

Backup und Recovery

- [Exportieren einer bestimmten Tabelle](#)
- [Recovery eines Controlfiles](#)
- [Recovery der Redo-Logs](#)
- [Allgemeines zu RMAN](#)
- [RMAN - Aufruf und Connect](#)
- [RMAN Format Codes](#)
- [Vollständiges Recovery](#)

Exportieren einer bestimmten Tabelle

Mit dem Befehl „exp“ lässt sich eine bestimmte Tabelle in eine Datei exportieren. Es handelt sich allerdings um ein Binärformat, welches nur mit dem „imp“-Tool wieder in eine Oracle-Datenbank importiert werden kann.

```
[oracle@oradb]$ /oracle/product/9.2.0/bin/exp user/pass file=dumpfile.dmp log=dumpfile.log  
tables=TABELLE1
```

Recovery eines Controlfiles

1. CTL-File aus Backup wiederherstellen

2. Datenbank in mount-Phase starten:

```
SQL> startup mount
```

3. Recover der Datenbank durchführen:

```
SQL> recover database until cancel using backup controlfile;
```

? Recovery schlägt fehl, da Oracle im Archivelog sucht, aber die benötigten Infos in den Redo-Logs zu finden sind

? manuelle Angabe des Filenamens der Redo-Logs (evtl. müssen alle nacheinander durchprobiert werden!)

4. Wenn erfolgreich kann die Datenbank geöffnet werden:

```
SQL> alter databse open resetlogs;
```

Recovery der Redo-Logs

- Verlust eines online Redo-Logs (nicht das current und inactive):

kann einfach aus einem Backup wiederhergestellt werden, danach ein

```
alter database clear logfile '<filename>;'
```

- Verlust des current Redo-Logs: Worst Case: Es kann kein Recovery dieses Redo-Logs durchgeführt werden, einzig möglicher Weg ist die Wiederherstellung der kompletten Datenbank aus einem Backup.

Allgemeines zu RMAN

Alle RMAN-Einstellungen anzeigen:

```
RMAN> show all;
```

Automatisches Backup der Controlfiles einschalten:

```
RMAN> CONFIGURE CONTROLFILE AUTOBACKUP ON;
```

Schnellverbindung zu RMAN und zur Katalog-Datenbank:

```
[oracle@student3 ~]$ rman target / catalog rman/rman@repo
```

RMAN konfigurieren und einen Recovery-Katalog einrichten:

1. User anlegen und berechtigen in ext. Oracle-Datenbank für den RMAN-Catalog

```
SQL> create user rman identified by rman default tablespace rman_ts;  
SQL> grant connect,resource,recovery_catalog_owner to rman;
```

2. Verbinden zur lokalen Datenbank und zum entfernten Katalog-Server:

```
[oracle@server ~]$ rman target / catalog rman/rman@repo  
...  
connected to target database: ORCL (DBID=1153117770)  
connected to recovery catalog database
```

3. Katalog erzeugen und Datenbank mit dem Katalog registrieren:

```
RMAN> create catalog;  
  
recovery catalog created  
  
RMAN> register database;  
  
database registered in recovery catalog  
starting full resync of recovery catalog  
full resync complete
```

RMAN - Aufruf und Connect

das Folgende wurde durch [Hermann Brunner](#) für unseren Oracle DBA-Kurs zusammengefasst:

Aufruf / Connect

```
$ rman
RMAN> connect target /
RMAN> connect catalog user/password@catdb
```

oder direkt beim Aufruf

```
$ rman target /
$ rman target / catalog user/password@catdb
$ rman target / log='/pfad/zum/log.log' [append] (mit Angabe einer Logdatei)
```

Einstellungen

```
RMAN> show all;
RMAN> configure [...]
```

Syntax wie in der SHOW Anzeige. Beispiele:

```
RMAN> configure retention policy to redundancy 3;
RMAN> configure controlfile autobackup on;
```

komplexere Beispiele:

```
RMAN> configure device type disk parallelism 3 backup type backupset;
RMAN> configure datafile backup copies for device type sbt to 2;
```

Channels vorkonfigurieren:

```
RMAN> configure channel device type disk format '/pfad/%U';
RMAN> configure channel n device type disk format '/home/backup_m/%d_%t_%s.bck'
```

Einstellungen auf default zurücksetzen:

```
RMAN> configure retention policy clean;
RMAN> configure controlfile autobackup clear;
```

Recovery Catalog

Recovery Catalog einrichten:

```
auf dem Katalog-Server:
SQL> create tablespace ts_rman datafile '/pfad/rman01.dbf' size 100m; ## Richtgröße 100MB pro Zi
SQL> create user rman identified by 'passwd' default tablespace ts_rman;
```

```
SQL> grant connect, resource, recovery_catalog_owner to rman;
SQL> grant select any dictionary to rman;

auf dem Ursprungs-Server:
$ rman target / catalog rman/rman@catdb ## (evtl. tnsnames.ora anpassen)
RMAN> create catalog;
RMAN> register database;

RMAN> report schema;
RMAN> report schema at time 'sysdate -3' ## (wie war die DB-Struktur vor 3 Tagen?)
```

Maintenance commands für den Katalog:

```
SQL> connect rman/rman@catdb;
SQL> select * from cat;

RMAN> resync catalog;
RMAN> unregister database;
RMAN> upgrade catalog; ## Katalog auf den Stand des rman-Clients bringen
RMAN> drop catalog;

RMAN> catalog backuppiece 'filename';
RMAN> catalog controlfilecopy 'filename';
RMAN> catalog datafilecopy 'filename';
RMAN> catalog archivelog 'filename';
RMAN> catalog recovery area noprompt;
RMAN> catalog start with '/u01/fra/archive' ## Start-Pfad, ab dem gesucht werden soll
```

Backups erstellen

```
RMAN> backup database;
RMAN> backup tablespace ts_name;
RMAN> backup datafile n;
RMAN> backup datafile df_name;
RMAN> backup current controlfile;
RMAN> backup spfile;
RMAN> backup archivelog [from sequence=nnn] [delete {ALL} input];

RMAN> backup as backupset [...];
RMAN> backup as compressed backupset [...];
RMAN> backup as copy [...];
RMAN> backup validate [...]; ## füllt die v$database_block_corruption Liste
RMAN> backup check logical [...];

RMAN> backup [...] plus archivelog;
RMAN> backup [...] include controlfile;
RMAN> backup [...] format '/pfad/zum/backup';
RMAN> backup [...] tag='freier Text';
RMAN> backup [...] not backed up nnn times;
RMAN> backup [...] duration hh:mm minimize load;
RMAN> backup [...] delete [ALL] input;
```

Alte Version 8i/9i Syntax:

```
RMAN> run {
    ALLOCATE CHANNEL c1 DEVICE TYPE disk FORMAT='/home/oracle/backup1/datafile_5.bck';
    BACKUP DATAFILE 5;
    RELEASE CHANNEL c1; } ## nur in rman8 nötig
```

Sinnvolle Technik für Full Backup mit anschließender Sicherung der Archive-Logs:

```
RMAN> backup database;
RMAN> sql 'alter system archive log current';
RMAN> backup archivelog all;
```

Seit Oracle 9i gehts einfacher:

```
RMAN> backup database plus archivelog; ## alternativ mit Option: delete [all] input
```

Zweistufiges Backup - z.B. Sicherung aller DISK Backups auf Band - Variante 1:

```
RMAN> backup device type disk as backupset database;  
Optional:  
RMAN> [...] plus archivelog;  
RMAN> [...] delete input;
```

Zweistufiges Backup - z.B. Sicherung aller DISK Backups auf Band - weitere Varianten:

```
RMAN> backup backupset completed before 'sysdate -1';  
RMAN> backup copy of database;
```

Trickreichere rman Commands:

```
RMAN> configure device type disk parallelism 4 backup type to copy;  
RMAN> backup database; ## erzeugt einzelne Datafiles in fra/datafile...
```

Incrementelles Backup:

```
RMAN> backup [as backupset] incremental level 0 [database|tablespace|datafile];  
RMAN> backup [as backupset] incremental level n [cumulative] [database|tablespace|datafile];
```

Incrementelles Updating einer Copy die mit <> gekennzeichneten Optionen sind nur in Oracle 9 nötig!

```
RMAN> backup as copy <incremental level 0> [database|tablespace|datafile] <format='somewhere/somewhere'>  
RMAN> backup incremental level 1 <for recover of copy with tag='TEST'> [database|tablespace|datafile]  
RMAN> recover copy of [database|tablespace|datafile] <with tag='TEST'>;
```

Restore / Recover Commands

```
RMAN> restore [database|tablespace|datafile];  
RMAN> restore [...] from tag 'xxx';  
RMAN> restore [...] to '/somewhere/somefile';  
RMAN> restore validate [database|tablespace|datafile]; ## prüft ob korrekte Backups vorhanden sind  
  
RMAN> recover database;  
RMAN> recover [tablespace|datafile]; ## database muss open, tablespace oder datafile muss offline sein
```

Weitere Recovery-Optionen Für Recovery ist nur beschränkter Platz für Archivelogs vorhanden:

```
RMAN> recover [...] delete archivelog maxsize 100M; ## braucht nur max. 100MB
```

Recovery an einen anderen Ort, weil Original-Location nicht erreichbar (z.B. weil neue Disk):

```
RMAN> set newname for datafile n to '/pfad/file.dbf';  
RMAN> restore datafile n;  
RMAN> switch datafile all; ## führt das SET NEWNAME im Katalog durch, Datenbank wird auf die neuen Dateien umgestellt  
RMAN> recover datafile n;  
RMAN> sql 'alter database datafile n online';
```

Unvollständiges Recovery Prinzip im rman: „UNTIL“-Klausel muss VOR dem RESTORE gesetzt werden!


```
RMAN> run { set until time 'time'; ## oder: set until change 'scn' | until sequence 'seq#' threa
            restore database;
            recover database;
            alter database open resetlogs; }
```

Alles Futsch? --> komplettes Recovery einer Datenbank

Alle zur Datenbank gehörenden Dateien sind verloren...
Prinzip:


```
RMAN> recover spfile from autobackup; ## nomount Phase
RMAN> recover controlfile from autobackup; ## nomount Phase
RMAN> restore database; ## mount Phase
RMAN> recover database; ## mount Phase
RMAN> alter database open resetlogs; ## Datenbank sollte jetzt wieder laufen
```

im Detail:

```
RMAN> startup nomount force; ## startup mit default-Parametern. Geht nur im RMAN!
RMAN> restore spfile from autobackup; ## funktioniert nur, wenn an default-locations zu finden)
oder:
RMAN> set dbid=1234567890
RMAN> restore spfile from 'location_of_autobackups';

dann weiter:
RMAN> shutdown abort;
RMAN> startup nomount; ## Datenbank startet wieder mit eigenem spfile
RMAN> restore controlfile from autobackup;
RMAN> alter database mount;

ab hier sind viele LIST-Commands möglich:
RMAN> list incarnation;
RMAN> list backup of database; ## etc..

also einfach den Rest restoren und recovern:
RMAN> restore database;
RMAN> recover database; ## führt zu Fehler bei letzter Log-Seq#, da das online Redo-Log ja futsch
daher am einfachsten/besten:
SQL> recover database until cancel using backup controlfile; ## bis zum Schluss, falls zufällig
SQL> alter database open resetlogs;

Fertig! An dieser Stelle sollte dann natürlich, wegen der neuen Datenbank-Incarnation, nochmal e
RMAN> backup database;
```

Umgang mit RMAN-Scripts

```
RMAN> create [global] script name {backup database;};
RMAN> create [global] script name from file 'file mit rman-commands';

RMAN> print script name;
RMAN> print script name to file 'filename';
RMAN> list [global] script names;

RMAN> replace script name { new script commands; }

RMAN> run { execute script name; }
```

Report, List und Maintenance Commands

```

RMAN> report schema;
RMAN> report obsolete;
RMAN> report obsolete orphan; ## aus der vorletzten incarnation oder älter
RMAN> report need backup [redundancy nn | days nn];
RMAN> report unrecoverable [...];

RMAN> list backup of [database|tablespace|datafile];
RMAN> list backup of [archivelog|controlfile|spfile];
RMAN> list backup of [...] summary; ## zeigt übersichtliches Kurzformat
RMAN> list copy of [database|tablespace|datafile];
RMAN> list incarnation [of database];

RMAN> crosscheck [backup of database|archivelog]; ## setzt nicht vorhandene Backupsets|Archivelo
RMAN> delete [noprompt] expired backupset; ## löscht expired Eintragungen aus dem Katalog
RMAN> delete expired archivelog;
RMAN> delete [noprompt] obsolete;
RMAN> delete obsolete [redundancy n];
RMAN> delete obsolete [orphan];
RMAN> delete backupset nnn;

```

Weitere Tips und Tricks mit RMAN

Direkte SQL-Commands, die RMAN selbst 'kann':

```

RMAN> startup;
RMAN> shutdown [...];
RMAN> alter database open [resetlogs];

```

Alle SQL Commands, die keinen Output benötigen, können über die SQL-Klausel gestartet werden

```

RMAN> sql 'alter system switch logfile';
RMAN> sql 'alter system archive log all';

```

Wenn man NLS_DATE_FORMAT außerhalb von RMAN setzt, kann man innerhalb von RMAN bessere Anzeigen von Backup-Zeitpunkten bekommen (z.B. in LIST Kommandos):

```

$ export NLS_LANG=german_germany ## für RMAN in 10g nicht mehr nötig
$ export NLS_DATE_FORMAT=yyy-mm-dd:hh24:mi:ss
RMAN> list backup of database;

```

Wenn man mit Hausmitteln die CTL-Files ausgetauscht hat und RMAN nicht durchblickt, weil die restaurierte CTL-File aus einer alten Incarnation stammt:

```

RMAN> reset incarnation to 2; ## ohne catalog DB
RMAN> reset database to incarnation 2; ## mit catalog DB

```

RMAN Format Codes

Restrictions and Usage Notes Any name that is legal as a sequential filename on the platform is allowed, so long as each backup piece or copy has a unique name. If backing up to disk, then any legal disk filename is allowed, provided it is unique.

Keywords and Parameters

Syntax Element	Description
%a	Specifies the activation ID of the database
%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
%e	Specifies the archived log sequence number
%f	Specifies the absolute file number
%F	Combines the DBID, day, month, year, and sequence into a unique and repeatable generated name. This variable translates into c-#####-YYYYMMDD-QQ , where ##### stands for the DBID. The DBID is printed in decimal so that it can be easily associated with the target database. YYYYMMDD is a time stamp in the Gregorian calendar of the day the backup is generated and QQ is the sequence in hexadecimal number that starts with 00 and has a maximum of FF (256)
%h	Specifies the archived redo log thread number
%l	Specifies the DBID
%M	Specifies the month in the Gregorian calendar in format MM
%N	Specifies the tablespace name
%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. 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: <tt>YYYYMMDD</tt>

Syntax Element	Description
%u	Specifies an 8-character name constituted by compressed representations of the backup set or image copy number and the time the backup set or image copy was created
%U	Specifies a system-generated unique filename (default). The meaning of %U is different for image copies and backup pieces. For a backup piece %U specifies a convenient shorthand for %u_%p_%c that guarantees uniqueness in generated backup filenames. If you do not specify a format when making a backup, then RMAN uses %U by default. For an image copy of a datafile %U means data-D-%d_id-%l_TS-%N_FNO-%f_%u . For an image copy of an archived redo log %U means arch-D_%d-id-%l_S-%e_T-%h_A-%a_%u . For an image copy of a control file %U means cf-D_%d-id-%l_%u
%Y	Specifies the year in the format YYYY
% %	Specifies the '%' character. For example, %%Y translates to the string %Y

Vollständiges Recovery

Ach du Scheiße! Alles in oradata (ctl-, db-, redo-files) ist weg! Aber zum Glück haben wir ja ein Backup:

```
RMAN> connect target /  
RMAN> connect catalog rman/rman@repo  
RMAN> startup nomount [force]  
RMAN> restore controlfile from autobackup;  
RMAN> alter database mount;  
RMAN> restore database;  
RMAN> recover database;
```

Der letzte Befehl bricht mit Fehler ab. Die Archivlogs sind zwar vorhanden und wurden bis zum letzten logswitch wiederhergestellt. Die online Redo-Logs sind leider futsch, d.h. wir müssen die Datenbank mit 'resetlogs' öffnen:

```
RMAN> alter database open resetlogs;
```