</p> <p>See Also: "LIST Output" on page 2-125 for an explanation of the column headings of the LIST output tables. Use the KEY column of the output to obtain the primary key usable in the CHANGE and DELETE commands.</p>
COPY	<p>Displays information about datafile copies, archived redo logs, and image copies of archived redo logs. By default, LIST COPY displays copies of all database files and archived redo logs. Both usable and unusable image copies are included in the output, even those that cannot be restored or are expired or unavailable.</p> <p>See Also: Table 2-15 and Table 2-17 for an explanation of the column headings of the LIST COPY output tables</p>
OF <i>listObjList</i>	<p>Restricts the list of objects operated on to the object type specified in the <i>listObjList</i> clause. If you do not specify an object, then LIST defaults to OF DATABASE CONTROLFILE ARCHIVELOG ALL.</p> <p>Note: The LIST BACKUP . . . LIKE command is not valid. The only valid exception is LIST BACKUP OF ARCHIVELOG LIKE.</p> <p>See Also: "listObjList" on page 2-140</p>
<i>archivelogRecordSpecifier</i>	<p>Displays information about a range of archived redo logs.</p>
<i>listBackupOption</i>	<p>Specifies the type of object or objects that you are listing.</p> <p>See Also: "recordSpec" on page 2-149</p>

Syntax Element	Description
BY BACKUP	Lists backup sets, then the contents of each backup set (pieces and files), and then proxy copies. This is the default option for <code>LIST BACKUP</code> . If you specify the <code>SUMMARY</code> option, then this command is equivalent to <code>LIST BACKUP SUMMARY</code> .
VERBOSE	Gives detailed description of contents of each backup set (default).
SUMMARY	Gives a one-line summary for each datafile (when using <code>BY FILE</code>) or backup (when using <code>BY BACKUP</code>).
BY FILE	Lists a datafile, then its backup sets (including recovery action info), and then proxy copies.
<i>maintQualifier</i>	Restricts the range of the listing. Refer to " <i>maintQualifier</i> " on page 2-142.
RECOVERABLE	Specifies datafile backups or copies whose status in the repository is <code>AVAILABLE</code> and which can be used for restore and recovery in the target database's current incarnation. This list includes all backups and copies except the following: <ul style="list-style-type: none"> ■ Incremental backups that have no valid parent to which the incremental can be applied. ■ Full backups that were taken before the most recent <code>RESETLOGS</code>, and which have been in online read/write status sometime between the time the backup was taken and the <code>RESETLOGS</code>. In other words, a backup taken prior to the <code>RESETLOGS</code> can be used in the current incarnation only if the file was offline clean or read at the time of the backup, and was never made read/write again before the <code>RESETLOGS</code>.
<i>untilClause</i>	Specifies an end time, SCN, or log sequence number. See " <i>untilClause</i> " on page 2-216.
<i>recordSpec</i>	Specifies the object or objects that you are listing. Refer to " <i>recordSpec</i> " on page 2-149.

LIST Output

The information that appears in the output is shown in the following tables:

- [Table 2-3, "List of Backup Sets \(for datafile backup sets\)"](#)
- [Table 2-4, "List of Backup Pieces \(for sets with only one piece\)"](#)
- [Table 2-5, "List of Datafiles in backup set ..."](#)
- [Table 2-6, "List of Archived Logs in backup set ..."](#)
- [Table 2-7, "Backup Set Copy ... of backup set ... \(only if multiple pieces\)"](#)

- Table 2–8, "List of Backup Pieces for backup set ... Copy ... (mult. pieces)"
- Table 2–9, "List of Proxy Copies"
- Table 2–10, "List of Backup Sets (LIST BACKUP ... SUMMARY)"
- Table 2–11, "List of Backup Pieces (LIST BACKUPPIECE ...)"
- Table 2–12, "List of Datafile Backups (LIST BACKUP ... BY FILE)"
- Table 2–13, "List of Archived Log Backups (LIST BACKUP ... BY FILE)"
- Table 2–14, "List of Controlfile Backups (LIST BACKUP ... BY FILE)"
- Table 2–15, "List of Datafile Copies"
- Table 2–16, "List of Controlfile Copies"
- Table 2–17, "List of Archived Log Copies"
- Table 2–18, "List of Database Incarnations"

Table 2–3 List of Backup Sets (for datafile backup sets)

Column	Indicates
BS Key	A unique key identifying this backup set. If you are connected to a recovery catalog, then BS Key is the primary key of the backup set in the catalog. It corresponds to BS_KEY in the RC_BACKUP_SET view. If you are connected in the default NOCATALOG mode, then BS Key displays the RECID from V\$BACKUP_SET.
Type	The type of backup: Full or Incr (incremental). Note: Column only included in datafile backup sets.
LV	The level of the backup: NULL for nonincrementals, level 0-4 for incrementals. Note: Column only included in datafile backup sets.
Size	The size of the backup in bytes. Note: Column only included in datafile backup sets.
Device Type	The type of device on which the backup was made, for example, DISK or sbt.
Elapsed Time	The duration of the backup.

Table 2–3 List of Backup Sets (for datafile backup sets) (Cont.)

Column	Indicates
Completion Time	The date and time that the backup set completed. Note that the format of this field depends on the <code>NLS_LANG</code> and <code>NLS_DATE_FORMAT</code> environment settings.

Table 2–4 List of Backup Pieces (for sets with only one piece)

Column	Indicates
BP Key	A unique identifier for this backup piece in the recovery catalog or target database control file. If you are connected to a recovery catalog, then <code>BP Key</code> is the primary key of the backup piece in the catalog. It corresponds to <code>BP_KEY</code> in the <code>RC_BACKUP_PIECE</code> view. If you are connected in <code>NOCATALOG</code> mode, then <code>BP Key</code> displays the <code>RECID</code> from <code>V\$BACKUP_PIECE</code> . Note: The values for <code>KEY</code> in the recovery catalog and the control file are different.
Status	The backup piece status: <code>AVAILABLE</code> , <code>UNAVAILABLE</code> , or <code>EXPIRED</code> (refer to the CHANGE , CROSSCHECK , and DELETE commands for an explanation of each status).
Tag	The tag applied to the backup set; <code>NULL</code> if none. Note that tag names are not case sensitive and display in all uppercase.
Piece Name	The filename or handle of the backup piece.
Controlfile Included	A control file is included in the backup. Note: This row appears only if the current control file is included in the backup.
SPFILE Included	A server parameter file is included in the backup.
Ckp SCN	The SCN of the backup control file checkpoint. All database changes recorded in the redo records before the specified SCN are reflected in this control file. Note: This row appears only if the current control file is included in the backup.
Ckp time	The time of the backup control file checkpoint. All database changes recorded in the redo records before the specified time are reflected in this control file. Note: This row appears only if the current control file is included in the backup.

Table 2–5 List of Datafiles in backup set ...

Column	Indicates
File	The number of the file that was backed up.
LV	The level of the backup: NULL for nonincrementals, level 0-4 for incrementals.
Type	The type of backup: Full or Incr (incremental).
Ckp SCN	The checkpoint of the datafile at the time it was backed up. All database changes prior to the SCN have been written to the file; changes after the specified SCN have not been written to the file.
Ckp Time	The checkpoint of the datafile at the time it was backed up. All database changes prior to the time have been written to the file; changes after the specified time have not been written to the file.
Name	The location where this file would be restored now if it were restored from this backup set and no SET NEWNAME command was entered. See Also: "SET" on page 2-195

Table 2–6 List of Archived Logs in backup set ...

Column	Indicates
Thrd	The thread number of the redo log.
Seq	The log sequence number of the archived log.
Low SCN	The lowest SCN in the archived log.
Low Time	The time when Oracle switched into the redo log having this sequence number.
Next SCN	The low SCN of the next archived log sequence.
Next Time	The low time of the next archived log sequence.

Table 2–7 Backup Set Copy ... of backup set ... (only if multiple pieces)

Column	Indicates
Device Type	The type of device on which the backup was made, for example, DISK or sbt.
Elapsed Time	The duration of the backup.

Table 2–7 Backup Set Copy ... of backup set ... (only if multiple pieces) (Cont.)

Column	Indicates
Completion Time	The date and time that the backup set completed. Note that the format of this field depends on the NLS_LANG and NLS_DATE_FORMAT environment settings.
Tag	The tag applied to the backup set; NULL if none. Note that tag names are not case sensitive and display in all uppercase.

Table 2–8 List of Backup Pieces for backup set ... Copy ... (mult. pieces)

Column	Indicates
BP Key	A unique identifier for this backup piece in the recovery catalog or target database control file. If you are connected to a recovery catalog, then BP Key is the primary key of the backup piece in the catalog. It corresponds to BP_KEY in the RC_BACKUP_PIECE view. If you are connected in NOCATALOG mode, then BP Key displays the RECID from V\$BACKUP_PIECE. Note: The values for KEY in the recovery catalog and the control file are different.
Pc#	The number of the backup piece in the backup set.
Status	The backup piece status: AVAILABLE, UNAVAILABLE, or EXPIRED (refer to the CHANGE , CROSSCHECK , and DELETE commands for an explanation of each status).
Piece Name	The filename or handle of the backup piece.

Table 2–9 List of Proxy Copies

Column	Indicates
PC Key	A unique key identifying this proxy copy. If you are connected to a catalog, then PC Key is the primary key of the proxy copy in the catalog. It corresponds to XDF_KEY in the RC_PROXY_DATAFILE view or XCF_KEY in the RC_PROXY_CONTROLFILE view. If you are connected in NOCATALOG mode, then PC Key displays the RECID from V\$PROXY_DATAFILE.
File	The absolute datafile number of the file that was copied.

Table 2–9 List of Proxy Copies (Cont.)

Column	Indicates
Status	The proxy copy status: AVAILABLE, UNAVAILABLE, or EXPIRED (see the CHANGE , CROSSCHECK , and DELETE commands for an explanation of each status).
Completion Time	The date and time that the backup set completed. Note that the format of this field depends on the NLS_LANG and NLS_DATE_FORMAT environment settings.
Ckp SCN	The SCN of the proxy copy control file checkpoint. All database changes recorded in the redo records before the specified SCN are reflected in this control file.
Ckp time	The time of the proxy copy control file checkpoint. All database changes recorded in the redo records before the specified time are reflected in this control file.
Datafile name	The location where this file would be restored now if it were restored from this backup set and no SET NEWNAME command was entered. See Also: "SET" on page 2-195
Handle	The media manager's handle for the proxy copy.
Tag	The tag applied to the proxy copy; NULL if none. Note that tag names are not case sensitive and display in all uppercase.

Table 2–10 List of Backup Sets (LIST BACKUP ... SUMMARY)

Column	Indicates
Key	A unique key identifying this backup set. If you are connected to a recovery catalog, then BS Key is the primary key of the backup set in the catalog. It corresponds to BS_KEY in the RC_BACKUP_SET view. If you are connected in NOCATALOG mode, then BS Key displays the RECID from V\$BACKUP_SET.
TY	The type of backup: backup set (B) or proxy copy (P).
LV	The level of the backup: NULL for nonincrementals, level 0-4 for incrementals.
S	The status of the backup: A (available), U (unavailable), or X (all backup pieces in set expired). Refer to the CHANGE , CROSSCHECK , and DELETE commands for an explanation of each status.

Table 2–10 List of Backup Sets (LIST BACKUP ... SUMMARY) (Cont.)

Column	Indicates
Device Type	The type of device on which the backup was made, for example, DISK or sbt.
Completion Time	The date and time that the backup set completed. Note that the format of this field depends on the NLS_LANG and NLS_DATE_FORMAT environment settings.
#Pieces	The number of backup pieces in the backup set.
#Copies	The number of copies made of each backup piece in the set. The number is 1 if no duplexing was performed. Otherwise, the value ranges from 2 to 4.
Tag	The tag applied to the backup set; NULL if none. An asterisk (*) indicates multiple copies with different tags. Note that tag names are not case sensitive and display in all uppercase.

Table 2–11 List of Backup Pieces (LIST BACKUPPIECE ...)

Column	Indicates
BP Key	<p>A unique identifier for this backup piece in the recovery catalog or target database control file.</p> <p>If you are connected to a catalog, then BP Key is the primary key of the backup piece in the catalog. It corresponds to BP_KEY in the RC_BACKUP_PIECE view. If you are connected in NOCATALOG mode, then BP Key displays the RECID from V\$BACKUP_PIECE.</p> <p>Note: The values for KEY in the recovery catalog and the control file are different.</p>
BS Key	<p>A unique key identifying this backup set.</p> <p>If you are connected to a recovery catalog, then BS Key is the primary key of the backup set in the catalog. It corresponds to BS_KEY in the RC_BACKUP_SET view. If you are connected in NOCATALOG mode, then BS Key displays the RECID from V\$BACKUP_SET.</p>
Pc#	The number of the backup piece in the backup set.
Cp#	The copy number of this backup piece in the backup set. The number is 1 if no duplexing was performed. Otherwise, the value ranges from 2 to 4.

Table 2–11 List of Backup Pieces (LIST BACKUPPIECE ...) (Cont.)

Column	Indicates
Status	The backup piece status: AVAILABLE, UNAVAILABLE, or EXPIRED (see the CHANGE , CROSSCHECK , and DELETE commands for an explanation of each status).
Device Type	The type of device on which the backup was made, for example, DISK or sbt.
Piece Name	The filename or handle of the backup piece.

Table 2–12 List of Datafile Backups (LIST BACKUP ... BY FILE)

Column	Indicates
File	The absolute datafile number.
Key	A unique key identifying this backup set. If you are connected to a recovery catalog, then <code>Key</code> is the primary key of the backup set in the catalog. It corresponds to <code>BS_KEY</code> in the <code>RC_BACKUP_SET</code> view. If you are connected in <code>NOCATALOG</code> mode, then <code>Key</code> displays the <code>RECID</code> from <code>V\$BACKUP_SET</code> .
TY	The type of backup: backup set (B) or proxy copy (P).
LV	The backup level: F for nonincrementals, level 0-4 for incrementals.
S	The status of the backup: A (available), U (unavailable), or X (all backup pieces in set expired). Refer to the CHANGE , CROSSCHECK , and DELETE commands for an explanation of each status.
Ckp SCN	The checkpoint of the datafile at the time it was backed up. All database changes prior to the SCN have been written to the file; changes after the specified SCN have not been written to the file.
Ckp Time	The checkpoint of the datafile at the time it was backed up. All database changes prior to the time have been written to the file; changes after the specified time have not been written to the file.
#Pieces	The number of backup pieces in the backup set.
#Copies	The number of copies made of each backup piece in the set. The number is 1 if no duplexing was performed. Otherwise, the value ranges from 2 to 4.

Table 2–12 List of Datafile Backups (LIST BACKUP ... BY FILE) (Cont.)

Column	Indicates
Tag	The tag applied to the backup set; NULL if none. Note that tag names are not case sensitive and display in all uppercase.

Table 2–13 List of Archived Log Backups (LIST BACKUP ... BY FILE)

Column	Indicates
Thrd	The thread number of the redo log.
Seq	The log sequence number of the archived log.
Low SCN	The lowest SCN in the archived log.
Low Time	The time when Oracle switched into the redo log having this sequence number.
BS Key	A unique key identifying this backup set. If you are connected to a recovery catalog, then BS Key is the primary key of the backup set in the catalog. It corresponds to BS_KEY in the RC_BACKUP_SET view. If you are connected in NOCATALOG mode, then BS Key displays the RECID from V\$BACKUP_SET.
S	The status of the backup: A (available), U (unavailable), or X (all backup pieces in set expired). Refer to the CHANGE , CROSSCHECK , and DELETE commands for an explanation of each status.
#Pieces	The number of backup pieces in the backup set.
#Copies	The number of copies made of each backup piece in the set. The number is 1 if no duplexing was performed. Otherwise, the value ranges from 2 to 4.
Tag	The tag applied to the backup set; NULL if none. Note that tag names are not case sensitive and display in all uppercase.

Table 2–14 List of Controlfile Backups (LIST BACKUP ... BY FILE)

Column	Indicates
CF Ckp SCN	Checkpoint SCN of the control file.
Ckp Time	The log sequence number of the archived log.
BS Key	A unique key identifying this backup set. If you are connected to a recovery catalog, then BS Key is the primary key of the backup set in the catalog. It corresponds to BS_KEY in the RC_BACKUP_SET view. If you are connected in NOCATALOG mode, then BS Key displays the RECID from V\$BACKUP_SET.
S	The status of the backup: A (available), U (unavailable), or X (all backup pieces in set expired). Refer to the CHANGE , CROSSCHECK , and DELETE commands for an explanation of each status.
#Pieces	The number of backup pieces in the backup set.
#Copies	The number of copies made of each backup piece in the set. The number is 1 if no duplexing was performed. Otherwise, the value ranges from 2 to 4.
Tag	The tag applied to the backup set; NULL if none. Note that tag names are not case sensitive and display in all uppercase.

Table 2–15 List of Datafile Copies

Column	Indicates
Key	The unique identifier for the datafile copy. Use this value in a CHANGE command to alter the status of the datafile copy. If you are connected to a recovery catalog, then Key is the primary key of the datafile copy in the catalog. It corresponds to CDF_KEY in the RC_DATAFILE_COPY view. If you are connected in NOCATALOG mode, then Key displays the RECID from V\$DATAFILE_COPY. Note: The values for KEY in the recovery catalog and the control file are different.
File	The file number of the datafile from which this copy was made.
S	The status of the copy: A (available), U (unavailable), or X (expired). Refer to the CHANGE , CROSSCHECK , and DELETE commands for an explanation of each status.

Table 2–15 List of Datafile Copies (Cont.)

Column	Indicates
Completion Time	The date and time that the copy completed. Note that the value of this field is sensitive to the <code>NLS_LANG</code> and <code>NLS_DATE_FORMAT</code> environment variables.
Ckp SCN	The checkpoint of this datafile when it was copied. All database changes prior to this SCN have been written to this datafile.
Ckp TIME	The checkpoint of this datafile when it was copied. All database changes prior to this time have been written to this datafile.
Name	The filename of the datafile copy.

Table 2–16 List of Controlfile Copies

Column	Indicates
Key	The unique identifier for the control file copy. Use this value in a <code>CHANGE</code> command to alter the status of the copy. If you are connected to a recovery catalog, then <code>Key</code> is the primary key of the control file copy in the catalog. It corresponds to <code>CCF_KEY</code> in the <code>RC_CONTROLFILE_COPY</code> view. If you are connected in <code>NOCATALOG</code> mode, then <code>Key</code> displays the <code>RECID</code> from <code>V\$DATAFILE_COPY</code> . Note: The values for <code>Key</code> in the recovery catalog and the control file are different.
S	The status of the copy: A (available), U (unavailable), or X (expired). Refer to the <code>CHANGE</code> , <code>CROSSCHECK</code> , and <code>DELETE</code> commands for an explanation of each status.
Completion Time	The date and time that the copy completed. Note that the value of this field is sensitive to the <code>NLS_LANG</code> and <code>NLS_DATE_FORMAT</code> environment variables.
Ckp SCN	The checkpoint of this control file when it was copied.
Ckp TIME	The checkpoint of this control file when it was copied.
Name	The filename of the control file copy.

Table 2–17 List of Archived Log Copies

Column	Indicates
Key	<p>The unique identifier for this archived redo log copy. Use this value in a CHANGE command to alter the status of the copy.</p> <p>If you are connected to a recovery catalog, then <code>Key</code> is the primary key of the backup set in the catalog. It corresponds to <code>AL_KEY</code> in the <code>RC_ARCHIVED_LOG</code> view. If you are connected in <code>NOCATALOG</code> mode, then <code>Key</code> displays the <code>RECID</code> from <code>V\$ARCHIVED_LOG</code>.</p> <p>Note: The values for <code>Key</code> in the recovery catalog and the control file are different.</p>
Thrd	The redo log thread number.
Seq	The log sequence number.
S	The status of the copy: <code>A</code> (available), <code>U</code> (unavailable), or <code>X</code> (expired). Refer to the CHANGE , CROSSCHECK , and DELETE commands for an explanation of each status.
Low Time	The time when Oracle switched into the redo log having this sequence number.
Name	The filename of the archived redo log copy.

Table 2–18 List of Database Incarnations

Column	Indicates
DB Key	When combined with the <code>Inc Key</code> , the unique key by which RMAN identifies the database incarnation in the recovery catalog. Use this key to unregister a database, that is, delete all the rows associated with that database from the recovery catalog.
Inc Key	When combined with <code>DB Key</code> , the unique key by which RMAN identifies the database incarnation in the recovery catalog. Use this key in <code>RESET DATABASE . . . TO INCARNATION</code> when recovering the database to a time before the most recent <code>RESETLOGS</code> .
DB Name	The database name as listed in the <code>DB_NAME</code> parameter.
DB ID	The database identification number, which Oracle generates automatically at database creation.
CUR	Whether the incarnation is the current incarnation of the database.

Table 2–18 List of Database Incarnations (Cont.)

Column	Indicates
Reset SCN	The SCN at which the incarnation was created.
Reset Time	the time at which the incarnation was created.

Examples

Listing Backups: Example This example lists all backups in default verbose mode:

```
LIST BACKUP;
```

```
List of Backup Sets
```

```
=====
```

```
BS Key Device Type Elapse Time Completion Time
```

```
-----
```

```
236 DISK 00:00:08 21-SEP-00
    BP Key: 237 Status: AVAILABLE Tag: TAG20011121T053733
    Piece Name: /oracle/oradata/09c5unih_1_1
```

```
List of Archived Logs in backup set 236
```

```
Thrd Seq Low SCN Low Time Next SCN Next Time
```

```
-----
```

```
1 141 49173 21-SEP-01 49784 21-SEP-01
1 142 49784 21-SEP-01 50331 21-SEP-01
```

```
BS Key Type LV Size Device Type Elapse Time Completion Time
```

```
-----
```

```
244 Full 61M DISK 00:00:18 21-SEP-01
    BP Key: 245 Status: AVAILABLE Tag: TAG20011121T053816
    Piece Name: /oracle/oradata/0ac5unj5_1_1
```

```
Controlfile Included: Ckp SCN: 51554 Ckp time: 21-SEP-01
```

```
SPFILE Included: Modification time: 21-SEP-01
```

```
List of Datafiles in backup set 244
```

```
File LV Type Ckp SCN Ckp Time Name
```

```
-----
```

```
1 Full 51555 21-SEP-01 /oracle/oradata/trgt/system01.dbf
2 Full 51555 21-SEP-01 /oracle/oradata/trgt/undotbs_01.dbf
```

```
List of Proxy Copies
```

```
=====
```

```
PC Key File Status Completion time Ckp SCN Ckp time
```

```
-----
```

```
552 1 AVAILABLE 10/07/2001 03:05:21 78022 10/07/2001 03:05:10
    Datafile name: /oracle/oradata/trgt/system01.dbf
```

Handle: 0jb81876_1_0

```
561      1  AVAILABLE  10/07/2001 03:38:22 78025      10/07/2001 03:38:09
Datafile name: /oracle/oradata/trgt/system01.dbf
Handle: 0lb81a51_1_0
Tag: WKLYBKUP
```

Listing a Summary of Backups: Example The following example lists a summarized version of all RMAN backups:

```
LIST BACKUP SUMMARY;
```

List of Backups

=====

Key	TY	LV	S	Device	Type	Completion Time	#Pieces	#Copies	Tag
35	B	A	A	SBT_TAPE		FEB 08 2002 05:37:37	1	1	TAG20020208T053733
42	B	F	A	SBT_TAPE		FEB 08 2002 05:38:21	1	1	TAG20020208T053744

Listing Backups by File: Example This example groups all backups by file:

```
LIST BACKUP BY FILE;
```

List of Datafile Backups

=====

File Key	TY	LV	S	Ckp SCN	Ckp Time	#Pieces	#Copies	Tag
1	502	B	0	A 37973	09/28/01 19:28:36	1	3	*
	552	P	F	X 78022	10/07/01 03:05:10	1	1	DF_1
	561	P	0	U 78025	10/07/01 03:38:09	2	1	DF_1
2	502	B	0	A 37973	09/28/01 19:28:36	1	2	*
	562	P	0	U 78027	10/07/01 03:38:22	1	1	DF_2

List of Archived Log Backups

=====

Thrd	Seq	Low SCN	Low Time	BS Key	S	#Pieces	#Copies	Tag
1	141	49463	14-SEP-01	213	A	1	1	TAG20011114T125431

List of Controlfile Backups

=====

CF Ckp SCN	Ckp Time	BS Key	S	#Pieces	#Copies	Tag
51593	14-SEP-01 222	A	1	1	1	

List of SPFILE Backups

=====

Modification Time	BS Key	S	#Pieces	#Copies	Tag
OCT 08 2001 05:38:55	251	A	1	1	

Listing Archived Redo Logs: Example The following example lists archived logs and copies of logs:

```
LIST COPY OF DATABASE ARCHIVELOG ALL;
```

List of Archived Log Copies

Key	Thrd	Seq	S	Low Time	Name
153	1	30	A	14-SEP-01	/oracle/oradata/trgt/arch/archive1_30.dbf
154	1	31	A	14-SEP-01	/oracle/oradata/trgt/arch/archive1_31.dbf

Listing Backups of Specific Datafiles: Example The following example lists backups of datafile 3 in summary mode:

```
LIST BACKUP OF DATAFILE 3 SUMMARY;
```

List of Backups

=====

Key	TY	LV	S	Device	Type	Completion Time	#Pieces	#Copies	Tag
180	B	0	A	DISK		14-SEP-01	1	2	TAG20011114T125431

Listing Database Incarnations: Example This example lists all database incarnations recorded in the recovery catalog:

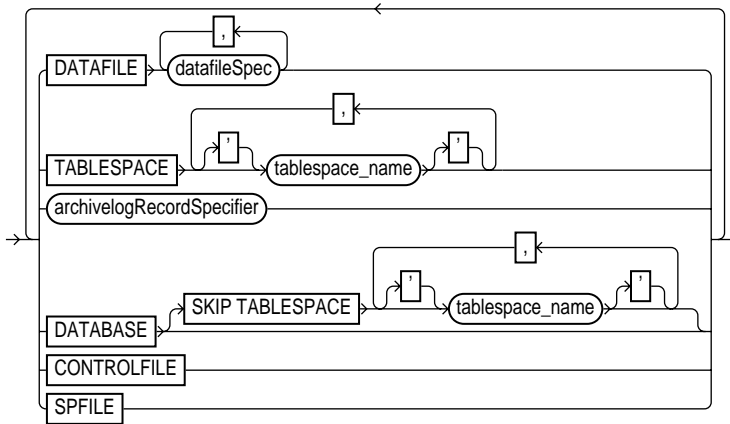
```
LIST INCARNATION;
```

List of Database Incarnations

DB Key	Inc Key	DB Name	DB ID	CUR	Reset SCN	Reset Time
1	2	RDBMS	674879952	YES	1	14-SEP-01

listObjList

Syntax



Purpose

A subclause used to specify database files and archived redo logs.

Restrictions and Usage Notes

Use this clause in the following commands:

- [CHANGE](#)
- [CROSSCHECK](#)
- [DELETE](#)
- [LIST](#)

Keywords and Parameters

Syntax Element	Description
DATAFILE <i>datafileSpec</i>	Specifies datafiles by filename or file number. The clause specifies datafile image copies or backup sets that contain at least one of the datafiles. See Also: " datafileSpec " on page 2-96

Syntax Element	Description
TABLESPACE <i>'tablespace_name'</i>	Specifies tablespace names. The clause specifies datafile image copies or backup sets that contain at least one of the datafile from the specified tablespace.
<i>archivelogRecordSpecifier</i>	Specifies a range of archived redo logs. See Also: " <i>archivelogRecordSpecifier</i> " on page 2-21
DATABASE	Specifies backup sets or image copies of all files in the current database.
SKIP TABLESPACE <i>'tablespace_name'</i>	Omits the specified tablespaces from the DATABASE specification.
CONTROLFILE	Specifies the current control file.
SPFILE	Specifies the current server parameter file.

Examples

Listing Datafile Copies: Example The following command lists image copies of all the files in the database, skipping the `temp` tablespace, which is a dictionary-managed temporary tablespace:

```
LIST COPY OF DATABASE SKIP TABLESPACE temp;
```

Crosschecking Archived Redo Logs: Example The following example queries the media manager for the status of server parameter file and archived redo log backups created in the last three months:

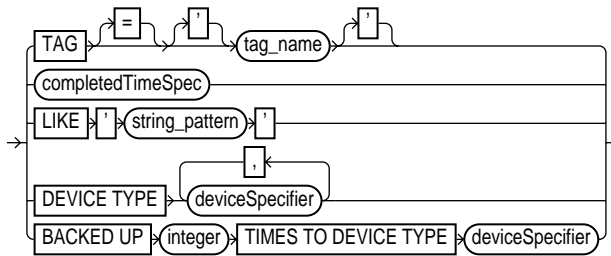
```
ALLOCATE CHANNEL FOR MAINTENANCE DEVICE TYPE sbt;
CROSSCHECK BACKUP OF SPFILE ARCHIVELOG FROM TIME 'SYSDATE-90';
```

Deleting Expired Control File Backup Sets: Example The following command deletes expired backups of the control file:

```
DELETE EXPIRED BACKUP OF CONTROLFILE;
```

maintQualifier

Syntax



Purpose

A subclause used to specify database files and archived redo logs.

Restrictions and Usage Notes

- Use this clause in the following commands:
 - [LIST](#)
 - [CROSSCHECK](#)
 - [DELETE](#)
- The `BACKED UP integer TIMES` clause applies only to archived redo logs.
- You cannot use `LIKE` with backup pieces.

Keywords and Parameters

Syntax Element	Description
TAG = 'tag_name'	Specifies the datafile copies and backup sets by tag. Tag names are not case sensitive and display in all uppercase. See Also: " BACKUP " on page 2-26 for a description of how a tag can be applied to an individual copy of a duplexed backup set, and also for a description of the default filename format for tags

Syntax Element	Description
<i>completedTimeSpec</i>	Specifies a range of time for completion of the backup or copy. See Also: " <i>completedTimeSpec</i> " on page 2-64
LIKE ' <i>string_pattern</i> '	Restricts datafile copies by specifying a filename pattern. The pattern can contain Oracle pattern matching characters % and _. RMAN only operates on those files whose name matches the pattern. Note: You cannot use the LIKE option with the LIST . . . ARCHIVELOG command.
DEVICE TYPE <i>deviceSpecifier</i>	Allocates automatic channels for the specified device type only. This option is valid only if you have configured automatic channels and have not manually allocated channels. For example, if you configure automatic disk and tape channels, and issue CHANGE . . . DEVICE TYPE DISK, then RMAN allocates only disk channels. See Also: " <i>deviceSpecifier</i> " on page 2-104
BACKED UP <i>integer</i> TIMES TO DEVICE TYPE <i>deviceSpecifier</i>	Restricts the command to archived logs that have been successfully backed up <i>integer</i> or more times to the specified media.

Example

Listing Backups on a Specific Device: Example The following command lists all backups located on tape and copies located in /tmp:

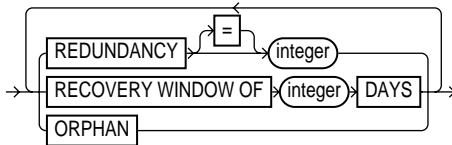
```
LIST BACKUP DEVICE TYPE sbt;
LIST COPY LIKE '/tmp';
```

Deleting Archived Logs That Are Already Backed Up: Example The following command deletes only those archived logs that have been successfully backed up three or more times to tape:

```
DELETE ARCHIVELOG ALL BACKED UP 3 TIMES TO DEVICE TYPE sbt;
```

obsOperandList

Syntax



Purpose

A subclause used to specify which criteria are used to mark backups and copies as obsolete.

Restrictions and Usage Notes

Use this clause in the following commands:

- `DELETE`
- `REPORT`

Keywords and Parameters

Syntax Element	Description
<code>REDUNDANCY = integer</code>	Specifies the minimum level of redundancy considered necessary for a backup or copy to be obsolete. A datafile copy is obsolete if there are at least <i>integer</i> more recent backups or image copies of this file; a datafile backup set is obsolete if there are at least <i>integer</i> more recent backups or image copies of each file contained in the backup set. For example, <code>REDUNDANCY 2</code> means that there must be at least two more recent backups or copies of a datafile for any other backup or copy to be obsolete.
<code>RECOVERY WINDOW OF integer DAYS</code>	Specifies that RMAN should report as obsolete those backups and copies that are not needed to recover the database to any point within the last <i>integer</i> days. See Also: " CONFIGURE " on page 2-66 for an explanation of the recovery window

Syntax Element	Description
ORPHAN	<p>Specifies as obsolete those backups and copies that are unusable because they belong to incarnations of the database that are not direct ancestors of the current incarnation. Note that RMAN displays orphaned backups <i>in addition to</i> the normal display of obsolete backups.</p> <p>See Also: <i>Oracle9i Recovery Manager User's Guide</i> for an explanation of orphaned backups</p>

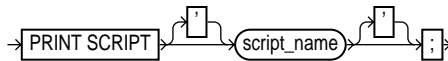
Example

Deleting Obsolete Backups: Example The following command deletes all backups and copies not needed to recover the database to a random point within the last 30 days:

```
DELETE OBSOLETE RECOVERY WINDOW OF 30 DAYS;
```

PRINT SCRIPT

Syntax



Purpose

To print a stored script to standard output or the RMAN message log. To print to a message log, specify the log filename with the `LOG` argument at the command line. If you do not specify this argument, Recovery Manager writes message output to standard output.

Note: You can also display the individual lines of the stored scripts by querying the `RC_STORED_SCRIPT_LINE` recovery catalog view.

Restrictions and Usage Notes

- Use this command only at the RMAN prompt.
- You must be connected to the target database that you connected to when you created or replaced the script.
- You must be connected to the recovery catalog, and the recovery catalog database must be open.

Keywords and Parameters

Syntax Element	Description
<code>'script_name'</code>	<p>Prints a stored script with the specified name to standard output or a message log. To obtain a listing of all stored scripts, use SQL*Plus connect to the recovery catalog as the catalog owner and issue the following query:</p> <pre>SQL> SELECT * FROM RC_STORED_SCRIPT;</pre> <p>Note: To run the script, use <code>EXECUTE SCRIPT</code> within the braces of the <code>RUN</code> command.</p> <p>See Also: "RC_STORED_SCRIPT" on page 3-29</p>

Examples

Printing a Script to the Message Log: Example The following shell script creates the backup_db recovery catalog script and prints it to rman_log. Finally, it executes the script:

```
#!/usr/bin/tcsh
rman TARGET / CATALOG rman/rman@rcatdb LOG = rman_log <<EOF
REPLACE SCRIPT backup_db
{
    ALLOCATE CHANNEL c1 DEVICE TYPE sbt;
    BACKUP DATABASE;
}
PRINT SCRIPT backup_db;
RUN
{
    EXECUTE SCRIPT backup_db;
}
EOF
```

Printing a Script to the Screen: Example This example prints a stored script to standard output (includes sample output):

```
PRINT SCRIPT backup_db;

printing stored script: backup_db
{
    ALLOCATE CHANNEL c1 DEVICE TYPE sbt
    BACKUP DATABASE;
}
```

QUIT

Syntax

→ QUIT →

Purpose

To shut down the Recovery Manager utility.

Restrictions and Usage Notes

Execute only at the RMAN prompt.

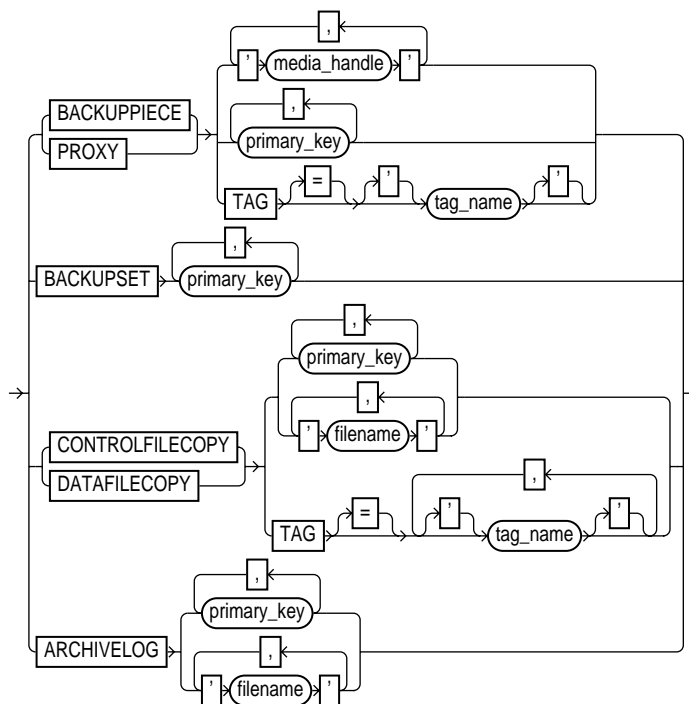
Example

Quitting RMAN: Example This example starts RMAN and then shuts it down:

```
% rman
RMAN> QUIT
```

recordSpec

Syntax



Purpose

A subclass that specifies which objects the [CHANGE](#), [CROSSCHECK](#), [DELETE](#), and [LIST](#) commands should operate on. Most *recordSpec* options allow you to specify a primary key. Run the `LIST` command to obtain the key.

Keywords and Parameters

Syntax Element	Description
BACKUPPIECE	Specifies a backup piece by ' <i>media_handle</i> ', <i>primary_key</i> , or <i>tag_name</i> .

Syntax Element	Description
PROXY	Specifies a proxy copy by ' <i>media_handle</i> ', <i>primary_key</i> , or <i>tag_name</i> .
BACKUPSET <i>primary_key</i>	Specifies a backup set by <i>primary_key</i> .
CONTROLFILECOPY	Specifies a control file copy by <i>primary_key</i> , filename pattern (' <i>filename</i> '), or TAG = <i>tag_name</i> . If you crosscheck a control file copy, you must specify a filename rather than a primary key.
DATAFILECOPY	Specifies a datafile copy by either <i>primary_key</i> , filename pattern (' <i>filename</i> '), or TAG = <i>tag_name</i> .
ARCHIVELOG	Specifies an archived redo log by either <i>primary_key</i> or ' <i>filename</i> '.

Examples

Crosschecking Backups: Example This example crosschecks backup sets specified by primary key:

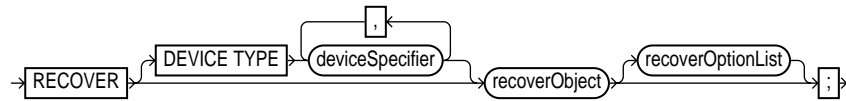
```
CROSSCHECK BACKUPSET 507, 508, 509;
```

Deleting Datafile Copies: Example This example deletes a specified datafile copy:

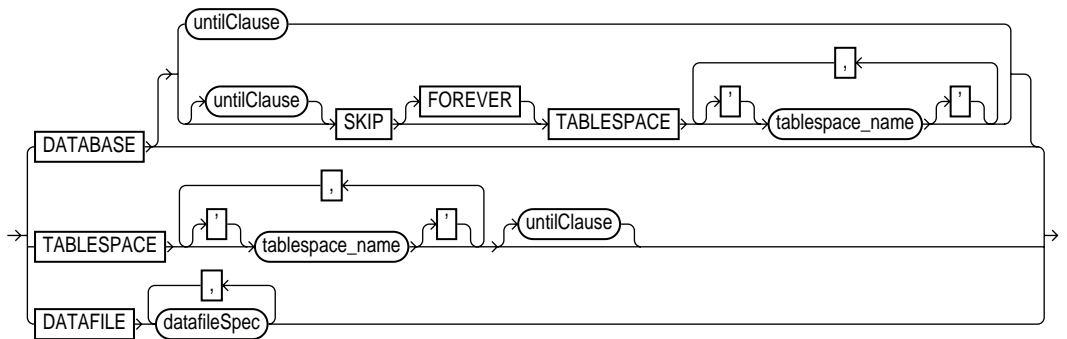
```
DELETE NOPROMPT DATAFILECOPY '?/oradata/users01.cpy';
```

RECOVER

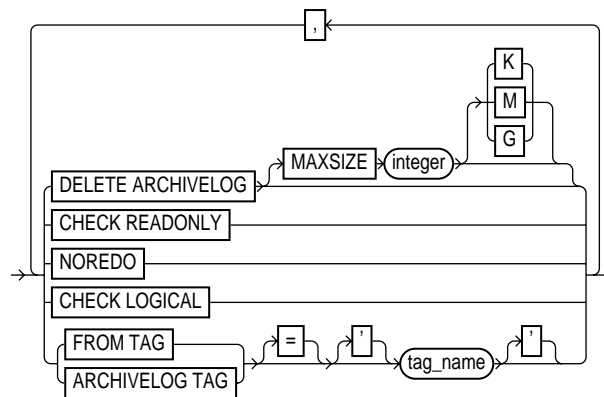
Syntax



recoverObject ::=



recoverOptionList ::=



Purpose

To apply redo logs or incremental backups to one or more restored datafiles in order to update them to a specified time.

RMAN first looks for archived logs or image copies on disk, and if none are available, then it restores logs from backups as needed for the recovery. If you specify `DELETE ARCHIVELOG`, then RMAN deletes the logs after restoring them.

If RMAN has a choice between applying an incremental backup or applying redo, then it always chooses the incremental backup. If overlapping levels of incremental backup are available, then RMAN automatically chooses the one covering the longest period of time. Note that RMAN can apply incremental backups to restored files that were not created as part of an incremental backup.

Note: When RMAN applies incremental backups, it recovers changes to objects created with the `NOLOGGING` option. Applying archived redo logs to datafiles does not recover these changes.

See Also: *Oracle9i Recovery Manager User's Guide* to learn how to recover datafiles

Restrictions and Usage Notes

- Recovery is possible with a backup control file and no recovery catalog, but requires manual intervention if you added files to the database after the backup control file was created.
- If you have not configured channels, then at least one `ALLOCATE CHANNEL` command must precede `RECOVER` unless you do not need to restore archived redo log or incremental backups.
- You do not have to allocate a channel for archived log recovery because RMAN uses the preconfigured `DISK` channel. If incremental backups need to be restored during recovery, then you must either use configured channels or manually allocate channels of the same type that created these backups.
- For datafile and tablespace recovery, the target database must be mounted. If it is open, then the datafiles or tablespaces to be recovered must be offline. For database recovery, the database must be mounted but not open.
- Only the current datafiles may be recovered or have incremental backups applied to them.

- If you want to perform incomplete recovery, the best practice is to enter a `SET UNTIL` command before both the `RESTORE` and `RECOVER` commands. If you run `SET UNTIL` after a `RESTORE` and before a `RECOVER`, then you may not be able to perform media recovery on the database to the time required because the restored files may have time stamps later than the specified time.
- The `RECOVER DATABASE` command does not recover any files that are offline normal or read-only at the point in time to which the files are being recovered. RMAN omits offline normal files with no further checking. If `CHECK READONLY` is specified, then RMAN checks each read-only file on disk to ensure that it is already current at the desired point in time. If `CHECK READONLY` is not specified, then RMAN omits read-only files.
- You must open the database with the `RESETLOGS` option after incomplete recovery or recovery with a backup control file.
- You cannot recover temporary tablespaces: you can only re-create them. Note that if you recover the database after restoring a backup control file, you must add new tempfiles to locally-managed temporary tablespaces.
- You must have already configured a device type with the `CONFIGURE DEVICE TYPE` command (except for `DISK`, which is preconfigured) before specifying the `DEVICE TYPE` option.
- You cannot manually allocate channels and then run `RECOVER` with the `DEVICE TYPE` option.

Keywords and Parameters

Syntax Element	Description
<code>DEVICE TYPE</code> <i>deviceSpecifier</i>	Allocates automatic channels for the specified device type only. For example, if you configure automatic disk and tape channels, and issue <code>RECOVER . . . DEVICE TYPE DISK</code> , then RMAN allocates only disk channels. See Also: "deviceSpecifier" on page 2-104
<code>DATABASE</code> <i>untilClause</i>	Specifies that the entire database is to be recovered. Unless you specify an <i>untilClause</i> , RMAN performs complete recovery. Specifies a noncurrent time, SCN, or log sequence number for termination of the <code>RECOVER</code> command. You must open the database with the <code>RESETLOGS</code> option after incomplete recovery. See Also: "untilClause" on page 2-216

Syntax Element	Description
SKIP [FOREVER] TABLESPACE	<p>Lists tablespaces that should not be recovered, which is useful for avoiding recovery of tablespaces containing only temporary data or for postponing recovery of some tablespaces. The <code>SKIP</code> clause takes the datafiles in the specified tablespaces offline before starting media recovery. These files are left offline after the media recovery is complete.</p> <p>If you perform incomplete recovery, then <code>SKIP</code> is not allowed. Instead, use <code>SKIP FOREVER</code>, with the intention of dropping the skipped tablespaces after opening the database with the <code>RESETLOGS</code> option. The <code>SKIP FOREVER</code> clause causes RMAN to take the datafiles offline with the <code>DROP</code> option. Only use <code>SKIP FOREVER</code> when the specified tablespaces will be dropped after opening the database.</p>
TABLESPACE 'tablespace_name' <i>untilClause</i>	<p>Specifies tablespaces by tablespace name.</p> <p>Specifies a noncurrent time, SCN, or log sequence number for termination of the <code>RECOVER</code> command in <code>TSPITR</code>.</p> <p>See Also: "untilClause" on page 2-216</p>
DATAFILE <i>datafileSpec</i>	<p>Specifies a list of one or more datafiles to recover. Specify datafiles by either filename (by using a quoted string) or absolute datafile number (by using an integer).</p> <p>If you are using the control file as the exclusive repository for RMAN metadata, then the filename must be the name of the datafile as known in the control file.</p> <p>If you are using a recovery catalog, then the filename of the datafile must be the most recent name recorded in the catalog. For example, assume that a datafile was renamed in the control file. The instance then fails before you can resynchronize the catalog. Specify the old name of the datafile in the <code>RECOVER</code> command, because this is the name recorded in the catalog.</p> <p>See Also: "datafileSpec" on page 2-96</p>
<i>recoverOptionList</i> DELETE ARCHIVELOG [MAXSIZE <i>integer</i> [K M G]]	<p>Specifies various recovery options.</p> <p>Deletes archived logs restored from backups or copies that are no longer needed. RMAN does not delete archived logs that were already on disk before the <code>RESTORE</code> command started.</p> <p>If you do not specify <code>MAXSIZE</code>, then RMAN deletes restored archived logs as they are applied. If you specify <code>MAXSIZE</code>, then RMAN will not use more than <i>integer</i> amount of disk space for restored archived logs. If recovery requires the restore of a log larger than the <code>MAXSIZE</code> value, then RMAN reports an error indicating that you should increase the <code>MAXSIZE</code> value. If <code>MAXSIZE</code> is smaller than the backup set containing the logs, then RMAN must read the backup set more than once. Hence, RMAN issues a warning that <code>MAXSIZE</code> should be increased.</p>

Syntax Element	Description
CHECK READONLY	Checks the headers of read-only files to ensure that they are current before omitting them from the recovery.
NOREDO	<p>Suppresses the application of redo logs—only incremental backups are applied. This option is intended for recovery of NOARCHIVELOG databases by using incremental backups. If you do not specify NOREDO when recovering a NOARCHIVELOG database, then Oracle terminates recovery and issues an error.</p> <p>Note: Incremental backups of NOARCHIVELOG databases must be taken after a consistent shutdown.</p>
CHECK LOGICAL	<p>Tests data and index blocks that pass physical corruption checks for logical corruption, for example, corruption of a row piece or index entry. If RMAN finds logical corruption, it logs the block in the <code>alert.log</code> and server session trace file.</p> <p>Provided the sum of physical and logical corruptions detected for a file remain below its <code>MAXCORRUPT</code> setting, the RMAN command completes and Oracle populates <code>V\$DATABASE_BLOCK_CORRUPTION</code> with corrupt block ranges. If <code>MAXCORRUPT</code> is exceeded, then the command terminates without populating the views.</p> <p>Note: The <code>MAXCORRUPT</code> setting represents the total number of physical and logical corruptions permitted on a file.</p>
FROM TAG = <i>tag_name</i>	<p>Specifies the tag for an incremental backup to be used during recovery. If the tagged backup does not contain all the necessary incrementals for recovery, then RMAN uses logs or incremental backups as needed from whatever is available. Note that tag names are not case sensitive and display in all uppercase.</p> <p>See Also: "BACKUP" on page 2-26 to learn how a tag can be applied to an individual copy of a duplexed backup set, and to learn about the default filename format for backup tags</p>
ARCHIVELOG TAG = <i>tag_name</i>	<p>Specifies the tag for an archived log backup to be used during recovery. If the tagged backup does not contain all the necessary logs for recovery, RMAN uses logs or incremental backups as needed from whatever is available. Note that tag names are not case sensitive and display in all uppercase.</p>

Examples

Recovering a Tablespace in an Open Database: Example The following example takes tablespace `tools` offline, uses automatic channels to restore and recover it (deleting the logs that it restored from tape), then brings it back online:

```
SQL "ALTER TABLESPACE tools OFFLINE IMMEDIATE";
RESTORE TABLESPACE tools;
```

```
# restore only 2M of logs at a time, then delete them
RECOVER TABLESPACE tools DELETE ARCHIVELOG MAXSIZE 2M;
SQL "ALTER TABLESPACE tools ONLINE";
```

Recovering Datafiles Restored to New Locations: Example The following example uses the preconfigured disk channel and manually allocates one media management channel to use datafile copies on disk and backups on tape, and restores one of the datafiles in tablespace `users` to a different location:

```
RUN
{
  ALLOCATE CHANNEL dev2 DEVICE TYPE sbt;
  SQL "ALTER TABLESPACE users OFFLINE IMMEDIATE";
  SET NEWNAME FOR DATAFILE '?/oradata/trgt/users01.dbf'
  TO '/tmp/users01.dbf';
  RESTORE TABLESPACE users;
  SWITCH DATAFILE ALL;
  RECOVER TABLESPACE users;
  SQL "ALTER TABLESPACE users ONLINE";
}
```

Performing DBPITR with a Backup Control File and Recovery Catalog: Example

Assume that all datafiles and control files as well as archived redo log 40 were lost due to a disk failure. Because you do not have incremental backups, you need to recover the database with available archived redo logs. You do not need to restore tablespace `history` because it has not changed since log 40. After connecting to the target and recovery catalog, you do:

```
STARTUP FORCE NOMOUNT;
RUN
{
  SET UNTIL SEQUENCE 530 THREAD 1;      # Recover database until log sequence 40
  RESTORE CONTROLFILE TO '/tmp/control01.ctl';
  RESTORE CONTROLFILE FROM '/tmp/control01.ctl'; # Replicates to CONTROL_FILES locations
  ALTER DATABASE MOUNT;
  RESTORE DATABASE SKIP TABLESPACE temp, history;
  RECOVER DATABASE SKIP FOREVER TABLESPACE temp;
}
ALTER DATABASE OPEN RESETLOGS;
# if the database uses locally-managed temporary tablespaces, then you must add
# tempfiles to these tablespaces after restoring a backup control file
SQL "ALTER TABLESPACE temp ADD TEMPFILE '?/oradata/trgt/temp01.dbf' REUSE";
```

REGISTER

Syntax

```
→ REGISTER DATABASE <db_name> ;
```

Purpose

To register the target database in the recovery catalog so that RMAN can access it. RMAN obtains all information it needs to register the target database from the target database itself.

Note: If you perform a `RESETLOGS` operation on a database and later register it in the recovery catalog, the catalog records the `DB_NAME` for the old incarnations as `UNKNOWN` because the old incarnations were not previously registered. You should not try to remove these records.

See Also: *Oracle9i Recovery Manager User's Guide*, and "[CREATE CATALOG](#)" on page 2-89

Restrictions and Usage Notes

- Execute this command only at the RMAN prompt.
- You must be connected to the target database and recovery catalog database.
- You can register multiple databases in the same recovery catalog, but the database identifiers of the databases must be unique.
- You can only register a database once in a given recovery catalog.
- The target database must be mounted or open.
- You should not register a standby database.
- The `REGISTER DATABASE` command fails when RMAN detects duplicate DBIDs. This situation can arise when databases are created by copying files from an existing database rather than by using the `DUPLICATE` command.

If this failure occurs, then you can change the DBID of the copied database with the standalone DBNEWID utility.

Note: If you are using RMAN with different target databases that have the same database name and DBID, be careful to always specify the correct recovery catalog schema when invoking RMAN.

See Also: *Oracle9i Database Utilities*, to learn how to use the DBNEWID utility

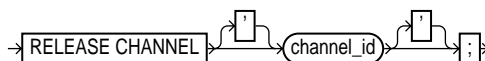
Example

Registering a Database: Example The following shell script registers a new target database, catalogs an existing datafile copy, then opens the database for use:

```
rman TARGET / CATALOG rman/rman@catdb <<EOF
STARTUP FORCE MOUNT;
REGISTER DATABASE;
CATALOG DATAFILECOPY '?/oradata/system01.cpy';
ALTER DATABASE OPEN;
EXIT;
EOF
```

RELEASE CHANNEL

Syntax



Purpose

To release a channel while maintaining the connection to the target database instance. Specify the channel name with the same identifier used in the [ALLOCATE CHANNEL](#) command. This command is optional because RMAN automatically releases all channels allocated when the [RUN](#) command terminates.

Requirements

Execute this command only within a [RUN](#) command.

Keywords and Parameters

Syntax Element	Description
<i>channel_id</i>	specifies the case-sensitive channel ID used in the ALLOCATE CHANNEL command.

Examples

Releasing a Channel: Example This example makes three identical backup sets of datafiles 1 through 4 to tape with channel `ch1`, then releases it. RMAN then makes three identical backups of datafiles 5 through 7 to tape with channel `ch2` and then releases it:

```

RUN {
    SET BACKUP COPIES = 3;
    ALLOCATE CHANNEL ch1 DEVICE TYPE sbt RATE = 1000;
    ALLOCATE CHANNEL ch2 DEVICE TYPE sbt MAXPIECESIZE = 5M;
    BACKUP CHANNEL ch1 DATAFILE 1,2,3,4;
    RELEASE CHANNEL ch1;
    BACKUP DATAFILE 5,6,7;
}

```

releaseForMaint

Syntax

```
→ RELEASE CHANNEL <n> ; >
```

Purpose

To release a sequential I/O device specified in an [ALLOCATE CHANNEL FOR MAINTENANCE](#) command. Note that maintenance channels are unaffected by [ALLOCATE CHANNEL](#) and [RELEASE CHANNEL](#) command issued within a [RUN](#) command.

Requirements

- Execute this command only at the RMAN prompt.
- You must have a maintenance channel allocated in order to release it.

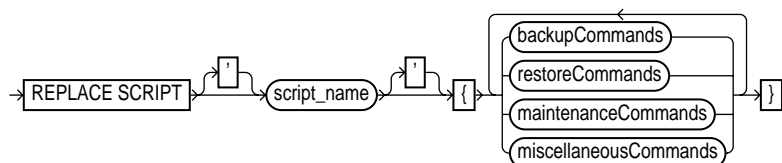
Examples

Releasing a Maintenance Channel After a Delete Operation: Example This example allocates and then releases a maintenance channel to the media manager:

```
ALLOCATE CHANNEL FOR MAINTENANCE DEVICE TYPE sbt;
DELETE NOPROMPT BACKUPPIECE 100;
RUN
{
  ALLOCATE CHANNEL ch1 DEVICE TYPE DISK;
  BACKUP DATAFILE 1;
  RELEASE CHANNEL ch1; # releases RUN channel but not maintenance channel
}
RELEASE CHANNEL; # releases maintenance channel
```

REPLACE SCRIPT

Syntax



See Also: The syntax diagrams for ["RUN"](#) on page 2-190 to determine which commands are included in the *backupCommands*, *restoreCommands*, *maintenanceCommands*, and *miscellaneousCommands* clauses

Purpose

To replace an existing script stored in the recovery catalog. If the script does not exist, then `REPLACE SCRIPT` creates it.

The stored script feature is provided primarily to provide a common repository for frequently executed collections of RMAN commands: use any command legal within a [RUN](#) command in the script. The script is not executed immediately; use the [EXECUTE SCRIPT](#) command to run it.

See Also:

- For descriptions of the individual commands that you can use in a stored script, see the appropriate entry, for example, ["BACKUP"](#) on page 2-26
- For information about the @ and @@ arguments, see ["CREATE SCRIPT"](#) on page 2-91
- For information about the `EXECUTE SCRIPT` command, see ["EXECUTE SCRIPT"](#) on page 2-115

Restrictions and Usage Notes

- Execute `REPLACE SCRIPT` only at the RMAN prompt.
- You must be connected to the target database. If you are replacing an existing script, then you must be connected to the target database that you connected to when you created the script.
- RMAN must be connected to the recovery catalog either through the `CATALOG` command-line option or the `CONNECT CATALOG` command, and the catalog database must be open.
- You cannot run `REPLACE SCRIPT` once to create a script and then use this same script on multiple target databases. You must connect to a specific target database and then run `REPLACE SCRIPT` to create a script for this specific target. Each script is associated with one and only one target database.
- The `@` and `@@` commands do not work within `REPLACE SCRIPT`.

Keywords and Parameters

For descriptions of the individual commands that you can use in a stored script, refer to the appropriate entry, for example, "[BACKUP](#)" on page 2-26.

Syntax Element	Description
<code>REPLACE SCRIPT</code> <code>'script_name'</code>	<p>Replaces the specified stored script with the new commands. The statements allowable within the parentheses of the <code>REPLACE SCRIPT 'script_name' (. . .)</code> command are the same allowable within the <code>RUN</code> command.</p> <p>To obtain a listing of all stored scripts, use SQL*Plus to connect to the recovery catalog database as the catalog owner and issue the following query:</p> <pre>SELECT * FROM RC_STORED_SCRIPT;</pre> <p>Note: To run the script, issue <code>EXECUTE SCRIPT</code> within the braces of the <code>RUN</code> command.</p> <p>See Also: "RC_STORED_SCRIPT" on page 3-29 for more information about <code>RC_STORED_SCRIPT</code></p>

Example

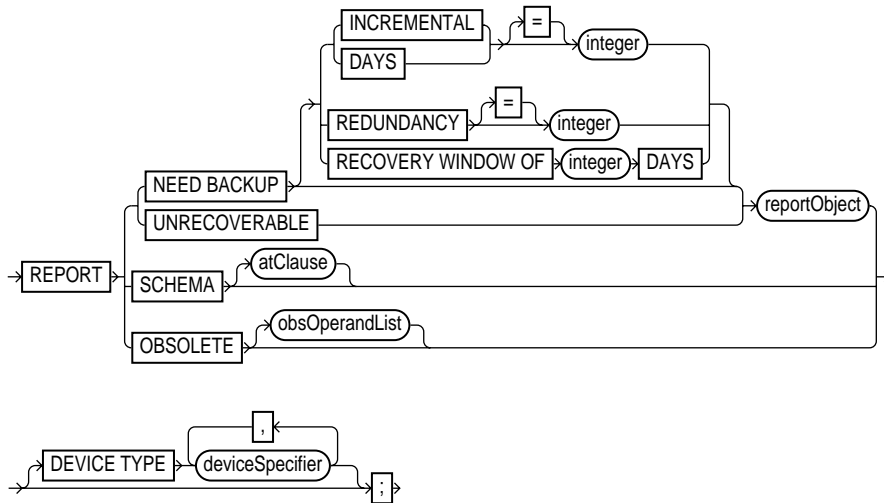
Replacing a Recovery Catalog Script: Example This example creates two distinct recovery catalog scripts called `backup_full`. Each script has different content and is associated with a different target database:

```
#!/usr/bin/tcsh
rman TARGET SYS/oracle@prod1 CATALOG rman/rman@catdb << SCRIPT1
# Create a script called backup_full that you can use to back up target
# database prod1. If a script already exists with the name backup_full,
# then this command overwrites it.
REPLACE SCRIPT backup_full
{
    # uses configured channel for default device type
    BACKUP DATABASE;
}
EXIT
SCRIPT1

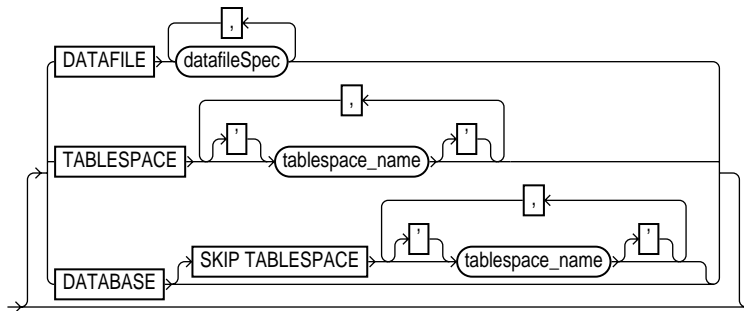
rman TARGET SYS/oracle2@prod2 CATALOG rman/rman@catdb <<SCRIPT2
# Creates a new script backup_full on database prod2. Even though it has the same name,
# this script is different from the backup_full script created on prod1 in the preceding
# example. Each catalog script is associated with one and only one target database.
REPLACE SCRIPT backup_full
{
    # uses configured sbt channel
    BACKUP DEVICE TYPE sbt DATABASE PLUS ARCHIVELOG;
}
# runs the backup_full script
RUN { EXECUTE SCRIPT backup_full; }
EXIT
SCRIPT2
```

REPORT

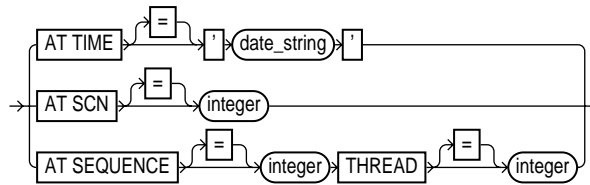
Syntax



reportObject ::=



atClause ::=



Purpose

To perform detailed analyses of the RMAN repository. Oracle writes the output from the `REPORT` command to standard output or the message log file.

Use the `REPORT` command to answer questions such as the following:

- Which files need a backup?
- Which files have not had a backup for some time?
- Which files are not recoverable due to unrecoverable operations?
- Which backup files can be deleted?
- What was the physical schema of the database at a previous time?

See Also: *Oracle9i Recovery Manager User's Guide* to learn how to use RMAN's reporting functionality

Restrictions and Usage Notes

- Execute this command only at the RMAN prompt.
- You must connect to a recovery catalog when issuing a `REPORT SCHEMA` command with the `AT TIME`, `AT SCN`, or `AT SEQUENCE` options. Otherwise, a recovery catalog is not required for the `REPORT` command.

Keywords and Parameters

Syntax Element	Description
NEED BACKUP	Lists all datafiles in need of a new backup. The report assumes that you will use the most recent backup for restore operations. If you do not specify any option, then RMAN uses the current retention policy configuration. If the retention policy is disabled (<code>CONFIGURE RETENTION POLICY TO NONE</code>), RMAN generates an error.
INCREMENTAL = <i>integer</i>	Specifies a threshold number of incremental backups required for recovery. If complete recovery of a datafile requires more than <i>integer</i> incremental backups, then the datafile requires a new full backup. The <code>REPORT</code> command, like the <code>RECOVER</code> command, uses the lowest level of incremental backup whenever there is a choice. This is the same strategy that RMAN would use if the file were actually being recovered by the <code>RECOVER</code> command. Note: Files for which no backups exist will not appear in this list: issue the <code>REPORT NEED BACKUP REDUNDANCY</code> command to display them.
DAYS = <i>integer</i>	Specifies a threshold number of days worth of logs needed to recover the file. For example, <code>REPORT NEED BACKUP DAYS 7 DATABASE</code> shows the datafiles whose recovery requires more than one week's worth of archived redo logs. If the target database control file is mounted and current, then RMAN makes the following optimizations to this report: <ul style="list-style-type: none"> Files that are offline and whose most recent backup contains all changes to the file are not included. Files that were offline and are now online, and whose most recent backup contains all changes up to the offline time, are only reported if they have been online for more than the specified number of days.
REDUNDANCY = <i>integer</i>	Specifies the minimum number of backups or copies that must exist for a datafile to be considered <i>not</i> in need of a backup. In other words, a datafile needs a backup if there are fewer than <i>integer</i> backups or copies of this file. For example, <code>REDUNDANCY 2</code> means that if there are fewer than two copies or backups of a datafile, then it needs a new backup.
RECOVERY WINDOW OF <i>integer</i> DAYS	Specifies a time window in which RMAN should be able to recover the database. The window stretches from the current time (<code>SYSDATE</code>) to the point of recoverability , which is the earliest date to which you want to recover. The point of recoverability is <i>integer</i> days in the past, that is, <code>SYSDATE - integer</code> .

Syntax Element	Description
UNRECOVERABLE	Lists all unrecoverable datafiles. A datafile is considered unrecoverable if an unrecoverable operation has been performed against an object residing in the datafile since the last backup of the datafile. Note: The nonexistence of any backup of a datafile is not sufficient reason to consider it unrecoverable. Such datafiles can be recovered through the use of the CREATE DATAFILE command, if redo logs starting from when the file was created still exist.
<i>reportObject</i>	Specifies the datafiles to be included in the report. The report can include the entire database (optionally skipping certain tablespaces), a list of tablespaces, or a list of datafiles.
DATAFILE <i>datafileSpec</i>	Lists the specified datafiles. RMAN reports on backups or datafile copies that contain at least one of the specified datafiles.
TABLESPACE ' <i>tablespace_name</i> '	Lists datafiles in the specified tablespace. RMAN reports on backups or datafile copies that include at least one datafile from a specified tablespace.
DATABASE	Lists backups or datafile copies of all files in the current database. Specify SKIP TABLESPACE <i>tablespace_name</i> to exclude the specified tablespace from the DATABASE specification.
SCHEMA	Lists the names of all datafiles and tablespaces at the specified point in time.
<i>atClause</i>	Specifies a point in time as a time, an SCN, or a log sequence number.
AT TIME = ' <i>date_string</i> '	Specifies a date. The NLS_LANG and NLS_DATE_FORMAT environment variables specify the format for the time.
AT SCN = <i>integer</i>	Specifies an SCN.
AT SEQUENCE = <i>integer</i> THREAD = <i>integer</i>	Specifies a log sequence number for a specified redo THREAD number. The integer indicates the time when the specified log and thread were first opened.
OBSOLETE <i>obsOperandList</i>	Lists full backups, datafile copies, and archived logs recorded in the RMAN repository that can be deleted because they are no longer needed. The subclause <i>obsOperandList</i> describes the criteria that RMAN uses to determine what is obsolete. If you do not specify parameters in <i>obsOperandList</i> , then RMAN uses the options specified in CONFIGURE RETENTION POLICY. If you use this option in conjunction with DEVICE TYPE, then RMAN only considers backups and copies created on the specified device.
DEVICE TYPE <i>deviceSpecifier</i>	Specifies the type of storage device. RMAN only considers backups and copies available on the specified device for its report.

Report Output

The information that appears in the output is shown in the following tables:

- [Table 2–19, "Report of Database Schema"](#)
- [Table 2–20, "Report of Obsolete Backups and Copies"](#)
- [Table 2–21, "Report of Files that Need Backup Due to Unrecoverable Operations"](#)
- [Table 2–22, "Report of Files with Less Than n Redundant Backups"](#)
- [Table 2–23, "Report of Files Whose Recovery Needs More Than n Days of Archived Logs"](#)
- [Table 2–24, "Report of Files That Need More than n Incrementals During Recovery"](#)

Table 2–19 *Report of Database Schema*

Column	Indicates
File	The absolute datafile number.
K-bytes	The size of the file in kilobytes.
Tablespace	The tablespace name.
RB segs	YES if rollback segments exist in the tablespace and NO if they do not (only if connected to the recovery catalog). If RMAN is not connected to the catalog, then *** is displayed.
Datafile Name	The filename of the datafile.

Table 2–20 *Report of Obsolete Backups and Copies*

Column	Indicates
Type	Whether the object is a backup set, backup piece, proxy copy, or datafile copy.
Key	A unique key that identifies this backup in the target database control file.
Completion Time	The time that the backup or copy completed.
Filename/handle	The filename or media handle of the backup or datafile copy.

Table 2–21 Report of Files that Need Backup Due to Unrecoverable Operations

Column	Indicates
File	The absolute number of the datafile that needs a new backup due to unrecoverable operations.
Type Of Backup Required	FULL or INCREMENTAL, depending on which type of backup is necessary to ensure the recoverability of all of the data in this file. If FULL, then create a full backup, level 0 backup, or a datafile copy. If INCREMENTAL, then a full or incremental backup will also suffice.
Name	The name of the datafile.

Table 2–22 Report of Files with Less Than *n* Redundant Backups

Column	Indicates
File	The absolute datafile number of a datafile with less than <i>n</i> redundant backups.
#bkps	The number of backups that exist for this file.
Name	The name of the file.

Table 2–23 Report of Files Whose Recovery Needs More Than *n* Days of Archived Logs

Column	Indicates
File	The absolute file number of a datafile that requires more than <i>n</i> days of archived redo logs for recovery.
Days	The number of days of archived redo data required for recovery.
Name	The name of the datafile.

Table 2–24 Report of Files That Need More than *n* Incrementals During Recovery

Column	Indicates
File	The absolute file number of a datafile that requires more than <i>n</i> incrementals for complete recovery.
Incrementals	The number of incremental backups required for complete recovery.
Name	The name of the datafile.

Examples

Reporting Database Schema: Example This example, which requires a recovery catalog, reports the names of all datafiles and tablespaces one week ago:

```
REPORT SCHEMA AT TIME 'SYSDATE-7';
```

```
Report of database schema
File K-bytes   Tablespace           RB segs Datafile Name
-----
1          307200 SYSTEM                ***    /oracle/oradata/trgt/system01.dbf
2           20480 UNDOTBS          ***    /oracle/oradata/trgt/undotbs01.dbf
3           10240 CWMLITE           ***    /oracle/oradata/trgt/cwmlite01.dbf
4           10240 DRSYS              ***    /oracle/oradata/trgt/drsys01.dbf
5           10240 EXAMPLE            ***    /oracle/oradata/trgt/example01.dbf
6           10240 INDX                 ***    /oracle/oradata/trgt/indx01.dbf
7           10240 TOOLS                ***    /oracle/oradata/trgt/tools01.dbf
8           10240 USERS                ***    /oracle/oradata/trgt/users01.dbf
```

Reporting Datafiles Needing Incremental Backups: Example This example reports all datafiles in the database that require the application of five or more incremental backups to be recovered to their current state:

```
REPORT NEED BACKUP INCREMENTAL 5 DATABASE;
```

```
Report of files that need more than 5 incrementals during recovery
File Incrementals Name
-----
```

```
2      9          /oracle/oradata/trgt/undotbs01.dbf
3      9          /oracle/oradata/trgt/cwmlite01.dbf
4      9          /oracle/oradata/trgt/drsys01.dbf
```

Reporting Datafiles Needing Backups: Example The following example reports all datafiles from tablespace SYSTEM that will need more than two days of archived redo logs to be applied during recovery after being restored from the most recent backup:

```
REPORT NEED BACKUP DAYS 2 TABLESPACE SYSTEM;
```

```
Report of files whose recovery needs more than 2 days of archived logs
File Days   Name
-----
```

```
1      3          /oracle/oradata/trgt/drsys01.dbf.f
```

Reporting Unrecoverable Datafiles: Example The following example reports all datafiles that cannot be recovered from existing backups because redo may be missing:

```
REPORT UNRECOVERABLE;
```

```
Report of files that need backup due to unrecoverable operations
File Type of Backup Required Name
```

```
-----
1    full                               /oracle/oradata/trgt/system01.dbf
```

Reporting Obsolete Backups and Copies: Example The following example reports obsolete backups and copies with a redundancy of 1:

```
REPORT OBSOLETE;
```

```
Report of obsolete backups and copies
```

```
Type                Key    Completion Time    Filename/Handle
-----
Backup Set          1      OCT 30 2001 15:54:56
  Backup Piece      1      OCT 30 2001 15:54:56 /oracle/dbs/01d7t0t9_1_1
Archive Log         1      OCT 30 2001 04:52:17 /oracle/oradata/trgt/arch/archive1_21.dbf
```

RESET DATABASE

Syntax



Purpose

To reset the target database in the RMAN repository, which means to do either of the following actions:

- Inform RMAN that the SQL statement `ALTER DATABASE OPEN RESETLOGS` has been executed and that a new incarnation of the target database has been created. Note that if you run the RMAN command `ALTER DATABASE OPEN RESETLOGS` (not the SQL statement with the same keywords), then RMAN resets the target database automatically so that you do not have to run `RESET DATABASE`. By resetting the database, RMAN considers the new incarnation as the current incarnation of the database.
- To reset the database to a previous incarnation. Typically, you would reset the incarnation when performing incomplete recovery to a point before a `RESETLOGS` operation, or when attempting to undo the affects of a `RESETLOGS` by restoring backups taken before a `RESETLOGS`.

Restrictions and Usage Notes

- Execute `RESET DATABASE` only at the RMAN prompt.
- You must be connected to the target database and a recovery catalog.
- You must issue a `RESET DATABASE` command before you can use RMAN with a target database that has been opened with the SQL statement `ALTER DATABASE OPEN RESETLOGS` option. If you do not, then RMAN refuses to access the recovery catalog because it cannot distinguish between a `RESETLOGS` operation and an accidental restore of an old control file. The `RESET DATABASE` command informs RMAN that you issued a `RESETLOGS` command.
- You cannot specify `TO INCARNATION` unless the database is started `NOMOUNT`. If you mount a control file from an incarnation after the desired incarnation, then `RESET DATABASE TO INCARNATION` fails because of a control file mismatch. If you mount the control file from the desired incarnation and then run `RESET`

DATABASE TO INCARNATION, then the connection to the target database and recovery catalog fails due to an incarnation mismatch.

Keywords and Parameters

Syntax Element	Description
TO INCARNATION <i>primary_key</i>	Changes the current incarnation to an older incarnation. Specify the primary key of the DBINC record for the database incarnation. Run LIST INCARNATION OF DATABASE to obtain the key. After you start the instance and issue RESET DATABASE TO INCARNATION, then you can mount or restore a control file from the desired incarnation and run RMAN commands.

Examples

Resetting RMAN to a Previous Incarnation: Example The following scenario makes an old incarnation of database `trgt` current again:

```
# step 1: obtain the primary key of old incarnation
LIST INCARNATION OF DATABASE trgt;
```

List of Database Incarnations

DB Key	Inc Key	DB Name	DB ID	CUR	Reset SCN	Reset Time
1	2	TRGT	1334358386	NO	154381	OCT 30 2001 16:02:12
1	116	TRGT	1334358386	YES	154877	OCT 30 2001 16:37:39

```
# step 2: start instance and reset database to incarnation key 2
STARTUP FORCE NOMOUNT;
RESET DATABASE TO INCARNATION 2;
```

```
# step 3: restore control file from previous incarnation, then shut down instance
# and mount control file
RESTORE CONTROLFILE;
STARTUP FORCE MOUNT;
```

```
# step 4: restore and recover the database to a point before the RESETLOGS
RESTORE DATABASE UNTIL SCN 154876;
RECOVER DATABASE UNTIL SCN 154876;
```

```
# step 5: make this incarnation the current incarnation and then list incarnations:
ALTER DATABASE OPEN RESETLOGS;
LIST INCARNATION OF DATABASE trgt;
```

RESET DATABASE

List of Database Incarnations

DB Key	Inc Key	DB Name	DB ID	CUR	Reset SCN	Reset Time
1	2	TRGT	1334358386	NO	154381	OCT 30 2001 16:02:12
1	116	TRGT	1334358386	NO	154877	OCT 30 2001 16:37:39
1	311	TRGT	1334358386	YES	154877	OCT 30 2001 17:17:03

Resetting the Database After Incomplete Recovery: Example This example assumes that an incomplete recovery or recovery with a backup control file was performed in NOCATALOG mode. Later, RMAN is started in CATALOG mode, but the RESYNC command fails because the incarnation has not been reset in the catalog.

```
% rman target / catalog rman/rman@catdb
```

```
Recovery Manager: Release 9.2.0.0.0 - Production
```

```
(c) Copyright 2001 Oracle Corporation. All rights reserved.
```

```
connected to target database: TRGT (DBID=1334531173)
```

```
connected to recovery catalog database
```

```
RMAN> RESYNC CATALOG;
```

```
RMAN-00571: =====
```

```
RMAN-00569: ===== ERROR MESSAGE STACK FOLLOWS =====
```

```
RMAN-00571: =====
```

```
RMAN-03009: failure of resync command on default channel at 11/01/2001 12:00:43
```

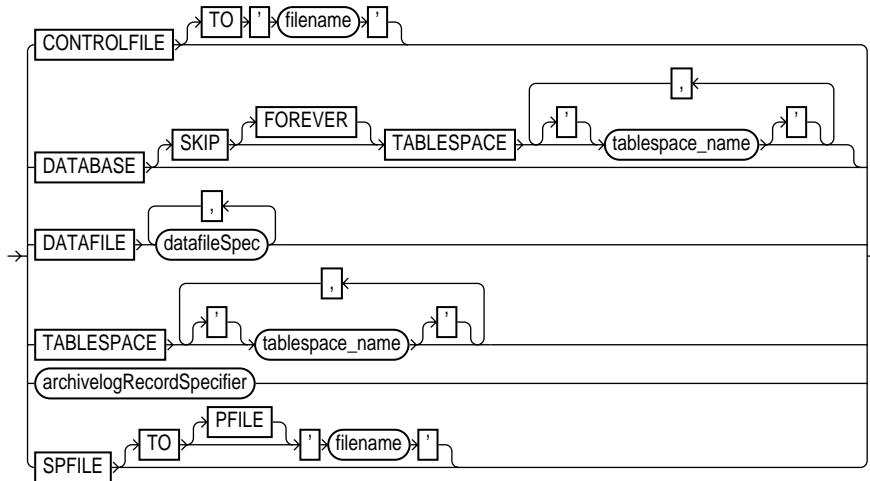
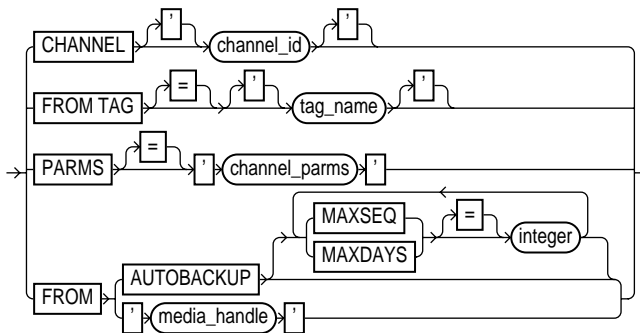
```
RMAN-20003: target database incarnation not found in recovery catalog
```

```
RMAN> RESET DATABASE;
```

```
new incarnation of database registered in recovery catalog
```

```
starting full resync of recovery catalog
```

```
full resync complete
```


restoreObject ::=**restoreSpecOperand ::=****Purpose**

To restore files from backups or image copies. By default, RMAN restores files to their default location. You can use the `SET NEWNAME` command to restore files to nondefault locations. RMAN restores backups from disk or tape and restores images copies from disk only.

Typically, you restore when a media failure has damaged a current datafile, control file, or archived log or prior to performing a point-in-time recovery. The `RESTORE`

command restores full backups, incremental backups (level 0 only), or copies of datafiles, control files, and archived redo logs. Because the `RECOVER` command automatically restores archived logs as needed, you should seldom need to restore logs manually. Possible reasons for manually restoring archived logs are to speed up recovery or to stage the logs to multiple destinations.

Note: In Oracle9i, unlike in previous RMAN releases, RMAN by default does not restore a datafile if the file is in the correct place and its header contains the expected data (RMAN does not scan the datafile body for corrupt blocks). The `FORCE` option overrides this behavior and restores the requested files unconditionally.

When you run `RESTORE` in `CATALOG` mode with a backup control file, RMAN automatically adjusts the control file to reflect the structure of the restored database.

If you restore to the default location, then RMAN overwrites files with the same filenames. If you restore to a new location, then issue `SET NEWNAME` commands to rename the files and issue a `SWITCH` command to make the restored files current. If you do not issue `SWITCH` commands, then RMAN considers the restored files as valid copies for use in future restore operations.

If you do not manually allocate channels, then RMAN allocates all automatic channels possibly needed by the `RESTORE` command. For example, assume you configure 3 separate `sbt` channels (each with different `PARMS`) and then configure parallelism for `DISK` and `sbt` as follows:

```
CONFIGURE DEVICE TYPE DISK PARALLELISM 2;  
CONFIGURE DEVICE TYPE sbt PARALLELISM 3;  
CONFIGURE DEFAULT DEVICE TYPE TO sbt;
```

During a restore, RMAN allocates three `sbt` channels and the two preconfigured `DISK` channels. For a restore, RMAN allocates all configured channels unless the `DEVICE TYPE` option restricts the device type from which RMAN restores files.

In a Real Application Clusters configuration, RMAN automatically restores backups, control file copies, and datafile copies from channels that can read the files on tape or a local file system. For example, if channel 1 connected to instance 1 can read log 1000 from its tape drive, but channel 2 connected to instance 2 cannot read the log from its tape drive, then channel 1 restores the log. Autolocation is automatically enabled when the channels meet any of the following criteria:

- Different `PARMS` settings

- Different `CONNECT` strings

See Also: *Oracle9i Recovery Manager User's Guide* to learn how to restore files

Restrictions and Usage Notes

- To restore datafiles to their current location, the database must be started, mounted, or open with the tablespaces or datafiles to be restored offline. If the database is started but not mounted, then it is recommended that you only restore the control file, if necessary (refer to "[Restrictions and Usage Notes for RESTORE CONTROLFILE](#)" on page 2-179). To restore other files, mount the database and then continue.

Note: When performing a database validation by using `RESTORE . . . VALIDATE`, the database can be open.

- To restore to a new location, run `SET NEWNAME` commands to rename the datafiles and `SWITCH` commands to make them the current database files. If you do not use `SWITCH`, then the repository lists restored datafiles as datafile copies.
- If you use the `FROM DATAFILECOPY` option, then the allocated channels must be of `DEVICE TYPE DISK`.
- If you use the `FROM BACKUPSET` operand, then the appropriate type of storage devices must be allocated for the backup sets that need to be restored. If the appropriate device is not allocated, then you may not be able to find a candidate backup set or copy to restore, and the `RESTORE` command fails.
- RMAN only restores backups that were created on the same type of channels that are allocated for the `RESTORE` command.

For example, if you made some backups of a datafile to `DISK` channels and others to `sbt` channels, and only a `DISK` channel is allocated for the `RESTORE` command, RMAN will not restore backups that were created on `sbt` channels.
- If datafile names are symbolic links, that is, files pointing to other files, then the control file stores the filenames of the link files but RMAN performs I/O on the datafiles pointed to by the link files. If a link file is lost and you `RESTORE` a datafile without re-creating the symbolic link, then RMAN restores the datafile to the location of the link file rather than to the location pointed to by the link.

- You can only restore from a previous incarnation when restoring the whole database. For example, you cannot restore one datafile of a previous incarnation while the current database is in a different incarnation.
- If the database is started but not mounted in NOCATALOG mode, then the `RESTORE SPFILE` command requires the `FROM AUTOBACKUP` clause.
- If you are restoring the server parameter file and the control file in a disaster recovery situation, you cannot run `RESTORE CONTROLFILE FROM AUTOBACKUP`, mount this control file, and then run `RESTORE SPFILE` *without* the `FROM AUTOBACKUP` clause.
- Do not specify a datafile more than once in a restore job. For example, the following command is illegal because `datafile 1` is both specified explicitly and implied by the `SYSTEM` tablespace:


```
RESTORE TABLESPACE SYSTEM DATAFILE 1;
```
- You must have already configured a device type by using `CONFIGURE` (except for `DISK`, which is preconfigured) before specifying the `DEVICE TYPE` option.
- You cannot manually allocate channels and then run `RECOVER DEVICE TYPE`.
- RMAN neither backs up nor restores locally-managed temporary tablespaces, although it can back up and restore dictionary-managed temporary tablespaces.

Restrictions and Usage Notes for RESTORE CONTROLFILE

- After you restore a backup control file, you must run `RECOVER DATABASE` and then open the database with the `RESETLOGS` option.
- After restoring a backup control file, entries for tempfiles in locally-managed temporary tablespaces are removed. Hence, you must add new tempfiles to these tablespaces after you open with the `RESETLOGS` option. If you do not, then Oracle can display the following error for when attempting to sort:


```
ORA-25153: Temporary Tablespace is Empty.
```
- If you restore a control file from a release earlier than release 8.1.6 on Windows NT that has not been normalized, then you must normalize it before mounting the database by following the procedure described in *Oracle9i Database Migration*. A flawed mechanism in releases prior to release 8.1.6 on Windows NT could allow two different filenames to refer to the same physical file.

[Table 2–25](#) indicates the restrictions that apply in different situations involving the `RESTORE CONTROLFILE` command.

Table 2–25 RESTORE CONTROLFILE Scenarios

	RESTORE CONTROLFILE;	RESTORE CONTROLFILE FROM AUTOBACKUP;	RESTORE CONTROLFILE ... TO 'filename';	RESTORE CONTROLFILE ... FROM 'media_handle' or TAG;
No catalog, target started	Error. Must specify FROM AUTOBACKUP.	First run <code>SET DBID</code> . Restores to CONTROL_FILES locations.	First run <code>SET DBID</code> . Must specify FROM AUTOBACKUP. Restores only to <i>filename</i> .	First run <code>SET DBID</code> . Restores from specified file (cannot restore from TAG). If TO <i>filename</i> not used, restores to all CONTROL_FILES locations.
No catalog, target mounted or open	Error. Must use TO <i>filename</i> , where <i>filename</i> is not in CONTROL_FILES list.	Error. Must use TO <i>filename</i> , where <i>filename</i> is not in CONTROL_FILES list.	Restores only to <i>filename</i> , where <i>filename</i> is not in CONTROL_FILES list.	Restores from specified file. If TO <i>filename</i> not used, restores to all CONTROL_FILES locations.
Catalog, target started	Restores to CONTROL_FILES locations. Run <code>SET DBID</code> only if DB_NAME not unique in catalog.	Do not use with catalog unless for testing purposes.	Restores only to <i>filename</i> , where <i>filename</i> is not in CONTROL_FILES list.	Restores from specified file. If TO <i>filename</i> not used, restores to all CONTROL_FILES locations.
Catalog, target mounted or open	Error. Must use TO <i>filename</i> , where <i>filename</i> is not in CONTROL_FILES list.	Do not use with catalog.	Restores only to <i>filename</i> , where <i>filename</i> is not in CONTROL_FILES list.	Restores from specified file. If TO <i>filename</i> not used, restores to all CONTROL_FILES locations.

Keywords and Parameters

Syntax Element	Description
<i>restoreObject</i>	Specifies the objects to be restored.
CONTROLFILE	Restores the current control file. See Also: Table 2–25 for restrictions and usage notes. Note: You must always run the <code>RECOVER</code> command after restoring a control file, and must also always open the database with the <code>RESETLOGS</code> option.

Syntax Element	Description
DATABASE	<p>Restores all datafiles in the database except those that are offline or read-only. Unlike <code>BACKUP DATABASE</code>, <code>RESTORE DATABASE</code> does <i>not</i> automatically include the control file and the server parameter file—you must issue additional <code>RESTORE CONTROLFILE</code> and <code>RESTORE SPFILE</code> commands to restore these files.</p> <p>If you specify the <code>CHECK READONLY</code> option, then RMAN examines the headers of all read-only datafiles and restores any that need restoring.</p> <p>Use an optional <code>SKIP TABLESPACE 'tablespace_name'</code> argument to avoid restoring specified tablespaces, which is useful when you want to avoid restoring tablespaces containing temporary data.</p> <p>If you specify <code>SKIP FOREVER TABLESPACE</code>, then RMAN specifies the <code>DROP</code> option of <code>ALTER DATABASE DATAFILE . . . OFFLINE</code> when taking the datafiles that belong to the tablespace offline before the restore. The <code>DROP</code> option indicates that RMAN does not intend to recover these files and intends to drop their tablespaces from the database after the database is opened again. In other words, <code>FOREVER</code> indicates that RMAN never intends to do anything with the skipped tablespaces again.</p>
DATAFILE <i>datafileSpec</i>	<p>Restores the datafiles specified by filename or absolute datafile number.</p> <p>See Also: "datafileSpec" on page 2-96</p>
TABLESPACE <i>'tablespace_name'</i>	Restores all datafiles in the specified tablespaces.
<i>archivelogRecordSpecifier</i>	<p>Restores the specified range of archived redo logs.</p> <p>See Also: "archivelogRecordSpecifier" on page 2-21</p> <p>Note: The database can be started, mounted, or open for this operation.</p>
SPFILE [TO [PFILE] <i>'filename'</i>]	<p>Restores the server parameter file to the location from which it was backed up (default). Alternatively, you can specify a different location with the <code>TO</code> clause. RMAN cannot overwrite a server parameter file currently in use by the target database.</p> <p>Specify <code>UNTIL</code> or <code>TAG</code> options of <code>RESTORE</code> command to restore older versions of the server parameter file. By default RMAN restores the most current server parameter file.</p> <p>If the server parameter file is lost, connect to the target (and catalog if used) and then run <code>SET DBID</code>. Run <code>STARTUP FORCE NOMOUNT</code> before running <code>RESTORE SPFILE</code> (with <code>FROM AUTOBACKUP</code> if in <code>NOCATALOG</code> mode). Then run <code>STARTUP FORCE</code> to restart the database with the restored server parameter file.</p> <p>See Also: "Restrictions and Usage Notes for the SET DBID Command" on page 2-197</p>

Syntax Element	Description
<i>restoreSpecOperand</i>	specifies options for the <i>restoreObject</i> clause. Note: These parameters override the parameters with the same name at the RESTORE command level.
CHANNEL 'channel_id'	Specifies the case-sensitive name of a channel to use for this restore operation. If you do not specify a channel, then RESTORE uses any available channel allocated with the correct device type.
FROM TAG = 'tag_name'	Overrides the default selection of the most recent backups or file copy available. The tag restricts the automatic selection to backup sets or file copies that were created with the specified tag. If multiple backup sets or file copies have a matching tag, then RMAN selects the most recent one. Note that tag names are not case sensitive. See Also: "BACKUP" on page 2-26 for a description of how a tag can be applied to an individual copy of a duplexed backup set, and for a description of the default filename format for tags
PARMS 'channel_parms'	Specifies a quoted string containing operating system-specific information. The string is passed to the operating system dependent layer each time a backup piece is restored.
FROM AUTOBACKUP	Restores a control file autobackup. You can only specify this option on the RESTORE CONTROLFILE and RESTORE SPFILE commands. When restoring either type of file in NOCATALOG mode, the FROM AUTOBACKUP clause is required. RMAN begins the search on the current day or on the day specified with the SET UNTIL. If no autobackup is found in the current or SET UNTIL day, RMAN checks the previous day starting with sequence 256 (or the sequence specified by MAXSEQ) until it reaches 0. The search continues up to MAXDAYS days (default of 7, maximum of 366) from the current or SET UNTIL day. If no autobackup is found within MAXDAYS days, then RMAN signals an error and the command stops. See Also: Table 2-25 for restrictions and usage notes.
FROM 'media_handle'	Specifies the name of the control file copy or backup piece containing a control file. The <i>media_handle</i> can be any backup piece that contains a backup of a control file: the control file backup does not need to be an autobackup. See Also: Table 2-25 for restrictions and usage notes.
CHANNEL 'channel_id'	Refer to the <i>restoreSpecOperand</i> clause.
PARMS 'channel_parms'	Refer to the <i>restoreSpecOperand</i> clause.
FROM {BACKUPSET DATAFILECOPY}	Specifies whether RMAN should restore from a DATAFILECOPY on disk or a BACKUPSET. By default RESTORE chooses the most recent backup set or file copy, that is, the file copy or backup set that needs the least media recovery.

Syntax Element	Description
<i>untilClause</i>	Limits the selection to those backup sets or file copies that would be suitable for performing an incomplete recovery to the specified time. In the absence of any other criteria, RMAN selects the most current file copy or backup set to restore. See Also: " <i>untilClause</i> " on page 2-216
FROM TAG = 'tag_name'	Refer to the <i>restoreSpecOperand</i> clause.
VALIDATE	Causes RMAN to decide which backup sets, datafile copies, and archived logs need to be restored and then scans them to verify their contents. This operation creates no output files. Specify this option periodically to verify that the copies and backup sets required to restore the specified files are intact and usable.
CHECK LOGICAL	Tests data and index blocks that pass physical corruption checks for logical corruption, for example, corruption of a row piece or index entry. If RMAN finds logical corruption, it logs the block in the <code>alert.log</code> and server session trace file. If the sum of physical and logical corruptions for a file remain below its <code>MAXCORRUPT</code> setting, the RMAN command completes and Oracle populates the <code>V\$DATABASE_BLOCK_CORRUPTION</code> view with corrupt block ranges. If <code>MAXCORRUPT</code> is exceeded, the command terminates without populating the views. Note: The <code>MAXCORRUPT</code> setting represents the total number of physical and logical corruptions permitted on a file.
CHECK READONLY	Checks the read-only datafiles to make sure they exist, are readable, and have the appropriate checkpoint. If any of these conditions is not met, then RMAN restores the files—whether or not they are read-only. By default, RMAN does not restore read-only files when you issue the <code>RESTORE DATABASE</code> command.
DEVICE TYPE <i>deviceSpecifier</i>	Allocates automatic channels for the specified device type only. For example, if you configure automatic disk and tape channels, and issue <code>RESTORE . . . DEVICE TYPE DISK</code> , then RMAN allocates only disk channels. See Also: " <i>deviceSpecifier</i> " on page 2-104
FORCE	Overrides the restartable restore feature and restores all files regardless of whether they need to be restored. If you do not specify <code>FORCE</code> , then RMAN restores a file only if its header information does not match the information in the control file.

Examples

Restoring a Tablespace: Example This example takes a tablespace offline, restores it, then performs media recovery:

```
SQL "ALTER TABLESPACE users OFFLINE IMMEDIATE";
RESTORE TABLESPACE users;
RECOVER TABLESPACE users;
SQL "ALTER TABLESPACE users ONLINE";
```

Restoring the Control File When Using a Recovery Catalog: Example This example restores the control file to its default location, replicates it automatically to all CONTROL_FILES locations, and mounts the database:

```
RUN
{ # SET DBID is not necessary when connected to a recovery catalog
  STARTUP FORCE NOMOUNT;
  RESTORE CONTROLFILE;
  ALTER DATABASE MOUNT;
}
```

Restoring the Control File with a Tag: Example This NOCATALOG example restores the control file specified by a tag, and then mounts the database:

```
CONNECT TARGET /
STARTUP NOMOUNT;
SET DBID 320066378; # required when restoring control file in NOCATALOG mode
RESTORE CONTROLFILE FROM TAG 'monday_cf_backup';
ALTER DATABASE MOUNT;
```

Restoring the Database with a Backup Control File: Example This example restores the control file, replicates it to all control file locations specified in the parameter file, and then restores and recovers the database:

```
CONNECT TARGET /
STARTUP NOMOUNT;
SET DBID 320066378; # required when restoring control file in NOCATALOG mode
RUN
{
  ALLOCATE CHANNEL c1 DEVICE TYPE sbt;
  RESTORE CONTROLFILE TO '/tmp/control01.ct1' FROM AUTOBACKUP;
  RESTORE CONTROLFILE FROM '/tmp/control01.ct1'; # restores to all CONTROL_FILES locations
  ALTER DATABASE MOUNT;
  RESTORE DATABASE;
  RECOVER DATABASE;
}
ALTER DATABASE OPEN RESETLOGS;
# if the database uses locally-managed temporary tablespaces, then add new tempfiles
```

```
# to these tablespaces after the RESETLOGS
SQL "ALTER TABLESPACE temp ADD TEMPFILE '"/oradata/trgt/temp01.dbf"' REUSE";
```

Restoring Archived Redo Logs to a New Location: Example This example restores all archived redo logs to the `/oracle/temp_restore` directory:

```
RUN
{
  SET ARCHIVELOG DESTINATION TO '/tmp';
  RESTORE ARCHIVELOG ALL;
}
```

Restoring a Control File Autobackup to a Nondefault Location: Example This example restores the latest control file autobackup made on or before June 23, 2000 with a nondefault format of `PROD_CF_AUTOBACKUP_%F`. It starts searching for backups with a sequence number of 20, and searches backward for 5 months:

```
SET DBID 320066378; # required when restoring control file in NOCATALOG mode
RUN
{
  SET UNTIL TIME '23-JUN-2001 00:00:00';
  SET CONTROLFILE AUTOBACKUP FORMAT FOR DEVICE TYPE sbt TO 'prod_cf_autobackup_%F';
  ALLOCATE CHANNEL CHANNEL_1 DEVICE TYPE sbt;
  RESTORE CONTROLFILE TO '/tmp/autobackup_20001002.dbf' FROM AUTOBACKUP
    MAXSEQ 20 MAXDAYS 150;
}
```

Restoring the Server Parameter File to Current Location: Example The following shell script restores the current server parameter file in NOCATALOG mode:

```
#!/usr/bin/tcsh
rman TARGET / <<EOF
SET DBID 1447326980 # set dbid to dbid of target database
STARTUP FORCE NOMOUNT; # start instance with dummy SPFILE
RUN
{
  ALLOCATE CHANNEL c1 DEVICE TYPE sbt;
  RESTORE SPFILE FROM AUTOBACKUP; # FROM AUTOBACKUP needed in NOCATALOG mode
}
STARTUP FORCE; # start with restored SPFILE and open database
EXIT
EOF
```

Restoring the Server Parameter File to a Nondefault Location: Example The following shell script restores a server parameter file that was backed up at least a month ago and restarts the database with the restored parameter file:

```
#!/usr/bin/tcsh
echo "SPFILE=/tmp/spfileTEMP.ora" > /tmp/initTEMP.ora
rman TARGET / <<EOF
STARTUP FORCE MOUNT; # make sure database is mounted so SET DBID not required
RESTORE SPFILE TO '/tmp/spfileTEMP.ora' UNTIL 'SYSDATE-31';
STARTUP FORCE PFILE=/tmp/initTEMP.ora; # start with restored SPFILE and open database
EXIT
EOF
```

RESYNC

Syntax



Purpose

To perform a full resynchronization of the recovery catalog.

Resynchronizations can be full or partial. When full, RMAN updates all changed records for the physical schema: datafiles, tablespaces, redo threads, and online redo logs. If the database is open, RMAN also obtains data about rollback segments. When partial, RMAN reads the current control file to update data, but does not resynchronize metadata about the physical schema or rollback segments.

When resynchronizing, RMAN creates a snapshot control file in order to obtain a read-consistent view of the control file, then updates the catalog with any new information from the snapshot. The `RESYNC CATALOG` command updates the classes or records described in the following table.

Record Type	Description
Log history	Records that are created whenever a log switch occurs. Note that log history records describe an online log switch, not a log archival.
Archived redo logs	Records associated with archived logs that were created by archiving an online redo log, copying an existing archived redo log, or restoring backups of archived redo logs.
Backups	Records associated with backup sets, backup pieces, backup set members, proxy copies, and image copies.
Physical schema	Records associated with datafiles and tablespaces. If the target database is open, then rollback segment information is also updated.

RMAN automatically executes a full or partial resynchronization as needed when you execute RMAN commands, so long as the control file is mounted and the recovery catalog database is available at command execution. RMAN reads the current control file and does not resynchronize metadata about the physical schema

unless it determines that this information has changed. If RMAN does detect a change, then it performs a full resynchronization.

Use `RESYNC CATALOG` to perform manual full resynchronizations when:

- The recovery catalog is unavailable when you issue any of the commands that automatically perform a resynchronization.
- You are running in `ARCHIVELOG` mode, since the catalog is *not* updated automatically when a log switch occurs or when an online redo log is archived.
- You have made changes to the physical structure of the target database such as adding or dropping a tablespace. As with archive operations, the recovery catalog is *not* updated automatically when the physical schema changes.

Restrictions and Usage Notes

- You must be connected to a recovery catalog.
- RMAN updates physical schema information in the recovery catalog only when the target database has the current control file mounted. If the target database has mounted a backup control file, a freshly created control file, or a control file that is less current than a control file that was used previously, then RMAN does not update physical schema information in the recovery catalog.

Keywords and Parameters

Syntax Element	Description
<code>FROM CONTROLFILECOPY</code> <code>'filename'</code>	Specifies the name of the control file copy to use for resynchronization. Physical schema information is not updated when you use this option.

Examples

Resynchronizing the Recovery Catalog in ARCHIVELOG Mode: Example This example performs a full resynchronization after archiving all unarchived redo logs:

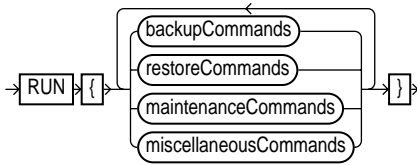
```
SQL "ALTER SYSTEM ARCHIVE LOG ALL";  
RESYNC CATALOG;
```

Resynchronizing the Recovery Catalog After a Structural Change: Example This example adds a datafile to tablespace `users` and then resynchronizes the catalog:

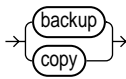
```
#!/usr/bin/tcsh
# connect in nocatalog mode and add datafile
rman TARGET / NOCATALOG <<EOF
SQL "ALTER TABLESPACE users ADD DATAFILE ''?/oradata/trgt/users03.dbf''
    SIZE 1M AUTOEXTEND ON NEXT 10K MAXSIZE 10M";
EXIT
EOF
# connect in catalog mode and resynchronize
rman TARGET / CATALOG rman/rman@catdb <<EOF
RESYNC CATALOG;
EOF
```

RUN

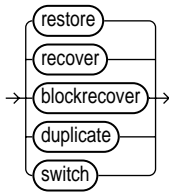
Syntax



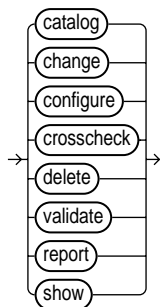
backupCommands ::=



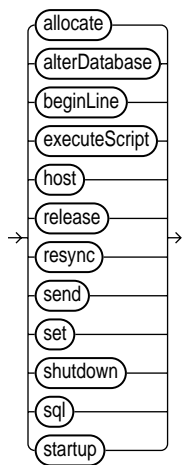
restoreCommands ::=



maintenanceCommands ::=



miscellaneousCommands ::=



Purpose

To compile and execute job commands, which are one or more statements executed within the braces of RUN. The RUN command compiles the list of job commands into one or more job steps and then executes them immediately.

Restrictions and Usage Notes

- Execute this command only at the RMAN prompt.
- You must precede and follow the list of job commands with an opening and closing brace.

Keywords and Parameters

Refer to individual entries for information about commands that you can run from the RMAN prompt.

Examples

Making a Backup: Example This example backs up a database by using a single manually allocated channel to perform the backup:

```
RUN
{
  ALLOCATE CHANNEL c1 TYPE sbt;
  BACKUP DATABASE;
}
```

Restoring and Recovering a Tablespace: Example This example takes tablespace `tools` offline, restores it, then performs complete media recovery:

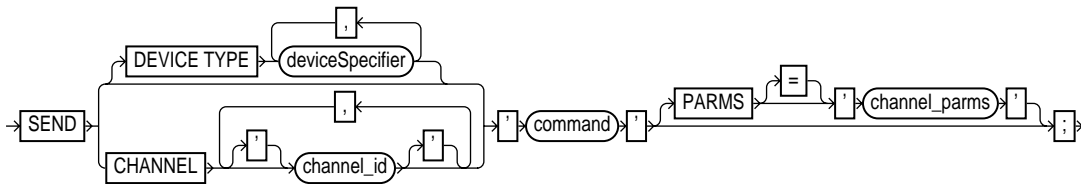
```
RUN
{
  SQL "ALTER TABLESPACE tools OFFLINE IMMEDIATE";
  RESTORE TABLESPACE tools;
  RECOVER TABLESPACE tools;
  SQL "ALTER TABLESPACE tools ONLINE";
}
```

Executing an RMAN Script: Example This example executes the stored script `backup_db`:

```
RUN { EXECUTE SCRIPT backup_db; }
```

SEND

Syntax



Purpose

To send a vendor-specific string to one or more channels. Refer to your media management documentation to determine which commands are supported.

Restrictions and Usage Notes

- Execute `SEND` at the RMAN prompt or within the braces of a `RUN` command.
- You must use a media manager to use `SEND` and only with commands supported by the media manager. The contents of the string are not interpreted by Oracle, but are passed unaltered to the media management subsystem.

Keywords and Parameters

Syntax Element	Description
<code>DEVICE TYPE</code> <i>deviceSpecifier</i>	Specifies the type of storage device and sends the command to all channels of the specified type. See Also: " deviceSpecifier " on page 2-104
<code>CHANNEL 'channel_id'</code>	Specifies which channel to use. If you do not specify <code>DEVICE TYPE</code> or <code>CHANNEL</code> , then RMAN uses all allocated channels. You must specify a case-sensitive channel ID, which is the name of the channel, after the <code>CHANNEL</code> keyword. Oracle uses the channel ID to report I/O errors.
<code>'command'</code>	Specifies a vendor-specific media management command. See Also: Your media management documentation to determine which commands are supported

Syntax Element	Description
PARMS <i>'channel_parms'</i>	Specifies parameters affecting the non-disk device you have allocated. The maximum length of the quoted string is 1000 bytes. Note: The only command from which the PARMS option will set environment variables in the server session is ALLOCATE CHANNEL or CONFIGURE CHANNEL.

Example

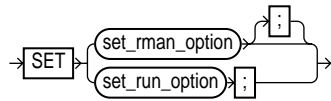
Sending a String to the Media Manager: Example This example sends vendor-specific commands to a media manager:

```
RMAN> SEND 'VAR=a82';  
sent command to channel: ORA_SBT_TAPE_1
```

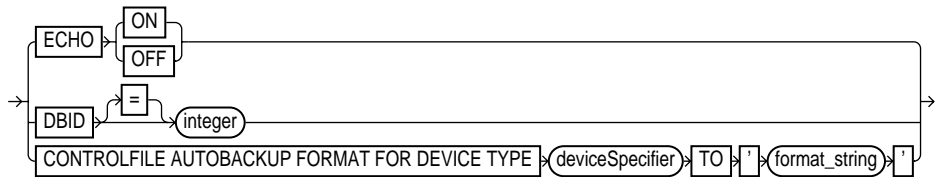
```
RMAN> BACKUP DATAFILE 2;
```

SET

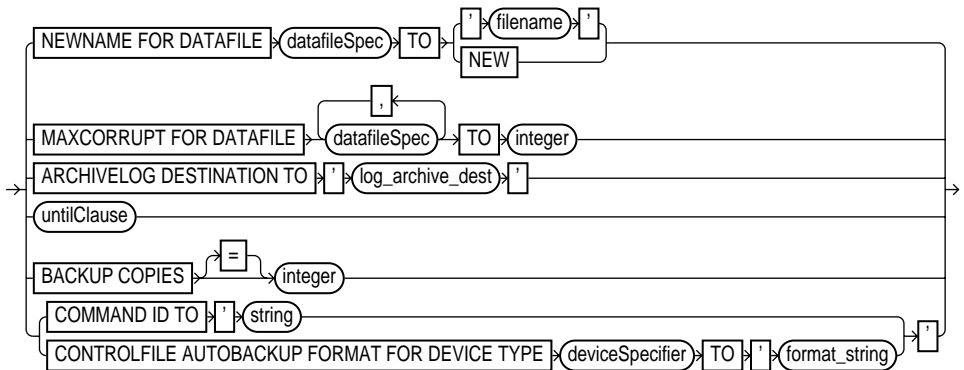
Syntax



set_rman_option ::=



set_run_option ::=



Purpose

To configure settings that apply *only* to the current RMAN session. The `SET` command contrasts with the `CONFIGURE` command, which configures persistent settings that apply to all RMAN sessions.

You can specify the `SET` command either at the RMAN prompt or within a `RUN` block. When you issue `SET` within a `RUN` block, the command sets attributes for a `RUN` command that persist until the end of the job. The specified attributes affect all statements within `RUN` that follow the `SET` command.

Use the `SET` specified at the RMAN prompt to:

- Display executed RMAN commands in the message log.
- Specify a database's database identifier (DBID).

Use `SET` specified within a `RUN` block to:

- Specify the filenames for the auxiliary database during TSPITR or database duplication.
- Specify a limit for the number of permissible block corruptions.
- Override default archived redo log destinations.
- Set an end time, SCN, or log sequence number for recovery.
- Specify that backups should be duplexed, that is, multiple copies should be created of each backup piece in the backup set.
- Determine which server session corresponds to which channel.
- Turn RMAN's automatic location feature on or off.
- Override the default format for control file autobackups at the session level.

Restrictions and Usage Notes for SET Command Within RUN

The following restrictions apply to `SET` when issued within a `RUN` command:

- The `SET BACKUP COPIES` command affects all backups in the `RUN` block after issuing the command and is in effect until explicitly disabled or changed. The `SET BACKUP COPIES` command does not affect previous backups.
- Include the `%F` substitution variable in the autobackup format.
- You cannot use `SET NEWNAME TO NEW` when creating a duplicate or standby database or performing RMAN TSPITR.

Restrictions and Usage Notes for the SET DBID Command

You should only run the `SET DBID` command in the following specialized circumstances:

- You are not connected to a recovery catalog and want to restore the control file or server parameter file.
- You are connected to a recovery catalog want to restore the control file, but the database name is not unique in the recovery catalog.
- The server parameter file is lost and you want to restore it.

See Also: [Table 2-25](#) for `RESTORE CONTROLFILE` usage notes.

Keywords and Parameters

Syntax Element	Description
<code>ECHO {ON OFF}</code>	<p>Controls whether RMAN commands are displayed in the message log. When reading commands from a command file, RMAN automatically echoes those commands to the message log. When reading commands from standard input, RMAN does not echo those commands to the message log unless the <code>SET ECHO ON</code> command is used.</p> <p>The command is useful only when <code>stdin</code> and <code>stdout</code> have been redirected. For example, in UNIX you can redirect RMAN's input and output in this manner:</p> <pre>% rman TARGET sys/sys_pwd@prodl CATALOG rman/rman@rcat < input_file > output_file</pre> <p>By running <code>SET ECHO ON</code>, you enable the commands contained in <code>input_file</code> to be visible in <code>output_file</code>.</p>
<code>DBID integer</code>	<p>Specifies the DBID, which is a unique 32-bit identification number computed when the database is created. RMAN displays the DBID upon connection to the target database. You can obtain the DBID by querying the <code>V\$DATABASE</code> view or the <code>RC_DATABASE</code> and <code>RC_DATABASE_INCARNATION</code> recovery catalog views.</p> <p>See Also: "Restrictions and Usage Notes for the SET DBID Command" on page 2-197</p>

Syntax Element	Description
CONTROLFILE AUTOBACKUP FORMAT FOR DEVICE TYPE <i>deviceSpecifier</i> TO ' <i>format_string</i> '	<p>Overrides the default filename format for the control file autobackup on the specified device type. The override occurs at the session level only. You can run this command either in RUN or at the RMAN prompt. The order of precedence is as follows:</p> <ol style="list-style-type: none"> 1. SET CONTROLFILE AUTOBACKUP executed within a RUN block 2. SET CONTROLFILE AUTOBACKUP executed at the RMAN prompt 3. CONFIGURE CONTROLFILE AUTOBACKUP FORMAT <p>See Also: The CONFIGURE command for an explanation of the autobackup format.</p>
NEWNAME FOR DATAFILE <i>datafileSpec</i> TO	<p>Sets the default name for all subsequent RESTORE or SWITCH commands that affect the specified datafile. If you do not issue this command before the datafile restore operation, then RMAN restores the file to its default location.</p> <p>After you restore a datafile to a new location, then you can run SWITCH to rename the file in the control file to the NEWNAME. If you do not run SWITCH, then the restored file functions as a datafile copy and is recorded as such in the repository.</p> <p>See Also: "datafileSpec" on page 2-96</p>
' <i>filename</i> '	Specifies a user-defined filename for the restored datafile.
NEW	<p>Creates an Oracle-managed file in the directory specified in <code>DB_CREATE_FILE_DEST</code>. You cannot use this option when using the <code>DUPLICATE</code> command or performing RMAN TSPITR.</p> <p>See Also: <i>Oracle9i Database Administrator's Guide</i> for information about Oracle Managed Files</p>
MAXCORRUPT FOR DATAFILE <i>datafileSpec</i> TO <i>integer</i>	<p>Sets a limit on the number of previously undetected physical block corruptions that Oracle will allow in a specified datafile or list of datafiles. If a BACKUP or COPY command detects more than the specified number of corruptions, then the command terminates. The default limit is zero, meaning that RMAN tolerates no corrupt blocks.</p> <p>Note: If you specify <code>CHECK LOGICAL</code>, then the <code>MAXCORRUPT</code> limit applies to logical corruptions as well.</p> <p>See Also: "datafileSpec" on page 2-96</p>

Syntax Element	Description
<pre>ARCHIVELOGDESTINATIONTO 'log_archive_dest'</pre>	<p>Overrides the LOG_ARCHIVE_DEST_1 initialization parameter in the target database when forming names for restored archive logs during subsequent RESTORE and RECOVER commands. RMAN restores the logs to the destination specified in 'log_archive_dest'. Use this parameter to restore archived redo logs that are not already on disk.</p> <p>Use this command to stage many archived logs to different locations while a database restore is occurring. RMAN knows where to find the newly restored archive logs; it does not require them to be in the destination specified by LOG_ARCHIVE_DEST_1. For example, if you specify a different destination from the one in the parameter file and restore archived redo log backups, subsequent restore and recovery operations will detect this new location. RMAN always looks for archived redo logs on disk first before restoring them from backup sets.</p>
<pre><i>untilClause</i></pre>	<p>Specifies an end time, SCN, or log sequence number for a subsequent RESTORE or RECOVER command.</p> <p>See Also: "untilClause" on page 2-216</p>
<pre>BACKUP COPIES = <i>integer</i></pre>	<p>Specifies the number of copies of each backup piece that the channels should create: 1, 2, 3, or 4. The SET BACKUP COPIES command, which affects only the BACKUP command, affects all channels allocated in the session. The order of precedence is as follows, with settings higher on the list overriding settings lower on the list:</p> <ul style="list-style-type: none"> ■ BACKUP COPIES ■ SET BACKUP COPIES ■ CONFIGURE . . . BACKUP COPIES <p>The names of the backup pieces are dependent on the FORMAT clause in the BACKUP command. You can specify up to four FORMAT strings. RMAN uses the second, third, and fourth values only when BACKUP COPIES, SET BACKUP COPIES, or CONFIGURE . . . BACKUP COPIES is in effect. When choosing which format to use for each backup piece, RMAN uses the first format value for copy 1, the second format value for copy 2, and so on. If the number of format values exceeds the number of copies, then the extra formats are not used. If the number of format values is less than the number of copies, RMAN reuses the format values, starting with the first one.</p> <p>Note: Control file autobackups on disk are a special case and are never duplexed: RMAN always writes one and only copy.</p>

Syntax Element	Description
COMMAND ID TO ' <i>string</i> '	<p>Enters the specified string into the <code>V\$SESSION.CLIENT_INFO</code> column of all channels. Use this information to determine which Oracle server sessions correspond to which RMAN channels. The <code>SET COMMAND ID</code> command applies only to channels that are already allocated.</p> <p>The <code>V\$SESSION.CLIENT_INFO</code> column contains information for each RMAN server session. The data appears in one of the following formats:</p> <ul style="list-style-type: none"> ▪ <code>id=string</code> ▪ <code>id=string, ch=channel_id</code> <p>The first form appears in the RMAN target database connection. The second form appears in all allocated channels. When the current job is complete, the <code>V\$SESSION.CLIENT_INFO</code> column will be cleared.</p> <p>See Also: <i>Oracle9i Database Reference</i> for more on <code>V\$SESSION.CLIENT_INFO</code></p>

Examples

Restoring the Control File When Databases Share the Same Name: Example The following shell script uses the DBID to restore the control file because multiple target databases share the same `DB_NAME` in the catalog. After you have restored the target control file, you can mount the database to restore the rest of the database:

```
#!/usr/bin/tcsh
rman TARGET / CATALOG rman/rman@catdb <<EOF
STARTUP FORCE NOMOUNT;
SET DBID = 862893450; # needed to distinguish target from others with same DB_NAME
RESTORE CONTROLFILE; # assuming catalog has automatic channel allocation information
ALTER DATABASE MOUNT;
EOF
```

Setting the Command ID: Example This example sets the command ID, backs up the `users` tablespace, then archives the online redo logs:

```
RUN
{
  ALLOCATE CHANNEL t1 DEVICE TYPE DISK FORMAT '/disk1/%U';
  ALLOCATE CHANNEL t2 DEVICE TYPE DISK FORMAT '/disk2/%U';
  SET COMMAND ID TO 'rman';
  BACKUP INCREMENTAL LEVEL 0 FILESPERSET 5 TABLESPACE users;
  SQL 'ALTER SYSTEM ARCHIVE LOG ALL';
}
```

Duplexing a Backup Set: Example Assume that you have used the CONFIGURE command to set duplexing as follows:

```
CONFIGURE ARCHIVELOG COPIES FOR DEVICE TYPE sbt TO 4;
CONFIGURE DATAFILE COPIES FOR DEVICE TYPE sbt TO 3;
```

The following example overrides these configurations and makes two copies of each datafile and archived log in the backup:

```
RUN
{
  ALLOCATE CHANNEL dev1 DEVICE TYPE sbt;
  SET BACKUP COPIES = 2;
  BACKUP FILESPERSET 1 DATAFILE 1,2,3,4,5;
  BACKUP FILESPERSET 10 ARCHIVELOG ALL;
}
```

Overriding the Autobackup Format During a Restore: Example This example sets the DBID, sets a boundary time for the restore, then restores a control file autobackup with a nondefault format. First start RMAN and then run:

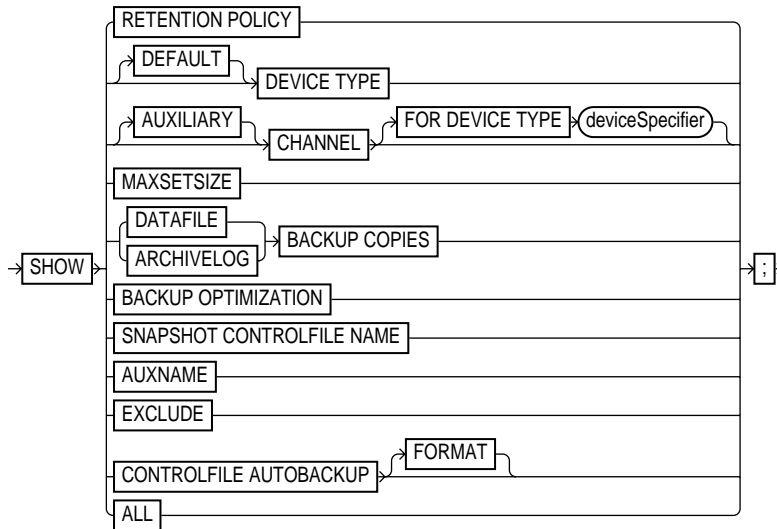
```
CONNECT TARGET / NOCATALOG
STARTUP FORCE NOMOUNT
SET DBID 676549873;
RUN
{
  SET UNTIL TIME '10/10/2001 13:45:00';
  SET CONTROLFILE AUTOBACKUP FORMAT FOR DEVICE TYPE DISK TO '?/oradata/cf_%F.bak';
  RESTORE CONTROLFILE FROM AUTOBACKUP MAXSEQ 100;
}
```

Restoring the Server Parameter File: Example This example restores a lost server parameter file:

```
CONNECT TARGET / CATALOG rman/rman@catdb
SET DBID 676549873; # set dbid so rman knows the database name
STARTUP FORCE NOMOUNT # rman starts database with a dummy server parameter file
RESTORE SPFILE;
STARTUP FORCE; # needed so that RMAN restarts database with restored server parameter file
```

SHOW

Syntax



Purpose

To display the current `CONFIGURE` command settings. The output of `SHOW` consists of the `CONFIGURE` commands used to set the configuration. RMAN default configurations are suffixed with `#default`.

Restrictions and Usage Notes

Execute this command at the RMAN prompt.

Keywords and Parameters

Syntax Element	Description
RETENTION POLICY	Displays the settings for CONFIGURE RETENTION POLICY for the current target database.
[DEFAULT] DEVICE TYPE	Displays the configured device types and parallelism settings. If DEFAULT is specified, then SHOW displays the default device type and settings.
[AUXILIARY] CHANNEL	Displays the CONFIGURE CHANNEL settings. You can specify a normal channel or an AUXILIARY channel.
FOR DEVICE TYPE <i>deviceSpecifier</i>	Specifies the device type of the channel. For example, SHOW CHANNEL FOR DEVICE TYPE DISK displays only channel settings for disk channels.
MAXSETSIZE	Displays the CONFIGURE MAXSETSIZE settings.
{DATAFILE ARCHIVELOG} BACKUP COPIES	Displays the CONFIGURE . . . BACKUP COPIES setting for datafiles and archived redo logs: 1, 2, 3, or 4.
BACKUP OPTIMIZATION	Displays the CONFIGURE BACKUP OPTIMIZATION settings: ON or OFF (default).
SNAPSHOT CONTROLFILE NAME	Displays the CONFIGURE SNAPSHOT CONTROLFILE settings.
AUXNAME	Displays the CONFIGURE AUXNAME settings.
EXCLUDE	Displays only the tablespaces that you have specified should be excluded.
CONTROLFILE AUTOBACKUP	Displays the CONFIGURE CONTROLFILE AUTOBACKUP settings: ON or OFF.
FORMAT	Displays the format for the control file autobackup file for configured devices.
ALL	Displays all user-entered CONFIGURE commands as well as default configurations.

Examples

Showing Channel Configurations: Example This example shows commands relevant for displaying automatic channel configurations:

```
SHOW CHANNEL;
SHOW DEVICE TYPE;
SHOW DEFAULT DEVICE TYPE;
SHOW MAXSETSIZE;
```

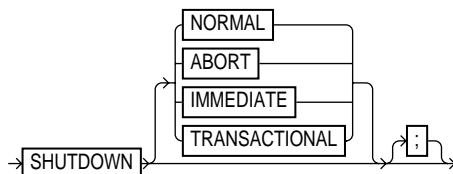
Showing All Configurations: Example This example shows all persistent configurations for the target database (and includes sample output):

```
SHOW ALL;
```

```
RMAN configuration parameters are:
CONFIGURE RETENTION POLICY TO REDUNDANCY 1; # default
CONFIGURE BACKUP OPTIMIZATION ON;
CONFIGURE DEFAULT DEVICE TYPE TO DISK; # default
CONFIGURE DEVICE TYPE "SBT" PARALLELISM 1;
CONFIGURE DEVICE TYPE DISK PARALLELISM 1; # default
CONFIGURE DATAFILE BACKUP COPIES FOR DISK TO 2;
CONFIGURE DATAFILE BACKUP COPIES FOR SBT TO 1; #default
CONFIGURE ARCHIVELOG BACKUP COPIES FOR SBT TO 1; # default
CONFIGURE ARCHIVELOG BACKUP COPIES FOR DISK TO 1; # default
CONFIGURE MAXSETSIZE TO 2097152K;
CONFIGURE SNAPSHOT CONTROLFILE NAME TO '/oracle/dbs/cf_snap.f';
```

SHUTDOWN

Syntax



Purpose

To shut down the target database without exiting RMAN. This command is equivalent to using the SQL*Plus `SHUTDOWN` statement.

See Also: *Oracle9i Database Administrator's Guide* for information on how to start up and shut down a database, and *SQL*Plus User's Guide and Reference* for `SHUTDOWN` syntax

Restrictions and Usage Notes

- Execute this command at the RMAN prompt or within the braces of a [RUN](#) command.
- You cannot use the RMAN `SHUTDOWN` command to shut down the recovery catalog database. To shut down this database, start a SQL*Plus session and issue a `SHUTDOWN` statement.
- The `NORMAL`, `TRANSACTIONAL`, and `IMMEDIATE` options all perform a clean close of the database. The `ABORT` option does not cleanly close the database; Oracle will perform instance recovery at startup.
- If the database operates in `NOARCHIVELOG` mode, then you must shut down the database cleanly and then issue a [STARTUP MOUNT](#) before a making a backup.

Keywords and Parameters

Syntax Element	Description
NORMAL	Shuts down the database with normal priority (default option), which means: <ul style="list-style-type: none">■ No new connections are allowed after the statement is issued.■ Before the database shuts down, Oracle waits for currently connected users to disconnect■ The next startup of the database will not require instance recovery.
ABORT	Shuts down the target instance, with the following consequences: <ul style="list-style-type: none">■ All current client SQL statements are immediately terminated.■ Uncommitted transactions are not rolled back until next startup.■ Oracle disconnects all connected users.■ Oracle will perform crash recovery on the database at next startup.
IMMEDIATE	Shuts down the target database immediately, with the following consequences: <ul style="list-style-type: none">■ Current client SQL statements being processed by Oracle are allowed to complete.■ Uncommitted transactions are rolled back.■ All connected users are disconnected.
TRANSACTIONAL	Shuts down the target database while minimizing interruption to clients, with the following consequences: <ul style="list-style-type: none">■ Clients currently conducting transactions are allowed to complete, that is, either commit or terminate before shutdown.■ No client can start a new transaction on this instance; any client attempting to start a new transaction is disconnected.■ After all transactions have either committed or terminated, any client still connected is disconnected.

Examples

Shutting Down a Database by Using the Immediate Option: Example This example waits for current SQL transactions to be processed before shutting down, then mounts the database:

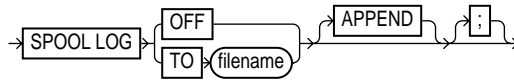
```
SHUTDOWN IMMEDIATE;  
STARTUP MOUNT;
```

Shutting Down a Database in NOARCHIVELOG Mode: Example This example backs up a database running in NOARCHIVELOG mode:

```
STARTUP FORCE DBA;  
SHUTDOWN IMMEDIATE;  
STARTUP MOUNT;  
# executing the preceding commands ensures that database is in proper state  
# for NOARCHIVELOG backups  
BACKUP COPIES 2 DATABASE;  
ALTER DATABASE OPEN;
```

SPOOL

Syntax



Purpose

To write RMAN output to a log file.

If the file does not already exist, then RMAN creates it. If the file does exist, then RMAN overwrites the file, unless `APPEND` is specified, in which case RMAN appends its output to the end of the file. The `SPOOL` command does not terminate if the specified file cannot be opened for writing. Instead, RMAN turns `SPOOL` to `OFF`.

See Also: ["cmdLine"](#) on page 2-59 for a description of `LOG` files

Restrictions and Usage Notes

Execute the `SQL` command at the RMAN prompt.

Keywords and Parameters

Syntax Element	Description
<code>OFF</code>	Turns off spooling.
<code>TO filename</code>	Specifies the name of the log file to which RMAN directs its output. RMAN creates the file if it does not exist, or overwrites the file if it does exist.
<code>APPEND</code>	Specifies that RMAN should append its output to the end of the existing log file.

Examples

Spooling RMAN Output to a File: Example This example directs RMAN output to standard output for the backup of datafile 1, then directs output to a log file for the backup of datafile 2, then directs output to a different log file for the whole database backup:

```
BACKUP DATAFILE 1;  
SPOOL LOG TO '/tmp/df2log.f';  
BACKUP DATAFILE 2;  
SPOOL LOG OFF;  
SPOOL LOG TO '/tmp/dblog.f';  
BACKUP DATABASE;  
SPOOL LOG OFF;
```

SQL

Syntax

→ SQL >'>' (command) >'>' ; >

Purpose

To execute a SQL statement or a PL/SQL stored procedure from within Recovery Manager.

Restrictions and Usage Notes

- Execute the SQL command at the RMAN prompt or within the braces of a [RUN](#) command.
- If the string that RMAN passes to PL/SQL contains a filename, then the filename must be enclosed in duplicate *single* quotes and the entire string following the SQL keyword must be enclosed in *double* quotes. For example, use the following syntax:

```
SQL "CREATE TABLESPACE temp1 DATAFILE ' '?/oradata/trgt/temp1.dbf' '  
    SIZE 10M TEMPORARY";
```

If you attempt to use single quotes for the string following the SQL keyword or use only one set of single quotes for the filename, then the command fails.

- You cannot execute SELECT statements.

See Also: For valid SQL syntax, see the *Oracle9i SQL Reference*

Keywords and Parameters

Syntax Element	Description
'command'	<p>Specifies a SQL statement for execution. For example, issue the following at the RMAN prompt to archive the online redo logs:</p> <pre>SQL 'ALTER SYSTEM ARCHIVE LOG ALL';</pre> <p>Because EXECUTE is a SQL*Plus command, you cannot execute a PL/SQL command by specifying EXECUTE within the RMAN SQL command. Instead, you must use the BEGIN and END keywords. For example, to execute a PL/SQL procedure named <code>rman.rman_purge</code> through the RMAN SQL command, issue the following:</p> <pre>SQL 'BEGIN rman.rman_purge; END;';</pre>

Examples

Archiving the Unarchived Online Logs: Example This example backs up a tablespace and then archives all unarchived online logs:

```
BACKUP TABLESPACE users;
SQL "ALTER SYSTEM ARCHIVE LOG CURRENT";
```

Specifying a Filename within a Quoted String: Example This example specifies a filename by using duplicate single quotes within the context of a double-quoted string:

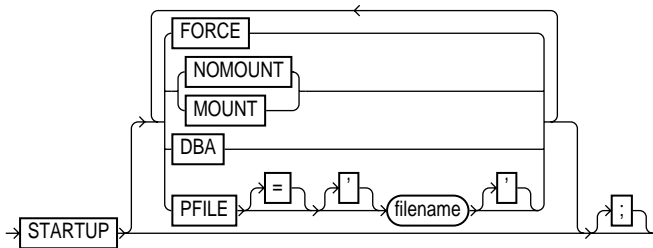
```
SQL "ALTER TABLESPACE tbs_1 ADD DATAFILE ' '/oracle/dbs/tbs_7.f' ' NEXT 10K MAXSIZE 100k;"
```

Executing a PL/SQL Stored Procedure Within RMAN: Example This example issues a PL/SQL stored procedure called `scott.update_log`:

```
RUN
{
  SQL ' BEGIN scott.update_log; END; ';
}
```

STARTUP

Syntax



Purpose

To start the target database from within the RMAN environment. This command is equivalent to using the SQL*Plus `STARTUP` command. You can:

- Start the instance without mounting a database.
- Start the instance and mount the database, but leave it closed.
- Start the instance, and mount and open the database in:
 - unrestricted mode (accessible to all users)
 - restricted mode (accessible to DBAs only)

Additionally, the RMAN `STARTUP` command can start an instance in `NOMOUNT` mode even if no server parameter file or initialization parameter file exists. This feature is useful when you need to restore a lost server parameter file.

See Also: *Oracle9i Database Administrator's Guide* to learn how to start up and shut down a database, and *SQL*Plus User's Guide and Reference* for SQL*Plus `STARTUP` syntax

Restrictions and Usage Notes

- Execute this command either at the RMAN prompt or within the braces of a [RUN](#) command.
- You cannot use the RMAN `STARTUP` command to open the recovery catalog database: execute a `STARTUP` statement in a SQL*Plus session instead.

Keywords and Parameters

If you do not specify any options, then RMAN mounts and opens the database with the default server parameter file.

Syntax Element	Description
STARTUP	If you specify only <code>STARTUP</code> with no other options, then Oracle starts the instance, then mounts and open the database.
FORCE	If the database is open, then <code>FORCE</code> shuts down the database with a <code>SHUTDOWN ABORT</code> statement before re-opening it. If the database is closed, then <code>FORCE</code> opens the database.
NOMOUNT	Starts the instance without mounting the database. If no parameter file exists, then RMAN starts the instance with a "dummy" parameter file. You can then run <code>RESTORE SPFILE</code> to restore a backup server parameter file.
MOUNT	Starts the instance, then mounts the database without opening it
DBA	Restricts access to the database to users with the <code>RESTRICTED SESSION</code> privilege.
<code>PFIL = 'filename'</code>	Specifies the filename of the <code>init.ora</code> file for the target database. If this parameter is not specified, then the default <code>init.ora</code> filename is used.

Examples

Opening the Database by Using the Default Parameter File: Example This example starts and opens the database:

```
STARTUP;
```

Mounting the Database While Specifying the Parameter File: Example This example forces a `SHUTDOWN ABORT` and then mounts the database with restricted access, specifying a nondefault parameter file location:

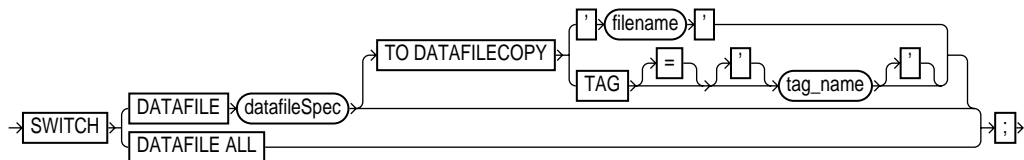
```
STARTUP FORCE MOUNT DBA PFIL=/tmp/initTRGT.ora;
```

Starting an Instance Without a Parameter File: Example The following example starts an instance without using a parameter file, then runs `RESTORE SPFILE`:

```
SET DBID 1447326980;
STARTUP FORCE NOMOUNT; # RMAN starts instance with dummy parameter file
RESTORE SPFILE FROM AUTOBACKUP; # restore a server parameter file
STARTUP FORCE; # restart instance with restored server parameter file
```

SWITCH

Syntax



Purpose

To specify that a datafile copy is now the **current datafile**, that is, the datafile pointed to by the control file. A switch is equivalent to using the `ALTER DATABASE RENAME FILE` statement: Oracle renames the files in the control file but does not actually rename them on the operating system. Note that this command deletes the records for the datafile copy from the recovery catalog and updates the control file records to status `DELETED`.

Restrictions and Usage Notes

- Execute `SWITCH` within the braces of a `RUN` command.
- If the control file is a restored backup control file, then `SWITCH` adds the datafile to the control file if it is not there already. You can only add datafiles through `SWITCH` that were created *after* the backup control file was created.

Keywords and Parameters

Syntax Element	Description
<code>DATAFILE</code> <i>datafileSpec</i>	<p>Specifies the datafile that you wish to rename. After the switch, the control file no longer views the specified file as current. For example, this command points the control file from <code>tbs_1.f</code> to <code>cp1.f</code>:</p> <pre>SWITCH DATAFILE '/oracle/dbs/tbs_1.f' TO DATAFILECOPY '/oracle/dbs/copies/cp1.f';</pre> <p>If you do not specify a <code>TO</code> option, then RMAN uses the filename specified on a prior <code>SET NEWNAME</code> command for this file number as the switch target.</p>

Syntax Element	Description
<pre>TO DATAFILECOPY {'filename' TAG = 'tag_name'}</pre>	<p>Specifies the input copy file for the switch, that is, the datafile copy that you wish to rename. Specify the file by filename or tag. For example, the following command sets <code>df2.copy</code> as the filename for datafile 2:</p> <pre>SWITCH DATAFILE 2 TO DATAFILECOPY '/oracle/dbs/df2.copy';</pre> <p>Note that if you specify a tag and more than one copy uses this tag name, then RMAN uses the most current copy, that is, the one needing the least recovery.</p> <p>The following command switches datafile 3 to the most recently created Monday evening copy:</p> <pre>SWITCH DATAFILE 3 TO DATAFILECOPY TAG mondayPMcopy;</pre>
DATAFILE ALL	<p>Specifies that all datafiles for which a SET NEWNAME FOR DATAFILE command has been issued in this job are switched to their new name.</p>

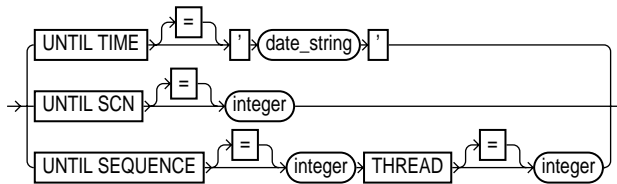
Example

Switching Datafile Filenames After a Restore: Example This example allocates one disk device and one tape device to allow RMAN to restore both from disk and tape.

```
RUN
{
  ALLOCATE CHANNEL dev1 DEVICE TYPE DISK RATE 1000;
  ALLOCATE CHANNEL dev2 DEVICE TYPE sbt;
  SQL "ALTER TABLESPACE tbs_1 OFFLINE IMMEDIATE";
  SET NEWNAME FOR DATAFILE '/disk7/oracle/tbs11.f'
    TO '/disk9/oracle/tbs11.f';
  RESTORE TABLESPACE tbs_1;
  SWITCH DATAFILE ALL;
  RECOVER TABLESPACE tbs_1;
  SQL "ALTER TABLESPACE tbs_1 ONLINE";
}
```

untilClause

Syntax



Purpose

A subclause that specifies an upper limit by time, SCN, or log sequence number for various RMAN operations.

See Also: *Oracle9i Recovery Manager User's Guide* to learn how to set the Globalization Support date format used by RMAN

Restrictions and Usage Notes

When specifying dates in RMAN commands, the date string must be either:

- A literal string whose format matches the `NLS_DATE_FORMAT` setting.
- A SQL expression of type `DATE`, for example, `'SYSDATE-10'` or `"TO_DATE('01/30/1997', 'MM/DD/YYYY')"`. Note that the second example includes its own date format mask and so is independent of the current `NLS_DATE_FORMAT` setting.

Following are examples of typical date settings in RMAN commands:

```

BACKUP ARCHIVELOG FROM TIME 'SYSDATE-31' UNTIL TIME 'SYSDATE-14';
RESTORE DATABASE UNTIL TIME "TO_DATE('09/20/00', 'MM/DD/YY')";
  
```

Keywords and Parameters

Syntax Element	Description
UNTIL TIME = 'date_string'	Specifies a time as an upper limit. RMAN selects only files that can be used to recover up to but not including the specified time. For example, <code>LIST BACKUP UNTIL TIME 'SYSDATE-7'</code> lists all backups that could be used to recover to a point one week ago.

Syntax Element	Description
UNTIL SCN = <i>integer</i>	Specifies an SCN as an upper limit. RMAN selects only files that can be used to recover up to but not including the specified SCN. For example, RESTORE DATABASE UNTIL SCN 1000 chooses only backups that could be used to recover to SCN 1000.
UNTIL SEQUENCE = <i>integer</i> THREAD = <i>integer</i>	Specifies a redo log sequence number and thread as an upper limit. RMAN selects only files that can be used to recover up to but not including the specified sequence number. For example, REPORT OBSOLETE UNTIL SEQUENCE 8000 THREAD 1 reports only backups that could be used to recover through log sequence 7999.

Examples

Performing Incomplete Recovery Until a Log Sequence Number: Example This example assumes that log sequence 1234 was lost due to a disk failure and the database needs to be recovered by using available archived redo logs.

```

RUN
{
  SET UNTIL SEQUENCE 1234 THREAD 1;
  RESTORE CONTROLFILE TO '?/oradata/cf.tmp';
  RESTORE CONTROLFILE FROM '?/oradata/cf.tmp'; # restores to all CONTROL_FILES locations
  ALTER DATABASE MOUNT;
  RESTORE DATABASE;
  RECOVER DATABASE; # recovers through log 1233
  ALTER DATABASE OPEN RESETLOGS;
  # you must add new tempfiles to locally-managed temporary tablespaces after restoring
  # a backup control file
  SQL "ALTER TABLESPACE temp ADD TEMPFILE '?/oradata/trgt/temp01.dbf' REUSE";
}

```

Performing Incomplete Recovery to a Specified SCN: Example This example (which assumes a mounted database) recovers the database until a specified SCN:

```

RUN
{
  ALLOCATE CHANNEL ch1 TYPE sbt;
  RESTORE DATABASE;
  RECOVER DATABASE UNTIL SCN 1000; # recovers through SCN 999
  ALTER DATABASE OPEN RESETLOGS;
}

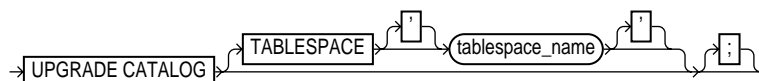
```

Reporting Obsolete Backups: Example This example assumes that you want to be able to recover to any point within the last week. It considers as obsolete all backups that could be used to recover the database to a point one week ago:

```
REPORT OBSOLETE UNTIL TIME 'SYSDATE-7';
```

UPGRADE CATALOG

Syntax



Purpose

To upgrade the recovery catalog schema from an older version to the version required by the RMAN executable. For example, if you use a release 8.0 recovery catalog with a release 8.1 version of RMAN, then you must upgrade the catalog.

Note that `UPGRADE CATALOG` does not run scripts to perform the upgrade. Instead, RMAN sends various SQL DDL statements to the recovery catalog to update the recovery catalog schema with new tables, views, columns, and so forth.

Restrictions and Usage Notes

- You must be connected to the catalog database, and the catalog database must be open. You do not have to be connected to the target database.
- You must enter the `UPGRADE` command twice in a row to confirm the upgrade.
- You will receive an error if the recovery catalog is already at a version greater than needed by the RMAN executable. RMAN permits the command to be run if the recovery catalog is already current, however, so that the packages can be re-created if necessary. RMAN displays all error messages generated during the upgrade in the message log.

Keywords and Parameters

Syntax Element	Description
TABLESPACE 'tablespace_name'	Specifies the tablespace in which to store the recovery catalog. If not specified, then no <code>TABLESPACE</code> parameter will be used for the new tables that may be created in the upgrade. New tables will be stored in the default tablespace for the catalog owner.

Example

Upgrading a Recovery Catalog: Example This example connects to recovery catalog database `recdb` at the operating system command line and then upgrades it to a more current version:

```
% rman CATALOG rcat/rcat@recdb

connected to recovery catalog database
PL/SQL package rcat.DBMS_RVCAT version 08.00.04 in RVCAT database is too old

RMAN> UPGRADE CATALOG

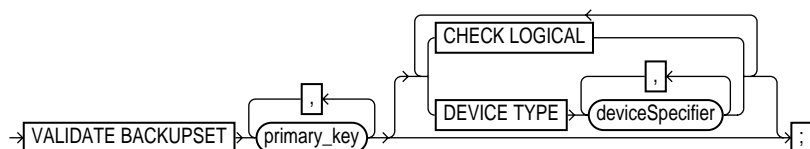
recovery catalog owner is rcat
enter UPGRADE CATALOG command again to confirm catalog upgrade

RMAN> UPGRADE CATALOG

recovery catalog upgraded to version 09.00.01
DEMS_RCVMAN package upgraded to version 09.00.01
DEMS_RVCAT package upgraded to version 09.00.01
```

VALIDATE

Syntax



Purpose

To examine a backup set and report whether it can be restored. RMAN scans all of the backup pieces in the specified backup sets and looks at the checksums to verify that the contents are intact so that the backup can be successfully restored if necessary.

Note: The `VALIDATE BACKUPSET` command tests whether the backup sets can be restored, whereas `CROSSCHECK` merely examines the headers of the specified files if they are on disk or queries the media management catalog if they are on tape.

Use this command when you suspect that one or more backup pieces in a backup set are missing or have been damaged. Use `VALIDATE BACKUPSET` to specify which backups to test; use the `VALIDATE` option of the `RESTORE` command to let RMAN choose which backups to validate.

Restrictions and Usage Notes

- Execute this command within the braces of a `RUN` command or at the RMAN prompt.
- If you do not have automatic channels configured, manually allocate at least one channel before executing a `VALIDATE BACKUPSET` statement.
- The target instance must be started.

Keywords and Parameters

Syntax Element	Description
<i>primary_key</i>	Specifies the backup sets to be validated by <i>primary_key</i> . Obtain the primary keys of backup sets by executing a LIST statement or, if you use a recovery catalog, by querying the RC_BACKUP_SET recovery catalog view.
CHECK LOGICAL	Tests data and index blocks that pass physical corruption checks for logical corruption, for example, corruption of a row piece or index entry. If RMAN finds logical corruption, then it logs the block in the <code>alert.log</code> and server session trace file. The RMAN command completes and Oracle populates the <code>V\$DATABASE_BLOCK_CORRUPTION</code> view with corrupt block ranges. Note: VALIDATE does not use MAXCORRUPT.
DEVICE TYPE <i>deviceSpecifier</i>	Allocates automatic channels for the specified device type only. This option is valid only if you have configured automatic channels and have not manually allocated channels. For example, if you configure automatic disk and tape channels, and run <code>VALIDATE . . . DEVICE TYPE DISK</code> , RMAN allocates only disk channels. See Also: " deviceSpecifier " on page 2-104

Example

Validating a Backup Set: Example This example validates the status of the backup set whose primary key is 218:

```
VALIDATE BACKUPSET 218;
# As the output indicates, RMAN determines whether it is possible to restore the
# specified backup set.

allocated channel: ORA_SBT_TAPE_1
channel ORA_SBT_TAPE_1: sid=14 devtype=SBT_TAPE
using channel ORA_DISK_1
channel ORA_SBT_TAPE_1: starting validation of datafile backupset
channel ORA_SBT_TAPE_1: restored backup piece 1
piece handle=09dg9kkl_1_1 tag=TAG20020208T125443 params=NULL
channel ORA_SBT_TAPE_1: validation complete
```

Recovery Catalog Views

This chapter contains descriptions of recovery catalog views. You can only access these views if you have created a recovery catalog. For a summary of the recovery catalog views, refer to "[Summary of RMAN Recovery Catalog Views](#)" on page 3-2.

Note: These views are not normalized, but are optimized for RMAN usage. Hence, most catalog views have redundant values that result from joining of several underlying tables.

Summary of RMAN Recovery Catalog Views

The following table provides a functional summary of RMAN recovery catalog views.

Table 3–1 Recovery Catalog Views

Recovery Catalog View	Corresponding V\$ View	Catalog View Describes ...
RC_ARCHIVED_LOG	V\$ARCHIVED_LOG	Archived and unarchived redo logs
RC_BACKUP_CONTROLFILE	V\$BACKUP_DATAFILE	Control files in backup sets
RC_BACKUP_CORRUPTION	V\$BACKUP_CORRUPTION	Corrupt block ranges in datafile backups
RC_BACKUP_DATAFILE	V\$BACKUP_DATAFILE	Datafiles in backup sets
RC_BACKUP_PIECE	V\$BACKUP_PIECE	Backup pieces
RC_BACKUP_REDOLOG	V\$BACKUP_REDOLOG	Archived redo logs in backup sets
RC_BACKUP_SET	V\$BACKUP_SET	Backup sets for all incarnations of the database
RC_BACKUP_SPFILE	V\$BACKUP_SPFILE	Server parameter files in backups
RC_CHECKPOINT		Deprecated in favor of RC_RESYNC
RC_CONTROLFILE_COPY	V\$DATAFILE_COPY	Control file copies on disk
RC_COPY_CORRUPTION	V\$COPY_CORRUPTION	Control file copies on disk
RC_DATABASE	V\$DATABASE	Databases registered in the recovery catalog
RC_DATABASE_BLOCK_CORRUPTION	V\$DATABASE_BLOCK_CORRUPTION	Database blocks marked as corrupted in the most recent RMAN backup or copy
RC_DATABASE_INCARNATION	V\$DATABASE_INCARNATION	Database incarnations registered in the recovery catalog
RC_DATAFILE	V\$DATAFILE	Datafiles registered in the recovery catalog
RC_DATAFILE_COPY	V\$DATAFILE_COPY	Datafile copies on disk
RC_LOG_HISTORY	V\$LOG_HISTORY	Online redo log history indicating when log switches occurred
RC_OFFLINE_RANGE	V\$OFFLINE_RANGE	Offline ranges for datafiles

Table 3–1 Recovery Catalog Views

Recovery Catalog View	Corresponding V\$ View	Catalog View Describes ...
RC_PROXY_CONTROLFILE	V\$PROXY_DATAFILE	Control file backups that were taken using the proxy copy functionality
RC_PROXY_DATAFILE	V\$PROXY_DATAFILE	Datafile backups that were taken using the proxy copy functionality
RC_REDO_LOG	V\$LOG and V\$LOGFILE	Online redo logs for all incarnations of the database since the last catalog resynchronization
RC_REDO_THREAD	V\$THREAD	All redo threads for all incarnations of the database since the last catalog resynchronization
RC_RESYNC	n/a	Recovery catalog resynchronizations
RC_RMAN_CONFIGURATION	V\$RMAN_CONFIGURATION	RMAN configuration settings
RC_STORED_SCRIPT	n/a	Names of scripts stored in the recovery catalog
RC_STORED_SCRIPT_LINE	n/a	Contents of the scripts stored in the recovery catalog
RC_TABLESPACE	V\$TABLESPACE	All tablespaces registered in the recovery catalog, all dropped tablespaces, and tablespaces that belong to old incarnations

RC_ARCHIVED_LOG

This view contains historical information about archived and unarchived redo logs. It corresponds to the V\$ARCHIVED_LOG view in the target database control file.

Oracle inserts an archived redo log record after the online redo log is successfully archived or cleared (`NAME` column is `NULL` if the log was cleared). If the log is archived multiple times, then the view will contain multiple archived log records with the same `THREAD#`, `SEQUENCE#`, and `RESETLOGS_CHANGE#`, but with a different name. An archived log record is also inserted when an archived log is restored from a backup set or a copy. Note that an archived log can have no record if the record ages out of the control file.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database. Use this column to form a join with almost any other catalog view.
DBINC_KEY	NUMBER	The primary key for the incarnation of the target database to which this record belongs. Use this column to form a join with <code>RC_DATABASE_INCARNATION</code> .
DB_NAME	VARCHAR2(8)	The <code>DB_NAME</code> of the database incarnation to which this record belongs.
AL_KEY	NUMBER	The primary key of the archived redo log in the recovery catalog. If you issue the <code>LIST</code> command while connected to the recovery catalog, this value appears in the <code>KEY</code> column of the output.
RECID	NUMBER	The archived redo log <code>RECID</code> from <code>V\$ARCHIVED_LOG</code> . <code>RECID</code> and <code>STAMP</code> form a concatenated primary key that uniquely identifies this record in the target database control file.
STAMP	NUMBER	The archived redo log stamp from <code>V\$ARCHIVED_LOG</code> . <code>RECID</code> and <code>STAMP</code> form a concatenated primary key that uniquely identifies this record in the target database control file.
NAME	VARCHAR2(1024)	The filename of the archived redo log.
THREAD#	NUMBER	The number of the redo thread.
SEQUENCE#	NUMBER	The log sequence number.
RESETLOGS_CHANGE#	NUMBER	The SCN of the most recent <code>RESETLOGS</code> when the record was created.
RESETLOGS_TIME	DATE	The time stamp of the most recent <code>RESETLOGS</code> when the record was created.
FIRST_CHANGE#	NUMBER	The first SCN of this redo log.
FIRST_TIME	DATE	The time when Oracle switched into the redo log.
NEXT_CHANGE#	NUMBER	The first SCN of the next redo log in the thread.
NEXT_TIME	DATE	The first time stamp of the next redo log in the thread.
BLOCKS	NUMBER	The size of this archived log in operating system blocks.

Column	Datatype	Description
BLOCK_SIZE	NUMBER	The size of the block in bytes.
COMPLETION_TIME	DATE	The time when the redo log was archived or copied.
ARCHIVED	VARCHAR2 (3)	Indicates whether the log was archived: YES (archived redo log) or NO (inspected file header of online redo log and added record to V\$ARCHIVED_LOG). Inspecting the online logs creates archived log records for them, which allows them to be applied during RMAN recovery. Oracle sets ARCHIVED to NO to prevent online logs from being backed up.
STATUS	VARCHAR2 (1)	The status of the archived redo log: A (available), U (unavailable), D (deleted), or X (expired).
IS_STANDBY	VARCHAR2 (3)	The database that archived this log: Y (belongs to a standby database) or N (belongs to the primary database).
DICTIONARY_BEGIN	VARCHAR2 (3)	Indicates whether this archived log contains the start of a LogMiner dictionary: YES or NO. If both DICTIONARY_BEGIN and DICTIONARY_END are YES, this log contains a complete LogMiner dictionary. If DICTIONARY_BEGIN is YES but DICTIONARY_END is NO, this log contains the start of the dictionary, and it continues through each subsequent log of this thread and ends in the log where DICTIONARY_END is YES.
DICTIONARY_END	VARCHAR2 (3)	Indicates whether this archived log contains the end of a LogMiner dictionary: YES or NO. See the description of DICTIONARY_BEGIN for an explanation of how to interpret this value.

RC_BACKUP_CONTROLFILE

This view lists information about control files in backup sets. Note that the V\$BACKUP_DATAFILE view contains both datafile and control file records: a backup datafile record with file number 0 represents the backup control file. In the recovery catalog, the RC_BACKUP_CONTROLFILE view contains only control file records, while the RC_BACKUP_DATAFILE view contains only datafile records.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database. Use this column to form a join with almost any other catalog view.
DBINC_KEY	NUMBER	The primary key for the incarnation of the target database. Use this column to form a join with RC_DATABASE_INCARNATION.
DB_NAME	VARCHAR2 (8)	The DB_NAME of the database incarnation to which this record belongs.
BCF_KEY	NUMBER	The primary key of the control file backup in the recovery catalog. If you issue the LIST command while connected to the recovery catalog, this value appears in the KEY column of the output.

Column	Datatype	Description
RECID	NUMBER	The RECID value from V\$BACKUP_DATAFILE. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
STAMP	NUMBER	The STAMP value from V\$BACKUP_DATAFILE. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
BS_KEY	NUMBER	The primary key of the backup set to which this record belongs in the recovery catalog. Use this column to form a join with RC_BACKUP_SET.
SET_STAMP	NUMBER	The SET_STAMP value from V\$BACKUP_SET. SET_STAMP and SET_COUNT form a concatenated key that uniquely identifies the backup set to which this record belongs in the target database control file.
SET_COUNT	NUMBER	The SET_COUNT value from V\$BACKUP_SET. SET_STAMP and SET_COUNT form a concatenated key that uniquely identifies the backup set to which this record belongs in the target database control file.
RESETLOGS_CHANGE#	NUMBER	The SCN of the most recent RESETLOGS when the record was created.
RESETLOGS_TIME	DATE	The time stamp of the most recent RESETLOGS when the record was created.
CHECKPOINT_CHANGE#	NUMBER	The control file checkpoint SCN.
CHECKPOINT_TIME	DATE	The control file checkpoint time.
CREATION_TIME	DATE	The control file creation time.
BLOCK_SIZE	NUMBER	The size of the blocks in bytes.
OLDEST_OFFLINE_RANGE	NUMBER	Internal use only.
STATUS	VARCHAR2(1)	The status of the backup set: A (available), U (unavailable), or D (deleted).
BS_RECID	NUMBER	The control file RECID of the backup set that contains this backup control file.
BS_STAMP	NUMBER	The control file stamp of the backup set that contains this backup control file.
BS_LEVEL	NUMBER	The incremental level (NULL, 0, 1, 2) of the backup set that contains this backup control file. Although an incremental backup set can contain the control file, it is always contains a complete copy of the control file. There is no such thing as an incremental control file backup.
COMPLETION_TIME	DATE	The date that the control file backup completed.
CONTROLFILE_TYPE	VARCHAR2(1)	The type of control file backup: B (normal backup) or S (standby backup).
BLOCKS	NUMBER	The number of blocks in the file.
AUTOBACKUP_DATE	DATE	The date of the control file autobackup.
AUTOBACKUP_SEQUENCE	NUMBER	The sequence of the control file autobackup: 1 - 255.

RC_BACKUP_CORRUPTION

This view lists corrupt block ranges in datafile backups. It corresponds to the V\$BACKUP_CORRUPTION view in the control file. Note that corruptions are not tolerated in control file and archived redo log backups.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database. Use this column to form a join with almost any other catalog view.
DBINC_KEY	NUMBER	The primary key for the incarnation of the target database. Use this column to form a join with RC_DATABASE_INCARNATION.
DB_NAME	VARCHAR2 (8)	The DB_NAME of the database incarnation to which this record belongs.
RECID	NUMBER	The record identifier from V\$BACKUP_CORRUPTION. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
STAMP	NUMBER	The stamp propagated from V\$BACKUP_CORRUPTION. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
BS_KEY	NUMBER	The primary key of the backup set to which this record belongs in the recovery catalog. Use this column to form a join with RC_BACKUP_SET.
SET_STAMP	NUMBER	The SET_STAMP value from V\$BACKUP_SET. SET_STAMP and SET_COUNT form a concatenated key that uniquely identifies the backup set to which this record belongs in the target database control file.
SET_COUNT	NUMBER	The SET_COUNT value from V\$BACKUP_SET. SET_STAMP and SET_COUNT form a concatenated key that uniquely identifies the backup set to which this record belongs in the target database control file.
PIECE#	NUMBER	The backup piece that contains this corrupt block.
BDF_KEY	NUMBER	The primary key for the datafile backup or copy in the recovery catalog. Use this key to join with RC_BACKUP_DATAFILE. If you issue the LIST command while connected to the recovery catalog, this value appears in the KEY column of the output.
BDF_RECID	NUMBER	The RECID value from V\$BACKUP_DATAFILE.
BDF_STAMP	NUMBER	The STAMP value from V\$BACKUP_DATAFILE.
FILE#	NUMBER	The absolute file number for the datafile that contains the corrupt blocks.
CREATION_CHANGE#	NUMBER	The creation SCN of the datafile containing the corrupt blocks.
BLOCK#	NUMBER	The block number of the first corrupted block in this range of corrupted blocks.
BLOCKS	NUMBER	The number of corrupted blocks found beginning with BLOCK#.
CORRUPTION_CHANGE#	NUMBER	For media corrupt blocks, this value is zero. For logically corrupt blocks, this value is the lowest SCN in the blocks in this corrupt range.

Column	Datatype	Description
MARKED_CORRUPT	VARCHAR2(3)	YES if this corruption was not previously detected by Oracle, or NO if Oracle had already discovered this corrupt block and marked it as corrupt in the database. Note that when a corrupt block is encountered in a backup, and was not already marked corrupt by Oracle, then the backup process does not mark the block as corrupt in the production datafile. Thus, this field may be YES for the same block in more than one backup set.
CORRUPTION_TYPE	VARCHAR2(9)	Same as RC_DATABASE_BLOCK_CORRUPTION.CORRUPTION_TYPE.

RC_BACKUP_DATAFILE

This view lists information about datafiles in backup sets. It corresponds to the V\$BACKUP_DATAFILE view. A backup datafile is uniquely identified by BDF_KEY.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database. Use this column to form a join with almost any other catalog view.
DBINC_KEY	NUMBER	The primary key for the incarnation of the target database. Use this column to form a join with RC_DATABASE_INCARNATION.
DB_NAME	VARCHAR2(8)	The DB_NAME of the database incarnation to which this record belongs.
BDF_KEY	NUMBER	The primary key of the datafile backup in the recovery catalog. If you issue the LIST command while connected to the recovery catalog, this value appears in the KEY column of the output.
RECID	NUMBER	The backup datafile RECID from V\$BACKUP_DATAFILE. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
STAMP	NUMBER	The backup datafile stamp from V\$BACKUP_DATAFILE. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
BS_KEY	NUMBER	The primary key of the backup set to which this record belongs in the recovery catalog. Use this column to form a join with RC_BACKUP_SET.
SET_STAMP	NUMBER	The SET_STAMP value from V\$BACKUP_SET. SET_STAMP and SET_COUNT form a concatenated key that uniquely identifies the backup set to which this record belongs in the target database control file.
SET_COUNT	NUMBER	The SET_COUNT value from V\$BACKUP_SET. SET_STAMP and SET_COUNT form a concatenated key that uniquely identifies the backup set to which this record belongs in the target database control file.
BS_RECID	NUMBER	The RECID from V\$BACKUP_SET.
BS_STAMP	NUMBER	The STAMP from V\$BACKUP_SET.
BACKUP_TYPE	VARCHAR2(1)	The type of the backup: D (full or level 0 incremental) or I (incremental level 1 or higher).

Column	Datatype	Description
INCREMENTAL_LEVEL	NUMBER	The level of the incremental backup: NULL or 0 - 4.
COMPLETION_TIME	DATE	The completion time of the backup.
FILE#	NUMBER	The absolute file number of the datafile.
CREATION_CHANGE#	NUMBER	The creation SCN of the datafile.
RESETLOGS_CHANGE#	NUMBER	The SCN of the most recent RESETLOGS in the datafile header.
RESETLOGS_TIME	DATE	The time stamp of the most recent RESETLOGS in the datafile header.
INCREMENTAL_CHANGE#	NUMBER	The SCN that determines whether a block will be included in the incremental backup. A block is only included if the SCN in the block header is greater than or equal to INCREMENTAL_CHANGE#. The range of redo covered by the incremental backup begins with INCREMENTAL_CHANGE# and ends with CHECKPOINT_CHANGE#.
CHECKPOINT_CHANGE#	NUMBER	The checkpoint SCN of this datafile in this backup set.
CHECKPOINT_TIME	DATE	The time associated with CHECKPOINT_CHANGE#.
ABSOLUTE_FUZZY_CHANGE#	NUMBER	The absolute fuzzy SCN.
DATAFILE_BLOCKS	NUMBER	The number of data blocks in the datafile.
BLOCKS	NUMBER	The number of data blocks written to the backup. This value is often less than DATAFILE_BLOCKS because for full backups, blocks that have never been used are not included in the backup, and for incremental backups, blocks that have not changed are not included in the backup. This value is never greater than DATAFILE_BLOCKS.
BLOCK_SIZE	NUMBER	The size of the data blocks in bytes.
STATUS	VARCHAR2(1)	The status of the backup set: A (all pieces available), D (all pieces deleted), O (some pieces are available but others are not, so the backup set is unusable).
BS_LEVEL	NUMBER	The incremental level (NULL or 0-4) specified when this backup was created. This value can be different from the INCREMENTAL_LEVEL column because if you run, for example, a level 2 incremental backup, but no previous level 0 backup exists for some files, a level 0 backup is automatically taken for these files. In this case, BS_LEVEL is 2 and INCREMENTAL_LEVEL is 0.
PIECES	NUMBER	The number of backup pieces in the backup set that contains this backup datafile.

RC_BACKUP_PIECE

This view lists information about backup pieces. This view corresponds to the V\$BACKUP_PIECE view. Each backup set contains one or more backup pieces.

Multiple copies of the same backup piece can exist, but each copy has its own record in the control file and its own row in the view.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database. Use this column to form a join with almost any other catalog view.
DB_ID	NUMBER	The database identifier.
BP_KEY	NUMBER	The primary key for the backup piece in the recovery catalog. If you issue the LIST command while connected to the recovery catalog, this value appears in the KEY column of the output.
RECID	NUMBER	The backup piece RECID from V\$BACKUP_PIECE. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
STAMP	NUMBER	The backup piece stamp propagated from V\$BACKUP_PIECE. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
BS_KEY	NUMBER	The primary key of the backup set to which this record belongs in the recovery catalog. Use this column to form a join with RC_BACKUP_SET.
SET_STAMP	NUMBER	The SET_STAMP value from V\$BACKUP_SET. SET_STAMP and SET_COUNT form a concatenated key that uniquely identifies the backup set to which this record belongs in the target database control file.
SET_COUNT	NUMBER	The SET_COUNT value from V\$BACKUP_SET. SET_STAMP and SET_COUNT form a concatenated key that uniquely identifies the backup set to which this record belongs in the target database control file.
BACKUP_TYPE	VARCHAR2(1)	The type of the backup: D (full or level 0 incremental), I (incremental level 1 or higher), L (archived redo log).
INCREMENTAL_LEVEL	NUMBER	The level of the incremental backup: NULL or 0 - 4.
PIECE#	NUMBER	The number of the backup piece. The first piece has the value of 1.
COPY#	NUMBER	The copy number of the backup piece.
DEVICE_TYPE	VARCHAR2(255)	The type of backup device, for example, DISK.
HANDLE	VARCHAR2(1024)	The filename of the backup piece.
COMMENTS	VARCHAR2(255)	Comments about the backup piece.
MEDIA	VARCHAR2(80)	A comment that contains further information about the media manager that created this backup.
MEDIA_POOL	NUMBER	The number of the media pool in which the backup is stored.

Column	Datatype	Description
CONCUR	VARCHAR2 (3)	Specifies whether backup media supports concurrent access: YES or NO.
TAG	VARCHAR2 (32)	The tag for the backup piece. Refer to description in BACKUP for default format for tag names.
START_TIME	DATE	The time when RMAN started to write the backup piece.
COMPLETION_TIME	DATE	The time when the backup piece was completed.
ELAPSED_SECONDS	NUMBER	The duration of the creation of the backup piece.
STATUS	VARCHAR2 (1)	The status of the backup piece: A (available), U (unavailable), D (deleted), or X (expired). Note that status D will not appear in Oracle9i unless an older recovery catalog is upgraded.

RC_BACKUP_REDOLOG

This view lists information about archived redo logs in backup sets. It corresponds to the V\$BACKUP_REDOLOG view.

You cannot back up online logs directly: you must first archive them to disk and then back them up. An archived log backup set contains one or more archived logs.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database. Use this column to form a join with almost any other catalog view.
DBINC_KEY	NUMBER	The primary key for the incarnation of the target database. Use this column to form a join with RC_DATABASE_INCARNATION.
DB_NAME	VARCHAR2 (8)	The DB_NAME of the database incarnation to which this record belongs.
BRL_KEY	NUMBER	The primary key of the archived redo log in the recovery catalog. If you issue the LIST command while connected to the recovery catalog, this value appears in the KEY column of the output.
RECID	NUMBER	The record identifier propagated from V\$BACKUP_REDOLOG. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
STAMP	NUMBER	The stamp from V\$BACKUP_REDOLOG. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
BS_KEY	NUMBER	The primary key of the backup set to which this record belongs in the recovery catalog. Use this column to form a join with RC_BACKUP_SET.
SET_STAMP	NUMBER	The SET_STAMP value from V\$BACKUP_SET. SET_STAMP and SET_COUNT form a concatenated key that uniquely identifies the backup set to which this record belongs in the target database control file.

RC_BACKUP_SET

Column	Datatype	Description
SET_COUNT	NUMBER	The SET_COUNT value from V\$BACKUP_SET. SET_STAMP and SET_COUNT form a concatenated key that uniquely identifies the backup set to which this record belongs in the target database control file.
BACKUP_TYPE	VARCHAR2(1)	The type of the backup: L (archived redo log).
COMPLETION_TIME	DATE	The time when the backup completed.
THREAD#	NUMBER	The thread number of the redo log.
SEQUENCE#	NUMBER	The log sequence number.
RESETLOGS_CHANGE#	NUMBER	The SCN of the most recent RESETLOGS when the record was created.
RESETLOGS_TIME	DATE	The time stamp of the most recent RESETLOGS when the record was created.
FIRST_CHANGE#	NUMBER	The SCN generated when Oracle switched into the redo log.
FIRST_TIME	DATE	The time when Oracle switched into the redo log.
NEXT_CHANGE#	NUMBER	The first SCN of the next redo log in the thread.
NEXT_TIME	DATE	The first time stamp of the next redo log in the thread.
BLOCKS	NUMBER	The number of operating system blocks written to the backup.
BLOCK_SIZE	NUMBER	The number of bytes in each block of this redo log.
STATUS	VARCHAR2(1)	The status of the backup set: A (all pieces available), D (all pieces deleted), O (some pieces are available but others are not, so the backup set is unusable).
BS_RECID	NUMBER	The RECID value from V\$BACKUP_SET.
BS_STAMP	NUMBER	The STAMP value from V\$BACKUP_SET. Note that BS_STAMP is different from SET_STAMP. BS_STAMP is the stamp of the backup set record when created in the control file, whereas SET_STAMP joins with SET_COUNT to make a unique identifier.
PIECES	NUMBER	The number of pieces in the backup set.

RC_BACKUP_SET

This view lists information about backup sets for all incarnations of the database. It corresponds to the V\$BACKUP_SET view. A backup set record is inserted after the backup has successfully completed.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database. Use this column to form a join with almost any other catalog view.
DB_ID	NUMBER	The unique database identifier.

Column	Datatype	Description
BS_KEY	NUMBER	The primary key of the backup set in the recovery catalog. If you issue the LIST command while connected to the recovery catalog, this value appears in the KEY column of the output.
RECID	NUMBER	The backup set RECID from V\$BACKUP_SET. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file. Use either RECID and STAMP or SET_STAMP and SET_COUNT to access V\$BACKUP_SET.
STAMP	NUMBER	The backup set STAMP from V\$BACKUP_SET. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file. Use either RECID and STAMP or SET_STAMP and SET_COUNT to access V\$BACKUP_SET.
SET_STAMP	NUMBER	The SET_STAMP value from V\$BACKUP_SET. SET_STAMP and SET_COUNT form a concatenated key that uniquely identifies this record in the target database control file. Use either RECID and STAMP or SET_STAMP and SET_COUNT to access V\$BACKUP_SET.
SET_COUNT	NUMBER	The SET_COUNT value from V\$BACKUP_SET. SET_STAMP and SET_COUNT form a concatenated key that uniquely identifies this record in the target database control file. Use either RECID and STAMP or SET_STAMP and SET_COUNT to access V\$BACKUP_SET.
BACKUP_TYPE	VARCHAR2 (1)	The type of the backup: D (full backup or level 0 incremental), I (incremental of level 1 or higher), L (archived redo log).
INCREMENTAL_LEVEL	NUMBER	The level of the incremental backup: NULL or 0 - 4.
PIECES	NUMBER	The number of backup pieces in the backup set.
START_TIME	DATE	The time when the backup began.
COMPLETION_TIME	DATE	The time when the backup completed.
ELAPSED_SECONDS	NUMBER	The duration of the backup in seconds.
STATUS	VARCHAR2 (1)	The status of the backup set: A (all backup pieces available), D (all backup pieces deleted), O (some backup pieces are available but others are not, so the backup set is unusable).
CONTROLFILE_INCLUDED	VARCHAR2 (7)	Possible values are NONE (backup set does not include a backup control file), BACKUP (backup set includes a normal backup control file), and STANDBY (backup set includes a standby control file).
INPUT_FILE_SCAN_ONLY	VARCHAR2 (3)	This backup set record was created by the BACKUP VALIDATE command. No real backup set exists. This record is only a placeholder used to keep track of which datafiles were scanned and which corrupt blocks (if any) were found in those files.
KEEP	VARCHAR2 (3)	Indicates whether this backup set has a retention policy different from the value for CONFIGURE RETENTION POLICY. Possible values are YES and NO.

RC_BACKUP_SPFILE

Column	Datatype	Description
KEEP_OPTIONS	VARCHAR2(10)	The KEEP options specified for this backup set. Options can be LOGS (RMAN keeps the logs needed to recover this backup), NOLOGS (RMAN does not keep the logs needed to recover this backup), or NULL (the backup has no KEEP options and will be made obsolete based on the retention policy).
KEEP_UNTIL	DATE	If the KEEP UNTIL TIME clause of the BACKUP command was specified, then this column shows the date after which this backup becomes obsolete. If the column is NULL and KEEP_OPTIONS is not NULL, the backup never becomes obsolete.

RC_BACKUP_SPFILE

This view lists information about server parameter files in backup sets.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database. Use this column to form a join with almost any other catalog view.
BSF_KEY	NUMBER	The primary key of the server parameter file in the recovery catalog. If you issue the LIST command while connected to the recovery catalog, this value appears in the KEY column of the output.
RECID	NUMBER	The record identifier propagated from V\$BACKUP_SPFILE. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
STAMP	NUMBER	The stamp from V\$BACKUP_SPFILE. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
BS_KEY	NUMBER	The primary key of the backup set to which this record belongs in the recovery catalog. Use this column to form a join with RC_BACKUP_SET.
SET_STAMP	NUMBER	The SET_STAMP value from V\$BACKUP_SET. SET_STAMP and SET_COUNT form a concatenated key that uniquely identifies the backup set to which this record belongs in the target database control file.
SET_COUNT	NUMBER	The SET_COUNT value from V\$BACKUP_SET. SET_STAMP and SET_COUNT form a concatenated key that uniquely identifies the backup set to which this record belongs in the target database control file.
MODIFICATION_TIME	DATE	The time when the server parameter file was last modified.
STATUS	VARCHAR2(1)	The status of the backup set: A (all backup pieces available), D (all backup pieces deleted), O (some backup pieces are available but others are not, so the backup set is unusable).
BS_RECID	NUMBER	The RECID value from V\$BACKUP_SET.

Column	Datatype	Description
BS_STAMP	NUMBER	The STAMP value from V\$BACKUP_SET. Note that BS_STAMP is different from SET_STAMP. BS_STAMP is the stamp of the backup set record when created in the control file, whereas SET_STAMP joins with SET_COUNT to make a unique identifier.
COMPLETION_TIME	DATE	The time when the backup set completed.
BYTES	NUMBER	The size of the backup set in bytes.

RC_CHECKPOINT

This view is deprecated. See [RC_RESYNC](#) on page 3-28 instead.

RC_CONTROLFILE_COPY

This view lists information about control file copies on disk. A datafile copy record with a file number of 0 represents the control file copy in V\$DATAFILE_COPY.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database. Use this column to form a join with almost any other catalog view.
DBINC_KEY	NUMBER	The primary key for the incarnation of the target database. Use this column to form a join with RC_DATABASE_INCARNATION.
DB_NAME	VARCHAR2 (8)	The DB_NAME of the database incarnation to which this record belongs.
CCF_KEY	NUMBER	The primary key of the control file copy in the recovery catalog. If you issue the LIST command while connected to the recovery catalog, this value appears in the KEY column of the output.
RECID	NUMBER	The record identifier from V\$DATAFILE_COPY. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
STAMP	NUMBER	The stamp from V\$DATAFILE_COPY. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
NAME	VARCHAR2 (1024)	The control file copy filename.
TAG	VARCHAR2 (32)	The tag of the control file copy. NULL if no tag used.
RESETLOGS_CHANGE#	NUMBER	The SCN of the most recent RESETLOGS when the record was created.
RESETLOGS_TIME	DATE	The time stamp of the most recent RESETLOGS when the record was created.
CHECKPOINT_CHANGE#	NUMBER	The control file checkpoint SCN.
CHECKPOINT_TIME	DATE	The control file checkpoint time.

RC_COPY_CORRUPTION

Column	Datatype	Description
CREATION_TIME	DATE	The control file creation time.
BLOCK_SIZE	NUMBER	The block size in bytes.
MIN_OFFR_RECID	NUMBER	Internal use only.
OLDEST_OFFLINE_RANGE	NUMBER	Internal use only.
COMPLETION_TIME	DATE	The time when the copy was generated.
STATUS	VARCHAR2(1)	The status of the copy: A (available), U (unavailable), X (expired), or D (deleted).
KEEP	VARCHAR2(3)	Indicates whether this copy has a retention policy different from the value for CONFIGURE RETENTION POLICY. Possible values are YES and NO.
KEEP_OPTIONS	VARCHAR2(10)	The KEEP options specified for this control file copy. Options can be LOGS (RMAN keeps the logs needed to recover this backup), NOLOGS (RMAN does not keep the logs needed to recover this backup), or NULL (the backup has no KEEP options and will be made obsolete based on the retention policy).
KEEP_UNTIL	DATE	If the KEEP UNTIL TIME clause of the COPY command was specified, then this column shows the date after which this file becomes obsolete. If the column is NULL and KEEP_OPTIONS is not NULL, the file never becomes obsolete.
CONTROLFILE_TYPE	VARCHAR2(1)	The type of control file copy: B (normal copy) or S (standby copy).

RC_COPY_CORRUPTION

This view lists corrupt block ranges in datafile copies. It corresponds to the V\$COPY_CORRUPTION view.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database. Use this column to form a join with almost any other catalog view.
DBINC_KEY	NUMBER	The primary key for the incarnation of the target database. Use this column to form a join with RC_DATABASE_INCARNATION.
DB_NAME	VARCHAR2(8)	The DB_NAME of the database incarnation to which this record belongs.
RECID	NUMBER	The record identifier from V\$COPY_CORRUPTION. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
STAMP	NUMBER	The stamp from V\$COPY_CORRUPTION. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.

Column	Datatype	Description
CDF_KEY	NUMBER	The primary key of the datafile copy in the recovery catalog. If you issue the LIST command while connected to the recovery catalog, this value appears in the KEY column of the output. Use this column to form a join with RC_DATAFILE_COPY.
COPY_RECID	NUMBER	The RECID from RC_DATAFILE_COPY. This value is propagated from the control file.
COPY_STAMP	NUMBER	The STAMP from RC_DATAFILE_COPY. This value is propagated from the control file.
FILE#	NUMBER	The absolute file number of the datafile.
CREATION_CHANGE#	NUMBER	The creation SCN of this data file. Because file numbers can be reused, FILE# and CREATION_CHANGE# are both required to uniquely identify a specified file over the life of the database.
BLOCK#	NUMBER	The block number of the first corrupted block in the file.
BLOCKS	NUMBER	The number of corrupted blocks found beginning with BLOCK#.
CORRUPTION_CHANGE#	NUMBER	For media corrupt blocks, this value is zero. For logically corrupt blocks, this value is the lowest SCN in the blocks in this corrupt range.
MARKED_CORRUPT	VARCHAR2(3)	YES if this corruption was not previously detected by the database server or NO if it was already known by the database server.
CORRUPTION_TYPE	VARCHAR2(9)	Same as RC_DATABASE_BLOCK_CORRUPTION.CORRUPTION_TYPE.

RC_DATABASE

This view gives information about the databases registered in the recovery catalog. It corresponds to the V\$DATABASE view.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the database. Use this column to form a join with almost any other catalog view.
DBINC_KEY	NUMBER	The primary key for the current incarnation. Use this column to form a join with RC_DATABASE_INCARNATION.
DBID	NUMBER	Unique identifier for the database obtained from V\$DATABASE.
NAME	VARCHAR2(8)	The DB_NAME of the database for the current incarnation.
RESETLOGS_CHANGE#	NUMBER	The SCN of the most recent RESETLOGS operation when the record was created.
RESETLOGS_TIME	DATE	The time stamp of the most recent RESETLOGS operation when the record was created.

RC_DATABASE_BLOCK_CORRUPTION

This view gives information about database blocks that were corrupted after the last backup. It corresponds to the V\$DATABASE_BLOCK_CORRUPTION view.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the database. Use this column to form a join with almost any other catalog view.
DBINC_KEY	NUMBER	The primary key for the current incarnation. Use this column to form a join with RC_DATABASE_INCARNATION.
FILE#	NUMBER	The absolute file number of the datafile.
BLOCK#	NUMBER	The block number of the first corrupted block in this range of corrupted blocks.
BLOCKS	NUMBER	The number of corrupted blocks found beginning with BLOCK#.
CORRUPTION_CHANGE#	NUMBER	For media corrupt blocks, this value is zero. For logically corrupt blocks, this value is the lowest SCN in the blocks in this corrupt range.
CORRUPTION_TYPE	VARCHAR2(9)	The type of block corruption in the datafile. Possible values are: <ul style="list-style-type: none"> ■ ALL ZERO. The block header on disk contained only zeros. The block may be valid if it was never filled and if it is in an Oracle7 file. The buffer will be reformatted to the Oracle8 standard for an empty block. ■ FRACTURED. The block header looks reasonable, but the front and back of the block are different versions. ■ CHECKSUM. The optional check value shows that the block is not self-consistent. It is impossible to determine exactly why the check value fails, but it probably fails because sectors in the middle of the block are from different versions. ■ CORRUPT. The block is wrongly identified or is not a data block (for example, the data block address is missing) ■ LOGICAL. Specifies the range is for logically corrupt blocks. CORRUPTION_CHANGE# will have a nonzero value.

RC_DATABASE_INCARNATION

This view lists information about all database incarnations registered in the recovery catalog. Oracle creates a new incarnation whenever you open a database

with the `RESETLOGS` option. Records about the current and immediately previous incarnation are also contained in the `V$DATABASE` view.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the database. Use this column to form a join with almost any other catalog view.
DBID	NUMBER	Unique identifier for the database.
DBINC_KEY	NUMBER	The primary key for the incarnation.
NAME	VARCHAR2 (8)	The <code>DB_NAME</code> for the database at the time of the <code>RESETLOGS</code> . The value is <code>UNKNOWN</code> if you have done at least one <code>RESETLOGS</code> before registering the target database with <code>RMAN</code> , because <code>RMAN</code> does not know the <code>DB_NAME</code> prior to the <code>RESETLOGS</code> .
RESETLOGS_CHANGE#	NUMBER	The SCN of the <code>RESETLOGS</code> operation that created this incarnation.
RESETLOGS_TIME	DATE	The time stamp of the <code>RESETLOGS</code> operation that created this incarnation.
CURRENT_INCARNATION	VARCHAR2 (3)	YES if it is the current incarnation; NO if it is not.
PARENT_DBINC_KEY	NUMBER	The <code>DBINC_KEY</code> of the previous incarnation for this database. The value is <code>NULL</code> if it is the first incarnation recorded for the database.

RC_DATAFILE

This view lists information about all datafiles registered in the recovery catalog. It corresponds to the `V$DATAFILE` view. A datafile is shown as dropped if its tablespace was dropped.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database.
DBINC_KEY	NUMBER	The primary key for the incarnation of the target database. Use this column to form a join with <code>RC_DATABASE_INCARNATION</code> .
DB_NAME	VARCHAR2 (8)	The <code>DB_NAME</code> of the database incarnation to which this record belongs.
TS#	NUMBER	The number of the tablespace to which the datafile belongs. The <code>TS#</code> may exist multiple times in the same incarnation if the tablespace is dropped and re-created.
TABLESPACE_NAME	VARCHAR2 (30)	The tablespace name. The name may exist multiple times in the same incarnation if the tablespace is dropped and re-created.
FILE#	NUMBER	The absolute file number of the datafile. The same datafile number may exist multiple times in the same incarnation if the datafile is dropped and re-created.
CREATION_CHANGE#	NUMBER	The SCN at datafile creation.

RC_DATAFILE_COPY

Column	Datatype	Description
CREATION_TIME	DATE	The time of datafile creation.
DROP_CHANGE#	NUMBER	The SCN recorded when the datafile was dropped. If a new datafile with the same file number is discovered then the DROP_CHANGE# is set to CREATION_CHANGE# for the datafile; otherwise the value is set to RC_CHECKPOINT.CKP_SCN.
DROP_TIME	DATE	The time when the datafile was dropped. If a new datafile with the same file number is discovered then the DROP_TIME is set to CREATION_TIME for the datafile; otherwise the value is set to RC_CHECKPOINT.CKP_TIME.
BYTES	NUMBER	The size of the datafile in bytes.
BLOCKS	NUMBER	The size of the datafile in blocks.
BLOCK_SIZE	NUMBER	The size of the data blocks in bytes.
NAME	VARCHAR2(1024)	The datafile filename.
STOP_CHANGE#	NUMBER	For offline or read-only datafiles, the SCN value such that no changes in the redo stream at an equal or greater SCN apply to this file.
STOP_TIME	DATE	For offline normal or read-only datafiles, the time beyond which there are no changes in the redo stream that apply to this datafile.
READ_ONLY	NUMBER	1 if the file is read-only; otherwise 0.
RFILE#	NUMBER	The relative file number of this datafile within its tablespace.
INCLUDED_IN_DATABASE_BACKUP	VARCHAR2(3)	Indicates whether this tablespace is included in whole database backups: YES or NO. The NO value occurs only if CONFIGURE EXCLUDE was run on the tablespace that owns this datafile.
AUX_NAME	VARCHAR2(1024)	Indicates the auxiliary name for the datafile as set by CONFIGURE AUXNAME.

RC_DATAFILE_COPY

This view lists information about datafile copies on disk. It corresponds to the V\$DATAFILE_COPY view.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database. Use this column to form a join with almost any other catalog view.
DBINC_KEY	NUMBER	The primary key for the incarnation of the target database. Use this column to form a join with RC_DATABASE_INCARNATION.
DB_NAME	VARCHAR2(8)	The DB_NAME of the database incarnation to which this record belongs.

Column	Datatype	Description
CDF_KEY	NUMBER	The primary key of the datafile copy in the recovery catalog. If you issue the LIST command while connected to the recovery catalog, this value appears in the KEY column of the output.
RECID	NUMBER	The datafile copy record from V\$DATAFILE_COPY. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
STAMP	NUMBER	The datafile copy stamp from V\$DATAFILE_COPY. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
NAME	VARCHAR2(1024)	The filename of the datafile copy.
TAG	VARCHAR2(32)	The tag for the datafile copy.
FILE#	NUMBER	The absolute file number for the datafile.
CREATION_CHANGE#	NUMBER	The creation SCN of the datafile.
RESETLOGS_CHANGE#	NUMBER	The SCN of the most recent RESETLOGS when the datafile was created.
RESETLOGS_TIME	DATE	The time stamp of the most recent RESETLOGS in the datafile header.
INCREMENTAL_LEVEL	NUMBER	The incremental level of the copy: 0 or NULL.
CHECKPOINT_CHANGE#	NUMBER	The SCN of the most recent datafile checkpoint.
CHECKPOINT_TIME	DATE	The time of the most recent datafile checkpoint.
ABSOLUTE_FUZZY_CHANGE#	NUMBER	The highest SCN in any block of the file, if known. Recovery must proceed to at least this SCN for the file to become not fuzzy.
RECOVERY_FUZZY_CHANGE#	NUMBER	The SCN to which recovery must proceed for the file to become not fuzzy. If not NULL, this file must be recovered at least to the specified SCN before the database can be opened with this file.
RECOVERY_FUZZY_TIME	DATE	The time that is associated with the RECOVERY_FUZZY_CHANGE#.
ONLINE_FUZZY	VARCHAR2(3)	YES/NO. If set to YES, this copy was made after an instance failure or OFFLINE IMMEDIATE (or is a copy that was taken improperly while the database was open). Recovery will need to apply all redo up to the next crash recovery marker to make the file consistent.
BACKUP_FUZZY	VARCHAR2(3)	YES/NO. If set to YES, this is a copy taken using the BEGIN BACKUP/END BACKUP technique. To make this copy consistent, Recovery needs to apply all redo up to the marker that is placed in the redo stream when the ALTER TABLESPACE END BACKUP command is used.
BLOCKS	NUMBER	The number of blocks in the datafile copy (also the size of the datafile when the copy was made).
BLOCK_SIZE	NUMBER	The size of the blocks in bytes.
COMPLETION_TIME	DATE	The time when the copy completed.
STATUS	VARCHAR2(1)	The status of the copy: A (available), U (unavailable), X (expired), or D (deleted).

RC_LOG_HISTORY

Column	Datatype	Description
KEEP	VARCHAR2(3)	Indicates whether this copy has a retention policy different from the value for CONFIGURE RETENTION POLICY. Possible values are YES and NO.
KEEP_UNTIL	DATE	If the KEEP UNTIL TIME clause of the COPY command was specified, then this column shows the date after which this datafile copy becomes obsolete. If the column is NULL and KEEP OPTIONS is not NULL, the copy never becomes obsolete.
KEEP_OPTIONS	VARCHAR2(10)	The KEEP options specified for this datafile copy. Options can be LOGS (RMAN keeps the logs needed to recover this backup), NOLOGS (RMAN does not keep the logs needed to recover this backup), or NULL (the backup has no KEEP options and will be made obsolete based on the retention policy).
SCANNED	VARCHAR2(3)	Whether RMAN scanned the file (YES or NO). If YES, then this copy was created by a server process that examined every block in the file, for example, by the RMAN COPY or RESTORE command. If NO, then RMAN did not examine every block in the file, as when RMAN inspects a non-RMAN generated image copy or restores by proxy copy. Whenever RMAN creates or restores a datafile copy, it adds rows to the V\$DATABASE_BLOCK_CORRUPTION view and RC_DATABASE_BLOCK_CORRUPTION view if it discovers corrupt blocks in the file. If RMAN has scanned the entire file, then the absence of corruption records for this copy means that no corrupt blocks exist in the file. If RMAN did not scan the file, then the absence of corruption records means that corrupt blocks may or may not exist in the file.

RC_LOG_HISTORY

This view lists historical information about the online redo logs. RMAN adds a new row during a catalog resynchronization whenever Oracle has switched out of the online redo log. This catalog view corresponds to the V\$LOG_HISTORY view.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database. Use this column to form a join with almost any other catalog view.
DBINC_KEY	NUMBER	The primary key for the incarnation of the target database. Use this column to form a join with RC_DATABASE_INCARNATION.
DB_NAME	VARCHAR2(8)	The DB_NAME of the database incarnation to which this record belongs.
RECID	NUMBER	The redo log history RECID from V\$LOG_HISTORY. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
STAMP	NUMBER	The redo log history stamp from V\$LOG_HISTORY. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.

Column	Datatype	Description
THREAD#	NUMBER	The thread number of the online redo log.
SEQUENCE#	NUMBER	The log sequence number of the redo log.
FIRST_CHANGE#	NUMBER	The SCN generated when switching into the redo log.
FIRST_TIME	DATE	The time stamp when switching into the redo log.
NEXT_CHANGE#	NUMBER	The first SCN of the next redo log in the thread.
CLEARED	VARCHAR2 (3)	YES if the redo log was cleared with the ALTER DATABASE CLEAR LOGFILE statement; otherwise, NULL. This statement allows a log to be dropped without archiving it first.

RC_OFFLINE_RANGE

This view lists the offline ranges for datafiles. It corresponds to the V\$OFFLINE_RANGE view.

An offline range is created for a datafile when its tablespace is first altered to be offline normal or read-only, and then subsequently altered to be online or read/write. Note that no offline range is created if the datafile itself is altered to be offline or if the tablespace is altered to be offline immediate.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database. Use this column to form a join with almost any other catalog view.
DBINC_KEY	NUMBER	The primary key for the incarnation of the target database. Use this column to form a join with RC_DATABASE_INCARNATION.
DB_NAME	VARCHAR2 (8)	The DB_NAME of the database incarnation to which this record belongs.
RECID	NUMBER	The record identifier for the offline range from V\$OFFLINE_RANGE. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
STAMP	NUMBER	The stamp for the offline range from V\$OFFLINE_RANGE. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
FILE#	NUMBER	The absolute file number of the datafile.
CREATION_CHANGE#	NUMBER	The SCN at datafile creation.
OFFLINE_CHANGE#	NUMBER	The SCN taken when the datafile was taken offline.
ONLINE_CHANGE#	NUMBER	The online checkpoint SCN.
ONLINE_TIME	DATE	The online checkpoint time.
CF_CREATE_TIME	DATE	The time of control file creation.

RC_PROXY_CONTROLFILE

This view contains descriptions of control file backups that were taken using the proxy copy functionality. It corresponds to the `V$PROXY_DATAFILE` view.

In a proxy copy, the media manager takes over the operations of backing up and restoring data. Each row represents a backup of one control file.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database. Use this column to form a join with almost any other catalog view.
DBINC_KEY	NUMBER	The primary key for the incarnation of the target database. Use this column to form a join with <code>RC_DATABASE_INCARNATION</code> .
DB_NAME	VARCHAR2(8)	The <code>DB_NAME</code> of the database incarnation to which this record belongs.
XCF_KEY	NUMBER	The proxy copy primary key in the recovery catalog. If you issue the <code>LIST</code> command while connected to the recovery catalog, this value appears in the <code>KEY</code> column of the output.
RECID	NUMBER	The proxy copy record identifier from <code>V\$PROXY_DATAFILE</code> . <code>RECID</code> and <code>STAMP</code> form a concatenated primary key that uniquely identifies this record in the target database control file.
STAMP	NUMBER	The proxy copy stamp from <code>V\$PROXY_DATAFILE</code> . <code>RECID</code> and <code>STAMP</code> form a concatenated primary key that uniquely identifies this record in the target database control file.
TAG	VARCHAR2(32)	The tag for the proxy copy.
RESETLOGS_CHANGE#	NUMBER	The <code>RESETLOGS</code> SCN of the database incarnation to which this datafile belongs.
RESETLOGS_TIME	DATE	The <code>RESETLOGS</code> time stamp of the database incarnation to which this datafile belongs.
CHECKPOINT_CHANGE#	NUMBER	Datafile checkpoint SCN when this copy was made.
CHECKPOINT_TIME	DATE	Datafile checkpoint time when this copy was made.
CREATION_TIME	DATE	The control file creation time.
BLOCK_SIZE	NUMBER	The block size for the copy in bytes.
MIN_OFFR_RECID	NUMBER	Internal use only.
OLDEST_OFFLINE_RANGE	NUMBER	Internal use only.
DEVICE_TYPE	VARCHAR2(255)	The type of sequential media device.
HANDLE	VARCHAR2(1024)	The name or "handle" for the proxy copy. <code>RMAN</code> passes this value to the operating system-dependent layer that identifies the file.
COMMENTS	VARCHAR2(255)	Comments about the proxy copy.
MEDIA	VARCHAR2(80)	A comment that contains further information about the media manager that created this backup.

Column	Datatype	Description
MEDIA_POOL	NUMBER	The number of the media pool in which the proxy copy is stored.
START_TIME	DATE	The time when proxy copy was initiated.
COMPLETION_TIME	DATE	The time when the proxy copy was completed.
ELAPSED_SECONDS	NUMBER	The duration of the proxy copy.
STATUS	VARCHAR2 (1)	The status of the backup set: A (available), U (unavailable), X (expired), or D (deleted).
KEEP	VARCHAR2 (3)	Indicates whether this proxy copy has a retention policy different from the value for CONFIGURE RETENTION POLICY. Possible values are YES and NO.
KEEP_OPTIONS	VARCHAR2 (10)	The KEEP options specified for this control file backup. Options can be LOGS (RMAN keeps the logs needed to recover this backup), NOLOGS (RMAN does not keep the logs needed to recover this backup), or NULL (the backup has no KEEP options and will be made obsolete based on the retention policy).
KEEP_UNTIL	DATE	If the KEEP UNTIL TIME clause of the BACKUP command was specified, then this column shows the date after which this control file backup becomes obsolete. If the column is NULL and KEEP OPTIONS is not NULL, the backup never becomes obsolete.
CONTROLFILE_TYPE	VARCHAR2 (1)	The type of control file copy: B (normal copy) or S (standby copy).

RC_PROXY_DATAFILE

This view contains descriptions of datafile backups that were taken using the proxy copy functionality. It corresponds to the V\$PROXY_DATAFILE view.

In a proxy copy, the media manager takes over the operations of backing up and restoring data. Each row represents a backup of one database file.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database. Use this column to form a join with almost any other catalog view.
DBINC_KEY	NUMBER	The primary key for the incarnation of the target database. Use this column to form a join with RC_DATABASE_INCARNATION.
DB_NAME	VARCHAR2 (8)	The DB_NAME of the database incarnation to which this record belongs.
XDF_KEY	NUMBER	The proxy copy primary key in the recovery catalog. If you issue the LIST command while connected to the recovery catalog, this value appears in the KEY column of the output.
RECID	NUMBER	The proxy copy record identifier from V\$PROXY_DATAFILE. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.

Column	Datatype	Description
STAMP	NUMBER	The proxy copy stamp from V\$PROXY_DATAFILE. RECID and STAMP form a concatenated primary key that uniquely identifies this record in the target database control file.
TAG	VARCHAR2 (32)	The tag for the proxy copy.
FILE#	NUMBER	The absolute file number of the datafile that is proxy copied.
CREATION_CHANGE#	NUMBER	The datafile creation SCN.
RESETLOGS_CHANGE#	NUMBER	The SCN of the most recent RESETLOGS in the datafile header.
RESETLOGS_TIME	DATE	The time stamp of the most recent RESETLOGS in the datafile header.
INCREMENTAL_LEVEL	NUMBER	0 if this copy is part of an incremental backup strategy, otherwise NULL.
CHECKPOINT_CHANGE#	NUMBER	Checkpoint SCN when the copy was made.
CHECKPOINT_TIME	DATE	Checkpoint time when the copy was made.
ABSOLUTE_FUZZY_CHANGE#	NUMBER	The highest SCN in any block of the file, if known. Recovery must proceed to at least this SCN for the file to become not fuzzy.
RECOVERY_FUZZY_CHANGE#	NUMBER	The SCN to which recovery must proceed for the file to become not fuzzy. If not NULL, this file must be recovered at least to the specified SCN before the database can be opened with this file.
RECOVERY_FUZZY_TIME	DATE	The time that is associated with the RECOVERY_FUZZY_CHANGE#.
ONLINE_FUZZY	VARCHAR2 (3)	YES/NO. If set to YES, this copy was made after an instance failure or OFFLINE IMMEDIATE (or is a copy of a copy which was taken improperly while the database was open). Recovery will need to apply all redo up to the next crash recovery marker to make the file consistent.
BACKUP_FUZZY	VARCHAR2 (3)	YES/NO. If set to YES, this is a copy taken using the BEGIN BACKUP/END BACKUP backup method. To make this copy consistent, recovery must apply all redo up to the marker that is placed in the redo stream when the ALTER TABLESPACE END BACKUP statement is issued.
BLOCKS	NUMBER	Size of the datafile copy in blocks (also the size of the datafile when the copy was made).
BLOCK_SIZE	NUMBER	The block size for the copy in bytes.
DEVICE_TYPE	VARCHAR2 (255)	The type of sequential media device.
HANDLE	VARCHAR2 (1024)	The name or "handle" for the proxy copy. RMAN passes this value to the operating system-dependent layer that identifies the file.
COMMENTS	VARCHAR2 (255)	Comments about the proxy copy.
MEDIA	VARCHAR2 (80)	A comment that contains further information about the media manager that created this backup.
MEDIA_POOL	NUMBER	The number of the media pool in which the proxy copy is stored.
START_TIME	DATE	The time when proxy copy was initiated.
COMPLETION_TIME	DATE	The time when the proxy copy was completed.

Column	Datatype	Description
ELAPSED_SECONDS	NUMBER	The duration of the proxy copy.
STATUS	VARCHAR2 (1)	The status of the backup set: A (available), U (unavailable), X (expired), or D (deleted).
KEEP	VARCHAR2 (3)	Indicates whether this proxy copy has a retention policy different from the value for CONFIGURE RETENTION POLICY (YES or NO).
KEEP_UNTIL	DATE	If the KEEP UNTIL TIME clause of the BACKUP command was specified, then this column shows the date after which this datafile backup becomes obsolete. If the column is NULL and KEEP OPTIONS is not NULL, the backup never becomes obsolete.
KEEP_OPTIONS	VARCHAR2 (10)	The KEEP options specified for this backup. Options can be LOGS (RMAN keeps logs needed to recover this backup), NOLOGS (RMAN does not keep logs needed to recover this backup), or NULL (the backup has no KEEP options and will be obsolete based on retention policy).

RC_REDO_LOG

This view lists information about the online redo logs for all incarnations of the database since the last catalog resynchronization. This view corresponds to a combination of the V\$LOG and V\$LOGFILE views.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database. Use this column to form a join with almost any other catalog view.
DBINC_KEY	NUMBER	The primary key for the incarnation of the target database. Use this column to form a join with RC_DATABASE_INCARNATION.
DB_NAME	VARCHAR2 (8)	The DB_NAME of the database incarnation to which this record belongs.
THREAD#	NUMBER	The number of the redo thread.
GROUP#	NUMBER	The number of the online redo log group.
NAME	VARCHAR2 (1024)	The name of the online redo log file.

RC_REDO_THREAD

This view lists data about all redo threads for all incarnations of the database since the last catalog resynchronization. This view corresponds to V\$THREAD.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database. Use this column to form a join with almost any other catalog view.

Column	Datatype	Description
DBINC_KEY	NUMBER	The primary key for the incarnation of the target database. Use this column to form a join with RC_DATABASE_INCARNATION.
DB_NAME	VARCHAR2(8)	The DB_NAME of the database incarnation to which this record belongs.
THREAD#	NUMBER	The redo thread number for the database incarnation.
STATUS	VARCHAR2(1)	The status of the redo thread: D (disabled), E (enabled), or O (open).
SEQUENCE#	NUMBER	The last allocated log sequence number.
ENABLE_CHANGE#	NUMBER	The SCN at which this thread was enabled.
ENABLE_TIME	DATE	The time at which this thread was enabled.
DISABLE_CHANGE#	NUMBER	The most recent SCN at which this thread was disabled. If the thread is still disabled, then no redo at or beyond this SCN exists for this thread. If the thread is now enabled, then no redo exists between the DISABLE_CHANGE# and the ENABLE_CHANGE# for this thread.
DISABLE_TIME	DATE	The most recent time at which this thread was disabled.

RC_RESYNC

This view lists information about recovery catalog resynchronizations. Every full resynchronization takes a snapshot of the target database control file and resynchronizes the recovery catalog from the snapshot.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database. Use this column to form a join with almost any other catalog view.
DBINC_KEY	NUMBER	The primary key for the incarnation of the target database. Use this column to form a join with RC_DATABASE_INCARNATION.
DB_NAME	VARCHAR2(8)	The DB_NAME of the database incarnation to which this record belongs.
RESYNC_KEY	NUMBER	The primary key for the resynchronization.
CONTROLFILE_CHANGE#	NUMBER	The control file checkpoint SCN from which the catalog was resynchronized.
CONTROLFILE_TIME	DATE	The control file checkpoint time stamp from which the catalog was resynchronized.
CONTROLFILE_SEQUENCE#	NUMBER	The control file sequence number.
CONTROLFILE_VERSION	DATE	The creation time for the version of the control file from which the catalog was resynchronized.
RESYNC_TYPE	VARCHAR2(7)	The type of resynchronization: FULL or PARTIAL.
DB_STATUS	VARCHAR2(7)	The status of the target database: OPEN or MOUNTED.

Column	Datatype	Description
RESYNC_TIME	DATE	The time of the resynchronization.

RC_RMAN_CONFIGURATION

This view lists information about RMAN persistent configuration settings. It corresponds to the V\$RMAN_CONFIGURATION view.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database corresponding to this configuration. Use this column to form a join with almost any other catalog view.
CONF#	NUMBER	A unique key identifying this configuration record within the target database that owns it.
NAME	VARCHAR2(65)	The type of configuration. All options of CONFIGURE command are valid types except: CONFIGURE EXCLUDE, (described in RC_TABLESPACE), CONFIGURE AUXNAME (described in RC_DATAFILE), and CONFIGURE SNAPSHOT CONTROLFILE (stored only in control file).
VALUE	VARCHAR2(1025)	The CONFIGURE command setting. For example: RETENTION POLICY TO RECOVERY WINDOW OF 1 DAYS.

RC_STORED_SCRIPT

This view lists information about scripts stored in the recovery catalog. The view contains one row for each stored script.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the database that owns this script. Use this column to form a join with almost any other catalog view.
DB_NAME	VARCHAR2(8)	The DB_NAME of the database incarnation to which this record belongs.
SCRIPT_NAME	VARCHAR2(100)	The name of the script.

RC_STORED_SCRIPT_LINE

This view lists information about lines of the scripts stored in the recovery catalog. The view contains one row for each line of each stored script.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the database that owns this script. Use this column to form a join with almost any other catalog view.
SCRIPT_NAME	VARCHAR2(100)	The name of the stored script.
LINE	NUMBER	The number of the line in the script. The line of a script is uniquely identified by SCRIPT_NAME and LINE.
TEXT	VARCHAR2(1024)	The text of the line of the script.

RC_TABLESPACE

This view lists all tablespaces registered in the recovery catalog, all dropped tablespaces, and tablespaces that belong to old incarnations. It corresponds to the V\$TABLESPACE view. The current value is shown for tablespace attributes.

Column	Datatype	Description
DB_KEY	NUMBER	The primary key for the target database. Use this column to form a join with almost any other catalog view.
DBINC_KEY	NUMBER	The primary key for the incarnation of the target database. Use this column to form a join with RC_DATABASE_INCARNATION.
DB_NAME	VARCHAR2(8)	The DB_NAME of the database incarnation to which this record belongs.
TS#	NUMBER	The tablespace ID in the target database. The TS# may exist multiple times in the same incarnation if the tablespace is dropped and re-created.
NAME	VARCHAR2(30)	The tablespace name. The name may exist multiple times in the same incarnation if the tablespace is dropped and re-created.
CREATION_CHANGE#	NUMBER	The creation SCN (from the first datafile).
CREATION_TIME	DATE	The creation time of the tablespace. NULL for offline tablespaces after creating the control file.
DROP_CHANGE#	NUMBER	The SCN recorded when the tablespace was dropped. If a new tablespace with the same TS# is discovered then the DROP_CHANGE# is set to CREATION_CHANGE# for the tablespace; otherwise, the value is set to RC_CHECKPOINT.CKP_SCN.
DROP_TIME	DATE	The date when the tablespace was dropped.

Column	Datatype	Description
INCLUDED_IN_DATABASE_ BACKUP	VARCHAR2(3)	Indicates whether this tablespace is included in whole database backups: YES or NO. The YES value occurs only if CONFIGURE EXCLUDE was run on the tablespace that owns this datafile.

Deprecated RMAN Commands

This appendix describes Recovery Manager syntax that is deprecated and describes preferred syntax if any exists.

Deprecated RMAN syntax continues to be supported in subsequent releases for backward compatibility. For example, the `SET AUXNAME` command replaced the `SET CLONENAME` command in Oracle8*i*, and the `CONFIGURE AUXNAME` command replaced the `SET AUXNAME` command in Oracle9*i*, but you can continue to run both `SET CLONENAME` and `SET AUXNAME` in all subsequent RMAN releases.

Table A–1 *Deprecated RMAN Syntax*

Deprecated in Release	Deprecated Syntax	Preferred Current Syntax
9.2	REPLICATE	RESTORE CONTROLFILE FROM . . .
9.2	SET AUTOLOCATE	Now enabled by default.
9.0.1	ALLOCATE CHANNEL FOR DELETE	n/a
9.0.1	ALLOCATE CHANNEL . . . TYPE	CONFIGURE CHANNEL . . . DEVICE TYPE
9.0.1	ALLOCATE CHANNEL . . . KBYTES	CONFIGURE CHANNEL . . . MAXPIECESIZE
9.0.1	ALLOCATE CHANNEL . . . READRATE	CONFIGURE CHANNEL . . . RATE
9.0.1	. . . ARCHIVELOG . . . LOGSEQ	. . . ARCHIVELOG . . . SEQUENCE
9.0.1	BACKUP . . . SETSIZE	BACKUP . . . MAXSETSIZE
9.0.1	CHANGE . . . CROSSCHECK	CROSSCHECK
9.0.1	CHANGE . . . DELETE	DELETE
9.0.1	REPORT . . . AT LOGSEQ	REPORT . . . AT SEQUENCE
9.0.1	SET AUXNAME	CONFIGURE AUXNAME

Table A-1 Deprecated RMAN Syntax (Cont.)

Deprecated in Release	Deprecated Syntax	Preferred Current Syntax
9.0.1	SET DUPLEX	SET BACKUP COPIES CONFIGURE BACKUP COPIES
9.0.1	SET LIMIT CHANNEL . . .	ALLOCATE CHANNEL . . . CONFIGURE CHANNEL . . .
9.0.1	SET SNAPSHOT	CONFIGURE SNAPSHOT
9.0.1	UNTIL LOGSEQ (see "untilClause")	UNTIL SEQUENCE (see "untilClause")
8.1.7	CONFIGURE COMPATIBLE	n/a
8.1.5	ALLOCATE CHANNEL CLONE	CONFIGURE AUXILIARY CHANNEL
8.1.5	CHANGE . . . VALIDATE	CROSSCHECK
8.1.5	CLONE (see "cmdLine")	AUXILIARY (see "cmdLine")
8.1.5	CONNECT CLONE	CONNECT AUXILIARY
8.1.5	MSGLOG (see "cmdLine")	LOG (see "cmdLine")
8.1.5	RCVCAT (see "cmdLine")	CATALOG (see "cmdLine")

RMAN Compatibility

This section contains these topics:

- [About RMAN Compatibility](#)
- [RMAN Compatibility Matrix](#)
- [RMAN Compatibility: Scenario](#)

About RMAN Compatibility

The RMAN environment can contain the following components:

- RMAN executable
- Recovery catalog database
- Recovery catalog schema in the recovery catalog database
- Target database
- Auxiliary database (that is, a duplicate or standby database)

Each component has a release number. For example, you can use a release 9.0.1 RMAN executable with:

- A release 9.0.1 target database
- A release 9.0.1 duplicate database
- A release 8.1.5 recovery catalog database whose catalog tables were created with RMAN release 9.0.1

RMAN Compatibility Matrix

In general, the rules of RMAN compatibility are as follows:

- You can create the recovery catalog schema in any Oracle database release 8.0 and higher (refer to "[Note 1: 8.1 or Later Catalog Schemas in 8.0 Catalog Databases](#)" on page B-3).
- The recovery catalog schema version must be greater than or equal to the RMAN executable version (refer to "[Note 2: 8.1 or Later Catalog Schemas and 8.0.4 or 8.0.5 RMAN Executables](#)" on page B-3).
- The versions of the RMAN executable and the target database should be the same (refer to [Table B-1](#) for other legal combinations).

Table B-1 shows version requirements for RMAN components.

Table B-1 RMAN Compatibility Table

Target/Auxiliary Database	RMAN Executable	Catalog Database	Catalog Schema
8.0.3	8.0.3	>=8.x	8.0.3
8.0.4	8.0.4	>=8.x	>= 8.0.4, see "Note 2: 8.1 or Later Catalog Schemas and 8.0.4 or 8.0.5 RMAN Executables"
8.0.5	8.0.5	>=8.x	>= 8.0.5, see "Note 2: 8.1 or Later Catalog Schemas and 8.0.4 or 8.0.5 RMAN Executables"
8.0.6	8.0.6	>=8.x	8.0.6
8.0.6	8.0.6	>=8.1.x	>= 8.1.x
8.1.5	8.1.5	>=8.1.x	>= 8.1.5
8.1.6	8.0.6.1	>=8.x	8.0.6
8.1.6	8.0.6.1	>=8.1.x	>= 8.1.x
8.1.6	8.1.5	>=8.1.x	>= RMAN executable
8.1.6	8.1.6	>=8.1.x	>= RMAN executable
8.1.7	8.0.6.1	>=8.x	8.0.6
8.1.7	8.0.6.1	>=8.1.x	>=8.1.x
8.1.7	8.1.x	>=8.1.x	>= RMAN executable
9.0.1	9.0.1	>=8.1.x	>= RMAN executable
9.2.0	>=9.0.3	>=8.1.x	>= RMAN executable

Note 1: 8.1 or Later Catalog Schemas in 8.0 Catalog Databases

RMAN cannot create release 8.1 or later catalog schemas in 8.0 catalog databases.

Note 2: 8.1 or Later Catalog Schemas and 8.0.4 or 8.0.5 RMAN Executables

You cannot use an 8.0.4 or 8.0.5 RMAN executable with a release 8.1 or later recovery catalog schema.

Note 3: 8.1.6 Catalog Schema and Pre-8.1.6 RMAN Executable

Using a pre-8.1.6 release of the RMAN executable with recovery catalog schema of release 8.1.6 (newly created by 8.1.6 RMAN executable with the `CREATE CATALOG` command) requires the following update at the catalog database:

```
SQL> UPDATE CONFIG SET VALUE='080004' WHERE NAME='COMPATIBLE';
```

RMAN Compatibility: Scenario

Assume that you maintain a production databases of the following releases:

- 8.0.3
- 8.0.4
- 8.0.5
- 8.0.6
- 8.1.6
- 8.1.7
- 9.0.1
- 9.2.0

You want to record metadata about these databases in a single recovery catalog database. According to [Table B-1](#), you can use a single 9.2.0 recovery catalog database with a 9.2.0 catalog schema for all but the 8.0.3, 8.0.4, and 8.0.5 target databases. The 8.0.3 target database requires an 8.0.3 catalog schema. "[Note 2: 8.1 or Later Catalog Schemas and 8.0.4 or 8.0.5 RMAN Executables](#)" on page B-3 indicates that in general 8.0.4 and 8.0.5 target databases should not use the 9.2.0 catalog schema.

RMAN permits you to create multiple catalog schemas in a single catalog database. Hence, the solution illustrated in [Table B-2](#) is to do the following:

- Use a single release 9.2.0 catalog database to store all metadata, but create three separate catalog schemas of the following releases: 8.0.3, 8.0.5, and 9.2.0
- Ensure that the RMAN executable version matches the release of the target database that it is backing up

Table B-2 RMAN Compatibility Scenario with Release 9.0.1 Catalog Database

Target Database	Catalog Schema
8.0.3	8.0.3
8.0.4	8.0.5
8.0.5	8.0.5
8.0.6	9.2.0
8.1.5	9.2.0
8.1.6	9.2.0
8.1.7	9.2.0
9.0.1	9.2.0
9.2.0	9.2.0

Index

Symbols

? symbol

in RMAN quoted strings, 1-4

@ command, 2-6

in RMAN quoted strings, 1-4

@@ command, 2-7

A

ALLOCATE CHANNEL command, 2-8, 2-15

and shared server, 2-9, 2-13

FOR MAINTENANCE option, 2-12

ALTER DATABASE command, 2-19

archivelogRecordSpecifier clause, 2-21

autobackups

control file, 2-75

B

BACKUP command, 2-26

BACKUP_TAPE_IO_SLAVES initialization
parameter, 2-30

BLOCKRECOVER command, 2-48

C

CATALOG command, 2-52

CHANGE command, 2-55

channels

allocating to shared server sessions, 2-9, 2-13

code examples

description of RMAN, 1-6

command line

arguments for RMAN, 2-59

commands, Recovery Manager

@, 2-6

@@, 2-7

ALLOCATE CHANNEL, 2-8, 2-15

ALLOCATE CHANNEL FOR
MAINTENANCE, 2-12

ALTER DATABASE, 2-19

archivelogRecordSpecifier clause, 2-21

BACKUP, 2-26

BLOCKRECOVER, 2-48

CATALOG, 2-52

CHANGE, 2-55

completedTimeSpec clause, 2-64

CONFIGURE, 2-66

CONNECT, 2-80

connectStringSpec clause, 2-82

COPY, 2-84

CREATE CATALOG, 2-89

CREATE SCRIPT, 2-91

CROSSCHECK, 2-93

DELETE, 2-98

DELETE SCRIPT, 2-103

deprecated, A-1

DROP CATALOG, 2-105

DUPLICATE, 2-106

EXECUTE SCRIPT, 2-115

EXIT, 2-117

HOST, 2-118

LIST, 2-122

listObjList clause, 2-140

PRINT SCRIPT, 2-146

QUIT, 2-148

recordSpec, 2-149

RECOVER, 2-151
REGISTER, 2-157
RELEASE CHANNEL, 2-159, 2-160
REPLACE SCRIPT, 2-161
REPORT, 2-164
RESET DATABASE, 2-172
RESTORE, 2-175
RESYNC CATALOG, 2-187
RUN, 2-190
SEND, 2-193
SET, 2-195
SHOW, 2-202
SHUTDOWN, 2-205
SPOOL, 2-208
SQL, 2-210
STARTUP, 2-212
summary, 2-2, 3-2
SWITCH, 2-214
untilClause, 2-216
UPGRADE CATALOG, 2-219
VALIDATE, 2-221

compatibility
recovery catalog, B-2
Recovery Manager, B-2

completedTimeSpec clause, 2-64
CONFIGURE command, 2-66
CONNECT command, 2-80
connectStringSpec clause, 2-82
control files
automatic backups, 2-75
COPY command, 2-84
corrupt datafile blocks
maximum acceptable number, 2-198
corruption detection
using SET MAXCORRUPT command, 2-198
CREATE CATALOG command, 2-89
CREATE SCRIPT command, 2-91
CROSSCHECK command, 2-93

D

dates
specifying in RMAN commands, 2-216
DELETE command, 2-98
DELETE SCRIPT command, 2-103

deprecated commands
Recovery Manager, A-1
DISKRATIO parameter
BACKUP command, 2-40
DROP CATALOG command, 2-105
DUPLICATE command, 2-106

E

environment variables
in RMAN strings, 1-4
EXECUTE SCRIPT command, 2-115
EXIT command, 2-117

F

FILESERSET parameter
BACKUP command, 2-37

H

HOST command, 2-118

I

initialization parameters
BACKUP_TAPE_IO_SLAVES, 2-30

K

keywords
in syntax diagrams, 1-3

L

LIST command, 2-122
listObjList clause, 2-140

M

MAXCORRUPT parameter
SET command, 2-198
MAXSETSIZE parameter
BACKUP command, 2-37

N

normalization

of pre-8.1.6 control file on NT, 2-179

P

parameters

in syntax diagrams, 1-3

PL/SQL stored procedures

executing within RMAN, 2-211

PRINT SCRIPT command, 2-146

Q

QUIT command, 2-148

quoted strings

environment variables, 1-4

R

RC_ARCHIVED_LOG view, 2-22, 3-4

RC_BACKUP_CONTROLFILE view, 3-5

RC_BACKUP_CORRUPTION view, 3-7

RC_BACKUP_DATAFILE view, 3-8

RC_BACKUP_PIECE view, 3-10

RC_BACKUP_REDOLOG view, 3-11

RC_BACKUP_SET view, 3-12

RC_BACKUP_SPFILE view, 3-14

RC_CHECKPOINT view, 3-15

RC_CONTROLFILE_COPY view, 3-15

RC_COPY_CORRUPTION view, 3-16

RC_DATABASE recovery catalog view, 3-17

RC_DATABASE view, 2-197

RC_DATABASE_BLOCK_CORRUPTION
view, 3-18

RC_DATABASE_INCARNATION view, 2-197,
3-18

RC_DATAFILE view, 3-19

RC_DATAFILE_COPY view, 3-20

RC_LOG_HISTORY view, 3-22

RC_OFFLINE_RANGE view, 3-23

RC_PROXY_CONTROLFILE view, 3-24

RC_PROXY_DATAFILE view, 3-25

RC_REDO_LOG view, 3-27

RC_REDO_THREAD view, 3-27

RC_RESYNC view, 3-28

RC_RMAN_CONFIGURATION view, 3-29

RC_STORED_SCRIPT view, 3-29

RC_STORED_SCRIPT_LINE view, 3-30

RC_TABLESPACE view, 3-30

recordSpec clause, 2-149

RECOVER command, 2-151

recovery catalog

views, 3-1

Recovery Manager

backups

control file autobackups, 2-75

commands

@, 2-6

@@, 2-7

ALLOCATE CHANNEL, 2-8, 2-15

ALLOCATE CHANNEL FOR

MAINTENANCE, 2-12

ALTER DATABASE, 2-19

archiveLogRecordSpecifier clause, 2-21

BACKUP, 2-26

BLOCKRECOVER, 2-48

CATALOG, 2-52

CHANGE, 2-55

completedTimeSpec, 2-64

CONFIGURE, 2-66

CONNECT, 2-80

connectStringSpec, 2-82

COPY, 2-84

CREATE CATALOG, 2-89

CREATE SCRIPT, 2-91

CROSSCHECK, 2-93

DELETE, 2-98

DELETE SCRIPT, 2-103

DROP CATALOG, 2-105

DUPLICATE, 2-106

EXECUTE SCRIPT, 2-115

EXIT, 2-117

HOST, 2-118

LIST, 2-122

listObjList, 2-140

PRINT SCRIPT, 2-146

QUIT, 2-148

recordSpec, 2-149

RECOVER, 2-151

- REGISTER, 2-157
- RELEASE CHANNEL, 2-159
- REPLACE SCRIPT, 2-161
- REPORT, 2-164
- RESET DATABASE, 2-172
- RESTORE, 2-175
- RESYNC, 2-187
- RUN, 2-190
- SEND, 2-193
- SET, 2-195
- SHOW, 2-202
- SHUTDOWN, 2-205
- SPOOL, 2-208
- SQL, 2-210
- STARTUP, 2-212
- SWITCH, 2-214
 - untilClause, 2-216
- UPGRADE CATALOG, 2-219
- VALIDATE, 2-221
 - compatibility, B-2
 - dates in commands, 2-216
 - symbolic links for filenames, 2-178
 - syntax conventions, 1-2
- REGISTER command, 2-157
- RELEASE CHANNEL command (RMAN), 2-159
 - releasing a maintenance channel, 2-160
- REPLACE SCRIPT command, 2-161
- REPORT command, 2-164
- RESET DATABASE command, 2-172
- RESTORE command, 2-175
- RESYNC CATALOG command, 2-187
- RUN command, 2-190

S

- SEND command, 2-193
- SET command, 2-195
- shared server
 - allocating channels, 2-9, 2-13
- SHOW command, 2-202
- SHUTDOWN command, 2-205
- SPOOL command, 2-208
- SQL command, 2-210
- STARTUP command, 2-212
- stored procedures

- executing within RMAN, 2-211
- SWITCH command, 2-214
- symbolic links
 - and RMAN, 2-178
- syntax conventions
 - Recovery Manager, 1-2
- syntax diagrams
 - explanation of, 1-2
 - keywords, 1-3
 - parameters, 1-3

U

- untilClause, 2-216
- UPGRADE CATALOG command, 2-219

V

- VSARCHIVED_LOG view, 2-22, 2-23, 2-24, 2-136
- V\$BACKUP_CORRUPTION view, 2-42
- V\$BACKUP_DEVICE view, 2-10, 2-13
- V\$BACKUP_PIECE view, 2-127, 2-129, 2-131
- V\$BACKUP_SET view, 2-126, 2-130, 2-131, 2-132, 2-133
- V\$DATABASE view, 2-197
- V\$DATABASE_BLOCK_CORRUPTION
 - view, 2-49, 2-50, 2-51, 2-86, 2-155, 2-183, 2-222
- V\$DATAFILE view, 2-96
- V\$DATAFILE_COPY view, 2-96, 2-134, 2-135
- V\$DATAFILE_HEADER view, 2-96
- V\$PROXY_DATAFILE view, 2-129
- V\$SESSION view, 2-200
- VALIDATE command, 2-221
- views
 - recovery catalog, 3-1