```
connect SYS/change_on_install as SYSDBA
set echo on
spool /var/oracle/OraHome2/assistants/dbca/logs/CreateDB.log
startup nomount pfile="/var/oracle/OraHome2/admin/galinux2/scripts/
init.ora";
CREATE DATABASE galinux2
MAXINSTANCES 1
MAXLOGHISTORY 1
MAXLOGFILES 5
MAXLOGMEMBERS 5
MAXDATAFILES 100
DATAFILE '/var/oracle/OraHome2/oradata/galinux2/system01.dbf' SIZE 325M
REUSE
AUTOEXTEND ON NEXT  10240K MAXSIZE UNLIMITED
UNDO TABLESPACE "UNDOTBS" DATAFILE
'/var/oracle/OraHome2/oradata/galinux2/undotbs01.dbf' SIZE 200M REUSE
AUTOEXTEND ON NEXT  5120K MAXSIZE UNLIMITED
CHARACTER SET US7ASCII
NATIONAL CHARACTER SET AL16UTF16
LOGFILE GROUP 1 ('/var/oracle/OraHome2/oradata/galinux2/redo01.log') SIZE
100M,
GROUP 2 ('/var/oracle/OraHome2/oradata/galinux2/redo02.log') SIZE 100M,
GROUP 3 ('/var/oracle/OraHome2/oradata/galinux2/redo03.log') SIZE 100M;
spool off
exit;
```

Quellcode 2.1: Beispielskript zum Erstellen einer Datenbank mit Database Creation Assistant (DBCA)

```
REM
REM NAME        : init_ora_rct.sql
REM FUNCTION    : Recreate the instance init.ora file
REM USE         : GENERAL
REM Limitations : None
REM
SET NEWPAGE 0 VERIFY OFF
SET ECHO OFF feedback off termout off PAGES 300 lines 80 heading off
column name  format a80 word_wrapped
column dbname new_value db noprint
select name dbname from v$database;
DEFINE OUTPUT = 'rep_out\&db\init.ora'
SPOOL &OUTPUT
SELECT '# Init.ora file from v$parameter' name from dual
union
select '# generated on:'||sysdate name from dual
union
select '# script by MRA 08/7/01 TUSC' name from dual
union
select '#' name from dual
UNION
SELECT name||' = '||value name  FROM V$PARAMETER
WHERE value is not null and ISDEFAULT='FALSE';
SPOOL OFF
CLEAR COLUMNS
SET NEWPAGE 0 VERIFY OFF
SET ECHO ON termout on PAGES 22 lines 80 heading on
SET TERMOUT ON
UNDEF OUTPUT
PAUSE Press enter to continue
```

Quellcode 2.2: Skript zum Neuerstellen der Initialisierungsdatei.

```
REM
REM ORA_KILL.SQL
REM FUNCTION: Kills nonessential Oracle sessions (those that aren't
REM owned)
REM          : by SYS or "NULL"
REM DEPENDENCIES: Depends on kill_session procedure
REM MRA 9/12/96
REM
SET HEADING OFF TERMOUT OFF VERIFY OFF ECHO OFF
SPOOL kill_all.sql
SELECT 'EXECUTE kill_session('||chr(39)||sid||chr(39)||','||
chr(39)||serial#||chr(39)||');' FROM v$session
WHERE username IS NOT NULL
OR username <> 'SYS'
/
SPOOL OFF
START kill_all.sql

The kill_session procedure Is defined as:

CREATE OR REPLACE PROCEDURE kill_session ( session_id in varchar2,
serial_num in varchar2)
AS
cur INTEGER;
ret INTEGER;
string VARCHAR2(100);
BEGIN
--
-- Comment out the following three lines to
-- not use KILL
--
string :=
'ALTER SYSTEM KILL SESSION' || CHR(10) ||
CHR(39)||session_id||','||serial_num||CHR(39);
--
-- Uncomment the following 4 lines to use DISCONNECT
--
-- string :=
--          'ALTER SYSTEM DISCONNECT SESSION' || CHR(10) ||
-- CHR(39)||session_id||','||serial_num||CHR(39)||CHR(10)||
--' POST_TRANSACTION';
cur := dbms_sql.open_cursor;
dbms_sql.parse(cur,string,dbms_sql.v7);
ret := dbms_sql.execute(cur)  ;
dbms_sql.close_cursor(cur);
EXCEPTION
WHEN OTHERS THEN
raise_application_error(-20001,'Error in execution',TRUE);
IF dbms_sql.is_open(cur) THEN
dbms_sql.close_cursor(cur); END IF;
END; /
```
Quellcode 2.3: Die ORA_KILL.SQL-Prozedur zum Abbrechen unwesentlicher Oracle-Sitzungen

```
#!/bin/ksh
ORATAB=/etc/oratab
trap 'exit' 1 2 3
# Set path if path not set (if called from /etc/rc)
case $PATH in
         "")     PATH=/bin:/usr/bin:/etc
           export PATH ;;
esac
rm kill.lis
rm proc.lis
touch kill.lis
touch proc.lis
#
# Loop for every entry in oratab
#
cat $ORATAB | while read LINE
do
        case $LINE in
        \#*)          ;;     #comment-line in oratab
       *)
       ORACLE_SID='echo $LINE | awk -F: '{print $1}' -'
     if [ "$ORACLE_SID" = '*' ] ; then
           ORACLE_SID=""
     fi
        esac
        if [ "$ORACLE_SID" <> '*' ] ; then
             proc_name='oracle'$ORACLE_SID
            ps -ef|grep $proc_name>>proc.lis
        fi
done
cat proc.lis | while read LINE2
do
           command='echo $LINE2 | awk -F: 'BEGIN { FS = ",[ \t]*|[ \t]+" }
                               { print $2}' -'
               test_it='echo $LINE2|awk -F: 'BEGIN { FS = ",[ \t]*|[ \t]+" }
                               { print $8}' -'
             if [ "$test_it" <> 'grep' ] ; then
                   command='kill -9 '$command
             echo $command>>kill.lis
       fi
done
rm proc.lis
chmod 755 kill.lis
kill.lis
rm kill.lis
```

Quellcode 2.4: Shell-Skript zum Abbrechen unwesentlicher Oracle-Prozesse von der Server-Seite

```
REM Script for getting undocumented init.ora
REM parameters from a 7.3, 8.0.x,8.1 or 9.0 instance
REM MRA - TUSC 4/23/97
REM
COLUMN parameter            FORMAT a37
COLUMN description          FORMAT a30 WORD_WRAPPED
COLUMN "Session Value"      FORMAT a10
COLUMN "Instance Value"     FORMAT a10
SET LINES 100
SET PAGES 0
SPOOL undoc.lis
SELECT
     a.ksppinm  "Parameter",
     a.ksppdesc "Description",
     b.ksppstvl "Session Value",
     c.ksppstvl "Instance Value"
FROM
     x$ksppi a,
     x$ksppcv b,
     x$ksppsv c
WHERE
     a.indx = b.indx
     AND a.indx = c.indx
     AND a.ksppinm LIKE '/_%' escape '/'
/
SPOOL OFF
SET LINES 80 PAGES 20
CLEAR COLUMNS
```
Quellcode 2.5: Skript zum Abrufen der undokumentierten Initailisierungsparameter

```
How to dump a segment header - by Don Burleson
set heading off;
spool dump_em.sql;
select
'alter session set events ''immediate trace name blockdump level '||
to_char((header_file*16777216)+header_block)||''';'
from
dba_segments
where
segment_name = 'VBAP';

spool off;

cat dump_em.sql
@dump_em
```
Quellcode 2.7: Der Einsatz von SET EVENTS auf Sitzungsebene

```
rem ************************************************************
rem NAME    : BOUND_OB.sql
rem FUNCTION: Show objects with extents bounded by freespace
rem ************************************************************
START title80 "Objects With Extents Bounded by Free Space"
SPOOL rep_out\&db\b_ob..lis
COLUMN e FORMAT a15        HEADING "TABLE SPACE"
COLUMN a FORMAT a6         HEADING "OBJECT|TYPE"
COLUMN b FORMAT a30        HEADING "OBJECT NAME"
COLUMN c FORMAT a10        HEADING "OWNER ID"
COLUMN d FORMAT 99,999,999 HEADING "SIZE|IN BYTES"
BREAK ON e SKIP 1 ON c
SET FEEDBACK OFF
SET VERIFY OFF
SET TERMOUT OFF
COLUMN bls NEW_VALUE block_size NOPRINT
SELECT blocksize bls
FROM sys.ts$
WHERE name='SYSTEM';

SELECT h.name e, g.name c, f.object_type a, e.name b,
       b.length*&&block_size d
 FROM sys.uet$ b, sys.fet$ c, sys.fet$ d, sys.obj$ e,
      sys.sys_objects f,sys.user$ g, sys.ts$ h
 WHERE b.block# = c.block# + c.length
   AND b.block# + b.length = d.block#
   AND f.header_file = b.segfile#
   AND f.header_block = b.segblock#
   AND f.object_id = e.obj#
   AND g.user# = e.owner#
   AND b.ts# = h.ts#
 ORDER BY 1,2,3,4
/

CLEAR COLUMNS
SET FEEDBACK ON
SET VERIFY ON
SET TERMOUT ON
TTITLE ''
TTITLE OFF
SPOOL OFF
CLEAR BREAKS
```
Quellcode 3.1: Skripten zur Ermittlung der eingeschlossenen Objekte

```
-- Zunächst wird der Paketrumpf erstellt
-- Paket wird verwendet, damit alle Prozeduren
-- am selben Speicherort vorhanden und gut zu
-- steuern sind
-- M. Ault 1/14/97 Rev 1.0
--
CREATE OR REPLACE PACKAGE cascade_update AS
--
-- Erstes Paket ist update_column
-- Dieses Paket führt die Arbeit mithilfe von
-- DBMS_SQL aus, um die UPDATEs zur Laufzeit für
-- jede Tabelle neu zu erstellen.
--
PROCEDURE update_column(
     old_value      IN VARCHAR2,
     new_value      IN VARCHAR2,
     table_name     IN VARCHAR2,
     update_column  IN VARCHAR2
);
--
-- Nächste Prozedur ist update_tables
-- Dies ist die Prozedur zur Schleifensteuerung
-- für den Trigger und ruft update_column auf
--
PROCEDURE update_tables(
     source_table   IN VARCHAR2,
     old_value      IN VARCHAR2,
     new_value      IN VARCHAR2
);
--
-- ENDE DES PAKET-HEADERS
--
END cascade_update;
/
--
-- Jetzt wird der Paketrumpf erstellt,
-- der die eigentlichen Prozeduren und
-- den Code enthält
--
CREATE OR REPLACE PACKAGE BODY cascade_update AS
PROCEDURE update_column(
     old_value      IN VARCHAR2,
     new_value      IN VARCHAR2,
     table_name     IN VARCHAR2,
     update_column  IN VARCHAR2)
AS
--
-- Statusvariablen für die dbms_sql-Prozeduren definieren
--
     cur INTEGER;
     rows_processed INTEGER;
--
-- Verarbeitung starten
-- (dbms_output-Aufrufe für das Debugging für
-- den normalen Betrieb auskommentieren!)
--
BEGIN
DBMS_OUTPUT.PUT_LINE(
'Table name: '||table_name||' Column: '||update_column);
     --
     -- Initialisierung der dynamischen Cursorposition
     -- für die Verarbeitung in dbms_sql
     --
     cur:=DBMS_SQL.OPEN_CURSOR;
     --
     -- initialisierten Speicherort mit der zu
     -- verarbeitenden Anweisung füllen
     --
DBMS_OUTPUT.PUT_LINE(
'UPDATE '||table_name||
' SET '||update_column||'='||chr(39)||new_value||chr(39)||
chr(10)||' WHERE '||
update_column||'='||chr(39)||old_value||chr(39)||
' AND 1=1');
     --
     dbms_sql.parse(cur,
     'UPDATE '||table_name||
' set '||update_column||'='||chr(39)||new_value||chr(39)||
chr(10)||' WHERE '||
update_column||'='||chr(39)||old_value||chr(39)||
' AND 1=1',dbms_sql.native);
     --
     -- dynamisch analysierte Anweisung ausführen
     --
     rows_processed:=DBMS_SQL.EXECUTE(cur);
     --
     -- dynamischen Cursor zur Vorbereitung auf
     -- die nächste Tabelle schließen
     --
     DBMS_SQL.CLOSE_CURSOR(cur);
--
-- ENDE DER PROZEDUR
--
END update_column;
--
PROCEDURE update_tables(
     source_table     IN VARCHAR2,
     old_value        IN VARCHAR2,
     new_value        IN VARCHAR2) as
--
-- Den Cursor zum Lesen der Datensätze aus
-- bbs_siteid_tables erstellen
-- Verwendung von *, um das Auslassen
-- einer Spalte zu verhindern
--
     CURSOR get_table_name IS
          SELECT
               *
          FROM
               bbs_update_tables
          WHERE
               main_table=source_table;
--
-- Variable vom Typ ROWTYPE für die Aufnahme eines
-- Datensatzes aus bbs_siteid_tables definieren.
-- Verwendung von ROWTYPE, um für zukünftige Änderungen
-- gewappnet zu sein.
--
     update_rec update_tables%ROWTYPE;
--
-- Verarbeitung beginnen
--
BEGIN
--
-- Cursor öffnen und Werte abrufen
--
   OPEN get_table_name;
   FETCH get_table_name INTO update_rec;
--
-- nun, da der Cursor geöffnet und die Werte in
-- Variablen eingelesen sind, kann die Schleife beginnen
--
```

```
LOOP
--
-- Unter Verwendung des Status NOTFOUND müssen wir den
-- Datensatz im Vorhinein füllen
--
   EXIT WHEN get_table_name%NOTFOUND;
--
-- Aufruf der Prozedur update_column initiieren
--
   update_column(old_value, new_value,
   update_rec.table_name, update_rec.column_name);
--
-- Nun wird der nächste Datensatz aus der Tabelle abgerufen
--
   FETCH get_table_name INTO update_rec;
--
-- Verarbeitung kehrt zur Schleifenanweisung zurück
--
 END LOOP;
--
-- Cursor schließen und beenden
--
 CLOSE get_table_name;
--
-- ENDE DER PROZEDUR
--
END update_tables;
--
-- ENDE DES PAKETRUMPFES
--
END cascade_update;
/
```
Quellcode 4.1: Paket für die Aktualisierungsweitergabe

```
CREATE OR REPLACE PROCEDURE get_bfiles(
  bfile_dir in   VARCHAR2,
  bfile_lis in   VARCHAR2,
  bfile_int_dir  VARCHAR2)
AS
  cur        INTEGER;
  bfile_int  VARCHAR2(100);
  sql_com    VARCHAR2(2000);
  file_proc  INTEGER;
  file_hand  utl_file.file_type;
  file_buff  VARCHAR2(1022);
  file_type  VARCHAR2(4);
BEGIN
  bfile_int:=UPPER(bfile_int_dir);
  file_hand:=utl_file.fopen(bfile_dir,bfile_lis,'R');
  LOOP
  BEGIN
    utl_file.get_line(file_hand,file_buff);
    cur:=dbms_sql.open_cursor;
    file_type:=SUBSTR(file_buff,INSTR(file_buff,'.')+1,3);
    file_type:=UPPER(file_type);
    IF file_type='GIF'
      THEN
       file_type:='GIF';
      ELSIF file_type='JPG'
        THEN file_type:='JPEG';
    END IF;
    sql_com:= 'INSERT INTO graphics_table '||CHR(10)||
              'VALUES (graphics_table_seq.NEXTVAL,'||CHR(39)||CHR(39)||
              ', bfilename('||
              CHR(39)||bfile_int||CHR(39)||','
              ||CHR(39)||file_buff||CHR(39)||
              ') ,'||CHR(39)||file_type||CHR(39)||')';
    dbms_output.put_line(sql_com);
    dbms_sql.parse(cur,sql_com,dbms_sql.v7);
    file_proc:=dbms_sql.execute(cur);
    dbms_sql.close_cursor(cur);
    EXCEPTION
      WHEN no_data_found THEN
        EXIT;
    END;
  END LOOP;
  utl_file.fclose(file_hand);
END;
/
```

Quellcode 4.2: Beispiel für eine Prozedur zum Einlesen eines `BFILE`

```
CREATE OR REPLACE PROCEDURE load_lob AS
  id         NUMBER;
  image1     BLOB;
  locator    BFILE;
  bfile_len  NUMBER;
  bf_desc    VARCHAR2(30);
  bf_name    VARCHAR2(30);
  bf_dir     VARCHAR2(30);
  bf_typ     VARCHAR2(4);
  ctr        INTEGER;
CURSOR get_id IS
SELECT bfile_id,bfile_desc,bfile_type
FROM graphics_table;
BEGIN
  open get_id;
 LOOP
   FETCH get_id INTO id, bf_desc, bf_typ;
   EXIT WHEN get_id%notfound;
   dbms_output.put_line('ID: '||to_char(id));
   SELECT bfile_loc
     INTO locator
     FROM graphics_table
     WHERE bfile_id=id;
   dbms_lob.filegetname(locator,bf_dir,bf_name);
   dbms_output.put_line('Dir: '||bf_dir);
   dbms_lob.fileopen(locator,dbms_lob.file_readonly);
   bfile_len:=dbms_lob.getlength(locator);
   dbms_output.put_line('ID: '||to_char(id)||' length: '||
     to_char(bfile_len));
   insert into dual_lob(x) values(empty_blob());
   select x into image1 from dual_lob;
   bfile_len:=dbms_lob.getlength(locator);
   dbms_lob.loadfromfile(image1,locator,bfile_len,1,1);
   IF bf_desc is null THEN
     bf_desc:=bf_name;
   END IF;
   insert into internal_graphics values (id,bf_desc,image1,bf_typ,'GENERAL');
   dbms_output.put_line(bf_desc||' Length: '||to_char(bfile_len)||
     ' Name: '||bf_name||' Dir: '||bf_dir||' '||bf_typ);
   dbms_lob.fileclose(locator);
   delete dual_lob;
 END LOOP;
END load_lob;
```

Quellcode 4.3: Beispiel für eine PL/SQL-Prozedur zum Verschieben von `BFILE`*s in* `BLOB`*s*

```
-- You may need to comment out the write_out procedure and
-- subsequenz calls to it, I like to track what tables need
-- analysis using a dba_running_stats table
--
CREATE OR REPLACE PROCEDURE check_tables (
   owner_name in varchar2,
   pchng IN NUMBER,
   lim_rows IN NUMBER) AS
--
CURSOR get_tab_count (own varchar2) IS
   SELECT table_name, nvl(num_rows,1)
   FROM dba_tables
   WHERE owner = upper(own);
--
tab_name    VARCHAR2(64);
rows        NUMBER;
string      VARCHAR2(255);
cur         INTEGER;
ret         INTEGER;
row_count   NUMBER;
com_string  VARCHAR2(255);
--
PROCEDURE write_out(
  par_name  IN VARCHAR2,
  par_value IN NUMBER,
  rep_ord   IN NUMBER,
  m_date    IN DATE,
  par_delta IN NUMBER) IS
 BEGIN
   INSERT INTO dba_running_stats VALUES(
   par_name,par_value,rep_ord,m_date,par_delta );
 END;
--
BEGIN
--
-- The next line is for schemas with many tables
-- If you don't lose the cursors you can exceed
-- open_cursor limits and flood the shared pool
--
DBMS_SESSION.SET_CLOSE_CACHED_OPEN_CURSORS(TRUE);
OPEN get_tab_count (owner_name);
LOOP
BEGIN
  FETCH get_tab_count INTO tab_name, rows;
  tab_name:=owner_name||'.'||tab_name;
  IF rows=0 THEN
    rows:=1;
  END IF;
EXIT WHEN get_tab_count%NOTFOUND;
DBMS_OUTPUT.PUT_LINE('Table name: '||tab_name||' rows: '||to_char(rows));
--
-- Need to have created the get_count procedure in the same schema
--
GET_COUNT(tab_name,row_count);
  IF row_count=0 THEN
    row_count:=1;
  END IF;
DBMS_OUTPUT.PUT_LINE('Row count for '||tab_name||': '||to_char(row_count));
DBMS_OUTPUT.PUT_LINE('Ratio: '||to_char(row_count/rows));
   IF (row_count/rows)>1+(pchng/100) OR (rows/row_count)>1+(pchng/100)
   THEN
     BEGIN
       IF (row_count<lim_rows) THEN
         string :=
           'ANALYZE TABLE '||tab_name||' COMPUTE STATISTICS ';
```

```
          ELSE
            string :=
              'ANALYZE TABLE '||tab_name||' ESTIMATE STATISTICS SAMPLE 30 PERCENT';
          END IF;
          cur := DBMS_SQL.OPEN_CURSOR;
          DBMS_OUTPUT.PUT_LINE('Beginning analysis');
          DBMS_SQL.PARSE(cur,string,dbms_sql.v7);
          ret := DBMS_SQL.EXECUTE(cur);
          DBMS_SQL.CLOSE_CURSOR(cur);
          DBMS_OUTPUT.PUT_LINE(' Table: '||tab_name||' had to be analyzed.');
          write_out(' Table: '||tab_name||' had to be analyzed.', row_count/
                      rows,33,sysdate,0);
          EXCEPTION
          WHEN OTHERS THEN
            raise_application_error(-20002,'Error in analyze: '||to_char(sqlcode)||'
                                         on '||tab_name,TRUE);
            write_out(' Table: '||tab_name||' error during analyze.
                       '||to_char(sqlcode), row_count/rows,33,sysdate,0);
            IF dbms_sql.is_open(cur) THEN
              dbms_sql.close_cursor(cur);
            END IF;
          END;
        END IF;
     EXCEPTION
     WHEN others THEN
     null;
   END;
   COMMIT;
   END LOOP;
   CLOSE get_tab_count;
   END;
```

Quellcode 4.4: Prozedur zum bedingten Analysieren von Tabellen

```
ACCEPT owner PROMPT 'Enter table owner name: '
ACCEPT table PROMPT 'Enter table name: '
SET HEADING OFF FEEDBACK OFF VERIFY OFF ECHO OFF RECSEP OFF PAGES 0
DEFINE cr = 'chr(10)'
SPOOL index_sz.sql
SELECT 'CREATE TABLE stat_temp AS SELECT * FROM index_stats WHERE
orwnum<1;'||&&cr
FROM dual;
SELECT
'ANALYZE INDEX '||owner||'.'||index_name||' VALIDATE STRUCTURE;'||&&cr||
'INSERT INTO stat_temp SELECT * FROM index_stats;'||&&cr||
'COMMIT;'
FROM dba_indexes
WHERE owner=upper('&owner')
AND table_name=upper('&table');
SPOOL OFF
@index_sz.sql
```
Quellcode 6.1: Codefragment für die Analyse aller Tabellenindices eines Besitzers

```
rem *****************************************************************
rem
rem NAME: brown.sql
rem
rem HISTORY:
rem Date Who What
rem -------- ---------- ---------------------------
rem 06/05/97 Mike Ault Updated for Oracle 7.x
rem 09/27/99 Mike Ault Verified for 8.x
rem 09/22/99 Mike Ault Verified for 9.x
rem FUNCTION: Will show index browning for all indexes for a
rem user.
rem INPUTS: owner = Table owner name.
Rem
*********************************************************************

column value noprint new_value blocksize
define cr=chr(10)
select value from v$parameter where name='db_block_size';
accept tab_owner prompt 'Enter Table Owner for Indexes:'
set heading off verify off termout off pages 0 recsep 0 feedback off
spool index_sz.sql
select
     'create table stat_temp as select * from index_stats;'||&&cr||
     'truncate table stat_temp;'
from dual;
select
     'analyze index '||owner||'.'||index_name||
     ' validate structure;'||&&cr||
     'insert into stat_temp select * from index_stats;'||&&cr||
     'commit;'
from dba_indexes
where
     owner=upper('&&tab_owner');
spool off
set feedback on termout on lines 80
start index_sz.sql
insert into temp_size_table select name,trunc(used_space/&&blocksize)
from stat_temp;
rem drop table stat_temp;
clear columns
column del_lf_rows_len format 999,999,999 heading 'Deleted Bytes'
column lf_rows_len format 999,999,999 heading 'Filled Bytes'
column browning format 999.90 heading 'Percent|Browned'
start ttitle "Index Browning Report"
spool rep_out/browning.lst
select
     name,del_lf_rows_len,lf_rows_len,
     (del_lf_rows_len/decode((lf_rows_len+del_lf_rows_len),0,1,
     lf_rows_len+del_lf_rows_len))*100 browning
from
     stat_temp
where
     del_lf_rows_len>0;
spool off
```

Quellcode 6.2: Verfallsbericht für Indices

```
rem ******************************************************
rem NAME: IN_ES_SZ.sql
rem HISTORY:
rem Date           Who                     What
rem ---------------- ------------------------------------ --------
--------------------
rem 01/20/93       Michael Brouillette     Creation
rem 09/22/01       Michael Ault            Upgraded to 9i
rem FUNCTION:   Compute the space used by an entry for an
rem             existing index.
rem NOTES:      Currently requires DBA.
rem INPUTS:
rem        tname  = Name of table.
rem        towner = Name of owner of table.
rem        clist  = List of columns enclosed in quotes.
rem                  i.e., 'ename', 'empno'
rem        cfile  = Name of output SQL Script file
rem ******************************************************
COLUMN name     NEW_VALUE     db NOPRINT
COLUMN dum1     NOPRINT
COLUMN isize    FORMAT 99,999.99
COLUMN rcount   FORMAT 999,999,999 NEWLINE
ACCEPT tname  PROMPT 'Enter table name: '
ACCEPT towner PROMPT 'Enter table owner name: '
ACCEPT clist  PROMPT 'Enter column list: '
ACCEPT cfile  PROMPT 'Enter name for output SQL file: '
SET HEADING OFF VERIFY OFF TERMOUT OFF PAGES 0 EMBEDDED ON
SET FEEDBACK OFF SQLCASE UPPER TRIMSPOOL ON SQLBL OFF
SET NEWPAGE 3
SELECT name FROM v$database;
SPOOL rep_out/&db/&cfile..sql
SELECT -1 dum1,
       'SELECT ''Proposed Index on table ''||'
  FROM dual
UNION
SELECT 0,
       '''&towner..&tname'||' has '',COUNT(*) rcount,
       '' entries of '', ('
  FROM dual
UNION
SELECT column_id,
       'SUM(NVL(vsize('||column_name||'),0)) + 1 +'
  FROM dba_tab_columns
 WHERE table_name = '&tname'
   AND owner = '&towner'
   AND column_name in (upper(&clist))
   AND column_id <> (SELECT MAX(column_id)
                       FROM dba_tab_columns
                      WHERE table_name = UPPER('&tname')
                        AND owner = UPPER('&towner')
                        AND column_name IN (upper(&clist)))
UNION
SELECT column_id,
       'SUM(NVL(VSIZE('||column_name||'),0)) + 1)'
  FROM dba_tab_columns
 WHERE table_name = upper('&tname')
   AND owner = upper('&towner') AND column_name IN (upper(&clist))
   AND column_id = (SELECT MAX(column_id)
                      FROM dba_tab_columns
                     WHERE table_name = upper('&tname')
                       AND owner = upper('&towner')
                       AND column_name IN (upper(&clist)))
UNION
SELECT 997, '/ COUNT(*) + 11 isize, '' bytes each.'''
  FROM dual
```

```
   UNION
   SELECT 999, 'FROM &towner..&tname.;' FROM dual;
   SPOOL OFF
   SET TERMOUT ON FEEDBACK 15 PAGESIZE 20 SQLCASE MIXED
   SET NEWPAGE 1
   START rep_out/&db/&cfile
   CLEAR COLUMNS
```
Quellcode 6.3: Skript zur Berechnung des erforderlichen Platzes für einen geplanten Index

```
rem *******************************************************
rem
rem NAME: IN_CM_SZ.sql
rem
rem HISTORY:
rem Date             Who           What
rem ---------------- ------------- ----------------
rem 01/20/93  Michael Brouillette  Creation
rem 09/22/01  Mike Ault            Updated to 9i
rem
rem FUNCTION: Compute the space used by an entry for an
rem           existing index.
Rem
rem NOTES:    Currently requires DBA.
Rem
rem INPUTS:
rem        tname  = Name of table.
rem        towner = Name of owner of table.
rem        iname  = Name of index.
rem        iowner = Name of owner of index.
rem        cfile  = Name of output file SQL Script.
rem *******************************************************
COLUMN dum1       NOPRINT
COLUMN isize      FORMAT 999,999,999.99
COLUMN rcount     FORMAT 999,999,999 NEWLINE
ACCEPT tname  PROMPT 'Enter table name: '
ACCEPT towner PROMPT 'Enter table owner name: '
ACCEPT iname  PROMPT 'Enter index name: '
ACCEPT iowner PROMPT 'Enter index owner name: '
ACCEPT cfile  PROMPT 'Enter name for output SQL file: '
SET PAGESIZE 0 HEADING OFF VERIFY OFF TERMOUT OFF
SET FEEDBACK OFF TRIMSPOOL ON SQLBL OFF
SET SQLCASE UPPER NEWPAGE 3
SPOOL &cfile..sql
SELECT -1 dum1,
       'SELECT ''Index '||'&iowner..&iname'||' on table '
  FROM dual
UNION
SELECT 0,
       '&towner..&tname'||' has '',
       nvl(COUNT(*),0) rcount,'' entries of '', ('
  FROM dual
UNION
SELECT column_id,
       'SUM(NVL(vsize('||column_name||'),0)) + 1 +'
  FROM dba_tab_columns
 WHERE table_name = '&tname'
   AND owner = upper('&towner') AND column_name IN
                 (SELECT column_name FROM dba_ind_columns
                   WHERE table_name = upper('&tname')
                     AND table_owner = upper('&towner')
                     AND index_name = upper('&iname')
                     AND index_owner = upper('&iowner'))
                     AND column_id <> (select max(column_id)
                                        FROM dba_tab_columns
                                       WHERE table_name = upper('&tname')
                                         AND owner = upper('&towner')
                                         AND column_name IN
                 (SELECT column_name FROM dba_ind_columns
                   WHERE table_name = upper('&tname')
                     AND table_owner = upper('&towner')
                     AND index_name = upper('&iname')
                     AND index_owner = upper('&iowner')))
UNION
SELECT column_id,
       'SUM(NVL(vsize('||column_name||'),0)) + 1)'
  FROM dba_tab_columns
 WHERE table_name = upper('&tname') AND owner = upper('&towner')
   AND column_name IN
                 (SELECT column_name FROM dba_ind_columns
                   WHERE table_name = upper('&tname')
                     AND table_owner = upper('&towner')
                     AND index_name = upper('&iname')
                     AND index_owner = upper('&iowner'))
                     AND column_id = (SELECT MAX(column_id)
                    FROM dba_tab_columns
                   WHERE table_name = upper('&tname')
                     AND owner = upper('&towner')
                     AND column_name IN
                     (SELECT column_name FROM dba_ind_columns
                       WHERE table_name = upper('&tname')
                         AND table_owner = upper('&towner')
                         AND index_name = upper('&iname')
                         AND index_owner = upper('&iowner')))
UNION
SELECT 997,
       '/ COUNT(*) + 11 isize, '' bytes each.''' from dual
UNION
SELECT 999, 'FROM &towner..&tname.;' FROM dual;
SPOOL OFF
SET TERMOUT ON FEEDBACK 15 PAGESIZE 20 SQLCASE MIXED
SET NEWPAGE 1
START &cfile
CLEAR columns
UNDEF tname
UNDEF towner
UNDEF iname
UNDEF iowner
UNDEF cfile
```
Quellcode 6.4: Skript zur Berechnung der durchschnittlichen Länge eines Indexeintrags

```
REM FUNCTION : SCRIPT FOR CREATING SYNONYMS
REM       This script must be run by a user with the DBA role.
REM       This script is intended to run with Oracle7 or Oracle8.
REM       Running this script will in turn create a script to build
REM       all the synonyms in the database. The created script,
REM       create_synonyms.sql, can be run by any user with the DBA
REM       role or with the 'CREATE ANY SYNONYM' and 'CREATE PUBLIC
REM       SYNONYM' system privileges.
REM NOTE: This script does not capture synonyms for tables
REM       owned by the 'SYS' user.
REM       Only preliminary testing of this script was performed.
REM       Be sure to test it completely before relying on it.
REM
SET VERIFY OFF FEEDBACK OFF TERMOUT OFF ECHO OFF PAGESIZE 0
SET TERMOUT ON
SELECT ''Creating synonym build script...'' FROM dual;
SET TERMOUT OFF
COLUMN dbname NEW_VALUE db NOPRINT
SELECT name dbname FROM v$database;
DEFINE cr=''chr(10)''
SPOOL rep_out\&db\crt_syns.sql

SELECT ''CREATE ''|| DECODE(owner,''PUBLIC'',''PUBLIC '',NULL) ||
     'SYNONYM ''|| DECODE(owner,''PUBLIC'',NULL, owner || ''.'') ||
     LOWER(synonym_name) || '' FOR '' || LOWER(table_owner) ||
     ''.'' || LOWER(table_name) ||
     DECODE(db_link,NULL,NULL,''@''||db_link) || '';''
 FROM sys.dba_synonyms
 WHERE table_owner != ''SYS''
 ORDER BY owner
/
SPOOL OFF
SET VERIFY ON FEEDBACK ON TERMOUT ON PAGESIZE 22
CLEAR COLUMNS
UNDEF cr
```
Quellcode 7.1: Skript zur Generierung eines Skripts für die erneute Erstellung von Synonymen

```
CREATE VIEW free_space
(tablespace, file_id, pieces, free_bytes, free_blocks,
largest_bytes,largest_blks) AS
SELECT tablespace_name, file_id, COUNT(*),
     SUM(bytes), SUM(blocks),
     MAX(bytes), MAX(blocks) FROM sys.dba_free_space
 GROUP BY tablespace_name, file_id;
```

Quellcode 7.2: Beispiel für eine Sicht mit Ausdrücken

```
REM Title       : DD_VIEW.SQL
REM Purpose     : View of the Data Dictionary caches
REM               showing only parameters that have usage
REM               and the percent of GETMISSES/GETS
REM USE         : Use as a selectable table only
REM Limitations : User must have access to V$ views.
REM Revisions   :
REM   Date     Modified By   Reason For change
REM   4/28/93  Mike Ault     Initial Creation
REM
CREATE VIEW dd_cache
AS SELECT parameter,gets,getmisses,
getmisses/gets*100 percent
,count,usage
FROM v$rowcache
WHERE gets > 100 AND getmisses > 0;
```

Quellcode 7.3: Beispiel für eine Sicht mit Ausdrücken und Filterregel

```
REM
REM NAME        : view_rct.sql
REM FUNCTION    : re-create database views by owner
REM USE         : Generate a report on database views
REM Limitations : If your view definitions are greater than 5000
REM               characters then increase the set long. This can
REM               be determined by querying the DBA_VIEWS table’s
REM               text_length column for the max value: select
REM               max(text_length) from dba_views;
REM
SET PAGES 59 LINES 79 FEEDBACK OFF ECHO OFF VERIFY
OFF
DEFINE cr=’chr(10)’
COLUMN text       FORMAT a80 word_wrapped
COLUMN view_name  FORMAT a20
COLUMN dbname NEW_VALUE db NOPRINT
UNDEF owner_name
UNDEF view_name
SELECT name dbname from v$database;
SET LONG 5000 HEADING OFF
SPOOL rep_out\&db\cre_view.sql
SELECT
     'rem Code for view: '||v.view_name||'instance: '||&&db||&&cr||
     'CREATE OR REPLACE VIEW '||v.owner||'.'||v.view_name||' AS '
     ||&&cr,
     v.text
 FROM
     dba_views v
 WHERE
     v.owner LIKE UPPER('&&owner_name%’)
     AND view_name LIKE UPPER('%&&view_name%’)
 ORDER BY
     v.view_name;
SPOOL OFF
SET HEADING ON PAGES 22 LINES 80 FEEDBACK ON
CLEAR COLUMNS
TTITLE OFF
PAUSE Press enter to continue
```

Quellcode 7.4: Skript zur Neuerstellung von Sichten

```
CREATE SNAPSHOT new_drugs
PCTFREE 10 PCTUSED 70
TABLESPACE clinical_tests
STORAGE (INITIAL 50K NEXT 50K PCTINCREASE 0)
REFRESH
START WITH ROUND(SYSDATE + 7) + 2/24
NEXT NEXT_DAY(TRUNC(SYSDATE, 'TUESDAY') + 2/24
AS SELECT * FROM appl_dba.test_drugs@kcgc
```

Quellcode 7.5: Beispielskript für den `CREATE SNAPSHOT`-Befehl mit einem einfachen Snapshot

```
CREATE SNAPSHOT trial_summary
PCTFREE 5 PCTUSED 60
TABLESPACE clinical_tests
STORAGE (INITIAL 100K NEXT 50K PCTINCREASE 0)
REFRESH COMPLETE
START WITH ROUND(SYSDATE + 14) + 6/24
NEXT NEXT_DAY(TRUNC(SYSDATE, 'FRIDAY’) + 19/24
AS
SELECT
     td.drug_name, s.trial_number, dr.doctor_id,
     s.comment_line,s.comment
 FROM
     appl_dba.test_drugs@kcgc td,
     appl_dba.trial_doctors@kcgc dr,
     appl_dba.trial_summaries@kcgc s
 WHERE
     td.drug_id = s.drug_id and
     s.trial_id = dr.trial_id and
     s.doctor_id = dr.doctor_id;
```
Quellcode 7.6: Beispielskript für den `CREATE SNAPSHOT`*-Befehl mit einem komplexen Snapshot*

```
rem Name    : inv_obj.sql
rem Purpose : Show all invalid objects in database
rem Mike Ault 7/2/96 TUSC
rem
COLUMN object_name  FORMAT A30 HEADING 'Object|Name'
COLUMN owner        FORMAT a10 HEADING 'Object|Owner'
COLUMN last_time    FORMAT a20 HEADING 'Last Change|Date'
SET LINES 80 FEEDBACK OFF PAGES 0 VERIFY OFF
START title80 'Invalid Database Objects'
SPOOL rep_out/&db/inv_obj
SELECT
     owner,
     object_name,
     object_type,
     TO_CHAR(last_ddl_time,'DD-MON-YY hh:mi:ss') Last_time
FROM
     dba_objects
WHERE
     status='INVALID'
/
PAUSE Press Enter to continue
SET LINES 80 FEEDBACK ON PAGES 22 VERIFY ON
CLEAR COLUMNS
TTITLE OFF
```
Quellcode 7.7: Beispielskript zur Überprüfung des Status von Datenbankobjekten

```
rem *****************************************************
rem NAME     : FPRC_RCT.sql
rem FUNCTION : Build a script to re-create functions,
rem            procedures, packages, or package bodies.
rem HISTORY  :
rem Date      Who           What
rem --------  ------------  ------------------------
rem 05/22/93  Michael Ault  Created
rem *******************************************************
SET VERIFY OFF FEEDBACK OFF LINES 80 PAGES 0 HEADING OFF
SPOOL cre_fprc.sql
SELECT 'CREATE '||s1.type||' '||s1.owner||'.'||s1.name,
substr(s2.text,1,80)||';'
 FROM
     dba_source s1,
     dba_source s2
 WHERE
     s1.type = UPPER('&object_type') AND
     s1.owner = UPPER('&object_owner') AND
     s1.type = s2.type AND
     s1.owner = s2.owner AND
     s1.name = UPPER('&object_name') AND
     s1.name = s2.name
 GROUP BY
     s1.owner,
     s1.name
 ORDER BY
     s2.line;
rem
SPOOL OFF
```
Quellcode 7.8: Beispielskript zur erneuten Erstellung von Funktionen, Prozeduren und Paketobjekten

```
CREATE OR REPLACE PROCEDURE BODY admin.employee_package AS
FUNCTION new_emp(ename CHAR, position CHAR, supervisor NUM,
category NUM, hiredate DATE)
RETURN NUMBER IS
emp_number number(5);
BEGIN
     .
     .
     .
END;
FUNCTION fire_them(
ename CHAR,reason VARCHAR2,term_date DATE)
RETURN NUMBER AS
years_of_service NUMBER (4,2);
BEGIN
     .
     .
     .
END;
PROCEDURE new_dept(ename CHAR, dept CHAR, new_dept CHAR,
date_of_change DATE)
IS
BEGIN
     .
     .
     .
END;
END employee_package
```

Quellcode 7.9: Beispielformat für einen Paketrumpf

```
set echo on
spool test_resource_plan.doc
-- Grant system privilege to plan administrator
--
execute
dbms_resource_manager_privs.grant_system_privilege('SYSTEM','ADMINISTER_RESOURCE_
MANAGER',TRUE);
--
--connect to plan administrator
--
CONNECT system/system_test@ortest1.world
--
-- Create Plan Pending Area
--
EXECUTE dbms_resource_manager.create_pending_area();
--
-- Create plan
--
execute dbms_resource_manager.create_plan('MASTER','Example Resource
Plan','EMPHASIS');
execute dbms_resource_manager.create_plan('USERS','Example Resource Sub
Plan','EMPHASIS');
execute dbms_resource_manager.create_plan('REPORTS','Example Resource Sub
Plan','EMPHASIS');
--
--Create tiers of groups in plan
--
EXECUTE dbms_resource_manager.create_consumer_group('ONLINE_USERS','3rd level
group','ROUND-ROBIN');
EXECUTE dbms_resource_manager.create_consumer_group('BATCH_USERS','3rd level
group','ROUND-ROBIN');
EXECUTE dbms_resource_manager.create_consumer_group('ONLINE_REPORTS','2nd
level group','ROUND-ROBIN');
EXECUTE dbms_resource_manager.create_consumer_group('BATCH_REPORTS','2nd
level group','ROUND-ROBIN');
--
-- Create plan directives
--
EXECUTE dbms_resource_manager.create_plan_directive('MASTER', 'USERS',
0,60,0,0,0,0,0,0,NULL);
EXECUTE dbms_resource_manager.create_plan_directive('MASTER', 'REPORTS',
0,20,0,0,0,0,0,0,NULL);
EXECUTE dbms_resource_manager.create_plan_directive('MASTER','OTHER_GROUPS',
0,20,0,0,0,0,0,0,NULL);
EXECUTE dbms_resource_manager.create_plan_directive('USERS', 'ONLINE_USERS',
0,0,70,0,0,0,0,0,NULL);
EXECUTE dbms_resource_manager.create_plan_directive('USERS', 'BATCH_USERS',
0,0,30,0,0,0,0,0,NULL);
EXECUTE
dbms_resource_manager.create_plan_directive('REPORTS','ONLINE_REPORTS',0,0,70
,0,0,0,0,0,NULL);
EXECUTE
dbms_resource_manager.create_plan_directive('REPORTS','BATCH_REPORTS',
0,0,30,0,0,0,0,0,NULL);
--
-- Verify Plan
--
EXECUTE dbms_resource_manager.validate_pending_area;
--
-- Submit Plan
--
EXECUTE dbms_resource_manager.submit_pending_area;
spool off
set echo off
```

Quellcode 9.1: Skript zum Erstellen eines beispielhaften Ressourcenplans

```
SQL> -- Grant system privilege to plan administrator
SQL> --
SQL> execute
dbms_resource_manager_privs.grant_system_privilege('SYSTEM','ADMINISTER_RESOU
RCE_MANAGER',TRUE);

PL/SQL procedure successfully completed.

SQL> --
SQL> --connect to plan administrator
SQL> --
SQL> CONNECT system/system_test@ortest1.world

Connected.

SQL> --
SQL> -- Create Plan Pending Area
SQL> --
SQL> EXECUTE dbms_resource_manager.create_pending_area();

PL/SQL procedure successfully completed.

SQL> --
SQL> -- Create plan
SQL> --
SQL> execute dbms_resource_manager.create_plan('MASTER','Example Resource
Plan','EMPHASIS');

PL/SQL procedure successfully completed.

SQL> execute dbms_resource_manager.create_plan('USERS','Example Resource Sub
Plan','EMPHASIS');

PL/SQL procedure successfully completed.

SQL> execute dbms_resource_manager.create_plan('REPORTS','Example Resource
Sub Plan','EMPHASIS');

PL/SQL procedure successfully completed.

SQL> --
SQL> --Create tiers of groups in plan
SQL> --
SQL> EXECUTE dbms_resource_manager.create_consumer_group('ONLINE_USERS','3rd
level group','ROUND-ROBIN');

PL/SQL procedure successfully completed.

SQL> EXECUTE dbms_resource_manager.create_consumer_group('BATCH_USERS','3rd
level group','ROUND-ROBIN');

PL/SQL procedure successfully completed.

SQL> EXECUTE
dbms_resource_manager.create_consumer_group('ONLINE_REPORTS','2nd level
group','ROUND-ROBIN');

PL/SQL procedure successfully completed.

SQL> EXECUTE dbms_resource_manager.create_consumer_group('BATCH_REPORTS','2nd
level group','ROUND-ROBIN');

PL/SQL procedure successfully completed.

SQL> --
SQL> -- Create plan directives
SQL> --
SQL> EXECUTE dbms_resource_manager.create_plan_directive('MASTER', 'USERS',
0,60,0,0,0,0,0,0,NULL);

PL/SQL procedure successfully completed.

SQL> EXECUTE dbms_resource_manager.create_plan_directive('MASTER', 'REPORTS',
0,20,0,0,0,0,0,0,NULL);

PL/SQL procedure successfully completed.

SQL> EXECUTE
dbms_resource_manager.create_plan_directive('MASTER','OTHER_GROUPS',
0,20,0,0,0,0,0,0,NULL);

PL/SQL procedure successfully completed.

SQL> EXECUTE dbms_resource_manager.create_plan_directive('USERS',
'ONLINE_USERS', 0,0,70,0,0,0,0,0,NULL);

PL/SQL procedure successfully completed.

SQL> EXECUTE dbms_resource_manager.create_plan_directive('USERS',
'BATCH_USERS', 0,0,30,0,0,0,0,0,NULL);

PL/SQL procedure successfully completed.

SQL> EXECUTE
dbms_resource_manager.create_plan_directive('REPORTS','ONLINE_REPORTS',0,0,70
,0,0,0,0,0,NULL);

PL/SQL procedure successfully completed.

SQL> EXECUTE
dbms_resource_manager.create_plan_directive('REPORTS','BATCH_REPORTS',
0,0,30,0,0,0,0,0,NULL);

PL/SQL procedure successfully completed.

SQL> --
SQL> -- Verify Plan
SQL> --
SQL> EXECUTE dbms_resource_manager.validate_pending_area;

PL/SQL procedure successfully completed.

SQL> --
SQL> -- Submit Plan
SQL> --
SQL> EXECUTE dbms_resource_manager.submit_pending_area;

PL/SQL procedure successfully completed.

SQL> spool off
```

Listing 9.2: Beispiel für die Ausführung des Skripts zum Erstellen eines Ressourcenplans

```
EXECUTE dbms_resource_manager.delete_plan('MASTER');
EXECUTE dbms_resource_manager.delete_plan('USERS');
EXECUTE dbms_resource_manager.delete_plan('REPORTS');
--
--delete tiers of groups in plan
--
EXECUTE dbms_resource_manager.delete_consumer_group('ONLINE_USERS');
EXECUTE dbms_resource_manager.delete_consumer_group('BATCH_USERS');
EXECUTE dbms_resource_manager.delete_consumer_group('ONLINE_REPORTS');
EXECUTE dbms_resource_manager.delete_consumer_group('BATCH_REPORTS');
```

Quellcode 9.2: Beispiel für eine Prozedur zum Löschen eines Ressourcenplans

```
CREATE OR REPLACE PACKAGE graphics_app AUTHID DEFINER AS
PROCEDURE get_graphics_function(usern IN VARCHAR2, graphics_function OUT
VARCHAR2);
PROCEDURE set_graphics_context(usern IN VARCHAR2);
END;
/
SET ARRAYSIZE 1
SHOW ERR
CREATE OR REPLACE PACKAGE BODY graphics_app AS
graphics_user VARCHAR2(32);
graphics_function VARCHAR2(32);
PROCEDURE get_graphics_function(usern IN VARCHAR2, graphics_function OUT
VARCHAR2) IS
BEGIN
SELECT user_function INTO graphics_function FROM graphics_dba.graphics_users
WHERE username=usern;
END get_graphics_function;
PROCEDURE set_graphics_context(usern IN VARCHAR2) IS
BEGIN
graphics_app.get_graphics_function(usern,graphics_function);
DBMS_SESSION.SET_CONTEXT('GRAPHICS_SEC','GRAPHICS_FUNCTION',graphics_function
);
DBMS_SESSION.SET_CONTEXT('GRAPHICS_SEC','GRAPHICS_USER',usern);
END set_graphics_context;
END graphics_app;
/
SHOW ERR
```
Quellcode 9.3: Beispiel für ein Paket zum Setzen des Kontextes

```
CREATE OR REPLACE TRIGGER set_graphics_context AFTER LOGON ON DATABASE
DECLARE
username VARCHAR2(30);
BEGIN
username:=SYS_CONTEXT('USERENV','SESSION_USER');
graphics_app.set_graphics_context(username);
EXCEPTION
WHEN OTHERS THEN
NULL;
END;
/
```

Quellcode 9.4: Beispiel für einen Anmeldungstrigger für die Datenbank

```
CREATE OR REPLACE PACKAGE graphics_sec AUTHID DEFINER AS
FUNCTION graphics_check(obj_schema VARCHAR2, obj_name VARCHAR2)
 RETURN VARCHAR2;
PRAGMA RESTRICT_REFERENCES(GRAPHICS_CHECK,WNDS);
END;
/
SET ARRAYSIZE 1
SHOW ERR
CREATE OR REPLACE PACKAGE BODY graphics_sec AS
FUNCTION graphics_check(obj_schema VARCHAR2, obj_name VARCHAR2)
 RETURN VARCHAR2 AS
d_predicate VARCHAR2(2000);
user_context VARCHAR2(32);
BEGIN
  user_context:=SYS_CONTEXT('graphics_sec','graphics_function');
  IF user_context = 'ADMIN' THEN
    d_predicate:=' 1=1';
dbms_output.put_line(d_predicate);
  ELSIF user_context = 'GENERAL USER' THEN
    d_predicate:=' graphics_usage='||chr(39)||'UNRESTRICTED'||chr(39);
dbms_output.put_line(d_predicate);
  ELSIF user_context='DEVELOPER' THEN
    d_predicate:=' 1=1';
dbms_output.put_line(d_predicate);
  ELSIF user_context IS NULL THEN
    d_predicate:='1=2';
  END IF;
  RETURN d_predicate;
END graphics_check;
END;
/
SHOW ERR
```

Quellcode 9.5: Beispiel für ein Paket zur Kontextprüfung

```
REM
REM NAME        : TABLE.SQL
REM FUNCTION    : GENERATE TABLE REPORT
REM Limitations : None
clear COLUMNs
COLUMN owner             FORMAT a15    HEADING 'Table | Owner'
COLUMN table_name        FORMAT a18    HEADING Table
COLUMN tablespace_name   FORMAT A13    HEADING Tablespace
COLUMN pct_increase                    HEADING 'Pct|Increase'
COLUMN init                            HEADING 'Initial|Extent'
COLUMN next                            HEADING 'Next|Extent'
COLUMN partitioned       FORMAT a4     HEADING 'Par?'
COLUMN iot_type          FORMAT a4     HEADING 'Iot?'
COLUMN nested            FORMAT a5     HEADING 'Nest?'
COLUMN temporary         FORMAT a5     HEADING 'Temp?'
COLUMN extern            FORMAT a8     Heading 'External?'
BREAK ON owner ON tablespace_name
SET PAGES 48 LINES 132
START TITLE132 "ORACLE TABLE REPORT"
SPOOL rep_out\&db\tab_rep
SELECT
     owner,
     tablespace_name,
     table_name,
     initial_extent Init,
     next_extent Next,
     pct_increase,
     partitioned,
     DECODE(iot_type,NULL,'No','Yes') iot_type,
     nested,
     DECODE(temporary,'N','No','Yes') temporary,
        DECODE(initial_extent, null,
              DECODE(iot_type,null,
              DECODE(temporary,'N','Yes')),'No') extern
FROM
     sys.dba_tables
WHERE
     owner NOT IN (
'SYSTEM','SYS','DBSNMP','AURORA$JIS$UTILITY$',
'AURORA$ORB$UNAUTHENTICATED','SCOTT','OSE$HTTP$ADMIN',
'OUTLN','LBACSYS','OE','QS','QS_CS','QS_CB','QS_CBADM',
'QS_OS','QS_ES','QS_WS','QS_ADM','SH','HR','WKSYS','ORDSYS',
'ORDPLUGINS','CTXSYS','MDSYS','PM')
ORDER BY
     owner,
     tablespace_name,
     table_name;
SPOOL OFF
CLEAR COLUMNS
PAUSE Press enter to continue
SET PAGES 22 LINES 80
TTITLE OFF
CLEAR COLUMNS
CLEAR BREAKS
```
Quellcode 10.1: Skript für einen Tabellenbericht

```
REM
REM NAME        : EXTENTS.SQL
REM FUNCTION    : GENERATE EXTENTS REPORT
REM USE         : FROM SQLPLUS OR OTHER FRONT END
REM LIMITATIONS : NONE
REM
CLEAR COLUMNS
COLUMN segment_name    HEADING 'Segment'     FORMAT A15
COLUMN tablespace_name HEADING 'Tablespace'  FORMAT A10
COLUMN owner           HEADING 'Owner'       FORMAT A10
COLUMN segment_type    HEADING 'Type'        FORMAT A10
COLUMN size            HEADING 'Size'        FORMAT 999,999,999
COLUMN extents         HEADING 'Current|Extents'
COLUMN max_extents     HEADING 'Max|Extents'
COLUMN bytes           HEADING 'Size|(Bytes)'
SET PAGESIZE 58 NEWPAGE 0 LINESIZE 130 FEEDBACK OFF
SET ECHO OFF VERIFY OFF
ACCEPT extents PROMPT 'Enter max number of extents: '
BREAK ON tablespace_name SKIP PAGE ON owner
START TITLE132 "Extents Report"
DEFINE output = rep_out\&db\extent
SPOOL &output
SELECT tablespace_name,
     segment_name,
     extents,
     max_extents,
     bytes,
     owner "owner",
     segment_type
FROM  dba_segments
WHERE extents >= &extents AND owner LIKE UPPER('%&owner%')
ORDER BY tablespace_name,owner,segment_type,segment_name;
SPOOL OFF
CLEAR COLUMNS
CLEAR BREAKS
SET TERMOUT ON FEEDBACK ON VERIFY ON
UNDEF extents
UNDEF owner
TTITLE OFF
UNDEF OUTPUT
PAUSE Press enter to continue
```

*Quellcode 10.2: SQL*Plus-Bericht zum Anzeigen der Extents für jede Tabelle in jedem Tablespace*

```
rem  *************************************************************
rem
rem  NAME: ACT_SIZE.sql
rem
rem  HISTORY:
rem  Date      Who                  What
rem  --------  -------------------- ---------------------------------
rem  09/??/90  Maurice C. Manton    Creation for IOUG
rem  12/23/92  Michael Brouillette  Assume TEMP_SIZE_TABLE exists.
rem                                 Use DBA info. Prompt for user
rem                                 name. Spool file = owner.
rem  07/15/96  Mike Ault        Updated for Oracle 7.x, added indexes
rem  06/12/97  Mike Ault        Updated for Oracle 8.x (use DBMS_ROWID)
rem  FUNCTION: Will show actual blocks used vs allocated for all tables
rem            for a user
rem  INPUTS:   owner = Table owner name.
rem  ************************************************************
ACCEPT owner PROMPT 'Enter table owner name: '
SET HEADING OFF FEEDBACK OFF VERIFY OFF ECHO OFF RECSEP OFF PAGES 0
COLUMN db_block_size NEW_VALUE blocksize NOPRINT
TTITLE OFF
DEFINE cr='chr(10)'
DEFINE qt='chr(39)'
TRUNCATE TABLE temp_size_table;
SELECT value db_block_size FROM v$parameter WHERE name='db_block_size';
SPOOL fill_sz.sql
SELECT
     'INSERT INTO temp_size_table'||&&cr||
     'SELECT '||&&qt||segment_name||&&qt||&&cr||
     ',COUNT(DISTINCT(dbms_rowid.rowid_block_number(rowid))) blocks'||&&cr||
     'FROM &&owner..'||segment_name, ';'
 FROM
     dba_segments
 WHERE
     segment_type ='TABLE'
     AND owner = UPPER('&owner');
SPOOL OFF
SPOOL index_sz.sql
SELECT
     'CREATE TABLE stat_temp AS SELECT * FROM index_stats;'||&&cr||
     'TRUNCATE TABLE stat_temp;'
 FROM
     dual;
SELECT
     'ANALYZE INDEX '||owner||'.'||index_name||' VALIDATE STRUCTURE;'||&&cr||
     'INSERT INTO stat_temp SELECT * FROM index_stats;'||&&cr||
     'COMMIT;'
 FROM
     dba_indexes
 WHERE
     owner=UPPER('&owner');
SPOOL OFF
SET FEEDBACK ON TERMOUT ON LINES 132
START index_sz.sql
INSERT INTO temp_size_table SELECT name,trunc(used_space/&&blocksize)
FROM stat_temp;
DROP TABLE stat_temp;
DEFINE temp_var = &&qt;
START fill_sz
HOST rm fill_size_table.sql
DEFINE bs = '&&blocksize K'
COLUMN t_date      NOPRINT NEW_VALUE t_date
COLUMN user_id     NOPRINT NEW_VALUE user_id
COLUMN segment_name      FORMAT A25         HEADING "SEGMENT|NAME"
COLUMN segment_type      FORMAT A7          HEADING "SEGMENT|TYPE"
COLUMN extents          FORMAT 999         HEADING "EXTENTS"
COLUMN kbytes           FORMAT 999,999,999 HEADING "KILOBYTES"
COLUMN blocks           FORMAT 9,999,999   HEADING "ALLOC.|&&bs|BLOCKS"
COLUMN act_blocks       FORMAT 9,999,990   HEADING "USED|&&bs|BLOCKS"
COLUMN pct_block        FORMAT 999.99      HEADING "PCT|BLOCKS|USED"
START title132 "Actual Size Report for &owner"
SET PAGES 55
BREAK ON REPORT ON segment_type SKIP 1
COMPUTE SUM OF kbytes ON segment_type REPORT
SPOOL rep_out\&db\&owner
SELECT
     segment_name,
     segment_type,
     SUM(extents) extents,
     SUM(bytes)/1024 kbytes,
     SUM(a.blocks) blocks,
     NVL(MAX(b.blocks),0) act_blocks,
    (MAX(b.blocks)/SUM(a.blocks))*100 pct_block
 FROM
     sys.dba_segments a,
     temp_size_table b
 WHERE
     segment_name = UPPER( b.table_name )
 GROUP BY
     segment_name,
     segment_type
 ORDER BY
     segment_type,
     segment_name;
SPOOL OFF
TRUNCATE TABLE temp_size_table;
SET TERMOUT ON FEEDBACK 15 VERIFY ON PAGESIZE 20 LINESIZE 80 SPACE 1
UNDEF qt
UNDEF cr
TTITLE OFF
CLEAR COLUMNS
CLEAR COMPUTES
PAUSE press enter to continue
```
Quellcode 10.3: Berichtsskript für die tatsächliche Größe

```
rem
rem Create temp_size_table for use by actsize.sql
rem
CREATE TABLE temp_size_table (
     table_name VARCHAR2(64),
     blocks NUMBER);
```

Quellcode 10.4: Skript zum Erstellen der Tabelle `TEMP_SIZE_TABLE`

```
rem
rem  NAME: tab_stat.sql
rem
rem  FUNCTION: Show table statistics for user's tables or all tables.
rem  10/08/01 Updated for 9i Mike Ault
rem
SET PAGES 56 LINES 132 NEWPAGE 0 VERIFY OFF ECHO OFF FEEDBACK OFF
rem
COLUMN owner            FORMAT a12          HEADING "Table Owner"
COLUMN table_name       FORMAT a20          HEADING "Table"
COLUMN tablespace_name  FORMAT a20          HEADING "Tablespace"
COLUMN num_rows         FORMAT 999,999,999  HEADING "Rows"
COLUMN blocks           FORMAT 999,999      HEADING "Blocks"
COLUMN empty_blocks     FORMAT 999,999      HEADING "Empties"
COLUMN space_full       FORMAT 999.99       HEADING “% Full"
COLUMN chain_cnt        FORMAT 999,999      HEADING "Chains"
COLUMN avg_row_len      FORMAT 99,999,999   HEADING "Avg Length|(Bytes)"
rem
START title132 "Table Statistics Report"
DEFINE OUTPUT = 'rep_out\&db\tab_stat..lis'
SPOOL &output
rem
BREAK ON OWNER SKIP 2 ON TABLESPACE_NAME SKIP 1;
SELECT owner, table_name, tablespace_name, num_rows, blocks,
     empty_blocks,
     100*((num_rows *
     avg_row_len)/((GREATEST(blocks,1)+empty_blocks)*value))
     space_full,
     chain_cnt, avg_row_len
 FROM dba_tables, v$parameter
 WHERE OWNER NOT IN ('SYS','SYSTEM')
     AND num_rows>0
     AND name='db_block_size'
 ORDER BY owner, tablespace_name;
SPOOL OFF
PAUSE Press enter to continue
SET PAGES 22 LINES 80 NEWPAGE 1 VERIFY ON FEEDBACK ON
CLEAR COLUMNS
CLEAR BREAKS
TTITLE OFF
```
Quellcode 10.5: Berichtsskript für zusätzliche Tabellenangaben

```
REM
REM  Name     : tab_rep.sql
REM  FUNCTION : Document table extended parameters
REM  Use      : From SQLPLUS
REM  MRA  6/13/97 Created for ORACLE8
REM  MRA  5/08/99 Updated for ORACLE8i
REM  MRA 10/08/01 Updated for Oracle9i
REM
COLUMN owner            FORMAT a10 HEADING 'Owner'
COLUMN table_name       FORMAT a15 HEADING 'Table'
COLUMN tablespace_name  FORMAT a13 HEADING 'Tablespace'
COLUMN table_type_owner FORMAT a10 HEADING 'Type|Owner'
COLUMN table_type       FORMAT a13 HEADING 'Type'
COLUMN iot_name         FORMAT a10 HEADING 'IOT|Overflow'
COLUMN iot_type         FORMAT a12 HEADING 'IOT or|Overflow'
COLUMN nested           FORMAT a6  HEADING 'Nested'
COLUMN extern           FORMAT a3  HEADING 'Ext'
UNDEF owner
SET LINES 130 VERIFY OFF FEEDBACK OFF PAGES 58
START title132 'Extended Table Report'
SPOOL rep_out\&&db\ext_tab.lis
SELECT
     owner,
     table_name,
     tablespace_name,
     iot_name,
     logging,
     partitioned,
     iot_type,
     'N/A' table_type_owner,
     'N/A' table_type,
     DECODE(temporary,'N','No',temporary),
     nested,
     'N/A' extern
 FROM
     dba_tables
 WHERE
     owner LIKE UPPER('%&&owner%')
UNION
SELECT
     owner,
     table_name,
     tablespace_name,
     iot_name,
     logging,
     partitioned,
     iot_type,
     table_type_owner,
     table_type,
     DECODE(temporary,'N','No',temporary),
     nested,
     'N/A' extern
 FROM
     dba_object_tables
 WHERE
     owner LIKE UPPER('%&&owner%')
UNION
SELECT
     Owner,
     'None' tablespace_name,
     'N/A' Iot_name,
     'N/A' logging,
     'N/A' partitioned,
     'N/A' Iot_type,
     type_owner table_type_owner,
     type_name table_type,
     'N/A' temporary,
     'N/A' nested,
     'Yes' extern
 FROM
     dba_external_tables
 WHERE
     Owner LIKE UPPER('%&&owner%');
SPOOL OFF
SET VERIFY ON LINES 80 PAGES 22 FEEDBACK ON
TTITLE OFF
UNDEF OWNER
CLEAR COLUMNS
```
Quellcode 10.6: Skript zur Dokumentation der erweiterten Tabellenparameter

```
rem
rem  NAME     : tab_stat.sql
rem
rem  FUNCTION :Show table statistics for a user's tables or all tables.
Rem
 set pages 56 lines 130 newpage 0 verify off echo off feedback off
rem
COLUMN owner                FORMAT a12          HEADING "Table Owner"
COLUMN table_name           FORMAT a17          HEADING "Table"
COLUMN tablespace_name      FORMAT a13          HEADING "Tablespace"
COLUMN num_rows             FORMAT 99,999,999   HEADING "Rows"
COLUMN blocks               FORMAT 99,999       HEADING "Blocks"
COLUMN empty_blocks         FORMAT 99,999       HEADING "Empties"
COLUMN space_full           FORMAT 999.99       HEADING "% Full"
COLUMN chain_cnt            FORMAT 99,999       HEADING "Chains"
COLUMN avg_row_len          FORMAT 9,999,999    HEADING "Avg|Length|(Bytes)"
COLUMN num_freelist_blocks FORMAT 99,999       HEADING "Num|Freelist|Blocks"
COLUMN avg_space_freelist_blocks FORMAT 99,999 HEADING "Avg|Space|Freelist
Blocks"
rem
START title132 "Table Statistics Report"
DEFINE OUTPUT = 'rep_out\&db\tab_stat..lis'
SPOOL &output
rem
BREAK ON OWNER SKIP 2 ON TABLESPACE_NAME SKIP 1;
SELECT
     owner, table_name, tablespace_name,
     num_rows, blocks,empty_blocks,
     100*((num_rows * avg_row_len)/((GREATEST(blocks,1) + empty_blocks)
     * 2048)) space_full,
     chain_cnt, avg_row_len,avg_space_freelist_blocks,
     num_freelist_blocks
 FROM
     dba_tables
 WHERE
     owner NOT IN ('SYS','SYSTEM')
UNION
SELECT
     owner, table_name, tablespace_name,
     num_rows, blocks,empty_blocks,
     100*((num_rows * avg_row_len)/((GREATEST(blocks,1) + empty_blocks)
     * 2048)) space_full,
     chain_cnt, avg_row_len,avg_space_freelist_blocks,
     num_freelist_blocks
 FROM
     dba_object_tables
 WHERE
     owner NOT IN ('SYS','SYSTEM')
 ORDER BY
     owner, tablespace_name;
SPOOL OFF
PAUSE Press enter to continue
SET PAGES 22 LINES 80 NEWPAGE 1 VERIFY ON FEEDBACK ON
CLEAR COLUMNS
CLEAR BREAKS
TTITLE OFF
```

Quellcode 10.7: Berichtsskript für statistische Werte zu Tabellen

```
rem
rem tab_col.sql
rem
rem FUNCTION: Report on Table and View Column Definitions
rem
rem MRA 9/18/96
rem MRA 6/14/97 Added table level selectivity
rem
COLUMN owner             FORMAT a10      HEADING Owner
COLUMN table_name        FORMAT a30      HEADING "Table or View Name"
COLUMN COLUMN_name       FORMAT a32      HEADING "Table or View|Attribute"
COLUMN data_type         FORMAT a15      HEADING "Data|Type"
COLUMN data_type_owner   FORMAT a13      HEADING "Type|Owner"
COLUMN data_length                       HEADING Length
COLUMN nullable          FORMAT a5       HEADING Null
BREAK ON owner ON table_name SKIP 1
SET LINES 132 PAGES 48 FEEDBACK OFF VERIFY OFF
START title132 "Table Columns Report"
SPOOL rep_out/&db/tab_col
SELECT
     a.owner,
     table_name||' '||object_type table_name,
     column_name,
     data_type,
     data_type_owner,
     data_length,
     DECODE(nullable,'N','NO','YES') nullable
 FROM
     dba_tab_columns a, dba_objects b
 WHERE
     a.owner=UPPER('&owner') AND
     a.owner=b.owner AND
     a.table_name LIKE UPPER('%&table%') AND
     a.table_name=b.object_name AND
     object_type IN ('TABLE','VIEW','CLUSTER')
 ORDER BY
     owner,
     object_type,
     table_name,
     column_id
/
SPOOL OFF
TTITLE OFF
SET LINES 80 PAGES 22 FEEDBACK ON VERIFY ON
```

Quellcode 10.8: Skript zur Ausgabe der Tabellenspalten nach Besitzer und Tabelle

```
rem
rem tab_col_stat.sql
rem
rem FUNCTION: Report on Table and View Column Definitions
rem
rem MRA 9/18/96
rem MRA 6/14/97 Added table level selectivity
rem MRA 5/8/99 Converted to do stats
rem
COLUMN owner            FORMAT a12      HEADING Owner
COLUMN table_name       FORMAT a20      HEADING "Table Name"
COLUMN COLUMN_name      FORMAT a13      HEADING "Table|Attribute"
COLUMN data_type        FORMAT a10      HEADING "Data|Type"
COLUMN avg_col_len      FORMAT 99,999   HEADING "Aver|Length"
COLUMN density          FORMAT 9.9999   HEADING "Density"
COLUMN last_analyzed                    HEADING "Analyzed"
COLUMN num_distinct                     HEADING "Distinct|Values"
COLUMN num_nulls                        HEADING "Num.|Nulls"
COLUMN sample_size                      HEADING "Sample|Size"
BREAK ON owner ON table_name SKIP 1
SET LINES 132 PAGES 48 FEEDBACK OFF VERIFY OFF
START title132 "Table Column Stats Report"
SPOOL rep_out/&db/tab_col
SELECT
     owner,table_name,column_name,data_type,
     num_distinct,density,num_nulls,
     TO_CHAR(last_analyzed,'dd-mon-yyyy hh24:mi') last_analyzed,
     sample_size, avg_col_len
 FROM
     dba_tab_columns
 WHERE
     owner LIKE UPPER('%&owner%')
     and table_name LIKE UPPER('%&tabname%')
/
SPOOL OFF
TTITLE OFF
SET LINES 80 PAGES 22 FEEDBACK ON VERIFY ON
```
Quellcode 10.9: Berichtsskript für statistische Werte zu den Tabellenspalten

```
rem  ****************************************************************
rem
rem  NAME     : CHAINING.sql
rem
rem  FUNCTION : Report number of CHAINED rows within a named table
rem
rem  NOTES    : Requires DBA privileges.
rem             Target table must have column that is the leading
rem             portion of an index and is defined as not null.
rem             Uses the V$SESSTAT table. USERNAME is the current
rem             user.
rem             A problem if > 1 session active with that USERID.
rem             V$SESSTAT may change between releases and
rem             platforms. Make sure that 'table fetch continued row'
rem             is a valid statistic.
rem             This routine can be run by AUTO_CHN.sql by remarking
rem             the two accepts and un-remarking the two defines.
Rem
rem  INPUTS   : obj_own = the owner of the table.
rem             obj_nam = the name of the table.
Rem
rem  ****************************************************************
ACCEPT obj_own PROMPT 'Enter the table owner''s name: '
ACCEPT obj_nam PROMPT 'Enter the name of the table: '

rem DEFINE obj_own = &1  fl Remove comment to use with auto_chain
rem DEFINE obj_nam = &2  fl Remove comment to use with auto_chain

SET TERMOUT OFF FEEDBACK OFF VERIFY OFF ECHO OFF HEADING OFF
SET EMBEDDED ON
COLUMN statistic# NEW_VALUE stat_no NOPRINT
SELECT
     statistic#
 FROM
     v$statname
 WHERE
     n.name = 'table fetch continued row'
/

rem Find out who we are in terms of sid
COLUMN sid NEW_VALUE user_sid
SELECT
     distinct sid
 FROM
     v$session
 WHERE
     audsid = USERENV('SESSIONID’)
/

rem Find the last col of the table and a not null indexed column
COLUMN column_name     NEW_VALUE last_col
COLUMN name            NEW_VALUE indexed_column
COLUMN value           NEW_VALUE before_count
SELECT
     column_name
 FROM
     dba_tab_columns
 WHERE
     table_name = upper('&&obj_nam')
     and owner = upper('&&obj_own')
 ORDER BY
     column_id
/
SELECT
     c.name
```

```
    FROM
        sys.col$ c,
        sys.obj$ idx,
        sys.obj$ base,
        sys.icol$ ic
    WHERE
        base.obj#     = c.obj#
           and ic.bo# = base.obj#
           and ic.col# = c.col#
           and base.owner# = (SELECT user# FROM sys.user$
                              WHERE name = UPPER('&&obj_own'))
           and ic.obj# = idx.obj#
           and base.name = UPPER('&&obj_nam')
           and ic.pos# = 1
           and c.null$ > 0
   /
   SELECT value
     FROM v$sesstat
    WHERE v$sesstat.sid = &user_sid
      AND v$sesstat.statistic# = &stat_no
   /
   rem Select every row from the target table
   SELECT &last_col xx
     FROM &obj_own..&obj_nam
    WHERE &indexed_column <= (SELECT MAX(&indexed_column)
                               FROM &obj_own..&obj_nam)
   /
   COLUMN value NEW_VALUE after_count
   SELECT value
     FROM v$sesstat
    WHERE v$sesstat.sid = &user_sid
      AND v$sesstat.statistic# = &stat_no
   /
   SET TERMOUT ON

   SELECT
        'Table '||UPPER('&obj_own')||'.'||UPPER('&obj_nam')||' contains '||
        (TO_NUMBER(&after_count) - TO_NUMBER(&before_count))||
        ' chained row'||
        DECODE(to_NUMBER(&after_count) - TO_NUMBER(&before_count),1,'.','s.')
     FROM dual
    WHERE RTRIM('&indexed_column') IS NOT NULL
   /

   rem If we don't have an indexed column this won't work so say so
   SELECT 'Table '||
        UPPER('&obj_own')||'.'||UPPER('&obj_nam')||
        ' has no indexed, not null columns.'
     FROM dual
    WHERE RTRIM('&indexed_column') IS NULL
   /

   SET TERMOUT ON FEEDBACK 15 VERIFY ON PAGESIZE 20 LINESIZE 80 SPACE 1
   SET HEADING ON
   UNDEF obj_nam
   UNDEF obj_own
   UNDEF before_count
   UNDEF after_count
   UNDEF indexed_column
   UNDEF last_col
   UNDEF stat_no
   UNDEF user_sid
   CLEAR COLUMNS
   CLEAR COMPUTES
```
Quellcode 10.10: Interaktives SQL-Skript zur Ermittlung verketteter Zeilen in einer Tabelle

```
   rem  **********************************************************
```

```
rem
rem  NAME     : AUTO_CHN.sql
rem
rem  FUNCTION : Run CHAINING.sql for all of a users tables.
Rem
rem  NOTES    : Requires mod to CHAINING.sql. See CHAINING.sql header
rem
rem  INPUTS   :
rem             tabown = Name of owner.
Rem
rem ************************************************************
rem
ACCEPT tabown PROMPT 'Enter table owner: '
rem
SET TERMOUT OFF FEEDBACK OFF VERIFY OFF ECHO OFF HEADING OFF PAGES 999
SET EMBEDDED ON
COLUMN name NEW_VALUE db NOPRINT
SELECT name FROM v$database;
SPOOL rep_out\auto_chn.gql
rem
SELECT 'start chaining &tabown '||table_name
  FROM dba_tables
 WHERE owner = UPPER('&tabown')
/

SPOOL OFF
SPOOL rep_out\&db\chaining
START rep_out\auto_chn.gql
SPOOL OFF
UNDEF tabown
SET TERMOUT ON FEEDBACK 15 VERIFY ON PAGESIZE 20 LINESIZE 80 SPACE 1
SET EMBEDDED OFF
HO del rep_out\auto_chn.gql
PAUSE Press enter to continue
```
Quellcode 10.11: Das Skript `auto_chn.sql` *zur Automatisierung der Verkettungsanalyse*

```
rem  *******************************************************************
rem  NAME     : db_tgnts.sql
rem
rem  FUNCTION : Produce report of table or procedure grants showing
rem             GRANTOR, GRANTEE or ROLE and specific GRANTS.
Rem
rem  INPUTS   : Owner name
rem  *******************************************************************
rem
COLUMN grantee         FORMAT A18   HEADING "Grantee|or Role"
COLUMN owner           FORMAT A18   HEADING "Owner"
COLUMN table_name      FORMAT A30   HEADING "Table|or Proc"
COLUMN grantor         FORMAT A18   HEADING "Grantor"
COLUMN privilege       FORMAT A10   HEADING "Privilege"
COLUMN grantable       FORMAT A19   HEADING "Grant|Option?"
rem
BREAK ON owner SKIP 4 ON table_name SKIP 1 ON grantee ON grantor ON REPORT
rem
SET LINESIZE 130 PAGES 56 VERIFY OFF FEEDBACK OFF
START title132 "TABLE GRANTS BY OWNER AND TABLE"
DEFINE OUTPUT = rep_out/&&db/db_tgnts
SPOOL &output
REM
SELECT
     owner,
     table_name,
     grantee,
     grantor,
     privilege,
     grantable
 FROM
     dba_tab_privs
 WHERE
     owner NOT IN ('SYS','SYSTEM')
 ORDER BY
     owner,
     table_name,
     grantor,
     grantee;
REM
SPOOL OFF
PAUSE Press enter to continue
```

Quellcode 10.12: SQL-Skript zur Anzeige der Zugriffsberechtigungen auf Objektebene

```
rem
rem  Name     : tab_part.sql
rem  Function : Report on partitioned table structure
rem  History  : MRA 6/13/97 Created
rem
COLUMN table_owner        FORMAT a10   HEADING 'Owner'
COLUMN table_name         FORMAT a15   HEADING 'Table'
COLUMN partition_name     FORMAT a15   HEADING 'Partition'
COLUMN tablespace_name    FORMAT a15   HEADING 'Tablespace'
COLUMN high_value         FORMAT a10   HEADING 'Partition|Value'
COLUMN subpartition_count FORMAT 9,999 HEADING 'Sub-Partitions'
SET LINES 130
START title132 'Table Partition Files'
BREAK ON table_owner ON table_name
SPOOL rep_out/&&db/tab_part.lis
SELECT
     table_owner,
     table_name,
     partition_name,
     sub_partition_count,
     high_value,
     tablespace_name,
     logging
 FROM sys.dba_tab_partitions
 ORDER BY table_owner,table_name
/
SPOOL OFF
```
Quellcode 10.13: Berichtsskript für die Strukturen partitionierter Tabellen

```
rem
rem  NAME     : Tab_pstor.sql
rem  FUNCTION : Provide data on part. table stor. charcacteristics
rem  HISTORY  : MRA 6/13/97 Created
rem
COLUMN table_owner        FORMAT a6      HEADING 'Owner'
COLUMN table_name         FORMAT a14     HEADING 'Table'
COLUMN partition_name     FORMAT a9      HEADING 'Partition'
COLUMN tablespace_name    FORMAT a11     HEADING 'Tablespace'
COLUMN pct_free           FORMAT 9999    HEADING '%|Free'
COLUMN pct_used           FORMAT 999     HEADING '%|Use'
COLUMN ini_trans          FORMAT 9999    HEADING 'Init|Tran'
COLUMN max_trans          FORMAT 9999    HEADING 'Max|Tran'
COLUMN initial_extent     FORMAT 9999999 HEADING 'Init|Extent'
COLUMN next_extent        FORMAT 9999999 HEADING 'Next|Extent'
COLUMN max_extent                        HEADING 'Max|Extents'
COLUMN pct_increase       FORMAT 999     HEADING '%|Inc'
COLUMN partition_position FORMAT 9999    HEADING 'Part|Nmbr'
SET LINES 130
START title132 'Table Partition File Storage'
BREAK ON table_owner on table_name
SPOOL rep_out/&&db/tab_pstor.lis
SELECT
     table_owner,
     table_name,
     tablespace_name,
     partition_name,
     partition_position,
     pct_free,
     pct_used,
     ini_trans,
     max_trans,
     initial_extent,
     next_extent,
     max_extent,
     pct_increase
 FROM sys.dba_tab_partitions
 ORDER BY table_owner,table_name
/
SPOOL OFF
```
Quellcode 10.14: Berichtsskript für die Speichereigenschaften einer Partition

```
rem
rem  Name     : tab_part_stat.sql
rem  Function : Report on partitioned table statistics
rem  History  : MRA 6/13/97 Created
rem
COLUMN table_name      FORMAT a15  HEADING 'Table'
COLUMN partition_name  FORMAT a15  HEADING 'Partition'
COLUMN num_rows                    HEADING 'Num|Rows'
COLUMN blocks                      HEADING 'Blocks'
COLUMN avg_space                   HEADING 'Avg|Space'
COLUMN chain_cnt                   HEADING 'Chain|Count'
COLUMN avg_row_len                 HEADING 'Avg|Row|Length'
COLUMN last_analyzed               HEADING 'Analyzed'
ACCEPT owner1 PROMPT 'Which Owner to report on?:'
SET LINES 130
START title132 'Table Partition Statistics For &owner1'
BREAK ON table_owner ON table_name ON partition_name
SPOOL rep_out/&&db/tab_part_stat.lis
SELECT
     table_name,
     partition_name,
     num_rows,
     blocks,
     avg_space,
     chain_cnt,
     avg_row_len,
     to_char(last_analyzed,'dd-mon-yyyy hh24:mi') last_analyzed
 FROM
     sys.dba_tab_partitions
 WHERE
     table_owner LIKE UPPER('%&&owner1%')
 ORDER BY
     table_owner,table_name
/
SPOOL OFF
CLEAR BREAKS
CLEAR COLUMNS
TTITLE OFF
UNDEF owner1
```
Quellcode 10.15: Berichtsskript für statistische Werte zu Partitionen

```
rem
rem  Name     : tab_subpart.sql
rem  Function : Report on partitioned table structure
rem  History  : MRA 6/13/97 Created
rem
COLUMN table_owner NEW_VALUE owner1 NOPRINT
COLUMN table_name      FORMAT a15   HEADING 'Table'
COLUMN partition_name  FORMAT a15   HEADING 'Partition'
COLUMN tablespace_name FORMAT a15   HEADING 'Tablespace'
COLUMN initial_extent  FORMAT 9,999 HEADING 'Initial|Extent (K)'
COLUMN next_extent     FORMAT 9,999 HEADING 'Next|Extent (K)'
COLUMN pct_increase    FORMAT 999   HEADING 'PCT|Increase'
SET LINES 130
START title132 'Table Sub-Partition Files For &owner1'
BREAK ON table_owner ON table_name ON partition_name
SPOOL rep_out/&&db/tab_subpart.lis
SELECT
     table_owner,
     table_name,
     partition_name,
     subpartition_name,
     tablespace_name,
     logging,
     initial_extent/1024 initial_extent,
     next_extent/1024 next_extent,
     pct_increase
FROM sys.dba_tab_subpartitions
ORDER BY table_owner,table_name,partition_name
/
SPOOL OFF
```
Quellcode 10.16: Berichtsskript für Teilpartitionen

```
rem
rem  Name     : tab_subpart_stat.sql
rem  Function : Report on partitioned table structure
rem  History  : MRA 6/13/97 Created
rem
COLUMN table_name         FORMAT a15  HEADING 'Table'
COLUMN partition_name     FORMAT a15  HEADING 'Partition'
COLUMN subpartition_name  FORMAT a15  HEADING 'Sub|Partition'
COLUMN num_rows                       HEADING 'Num|Rows'
COLUMN blocks                         HEADING 'Blocks'
COLUMN avg_space                      HEADING 'Avg|Space'
COLUMN chain_cnt                      HEADING 'Chain|Count'
COLUMN avg_row_len                    HEADING 'Avg|Row|Length'
COLUMN last_analyzed                  HEADING 'Analyzed'
ACCEPT owner1 PROMPT 'Owner to Report On?: '
SET LINES 130
START title132 'Table Sub-Partition Statistics For &owner1'
BREAK ON table_owner ON table_name ON partition_name
SPOOL rep_out/&&db/tab_subpart_stat.lis
SELECT
     table_owner,
     table_name,
     partition_name,
     subpartition_name,
     num_rows,
     blocks,
     avg_space,
     chain_cnt,
     avg_row_len,
     to_char(last_analyzed,'dd-mon-yyyy hh24:mi') last_analyzed
 FROM
     sys.dba_tab_subpartitions
 WHERE
     Table_owner LIKE UPPER('%&&owner1%')
 ORDER BY
     table_owner,table_name,partition_name
/
SPOOL OFF
CLEAR COLUMNS
TTITLE OFF
UNDEF owner1
```

Quellcode 10.17: Berichtsskript für statistische Werte zu Teilpartitionen

```
rem
rem  NAME    : tab_nest.sql
rem  PURPOSE : Report on Nested Tables
rem  HISTORY : MRA 6/14/97 Created
rem                5/08/99 Updated to Oracle8i
rem
COLUMN owner               FORMAT a10 HEADING 'Owner'
COLUMN table_name          FORMAT a15 HEADING 'Store Table'
COLUMN table_type_owner    FORMAT a10 HEADING 'Type|Owner'
COLUMN table_type_name     FORMAT a15 HEADING 'Type|Name'
COLUMN parent_table_name   FORMAT a25 HEADING 'Parent|Table'
COLUMN parent_table_column FORMAT a12 HEADING 'Parent|Column'
COLUMN storage_spec        FORMAT a15 HEADING 'Storage|Spec'
COLUMN return_type         FORMAT a7  HEADING 'Return|Type'
SET PAGES 58 LINES 132 VERIFY OFF FEEDBACK OFF
START title132 'Nested Tables'
BREAK ON owner
SPOOL rep_out\&db\tab_nest.lis
SELECT
     owner,
     table_name,
     table_type_owner,
     table_type_name,
     parent_table_name,
     parent_table_column,
     LTRIM(storage_spec) storage_spec,
     LTRIM(return_type) return_type
 FROM
     sys.dba_nested_tables
 ORDER BY
     owner;
SPOOL OFF
```
Quellcode 10.18: Berichtsskript zur Überwachung von verschachtelten Tabellen

```
REM  EXT_TAB.SQL
REM  MRA 10/08/01 Initial Creation
REM  Script to monitor external tables
REM
COLUMN owner                   FORMAT a8   HEADING 'Owner'
COLUMN table_name              FORMAT a15  HEADING 'Table'
COLUMN type_owner              FORMAT a8   HEADING 'Type|Owner'
COLUMN type_name               FORMAT a13  HEADING 'Type|Name'
COLUMN default_directory_owner FORMAT a10  HEADING 'Dir|Owner'
COLUMN default_directory_name  FORMAT a10  HEADING 'Dir|Name'
COLUMN reject_limit            FORMAT a9   HEADING 'Reject|Limit'
COLUMN access_type             FORMAT a6   HEADING 'Access|Type'
COLUMN access_parameters FORMAT a35 WORD_WRAPPED HEADING 'Access Parameters'
SET LINES 132 PAGES 55
START title132 'External Tables'
SPOOL rep_out/&db/ext_tab
SELECT
     owner,
     table_name,
     type_owner,
     type_name,
     default_directory_owner,
     default_directory_name,
     reject_limit,
     access_type,
     access_parameters
 FROM dba_external_tables
/
SPOOL OFF
SET lines 80 Pages 22
```

Quellcode 10.19: Berichtsskript zur Überwachung externer Tabellen

```
rem
rem  NAME     : ind_rep.sql
rem  FUNCTION : Report on indexes
rem  HISTORY  : MRA 6/14/97 Creation
rem
COLUMN owner           FORMAT a8   HEADING 'Index|Owner'
COLUMN index_name      FORMAT a27  HEADING 'Index'
COLUMN index_type      FORMAT a6   HEADING 'Type|Index'
COLUMN table_owner     FORMAT a8   HEADING 'Table|Owner'
COLUMN table_name      FORMAT a24  HEADING 'Table Name'
COLUMN table_type      FORMAT a10  HEADING 'Table|Type'
COLUMN uniqueness      FORMAT a1   HEADING 'U|n|i|q|u|e'
COLUMN tablespace_name FORMAT a13  HEADING 'Tablespace'
COLUMN column_name     FORMAT a25  HEADING 'Col. Name'
SET PAGES 58 LINES 130 FEEDBACK OFF VERIFY OFF
BREAK ON owner
START title132 'Expandeded Index Report'
SPOOL rep_out\&db\ind_exp.lis
SELECT
     a.owner,
     a.index_name,
     a.index_type,
     a.table_owner,
     a.table_name,
     a.table_type,
     DECODE
       (a.uniqueness, 'UNIQUE', 'U','NONUNIQUE','N') uniqueness,
     a.tablespace_name,
     b.column_name
 FROM
     dba_indexes a, dba_ind_columns b
 WHERE
     owner LIKE UPPER('%&owner%')
```

```
        AND a.owner=b.index_owner(+)
        AND a.index_name=b.index_name(+)
    ORDER BY
        owner, index_type;
   SPOOL OFF
```
Quellcode 10.20: Berichtsskript für statistische Werte zu den Tabellenspalten

```
rem
rem  NAME     : brown.sql
rem  FUNCTION : Analyze indexes and produce stat report
rem  FUNCTION : Including browning indicator
rem
rem  HISTORY  : MRA 6/15/97 Created
rem
COL del_lf_rows_len   FORMAT 999,999,999 HEADING 'Deleted Bytes'
COL lf_rows_len       FORMAT 999,999,999 HEADING 'Filled Bytes'
COL browning          FORMAT 999.90      HEADING 'Percent|Browned'
COL height            FORMAT 999,999     HEADING 'Height'
COL blocks            FORMAT 999,999     HEADING 'Blocks'
COL distinct_keys     FORMAT 999,999,999 HEADING '#|Keys'
COL most_repeated_key FORMAT 999999999   HEADING 'Most|Repeated|Key'
COL used_space        FORMAT 999999999   HEADING 'Used|Space'
COL rows_per_key      FORMAT 999999      HEADING 'Rows|Per|Key'
ACCEPT owner PROMPT 'Enter table owner name: '
SET HEADING OFF FEEDBACK OFF VERIFY OFF ECHO OFF RECSEP OFF
SET PAGES 0
TTITLE OFF
DEFINE cr='CHR(10)'
SPOOL index_sz.sql
SELECT
     'CREATE TABLE stat_temp AS SELECT * FROM index_stats;'||&&cr||
     'TRUNCATE TABLE stat_temp;'
 FROM dual;
SELECT
     'ANALYZE INDEX '||owner||'.'||index_name||
     ' VALIDATE STRUCTURE;'||&&cr||
     'INSERT INTO stat_temp SELECT * FROM index_stats;'||&&cr||
     'COMMIT;'
 FROM
     dba_indexes
 WHERE
     owner=UPPER('&owner');
SPOOL OFF
PROMPT 'Analyzing Indexes'
SET FEEDBACK OFF TERMOUT OFF LINES 132 VERIFY OFF
START index_sz.sql
SET TERMOUT ON FEEDBACK ON VERIFY ON LINES 132 PAGES 58
START title132 "Index Statistics Report"
SPOOL rep_out/&db/browning.lst
SELECT
     name,
     del_lf_rows_len,
     lf_rows_len,
     (del_lf_rows_len/
      DECODE((lf_rows_len+del_lf_rows_len),0,1,lf_rows_len+
      del_lf_rows_len))*100 browning,
     height,
     blocks,
     distinct_keys,
     most_repeated_key,
     used_space,
     rows_per_key
 FROM
     stat_temp
 WHERE rows_per_key>0;
SPOOL OFF
SET FEEDBACK ON TERMOUT ON LINES 80 VERIFY ON
HOST del stat_temp
```

Quellcode 10.21: Skript zur Ermittlung der statistischen Werte zu Indices mithilfe des `ANALYZE INDEX`*-Befehls*

```
rem  NAME     : IN_STAT.sql
rem
rem  FUNCTION : Report on index statistics
rem  INPUTS   : 1 = Index owner     2 = Index name
rem
DEF iowner = '&OWNER'
DEF iname = '&INDEX'
SET PAGES 56 LINES 130 VERIFY OFF FEEDBACK OFF
COLUMN owner                    FORMAT a8          HEADING "Owner"
COLUMN index_name               FORMAT a25         HEADING "Index"
COLUMN status                   FORMAT a7          HEADING "Status"
COLUMN blevel                   FORMAT 9,999       HEADING "Tree|Level"
COLUMN leaf_blocks              FORMAT 999,999,999 HEADING "Leaf Blk"
COLUMN distinct_keys            FORMAT 999,999,999 HEADING "# Keys"
COLUMN avg_leaf_blocks_per_key FORMAT 9,999       HEADING "Avg.|LB/Key"
COLUMN avg_data_blocks_per_key FORMAT 9,999       HEADING "Avg.|DB/Key"
COLUMN clustering_factor        FORMAT 999,999     HEADING "Clstr|Factor"
COLUMN num_rows                 FORMAT 999,999,999 HEADING "Number|Rows"
COLUMN sample_size              FORMAT 99,999      HEADING "Sample|Size"
COLUMN last_analyzed                               HEADING 'Analysis|Date'
rem
BREAK ON owner
START title132 "Index Statistics Report"
SPOOL rep_out\&db\ind_stat
rem
SELECT
     owner, index_name, status, blevel, leaf_blocks,
     distinct_keys, avg_leaf_blocks_per_key,
     avg_data_blocks_per_key, clustering_factor,
     num_rows, sample_size, last_analyzed
 FROM
     dba_indexes
 WHERE
     owner LIKE UPPER('&&iowner')
     AND index_name LIKE UPPER('&&iname')
     AND num_rows>0
 ORDER BY
     1,2;
rem
SPOOL OFF
SET PAGES 22 LINES 80 VERIFY ON FEEDBACK ON
CLEAR COLUMNS
UNDEF iowner
UNDEF iname
UNDEF owner
UNDEF name
TTITLE OFF
```
Quellcode 10.22: Bericht für statistische Werte zu Indices unter Oracle8, Oracle8i und Oracle9i

```
rem
rem  Name     : ind_part.sql
rem  Function : Report on partitioned index structure
rem  History  : MRA 6/14/97 Created
rem             MRA 5/10/99 Updated for Subpartitions
rem
COLUMN index_owner       FORMAT a10  HEADING 'Owner'
COLUMN index_name        FORMAT a15  HEADING 'Index'
COLUMN partition_name    FORMAT a15  HEADING 'Partition'
COLUMN subpartition_name FORMAT a15  HEADING 'Sub|Partition'
COLUMN tablespace_name   FORMAT a15  HEADING 'Tablespace'
COLUMN high_value        FORMAT a10  HEADING 'Partition|Value'
COLUMN status            FORMAT a10  Heading 'Status'
SET LINES 130
START title132 'Index Partition Files'
BREAK ON index_owner ON index_name
SPOOL rep_out/&&db/ind_part.lis
SELECT
     a.index_owner,
     a.index_name,
     a.partition_name,
     a.high_value,
     b.subpartition_name,
     b.tablespace_name,
     b.logging,
     b.status
 FROM sys.dba_ind_partitions a, sys.dba_ind_subpartitions b
 WHERE a.owner=b.owner
     AND a.index_name=b.index_name
     And a.partition_name=b.partition_name
 ORDER BY a.index_owner,a.index_name,a.partition_name,
     b.subpartition_position
/
SPOOL OFF
```

Quellcode 10.23: Berichtsskript zur Überwachung der Partitionsdateien eines Index

```
rem
rem  NAME     : ind_pstor.sql
rem  FUNCTION : Provide data on partitioned index storage charcacteristics
rem  HISTORY  : MRA 6/13/97 Created
rem
COLUMN owner             FORMAT a6      HEADING 'Owner'
COLUMN index_name        FORMAT a14     HEADING 'Index'
COLUMN partition_name    FORMAT a9      HEADING 'Partition'
COLUMN tablespace_name   FORMAT a11     HEADING 'Tablespace'
COLUMN pct_free          FORMAT 9999    HEADING '%|Free'
COLUMN ini_trans         FORMAT 9999    HEADING 'Init|Tran'
COLUMN max_trans         FORMAT 9999    HEADING 'Max|Tran'
COLUMN initial_extent    FORMAT 9999999 HEADING 'Init|Extent'
COLUMN next_extent       FORMAT 9999999 HEADING 'Next|Extent'
COLUMN max_extent                       HEADING 'Max|Extents'
COLUMN pct_increase      FORMAT 999     HEADING '%|Inc'
COLUMN distinct_keys     FORMAT 9999999 HEADING '#Keys'
COLUMN clustering_factor FORMAT 999999  HEADING 'Clus|Fact'
SET LINES 130
START title132 'Index Partition File Storage'
BREAK ON index_owner on index_name
SPOOL rep_out/&&db/ind_pstor.lis
SELECT
     index_owner,
     index_name,
     tablespace_name,
     partition_name,
     pct_free,
     ini_trans,
     max_trans,
     initial_extent,
     next_extent,
     max_extent,
     pct_increase,
     distinct_keys,
     clustering_factor
 FROM sys.dba_ind_partitions
 ORDER BY index_owner,index_name
/
SPOOL OFF
```
Quellcode 10.24: Berichtsskript für statistische Werte und Speicherangaben von partitionierten Indices

```
rem
rem  NAME     : ind_subpstor.sql
rem  FUNCTION : Get data on subpartitioned index charcacteristics
rem  HISTORY  : MRA 5/10/99 Created
rem
COLUMN owner             FORMAT a6      HEADING 'Owner'
COLUMN index_name        FORMAT a14     HEADING 'Index'
COLUMN partition_name    FORMAT a9      HEADING 'Partition'
COLUMN subpartition_name FORMAT a9      HEADING 'Sub|Partition'
COLUMN tablespace_name   FORMAT a11     HEADING 'Tablespace'
COLUMN pct_free          FORMAT 9999    HEADING '%|Free'
COLUMN ini_trans         FORMAT 9999    HEADING 'Init|Tran'
COLUMN max_trans         FORMAT 9999    HEADING 'Max|Tran'
COLUMN initial_extent    FORMAT 9999999 HEADING 'Init|Extent'
COLUMN next_extent       FORMAT 9999999 HEADING 'Next|Extent'
COLUMN max_extent                       HEADING 'Max|Extents'
COLUMN pct_increase      FORMAT 999     HEADING '%|Inc'
COLUMN distinct_keys     FORMAT 9999999 HEADING '#Keys'
COLUMN clustering_factor FORMAT 999999  HEADING 'Clus|Fact'
COLUMN num_rows          FORMAT 9999999 HEADING 'Number|Rows'
SET LINES 130
START title132 'Index SubPartition File Storage'
BREAK ON index_owner on index_name
SPOOL rep_out/&&db/ind_pstor.lis
SELECT
     index_owner,
     index_name,
     partition_name,
     sub_partition_name,
     tablespace_name,
     pct_free,
     ini_trans,
     max_trans,
     initial_extent,
     next_extent,
     max_extent,
     pct_increase,
     distinct_keys,
     clustering_factor,
     num_rows
 FROM sys.dba_ind_subpartitions
 ORDER BY index_owner,index_name,partition_name,subpartition_position
/
SPOOL OFF
```
Quellcode 10.25: Berichtsskript für statistische Werte und Speicherangaben für Teilpartitionen von Indice

```
rem
rem  NAME     : ind_func.sql
rem  FUNCTION : Get data on functional index charcacteristics
rem  HISTORY  : MRA 5/12/99 Created
rem
COLUMN owner             FORMAT a6            HEADING 'Owner'
COLUMN index_name        FORMAT a14           HEADING 'Index'
COLUMN table_name        FORMAT a20           HEADING 'Table'
COLUMN column_expression FORMAT a80 WORD_WRAPPED HEADING 'Expression'
SET LINES 130
START title132 'Functional Index Report'
BREAK ON index_owner on index_name
SPOOL rep_out/&&db/ind_func.lis
SELECT
     Index_owner,
     index_name,
     table_name,
     column_expression
 FROM
     Dba_ind_expressions
 WHERE
     Index_owner LIKE '%&&owner%'
     And index_name like '%&&index%'
 ORDER BY
     Index_owner,index_name,column_position;
SPOOL OFF
TTITLE OFF
```

Quellcode 10.26: Berichtsskript für funktionsbasierte Indices

```
REM  bmj_Index.sql
REM  MRA 10/10/01
COLUMN owner       FORMAT a10  HEADING 'Index|Owner'
COLUMN index_name  FORMAT a25  HEADING 'Index|Name'
COLUMN table_owner FORMAT a10  HEADING 'Table|Owner'
COLUMN table_name  FORMAT a15  HEADING 'Table|Name'
COLUMN column_name FORMAT a15  HEADING 'Column|Name'
SET LINES 132
START title132 'Bitmap Join Indexes'
SPOOL rep_out/&db/bmj_index
SELECT a.owner, a.index_name, b.table_owner, b.table_name, b.column_name
 FROM dba_indexes a, dba_join_ind_columns b
 WHERE a.owner = UPPER('&owner')
     and a.join_index='YES'
     and a.owner=b.index_owner
     and a.index_name=b.index_name
/
SPOOL OFF
SET LINES 80
TTITLE OFF
```
Quellcode 10.27: Berichtsskript für Bitmap-Join-Indices

```
rem
rem File    : CLU_REP.SQL
rem Purpose : Document Cluster Data
rem Use     : From user with access to DBA_ views
rem
rem When     Who        What
rem ----     --------   ------------------------
rem 5/27/93  Mike Ault  Initial Creation
rem 6/15/97  Mike Ault  Verified against Oracle8
rem 10/11/01 Mike Ault  Verified against oracle9i
rem
COLUMN owner            FORMAT a10
COLUMN cluster_name     FORMAT a15 HEADING "Cluster"
COLUMN tablespace_name  FORMAT a20 HEADING "Tablespace"
COLUMN table_name       FORMAT a20 HEADING "Table"
COLUMN tab_column_name  FORMAT a20 HEADING "Table Column"
COLUMN clu_column_name  FORMAT a20 HEADING "Cluster Column"
SET PAGES 56 LINES 130 FEEDBACK OFF
START title132 "Cluster Report"
BREAK ON owner SKIP 1 ON cluster ON tablespace
SPOOL rep_out\&db\cluster
SELECT
     a.owner,a.cluster_name,tablespace_name,
     table_name,tab_column_name,clu_column_name
 FROM
     dba_clusters a,dba_clu_columns b
 WHERE
     a.owner = b.owner and
     a.cluster_name=b.cluster_name
ORDER BY 1,2,3,4
/
SPOOL OFF
```
Quellcode 10.28: Berichtsskript für Cluster

```
rem
rem Name     : clus_siz.sql
rem FUNCTION : Generate a cluster sizing report
rem
COLUMN owner           FORMAT a10
COLUMN cluster_name    FORMAT a15       HEADING "Cluster"
COLUMN tablespace_name FORMAT a15       HEADING "Tablespace"
COLUMN pct_free        FORMAT 999       HEADING "%|Fre"
COLUMN pct_used        FORMAT 999       HEADING "%|Use"
COLUMN key_size        FORMAT 999999    HEADING "Key Size"
COLUMN ini_trans       FORMAT 999       HEADING "Ini|Trn"
COLUMN max_trans       FORMAT 999       HEADING "Max|Trn"
COLUMN initial_extent  FORMAT 999999999 HEADING "Init Ext"
COLUMN next_extent     FORMAT 999999999 HEADING "Next Ext"
COLUMN min_extents     FORMAT 999       HEADING "Min|Ext"
COLUMN max_extents     FORMAT 999       HEADING "Max|Ext"
COLUMN pct_increase    FORMAT 999       HEADING "%|Inc"
SET PAGES 56 LINES 130 FEEDBACK OFF
START title132 "Cluster Sizing Report"
BREAK ON owner ON tablespace_name
SPOOL rep_out\&db\cls_sze
SELECT
     owner,
     tablespace_name,
     cluster_name,
     pct_free,
     pct_used,
     key_size,
     ini_trans,
     max_trans,
     initial_extent,
     next_extent,
     min_extents,
     max_extents,
     pct_increase
 FROM
     dba_clusters
 ORDER BY
     1,2,3
/
SPOOL OFF
CLEAR COLUMNS
CLEAR BREAKS
SET PAGES 22 LINES 80 FEEDBACK ON
PAUSE Press enter to continue
```

Quellcode 10.29: Berichtsskript für die Größenermittlung von Clustern

```
rem  Name    : clu_stat.sql
rem  Purpose : Report on new DBA_CLUSTER columns
rem  Use     : From an account that accesses DBA_ views
rem
COLUMN owner              FORMAT a10    HEADING "Owner"
COLUMN cluster_name       FORMAT a15    HEADING "Cluster"
COLUMN tablespace_name    FORMAT a10    HEADING "Tablespace"
COLUMN avg_blocks_per_key FORMAT 999999 HEADING "Blocks per Key"
COLUMN cluster_type       FORMAT a8     HEADING "Type"
COLUMN function           FORMAT 999999 HEADING "Function"
COLUMN hashkeys           FORMAT 99999  HEADING "# of Keys"
SET PAGES 56 LINES 79 FEEDBACK OFF
START title80 "Cluster Statistics Report"
SPOOL report_output/&db/clu_type
SELECT
     owner,
     cluster_name,
     tablespace_name,
     avg_blocks_per_key,
     cluster_type,
     function,
     hashkeys
 FROM
     dba_clusters
 ORDER BY 2
 GROUP BY owner, tablespace, type
/
SPOOL OFF
SET PAGES 22 LINES 80 FEEDBACK ON
CLEAR COLUMNS
TTITLE OFF
```
Quellcode 10.30: Berichtsskript für die neuen DBA_CLUSTERS-Spalten

```
rem
rem  Name    : mv_rep.sql
rem  Purpose :Report on database Materialized views
rem  Use     : From an account that accesses DBA_MVIEWS
rem
rem When      Who        What
rem -------   ---------  ----------------
rem 5/27/93   Mike Ault  Initial Creation
rem 10/10/01  Mike Ault  Update to 9i
rem
SET PAGES 56 LINES 130 FEEDBACK OFF VERIFY OFF
rem
COLUMN mv        FORMAT a30   HEADING "Materialized|View"
COLUMN source    FORMAT a30   HEADING "Source Table"
COLUMN log                    HEADING "Use|Log?”
COLUMN type      FORMAT a10   HEADING "Ref|Type"
COLUMN refreshed              HEADING "Last Refresh"
COLUMN start     FORMAT a13   HEADING "Start Refresh"
COLUMN error                  HEADING "Error"
COLUMN next      FORMAT a13   HEADING "Next Refresh"
rem
PROMPT Percent signs are wild card
ACCEPT mv_owner PROMPT Enter the materialized view owner
START title132 "Materialized View Report for &mv_owner"
SPOOL rep_out/&db/mv_rep&db
rem
SELECT
     Owner||'.'||mview_name mv, master_view,
     master_link Source,
     substr(query,1,query_len) query,
     updatable,
     update_log Log, last_refresh_date Refreshed,
     DECODE(refresh_mode,'FAST','F','COMPLETE','C','FORCE',
            'FR','COMMIT','CM'),
     query,
     master_rollback_segment rbk
 FROM dba_mviews
 WHERE owner LIKE UPPER('%&mv_owner%')
 ORDER BY owner,mview_name;
rem
SPOOL OFF
```

Quellcode 10.31: Skript zur Dokumentation von materialisierten Sichten und Log-Dateien zu materialisierten Sichten

```
rem
rem  Name    : mv_log_rep.sql
rem  Purpose : Report on database materialized view Logs
rem  Use     : From an account that accesses DBA_ views
rem
rem  When      Who        What
rem  -------   ---------  -------------------
rem  5/27/93   Mike Ault  Initial Creation
rem  10/10/01  Mike Ault  Updated to oracle9i
rem
SET PAGES 56 LINES 130 FEEDBACK OFF
START title132 "Materialized View Log Report"
SPOOL rep_out/&db/mv_log_rep&db
rem
COLUMN log_owner  FORMAT a10   HEADING "Owner"
COLUMN master     FORMAT a20   HEADING "Master"
COLUMN log_table  FORMAT a20   HEADING "Materialized View"
COLUMN trigger    FORMAT a20   HEADING "Trigger Text"
COLUMN current                 HEADING "Last Refresh"
rem
```

```
SELECT
     log_owner, master, log_table table,
     log_trigger trigger, rowids, filter_columns filtered,
     object_id id, sequence seq,Include_new_values new
 FROM
     dba_mview_logs
 ORDER BY 1;
rem
SPOOL OFF
CLEAR COLUMNS
SET FEEDBACK ON
TTITLE OFF
```
Quellcode 10.32: Berichtsskript für Log-Dateien zu materialisierten Sichten

```
rem
rem  NAME     : types.sql
rem  FUNCTION : Provide basic report of all database types
rem             for a specific owner or all owners
rem  HISTORY  : MRA 6/15/97 Created
rem
COLUMN owner            FORMAT a10   HEADING 'Type|Owner'
COLUMN type_name        FORMAT a15   HEADING 'Type|Name'
COLUMN typecode         FORMAT a11   HEADING 'Type|Code'
COLUMN predefined       FORMAT a3    HEADING Pre?
COLUMN incomplete       FORMAT a3    HEADING Inc?
COLUMN methods          FORMAT 9999  HEADING '#|Methods'
COLUMN attributes       FORMAT 9999  HEADING '#|Attrib'
COLUMN final            FORMAT A5    HEADING 'Final'
COLUMN instantiable     FORMAT A5    HEADING 'Inst.'
COLUMN supertype_owner  FORMAT a10   HEADING 'SuperType|Owner'
COLUMN supertype_name   FORMAT a15   HEADING 'SuperType|Name'
COLUMN local_attributes FORMAT 99999 HEADING 'Local|Attri'
COLUMN local_methods    FORMAT 99999 HEADING 'Local|Meth'
SET LINES 130 PAGES 58 VERIFY OFF FEEDBACK OFF
BREAK ON owner
START title132 'Database Types Report'
SPOOL rep_out\&db\types.lis
SELECT
     DECODE(owner, null,’SYS-GEN',owner) owner,
     type_name,
     typecode,
     attributes,
     methods,
     predefined,
     incomplete,
     final,
     Instantiable,
     Supertype_owner,
     Supertype_name,
     local_attributes,
     local_methods
 FROM dba_types
 WHERE owner LIKE '%&owner%'
 ORDER BY owner, type_name;
SPOOL OFF
TTITLE OFF
SET VERIFY ON FEEDBACK ON LINES 80 PAGES 22
CLEAR COLUMNS
CLEAR BREAKS
```
Quellcode 10.33: Berichtsskript für Typen

```
rem
rem NAME     : coll_type.sql
rem FUNCTION : Document the collection types in the database
rem            for a specified user or all users
rem HISTORY  : MRA  6/15/97  Created
rem            MRA 10/10/01  Updated to 9i
rem
COL owner           FORMAT a10  HEADING 'Collec.|Owner'
COL type_name       FORMAT a16  HEADING 'Type|Name'
COL coll_type       FORMAT a15  HEADING 'Collec.|Type'
COL upper_bound                 HEADING 'VARRAY|Limit'
COL elem_type_owner FORMAT a10  HEADING 'Elementary|Type|Owner'
COL elem_type_name  FORMAT a11  HEADING 'Elementary|Type|Name'
SET PAGES 58 LINES 130 VERIFY OFF FEEDBACK OFF
START title132 'Collection Type Report'
SPOOL rep_out\&db\col_type.lis
SELECT
     owner,
     type_name,
     coll_type,
     upper_bound,
     elem_type_mod,
     elem_type_owner,
     elem_type_name,
     length,
     precision,
     scale,
     elem_storage,
     nulls_stored
 FROM dba_coll_types
 WHERE owner LIKE '%&owner%'
/
SPOOL OFF
CLEAR COLUMNS
CLEAR BREAKS
TTILTE OFF
SET VERIFY ON FEEDBACK ON
```

Quellcode 10.34: Dieses Skript erzeugt einen Bericht wie den in Listing 10.32

```
rem
rem NAME     : typ_meth.sql
rem FUNCTION : Create a report of type methods for a
rem            specific user or all users
rem HISTORY  : MRA  6/16/97  Created
rem            MRA 10/10/01  Updated to 9i
rem
COLUMN owner         FORMAT a10    HEADING 'Owner'
COLUMN type_name     FORMAT a25    HEADING 'Type|Name'
COLUMN method_name   FORMAT a25    HEADING 'Method|Name'
COLUMN method_type                 HEADING 'Method|Type'
COLUMN parameters    FORMAT 99999  HEADING '#|Param'
COLUMN results       FORMAT 99999  HEADING '#|Results'
COLUMN method_no     FORMAT 99999  HEADING 'Meth.|Number'
COLUMN final         FORMAT A5     HEADING 'Final'
COLUMN Instantiable  FORMAT A6     HEADING 'Instan'
COLUMN overriding    FORMAT A6     HEADING 'ORide?'
COLUMN Inherited     FORMAT A9     HEADING 'Inherited'
BREAK ON owner ON type_name
SET LINES 132 PAGES 58 VERIFY OFF FEEDBACK OFF
START title132 'Type Methods Report'
SPOOL rep_out\&db\typ_meth.lis
SELECT
     owner,
     type_name,
     method_name,
     method_no,
     method_type,
     parameters,
     results,
     final,
     Instantiable,
     Overriding,
     Inherited
 FROM dba_type_methods
 WHERE owner LIKE UPPER('%&owner%')
 ORDER BY owner, type_name;
SPOOL OFF
CLEAR COLUMNS
CLEAR BREAKS
SET VERIFY ON FEEDBACK ON LINES 80 pages 22
TTITLE OFF
```
Quellcode 10.35: Berichtsskript für Typmethoden

```
rem
rem NAME     : tab_ref.sql
rem FUNCTION : Generate a lit of all REF columns in the database
rem            for a specific user or all users
rem HISTORY  : MRA 6/16/97 Created
rem
COLUMN owner             FORMAT a8  HEADING 'Owner'
COLUMN table_name        FORMAT a23 HEADING 'Table|Name'
COLUMN column_name       FORMAT a15 HEADING 'Column|Name'
COLUMN with_rowid        FORMAT a5  HEADING 'With|Rowid'
COLUMN is_scoped         FORMAT a6  HEADING 'Scoped'
COLUMN scope_table_owner FORMAT a8  HEADING 'Scope|Table|Owner'
COLUMN scope_table_name  FORMAT a15 HEADING 'Scope|Table|Name'
BREAK ON owner
SET PAGES 58 LINES 130 FEEDBACK OFF VERIFY OFF
START title132 'Database REF Report'
SPOOL rep_out\&db\tab_ref.lis
SELECT
     owner,
     table_name,
     column_name,
     with_rowid,
     is_scoped,
     scope_table_owner,
     scope_table_name
 FROM
     dba_refs
 WHERE
     Owner LIKE UPPER('%&owner%')
 ORDER BY
     owner;
SPOOL OFF
SET FEEDBACK ON VERIFY ON
CLEAR COLUMNS
CLEAR BREAKS
TTITLE OFF
```
Quellcode 10.36: Berichtsskript für REF-Spalten

```
rem
rem NAME     : operator.sql
rem FUNCTION : Generate a lit of all OPERATORS in the database
rem            for a specific user or all users
rem HISTORY  : MRA 5/12/98 Created
rem
COLUMN owner           FORMAT a8   HEADING 'Owner'
COLUMN operator_name   FORMAT a10  HEADING 'Operator|Name'
COLUMN number_of_binds FORMAT 9999 HEADING 'Binds'
COLUMN position                    HEADING 'Position'
COLUMN argument_type   FORMAT A20  HEADING 'Argument|Type'
COLUMN function_name   FORMAT A20  HEADING 'Binding|Argument'
COLUMN return_schema   FORMAT A10  HEADING 'Return|Schema'
COLUMN return_type     FORMAT A20  HEADING 'Return|Type'
BREAK ON owner ON operator_name ON number_of_binds
SET PAGES 58 LINES 130 FEEDBACK OFF VERIFY OFF
START title132 'Database OPERATOR Report'
SPOOL rep_out\&db\operator.lis
SELECT
     a.owner,
     a.operator_name,
     a.number_of_bindings,
     b.position,
     b.argument_type,
     c.function_name,
     DECODE(c.return_schema,NULL,'Internal',c.return_schema) return_schema,
     c.return_type
 FROM
     Dba_operators a, dba_oparguments b, dba_opbindings c
 WHERE
     Owner LIKE '%&owner%'
     AND a.owner=b.owner
     AND a.operator_name=b.operator_name
     AND a.owner=c.owner
     AND a.operator_name=c.operator_name
     AND b.binding#=c.binding#;
SPOOL OFF
CLEAR BREAKS
CLEAR COLUMNS
TTITLE OFF
SET FEEDBACK ON VERIFY ON
```

Quellcode 10.37: Berichtsskript für Operatoren in der Datenbank

```
rem
rem NAME     : dim_level.sql
rem FUNCTION : Generate a lit of all Dimensions and levels in the
rem            database for a specific user or all users
rem HISTORY  : MRA 5/12/98 Created
rem
COLUMN owner           FORMAT a8    HEADING 'Owner'
COLUMN dimension_name  FORMAT a10   HEADING 'Dimension|Name'
COLUMN level_name      FORMAT a10   HEADING 'Level|Name'
COLUMN column_name     FORMAT a20   HEADING 'Column|Name'
COLUMN key_position    FORMAT 9999  HEADING 'Key|Position'
BREAK ON owner ON operator_name ON number_of_binds
SET PAGES 58 LINES 130 FEEDBACK OFF VERIFY OFF
START title132 'Database Dimension Levels Report'
SPOOL rep_out\&db\dim_level.lis
SELECT
     a.owner,
     a.dimension_name,
     b.level_name,
     c.column_name,
     c.key_position
 FROM
     Dba_dimensions a, dba_dim_levels b, dba_dim_level_key c
 WHERE
     a.Owner LIKE '%&owner%'
     AND a.owner=b.owner
     AND a.dimension_name=b.dimension_name
     AND a.owner=c.owner
     AND a.dimension_name=c.dimension_name
     AND b.level_name=c.level_name
 ORDER BY
     a.owner,
     a.dimension_name,
     b.level_name;
SPOOL OFF
CLEAR BREAKS
CLEAR COLUMNS
TTITLE OFF
SET FEEDBACK ON VERIFY ON
```
Quellcode 10.38: Berichtsskript für Dimensionsebenen einer Datenbank

```
rem
rem NAME     : dim_hierarchies.sql
rem FUNCTION : Generate a lit of all dimensions and hierarchies in the
rem            database for a specific user or all users
rem HISTORY  : MRA 5/12/98 Created
rem
COLUMN owner              FORMAT a8   HEADING 'Owner'
COLUMN dimension_name     FORMAT a10  HEADING 'Dimension|Name'
COLUMN column_name        FORMAT a10  HEADING 'Column|Name'
COLUMN hierarchy_name     FORMAT a10  HEADING 'Hierarchy|Name'
COLUMN parent_level_name  FORMAT a10  HEADING 'Parent|Level'
COLUMN child_level_name   FORMAT a10  HEADING 'Child|Level'
COLUMN join_key_id        FORMAT a20  HEADING 'Join Key|ID'
BREAK ON owner ON dimension_name
SET PAGES 58 LINES 78 FEEDBACK OFF VERIFY OFF
START title80 'Database Dimension Hierarchy Report'
SPOOL rep_out\&db\dim_hierarchies.lis
SELECT
     a.owner,
     a.dimension_name,
     b.hierarchy_name,
     c.parent_level_name,
     c.child_level_name,
     c.join_key_id
 FROM
     Dba_dimensions a, dba_dim_hierarchies b, dba_dim_child_of c
 WHERE
     a.Owner LIKE '%&owner%'
     AND a.owner=b.owner
     AND a.dimension_name=b.dimension_name
     AND a.owner=c.owner
     AND a.dimension_name=c.dimension_name
     AND b.hierarchy_name=c.hierarchy_name
 ORDER BY
     a.owner,
     a.dimension_name,
     b.hierarchy_name;
SPOOL OFF
CLEAR BREAKS
CLEAR COLUMNS
TTITLE OFF
SET FEEDBACK ON VERIFY ON
```

Quellcode 10.39: Berichtsskript für die Dimensionshierarchie einer Datenbank

```
rem
rem NAME     : dim_attribute.sql
rem FUNCTION : Generate a lit of all Dimensions and atrributes in the
rem            database for a specific user or all users
rem HISTORY  : MRA 5/12/98 Created
rem
COLUMN owner           FORMAT a8   HEADING 'Owner'
COLUMN dimension_name  FORMAT a10  HEADING 'Dimension|Name'
COLUMN column_name     FORMAT a20  HEADING 'Column|Name'
COLUMN level_name      FORMAT a20  HEADING 'Level|Name'
COLUMN inferred        FORMAT a10  HEADING 'Inferred'
BREAK ON owner ON level_name
SET PAGES 58 LINES 78 FEEDBACK OFF VERIFY OFF
START title80 'Database OPERATOR Report'
SPOOL rep_out\&db\dim_attribute.lis
SELECT
     a.owner,
     a.dimension_name,
     b.level_name,
     c.column_name,
     c.inferred
 FROM
     Dba_dimensions a, dba_dim_levels b, dba_dim_attributes c
 WHERE
     a.owner LIKE '%&owner%'
     AND a.owner=b.owner
     AND a.dimension_name=b.dimension_name
     AND a.owner=c.owner
     AND a.dimension_name=c.dimension_name
     AND b.level_name=c.level_name
 ORDER BY
     a.owner,
     a.dimension_name,
     b.level_name;
SPOOL OFF
CLEAR BREAKS
CLEAR COLUMNS
TTITLE OFF
SET FEEDBACK ON VERIFY ON
```
Quellcode 10.40: Berichtsskript für die Dimensionsattribute einer Datenbank

```
rem
rem NAME     : outline.sql
rem FUNCTION : Generate a lit of all outlines in the
rem            database for a specific user or all users
rem HISTORY  : MRA 5/13/98 Created
rem
COLUMN owner      FORMAT a8   HEADING 'Owner'
COLUMN name       FORMAT a13  HEADING 'Outline|Name'
COLUMN category   FORMAT a8   HEADING 'Category|Name'
COLUMN used       FORMAT a7   HEADING 'Used?'
COLUMN timestamp  FORMAT a16  HEADING 'Date Last|Used'
COLUMN version    FORMAT a9   HEADING 'Version'
COLUMN sql_text   FORMAT a40  HEADING 'SQL Outlined' WORD_WRAPPED
BREAK ON owner ON category
SET PAGES 58 LINES 130 FEEDBACK OFF VERIFY OFF
START title132 'Database OUTLINE Report'
SPOOL rep_out\&db\outline.lis
SELECT
     owner,
     name,
     category,
     used,
     to_char(timestamp,'dd/mm/yyyy hh24:mi') timestamp,
     version,
     sql_text
 FROM
     Dba_outlines
 WHERE
     Owner LIKE '%&owner%'
 ORDER BY
     owner,category;
SPOOL OFF
CLEAR BREAKS
TTITLE OFF
SET FEEDBACK ON VERIFY ON
```
Quellcode 10.41: Berichtsskript für Outlines der Datenbank

```
rem
rem NAME     : outline_hint.sql
rem FUNCTION : Generate a lit of all outlines in the
rem            database for a specific user and outline
rem            or all users and outlines
rem HISTORY  : MRA 5/13/98  Created
rem
COLUMN owner     FORMAT a8    HEADING 'Owner'
COLUMN name      FORMAT a13   HEADING 'Outline|Name'
COLUMN category  FORMAT a10   HEADING 'Category|Name'
COLUMN node      FORMAT 9999  HEADING 'Node'
COLUMN join_pos  FORMAT 9999  HEADING 'Join|Pos'
COLUMN hint      FORMAT A27   HEADING 'Hint Text' WORD_WRAPPED
BREAK ON owner ON category ON name
SET PAGES 58 LINES 78 FEEDBACK OFF VERIFY OFF
START title80 'Database OUTLINE Report'
SPOOL rep_out\&db\outline_hint.lis
SELECT
     a.owner, a.name,
     a.category, b.node,
     b.join_pos, b.hint
 FROM
     Dba_outlines a, dba_outline_hints b
 WHERE
     a.Owner LIKE UPPER('%&owner%')
     AND a.name LIKE UPPER('%&outline%')
     AND a.owner=b.owner
     AND a.name=b.name
 ORDER BY
     owner,category,name,b.node;
SPOOL OFF
CLEAR BREAKS
TTITLE OFF
SET FEEDBACK ON VERIFY ON
```

Quellcode 10.42: Berichtsskript für die Outline-Hinweise einer Datenbank

```
REM
REM NAME        : DB_USER.SQL
REM
REM FUNCTION    : GENERATE USER_REPORT
REM Limitations : None
REM
REM Updates     : MRA 6/10/97 added Oracle8 account status
REM               MRA 5/14/99 Added Oracle8i  Resource Group
REM               MRA 5/22/99 Removed expiry data to new report
REM
SET PAGESIZE 58  LINESIZE 131 FEEDBACK OFF
REM
COLUMN username                 FORMAT a12 HEADING User
COLUMN account_status           FORMAT a6  HEADING Status
COLUMN default_tablespace       FORMAT a14 HEADING Default
COLUMN temporary_tablespace     FORMAT a10 HEADING Temporary
COLUMN granted_role             FORMAT a22 HEADING Roles
COLUMN default_role             FORMAT a8  HEADING Default?
COLUMN admin_option             FORMAT a6  HEADING Admin?
COLUMN profile                  FORMAT a10 HEADING Profile
COLUMN initial_rsrc_consumer_group FORMAT a22 HEADING 'Resource|Group'
REM
START title132 'ORACLE USER REPORT'
DEFINE output = rep_out\&db\db_user
BREAK ON username SKIP 1 ON default_tablespace ON temporary_tablespace ON profile
ON account_status ON initial_rsrc_consumer_group
SPOOL &output
REM
```

```
SELECT a.username,
       a.default_tablespace,a.temporary_tablespace,
       a.profile,a.account_status,
       a.initial_rsrc_consumer_group,
       b.granted_role,b.admin_option,
       b.default_role
 FROM sys.dba_users a,
      sys.dba_role_privs b
 WHERE a.username = b.grantee
 ORDER BY username,
          default_tablespace,
          temporary_tablespace,
          profile,
          granted_role;
REM
SPOOL OFF
SET TERMOUT ON FLUSH ON FEEDBACK ON VERIFY ON CLEAR COLUMNS
CLEAR BREAKS
PAUSE Press Enter to continue
```
Quellcode 11.1: Beispielskript für einen Benutzerbericht

```
REM
REM NAME        : DB_USER.SQL
REM
REM FUNCTION    : GENERATE USER_REPORT
REM Limitations : None
REM
REM Updates     : MRA 6/10/97 added Oracle8 account status
REM               MRA 5/14/99 Added Oracle8i Resource Group
REM
SET PAGESIZE 58 LINESIZE 131 FEEDBACK OFF
REM
COLUMN username                 FORMAT a10 HEADING User
COLUMN account_status           FORMAT a10 HEADING Status
COLUMN default_tablespace       FORMAT a15 HEADING Default
COLUMN temporary_tablespace     FORMAT a15 HEADING Temporary
COLUMN granted_role             FORMAT a21 HEADING Roles
COLUMN default_role             FORMAT a9  HEADING Default?
COLUMN admin_option             FORMAT a7  HEADING Admin?
COLUMN profile                  FORMAT a15 HEADING Profile
COLUMN initial_rsrc_consumer_group FORMAT a10 HEADING 'Resource|Group'
COLUMN lock_date                           HEADING 'Date|Locked'
COLUMN expiry_date                         HEADING 'Expiry_date'
REM
START title132 'ORACLE USER REPORT'
DEFINE output = rep_out\&db\db_user
BREAK ON username SKIP 1 ON account_status ON default_tablespace
ON temporary_tablespace ON profile
SPOOL &output
REM
SELECT a.username,
       a.account_status,
       TO_CHAR(a.lock_date,'dd-mon-yyyy hh24:mi') lock_date,
       TO_CHAR(a.expiry_date,'dd-mon-yyyy hh24:mi') expiry_date,
       a.default_tablespace,a.temporary_tablespace,
       a.profile,b.granted_role,
       b.admin_option,b.default_role,
       a.initial_rsrc_consumer_group
 FROM sys.dba_users a,
      sys.dba_role_privs b
 WHERE a.username = b.grantee
 ORDER BY username,
          default_tablespace,temporary_tablespace,
          profile, granted_role;
REM
SPOOL OFF
SET TERMOUT ON FLUSH ON FEEDBACK ON VERIFY ON
```

```
   CLEAR COLUMNS
   CLEAR BREAKS
   PAUSE Press Enter to continue
```
Quellcode 11.2: Beispielskript für einen Benutzerbericht

```
REM
REM NAME        : USER_EXPIRE.SQL
REM
REM FUNCTION    : GENERATE USER EXPIRY DATA REPORT
REM Limitations : None
REM
REM Updates     : MRA 5/22/99 Created
REM
COLUMN account_status        FORMAT a15 HEADING Status
COLUMN default_tablespace    FORMAT a14 HEADING Default
COLUMN temporary_tablespace  FORMAT a10 HEADING Temporary
COLUMN username              FORMAT a12 HEADING User
COLUMN lock_date             FORMAT a11 HEADING 'Date|Locked'
COLUMN expiry_date           FORMAT a11 HEADING 'Expiry|Date'
COLUMN profile               FORMAT a15 HEADING Profile
SET PAGESIZE 58 LINESIZE 131 FEEDBACK OFF
START title132 'ORACLE USER EXPIRATION REPORT'
BREAK ON username SKIP 1 ON default_tablespace ON temporary_tablespace ON
profile ON account_status
SPOOL rep_out\&db\user_expire
rem
SELECT username,
       default_tablespace,temporary_tablespace,
       profile,account_status,
       TO_CHAR(lock_date,'dd-mon-yyyy') lock_date,
       TO_CHAR(expiry_date,'dd-mon-yyyy') expiry_date
 FROM sys.dba_users
 ORDER BY username,
          default_tablespace,temporary_tablespace,
          profile, account_status;
rem
SPOOL OFF
SET TERMOUT ON FLUSH ON FEEDBACK ON VERIFY ON
CLEAR COLUMNS
CLEAR BREAKS
PAUSE Press Enter to continue
```

Quellcode 11.3: Beispielskript für einen Bericht über den Verfallsstatus von Accounts

```
REM
REM NAME        : sys_role.SQL
REM PURPOSE     : GENERATE SYSTEM GRANTS and ROLES REPORT
REM USE         : CALLED BY SQLPLUS
REM Limitations : None
REM Revisions   :
REM Date          Modified by  Reason for change
REM 08-Apr-1993   MIKE AULT    INITIAL CREATE
REM 10-Jun-1997   Mike Ault    Update to Oracle8
REM 15-May-1999   Mike Ault    No changes for Oracle8i
REM
SET FLUSH OFF TERM OFF PAGESIZE 58 LINESIZE 78
COLUMN grantee          HEADING 'User or Role'
COLUMN admin_option     HEADING Admin?
START title80 'SYSTEM GRANTS AND ROLES REPORT'
DEFINE output = rep_out\&&db\role_report
SPOOL &output
SELECT
     grantee,
     privilege,
     admin_option
 FROM
     sys.dba_sys_privs
 GROUP BY
     grantee;
SPOOL OFF
SET FLUSH ON TERM ON
CLEAR COLUMNS
TTITLE OFF
```

Quellcode 11.4: Beispielskript für einen Rollenbericht unter Oracle7, Oracle8 und Oracle8i

```
REM NAME       : PROFILE_REPORT.SQL
REM PURPOSE    : GENERATE USER PROFILES REPORT
REM Revisions  :
REM Date         Modified by     Reason for change
REM 08-Apr-1993  MIKE AULT       INITIAL CREATE
REM 14-May-1999  MIKE AULT       Added resource_type
REM
SET FLUSH OFF TERM OFF PAGESIZE 58 LINESIZE 78 VERIFY OFF FEEDBACK OFF
COLUMN profile          FORMAT a15  HEADING Profile
COLUMN resource_name    FORMAT A25  HEADING 'Resource:'
COLUMN resource_type    FORMAT A9   HEADING 'Resource|Affects'
COLUMN limit            FORMAT a20  HEADING Limit
START title80 'ORACLE PROFILES REPORT'
BREAK ON profile
SPOOL rep_out/&&db/prof_rep
SELECT
     profile,resource_name,
     resource_type,limit
 FROM
     sys.dba_profiles
 WHERE
     profile LIKE UPPER('%&profile_name%')
 ORDER BY
     profile,resource_type,resource_name;
SPOOL OFF
CLEAR COLUMNS
SET FLUSH ON TERM ON VERIFY ON FEEDBACK ON
TTITLE OFF
```

Quellcode 11.5: wSkript zum Generieren eines Ressourcenberichts für Benutzerprofile unter Oracle8, Oracle8i und Oracle9i

```
REM NAME       : RESOURCE_PLAN.SQL
REM PURPOSE    : GENERATE DATABASE RESOURCE PLAN REPORT
REM Revisions  :
REM Date         Modified by     Reason for change
REM 15-May-1999  MIKE AULT       initial creation
REM
COLUMN plan             FORMAT a16 HEADING 'Plan|Name'
COLUMN cpu_method1      FORMAT a8  HEADING 'CPU|Method'
COLUMN mandatory1       FORMAT a4  HEADING 'Man?'
COLUMN group_or_subplan FORMAT a12 HEADING 'Group or|Subplan Name'
COLUMN type             FORMAT a8  HEADING 'Group or|Subplan'
COLUMN cpu_method2      FORMAT a8  HEADING 'CPU|Method2'
COLUMN plan2 NOPRINT
COLUMN queue_meth1      FORMAT A12
COLUMN queue_meth2      FORMAT A12
COLUMN session_pool1    FORMAT A25 HEADING 'Sessions 1'
COLUMN session_pool2    FORMAT A25 HEADING 'Sessions 2'
REM
SET LINES 228 PAGES 55 VERIFY OFF FEEDBACK OFF
BREAK ON plan ON cpu_method1 ON mandatory1 ON num_plan_directives
START title132 'Resource Plan Report'
SPOOL rep_out\&&db\resource_plan.lis
REM
SELECT DISTINCT
     decode(b.plan,'',a.plan,b.plan) plan,
     a.active_sess_pool_mth session_pool1,
     a.parallel_degree_limit_mth parallel_meth1,
     a.queueing_mth queue_meth1,
     b.plan plan2,
     a.cpu_method cpu_method1,
     a.mandatory mandatory1,
     DECODE(b.group_or_subplan,'',d.consumer_group,
       b.group_or_subplan) group_or_subplan,
     DECODE(b.type,'CONSUMER_GROUP','GROUP',b.type) type,
     c.active_sess_pool_mth session_pool2,
     c.parallel_degree_limit_mth parallel_meth2,
     c.queueing_mth queue_meth2,
     decode(c.cpu_method,'',d.cpu_method,c.cpu_method) cpu_method2
 FROM
     dba_rsrc_plans a, dba_rsrc_plan_directives b, dba_rsrc_plans c,
     dba_rsrc_consumer_groups d
 WHERE
     a.plan=b.plan
     AND ((b.group_or_subplan = c.plan OR
     b.group_or_subplan = d.consumer_group))
 ORDER BY
     2,5;
SPOOL OFF
CLEAR COLUMNS
SET VERIFY ON FEEDBACK ON LINES 80 PAGES 22
TTITLE OFF
```

Quellcode 11.6: Berichtsskript für Ressourcenpläne unter Oracle8i und Oracle9i

```
REM NAME       : PLAN_DIRECTIVES.SQL
REM PURPOSE    : GENERATE DATABASE RESOURCE PLAN DIRECTIVES REPORT
REM Revisions  :
REM Date         Modified by     Reason for change
REM 15-May-1999  MIKE AULT       initial creation
REM 13-Oct-2001  Mike Ault       Update to 9i
REM
COLUMN plan                     FORMAT a17     HEADING 'Plan|Name'
COLUMN cpu_method1              FORMAT a8      HEADING 'CPU|Method'
COLUMN mandatory1               FORMAT a3      HEADING 'Man|?'
COLUMN num_plan_directives      FORMAT 999     HEADING 'Num|Dir'
COLUMN group_or_subplan         FORMAT a17     HEADING 'Group or|Subplan
Name'
COLUMN type                     FORMAT a5      HEADING 'Type'
COLUMN cpu_method2              FORMAT a8      HEADING 'CPU|Method'
COLUMN cpu_p1                   FORMAT 999     HEADING 'CPU|1%'
COLUMN cpu_p2                   FORMAT 999     HEADING 'CPU|2%'
COLUMN cpu_p3                   FORMAT 999     HEADING 'CPU|3%'
COLUMN cpu_p4                   FORMAT 999     HEADING 'CPU|4%'
COLUMN cpu_p5                   FORMAT 999     HEADING 'CPU|5%'
COLUMN cpu_p6                   FORMAT 999     HEADING 'CPU|6%'
COLUMN cpu_p7                   FORMAT 999     HEADING 'CPU|7%'
COLUMN cpu_p8                   FORMAT 999     HEADING 'CPU|8%'
COLUMN parallel_degree_limit_p1 FORMAT 9999999 HEADING 'Par|Degree'
COLUMN switch_group             FORMAT a15     HEADING 'Switch|Group'
COLUMN switch_time                             HEADING 'Switch|Time'
COLUMN switch_estimate                         HEADING 'Switch|Estimate'
COLUMN max_est_exec_time                       HEADING 'Max Est|Exec Time'
COLUMN undo_pool                               HEADING 'Undo|Pool'
COLUMN active_sess_pool_p1                     HEADING 'Active|Session|Pool'
COLUMN queueing_p1                             HEADING 'Queueing'
REM
SET LINES 200 PAGES 55 VERIFY OFF FEEDBACK OFF
BREAK ON plan on cpu_method1 on mandatory1 on num_plan_directives
START title132 'Resource Plan Directives Report'
SPOOL rep_out\&&db\plan_directives.lis
REM
SELECT DISTINCT
     a.plan,
     a.cpu_method cpu_method1,
     a.mandatory mandatory1,
     b.group_or_subplan,
     DECODE(b.type,'CONSUMER_GROUP','GROUP',b.type) type,
     c.cpu_method cpu_method2,
     b.cpu_p1,b.cpu_p2,b.cpu_p3,b.cpu_p4,
     b.cpu_p5,b.cpu_p6,b.cpu_p7,b.cpu_p8,
     b.active_sess_pool_p1,b.queueing_p1,
     b.parallel_degree_limit_p1,
     b.switch_group,b.switch_time,
     b.switch_estimate,b.max_est_exec_time,
     b.undo_pool
 FROM
     dba_rsrc_plans a, dba_rsrc_plan_directives b, dba_rsrc_plans c,
     dba_rsrc_consumer_groups d
 WHERE
     a.plan=b.plan
     AND ((b.group_or_subplan = c.plan OR
         b.group_or_subplan=d.consumer_group))
     AND b.status='ACTIVE'
 ORDER BY
     1,4,5;
SPOOL OFF
CLEAR COLUMNS
SET VERIFY ON FEEDBACK ON
TTITLE OFF
```
Quellcode 11.7: Berichtsskript für Plandirektiven unter Oracle8i und Oracle9i

```
   REM NAME       : PLAN_SYS_GRANTS.SQL
   REM PURPOSE    : GENERATE DATABASE RESOURCE PLAN SYSTEM GRANTS REPORT
   REM Revisions  :
   REM Date         Modified by     Reason for change
   REM 15-May-1999  MIKE AULT       initial creation
   REM
   COLUMN privilege     FORMAT a30  HEADING 'Plan System Privilege'
   COLUMN grantee       FORMAT a30  HEADING 'User or Role'
   COLUMN admin_option  FORMAT a7   HEADING 'Admin?'
   BREAK ON privilege
   SET LINES 78 VERIFY OFF FEEDBACK OFF
   START title80 'Resource Plan System Grants'
   SPOOL rep_out\&&db\plan_sys_grants.lis
   REM
   SELECT
        privilege, grantee, admin_option
    FROM
        Dba_rsrc_manager_system_privs
    ORDER BY
        Privilege;
   SPOOL OFF
   SET VERIFY ON FEEDBACK ON
   TTITLE OFF
```
Quellcode 11.8: Berichtsskript für Systemprivilegien bei Ressourcenplänen unter Oracle8i und Oracle9i

```
REM NAME       : PLAN_GROUP_GRANTS.SQL
REM PURPOSE    : GENERATE DATABASE RESOURCE PLAN GROUP GRANTS REPORT
REM Revisions  :
REM Date         Modified by     Reason for change
REM 15-May-1999  MIKE AULT       initial creation
REM
COLUMN granted_group  FORMAT a30  HEADING 'Granted Group'
COLUMN grantee        FORMAT a30  HEADING 'User or Role'
COLUMN grant_option   FORMAT a7   HEADING 'Admin?'
COLUMN initial_group  FORMAT a8   HEADING 'Initial?'
BREAK ON granted_group
SET LINES 78 VERIFY OFF FEEDBACK OFF
START title80 'Resource Plan Group Grants'
SPOOL rep_out\&&db\plan_group_grants.lis
REM
SELECT
     Granted_group, grantee, grant_option, initial_group
 FROM
     Dba_rsrc_consumer_group_privs
 ORDER BY
     Granted_group;
SPOOL OFF
SET VERIFY ON FEEDBACK ON
TTITLE OFF
```

Quellcode 11.9: Berichtsskript für Gruppenrechte bei Ressourcenplänen unter Oracle8i und Oracle9i

```
rem PURPOSE     : Produce report of table grants showing
rem               GRANTOR, GRANTEE and specific GRANTS.
rem LIMITATIONS : User must have access to DBA_TAB_PRIVS
rem INPUTS      : Owner name
rem OUTPUTS     : Report of table grants
rem
rem HISTORY     :
rem Who:          What:                 Date:
rem Mike Ault     Initial creation      3/2/95
rem Mike Ault     Oracle8 verified      6/10/97
rem Mike Ault     Oracle8i verified     5/15/99
rem Mike Ault     Oracle9i Updated      13/10/01
rem
rem
COLUMN GRANTEE     FORMAT A19  HEADING "Grantee"
COLUMN OWNER       FORMAT A10  HEADING "Owner"
COLUMN TABLE_NAME  FORMAT A26  HEADING "Table"
COLUMN GRANTOR     FORMAT A10  HEADING "Grantor"
COLUMN PRIVILEGE   FORMAT A10  HEADING "Privilege"
COLUMN GRANTABLE   FORMAT A6   HEADING "With|Grant|Option?"
COLUMN HIERARCHY   FORMAT A3   HEADING 'HRY'
REM
BREAK ON owner SKIP 2 ON table_name ON grantee ON grantor ON REPORT
REM
SET LINESIZE 100 PAGES 56 VERIFY OFF FEEDBACK OFF
START title132 "TABLE GRANTS BY OWNER AND TABLE"
SPOOL rep_out\&db\tab_grants
REM
SELECT
     owner,table_name,grantee,grantor,
     privilege,grantable,hierarchy
 FROM
     dba_tab_privs
 WHERE
     owner LIKE UPPER('%&owner&')
     AND privilege !='EXECUTE'
 ORDER BY
     owner,table_name,grantor,grantee;
REM
SPOOL OFF
PAUSE Press Enter to continue
SET LINESIZE 80 PAGES 22 VERIFY ON FEEDBACK ON
CLEAR BREAKS
CLEAR COLUMNS
TTITLE OFF
```

Quellcode 11.10: Berichtsskript für Tabellenrechte unter Oracle9i

```
REM FUNCTION: SCRIPT FOR CAPTURING TABLE COLUMN GRANTS
REM
REM This script is intended to run with Oracle7,Oracle8 or Oracle9.
REM
REM Running this script will create a script of all the grants
REM on columns
REM
REM Grants must be made by the original grantor so the script
REM connects as that user using the username as the password
REM edit the proper password in at time of running
REM
REM NOTE: Grants made to 'SYS','CONNECT','RESOURCE','DBA',
REM       'EXP_FULL_DATABASE','IMP_FULL_DATABASE' are not captured.
REM
REM       Only preliminary testing of this script was performed.
REM       Be sure to test it completely before relying on it.
REM
SET VERIFY OFF FEEDBACK OFF TERMOUT OFF ECHO OFF PAGESIZE 0
SET EMBEDDED ON HEADING OFF
SET TERMOUT ON
PROMPT Creating table grant script...
SET TERMOUT OFF
DEFINE cr=CHR(10);
BREAK ON line1
COLUMN dbname NEW_VALUE db NOPRINT
SELECT name dbname FROM v$database;
SPOOL rep_out\&db\grt_cols.sql
rem
SELECT
     'CONNECT '||grantor||'/'||grantor line1,
     'GRANT '||&&cr||lower(privilege)||'('||column_name||
     ') ON '||owner||'.'||table_name||&&cr||
     ' TO '|| lower(grantee) ||&&cr||
     decode(grantable,'YES',' WITH ADMIN OPTION;',';')
 FROM
     sys.dba_col_privs
 WHERE
     grantee NOT IN ('SYS','CONNECT','RESOURCE','DBA',
       'EXP_FULL_DATABASE','IMP_FULL_DATABASE')
 ORDER BY grantor, grantee
/
SPOOL OFF
SET VERIFY ON FEEDBACK ON TERMOUT ON PAGESIZE 22 EMBEDDED OFF
CLEAR COLUMNS
CLEAR COMPUTES
CLEAR BREAKS
```

Quellcode 11.11: Skript für die Aufzeichnung von Zugriffsrechten auf der Ebene von Tabellenspalten

```
rem PURPOSE    : Produce report of db policies
rem              used to implement row level grants
rem
rem LIMITATIONS: User must have access to DBA_POLICIES
rem
rem HISTORY:
rem Who:         What:                 Date:
rem Mike Ault    Initial creation      5/23/99
rem Mike Ault    Updated to Oracle9i   10/13/01
rem
COLUMN object_owner  FORMAT A10  HEADING 'Object|Owner'
COLUMN object_name   FORMAT A19  HEADING 'Object|Name'
COLUMN policy_group  FORMAT A12  HEADING 'Policy|Group'
COLUMN policy_name   FORMAT A16  HEADING 'Policy|Name'
COLUMN pf_owner      FORMAT A10  HEADING 'Policy|Function|Owner'
COLUMN function      FORMAT A15  HEADING 'Function|Name'
COLUMN sel           FORMAT A3   HEADING 'Sel|?'
COLUMN ins           FORMAT A3   HEADING 'Ins|?'
COLUMN upd           FORMAT A3   HEADING 'Upd|?'
COLUMN del           FORMAT A3   HEADING 'Del|?'
COLUMN chk_option    FORMAT A3   HEADING 'Check|Option'
COLUMN enable        FORMAT A3   HEADING 'Enabled?'
COLUMN static_policy FORMAT A7   HEADING 'Static?'
SET LINES 132 VERIFY OFF FEEDBACK OFF PAGES 47
START title132 'DB Policies Report'
BREAK ON object_owner
SPOOL rep_out\&db\db_policies
SELECT
     object_owner, object_name,policy_group,
     policy_name,pf_owner,function,
     sel,ins,upd,del,chk_option,
     enable,static_policy
 FROM
     dba_policies
 ORDER BY
     1,2,3;
SPOOL OFF
SET LINES 80 VERIFY ON FEEDBACK ON PAGES 22
CLEAR BREAKS
CLEAR COLUMNS
TTITLE OFF
```
Quellcode 11.12: Berichtsskript zur Überwachung von Sicherheitsrichtlinien auf Zeilenebene

```
REM
REM  Name     : pid.sql
REM
REM  FUNCTION : Generate a list of current oracle sids/pids
REM
COLUMN terminal FORMAT a10   HEADING 'Terminal'
COLUMN program  FORMAT a30   HEADING 'Program'
COLUMN pid      FORMAT 9999  HEADING 'Process|ID'
COLUMN sid      FORMAT 9999  HEADING 'Session|ID'
COLUMN osuser   FORMAT A15   HEADING 'Operating|System|User'
COLUMN spid     FORMAT A7    HEADING 'OS|Process|ID'
COLUMN serial#  FORMAT 99999 HEADING 'Serial|Number'
SET LINES 132 PAGES 58
BREAK ON username
COMPUTE COUNT OF pid ON username
START title132 "Oracle Processes"
SPOOL rep_out\&db\cur_proc
SELECT
     NVL(a.username,'Null') username,
     b.pid,a.sid,
     DECODE(a.terminal,'?','Detached',a.terminal) terminal,
     b.program,b.spid,a.osuser,a.serial#
 FROM
     v$session a,
     v$process b
 WHERE
     a.PADDR = b.ADDR
 ORDER by
     a.username,
     b.pid
/
SPOOL OFF
CLEAR BREAKS
CLEAR COLUMNS
SET PAGES 22
TTITLE OFF
PAUSE Press Enter to continue
```

Quellcode 11.13: Berichtsskript zur Anzeige der aktiven Benutzer

```
rem
rem Name     : free_space.sql
rem
rem FUNCTION : Provide data on tablespace extent status
rem FUNCTION : this report uses the free_space2 view
rem FUNCTION : includes fsfi from DBA Handbook
rem
SET FEED OFF
SET FLUSH OFF
SET VERIFY OFF
set pages 58 LINES 132
COLUMN tablespace    HEADING Name            FORMAT a30
COLUMN files         HEADING '#Files'        FORMAT 9,999
COLUMN pieces        HEADING 'Frag'          FORMAT 9,999
COLUMN free_bytes    HEADING 'Free|Byte'     FORMAT 9,999,999,999
COLUMN free_blocks   HEADING 'Free|Blk'      FORMAT 999,999
COLUMN largest_bytes HEADING 'Biggest|Bytes' FORMAT 9,999,999,999
COLUMN largest_blks  HEADING 'Biggest|Blks'  FORMAT 999,999
COLUMN ratio         HEADING 'Percent'       FORMAT 999.999
COLUMN average_fsfi  HEADING 'Average|FSFI'  FORMAT 999.999
START title132 "FREE SPACE REPORT"
DEFINE 1 = report_output/&&db/free_spc
SPOOL &1
SELECT
```

```
        tablespace,
        COUNT(*) files,
        SUM(pieces) pieces,
        SUM(free_bytes) free_bytes,
        SUM(free_blocks) free_blocks,
        SUM(largest_bytes) largest_bytes,
        SUM(largest_blks) largest_blks,
        SUM(largest_bytes)/sum(free_bytes)*100 ratio,
        SUM(fsfi)/COUNT(*) average_fsfi
    FROM
        free_space
    GROUP BY
        tablespace;
   SPOOL OFF
   CLEAR COLUMNS
   TTITLE OFF
   SET FEED ON
   SET FLUSH ON
   SET VERIFY ON
   SET PAGES 22 LINES 80
   PAUSE Press Enter to continue
```

Quellcode 11.14: Berichtsskript für die Nutzung und Fragmentierung von Tablespaces

```
   rem
   rem Name     : free_space_view.sql
   rem
   rem FUNCTION : Create free_space view for use by freespc reports
   rem
   CREATE VIEW free_space
   (tablespace, file_id, pieces, free_bytes, free_blocks,
   largest_bytes,largest_blks, fsfi) AS
   SELECT tablespace_name, file_id, COUNT(*),
        SUM(bytes), SUM(blocks),
        MAX(bytes), MAX(blocks),
        SQRT(MAX(blocks)/SUM(blocks))*(100/SQRT(SQRT(COUNT(blocks))))
   FROM sys.dba_free_space
   GROUP BY tablespace_name, file_id, relative_fno;
```

Quellcode 11.15: Berichtsskript der Sicht für freien Speicherplatz

```
CREATE VIEW dba_file_data AS
SELECT
     a.name tablespace,a.dflminext min_extents,
     a.dflmaxext max_extents,
     a.dflinit init,a.dflincr next,
     a.dflextpct pct_increase, d.name datafile,
     b.blocks datafile_size, c.maxextend max_extend,
     c.inc ext_incr
 FROM
     ts$ a, file$ b, filext$ c, v$dbfile d
 WHERE
     a.ts#=b.ts# and b.file#=c.file# and b.file#=d.file#
/
```
Quellcode 11.16: Skript zum Erstellen einer Sicht für Datendateien

```
REM
REM  Name     : dbfiles.sql
REM  FUNCTION : Document file sizes and locations
REM  Use      : From SQLPLUS
REM  MRA 05/16/99 Added autoextend monitoring
REM  MRA 10/14/99 Added temp file monitoring 9i
REM
CLEAR COMPUTES
COLUMN file_name       FORMAT A51       HEADING 'File Name'
COLUMN tablespace_name FORMAT A15       HEADING 'Tablespace'
COLUMN meg             FORMAT 99,999.90 HEADING 'Megabytes'
COLUMN status          FORMAT A10       HEADING 'Status'
COLUMN autoextensible  FORMAT A3        HEADING 'AE?'
COLUMN maxmeg          FORMAT 99,999    HEADING 'Max|Megabytes'
COLUMN Increment_by    FORMAT 9,999     HEADING 'Inc|By'
SET LINES 130 PAGES 47 VERIFY OFF FEEDBACK OFF
START title132 'DATABASE DATA FILES'
SPOOL rep_out\&db\datafile
BREAK ON tablespace_name SKIP 1 ON REPORT
COMPUTE SUM OF meg ON tablespace_name
COMPUTE SUM OF meg ON REPORT
SELECT
     tablespace_name,file_name,
     bytes/1048576 meg,
     status,autoextensible,
     maxbytes/1048576 maxmeg,
     increment_by
 FROM
     dba_data_files
UNION
SELECT
     tablespace_name,file_name,
     bytes/1048576 meg,
     status,autoextensible,
     maxbytes/1048576 maxmeg,
     increment_by
 FROM
     dba_temp_files
 ORDER BY
     tablespace_name
/
SPOOL OFF
SET VERIFY ON FEEDBACK ON
TTITLE OFF
CLEAR COLUMNS
CLEAR COMPUTES
PAUSE Press Enter to continue
```

Quellcode 11.17: Skript zur Dokumentation von Datendateien für Tablespaces

```
rem
rem Name     : mapper.sql
rem Function : create an extent map for a specific tablespace
rem            Based on a technique from DBA Handbook
rem Mike Ault 7/19/96 TUSC
rem
SET PAGES 47 LINES 132 VERIFY OFF FEEDBACK OFF
COLUMN file_id                        HEADING 'File|id'
COLUMN value          NEW_VALUE dbblksiz NOPRINT
COLUMN meg            FORMAT 9,999.99 HEADING 'Meg'
COLUMN partition_name FORMAT a30      HEADING 'Partition|Name'
SELECT value FROM v$parameter WHERE name='db_block_size';
START title132 '&&ts Mapping Report'
SPOOL rep_out/&db/ts_map
SELECT
     'free space' owner, '      ' object,'Not Part.' partition
     file_id, block_id, blocks,
     (blocks*&dbblksiz)/(1024*1024) meg
 FROM
     dba_free_space
 WHERE
     tablespace_name=UPPER('&&ts')
UNION
SELECT
     SUBSTR(owner,1,20), SUBSTR(segment_name, 1,32),partition_name
     file_id, block_id, blocks,
     (blocks*&dbblksiz)/(1024*1024) meg
 FROM
     dba_extents
 WHERE
     tablespace_name = UPPER('&&ts')
 ORDER BY 3,4;
SPOOL OFF
UNDEF ts
SET PAGES 22 LINES 80 VERIFY ON FEEDBACK ON
CLEAR COLUMNS
TTITLE OFF
```

Quellcode 11.18: Skript zur Dokumentation des freien Platzes in den Extents eines Tablespaces

```
rem NAME    : Seq_rep.sql
rem
rem HISTORY :
rem Date      Who                       What
rem --------  ----------------------   --------------
rem 5/10/93   Mike Ault                 Creation
rem 5/16/99   Mike Ault                 Verified for Oracle8i
rem FUNCTION: Generate report on Sequences
rem INPUTS  :
rem           1 - Sequence Owner or Wild Card
rem           2 - Sequence Name or Wild Card
rem
rem **************************************************************
SET HEADING OFF VERIFY OFF PAUSE OFF
PROMPT ** Sequence Report **
PROMPT
PROMPT Percent signs are wild
ACCEPT sequence_owner char PROMPT 'Enter account to report on (or pct
sign):';
ACCEPT sequence_name char PROMPT 'Enter sequence to report on (or pct
sign):';
PROMPT
PROMPT Report file name is SEQUENCE.LIS
SET HEADING ON
SET LINESIZE 80 PAGESIZE 56 NEWPAGE 0 TAB OFF SPACE 1
SET TERMOUT OFF VERIFY OFF FEEDBACK OFF
BREAK ON sequence_owner SKIP 2
COLUMN sequence_owner  FORMAT A10    HEADING 'Sequence|Owner'
COLUMN sequence_name   FORMAT A16    HEADING 'Sequence|Name'
COLUMN min_value                     HEADING 'Minimum'
COLUMN max_value                     HEADING 'Maximum'
COLUMN increment_by    FORMAT 999    HEADING 'Inc'
COLUMN cycle_flag                    HEADING 'Cycle'
COLUMN order_flag                    HEADING 'Order'
COLUMN cache_size      FORMAT 99999  HEADING 'Cache'
COLUMN last_number     FORMAT 99999  HEADING 'Last|Value'
START title80 "SEQUENCE REPORT"
SPOOL rep_out/&&db/seq_rep
SELECT
     sequence_owner,sequence_name,
     min_value,max_value,
     increment_by,
     DECODE(cycle_flag,'Y','YES','N','NO') cycle_flag,
     DECODE(order_flag,'Y','YES','N','NO') order_flag,
     cache_size,last_number
 FROM
     dba_sequences
 WHERE
     sequence_owner LIKE UPPER('&sequence_owner') AND
     sequence_name LIKE UPPER('&sequence_name')
 ORDER BY
     1,2;
SPOOL OFF
SET LINESIZE 80 PAGESIZE 22 NEWPAGE 0 TAB ON SPACE 1
SET TERMOUT ON VERIFY ON FEEDBACK ON
CLEAR BREAKS
CLEAR COLUMNS
TTITLE OFF
```

Quellcode 11.19: Berichtsskript für Sequenzen

```
REM
REM NAME        : SYNONYM.SQL
REM PURPOSE     : GENERATE REPORT OF A USERS SYNONYMS
REM USE         : FROM SQLPLUS
REM Limitations : None
REM Revisions   :
REM Date          Modified by       Reason for change
REM 12/MAY/93     Mike Ault         Initial Creation
REM 15/Jun/97     Mike Ault         Verified for Oracle8
REM 16/May/99     Mike Ault         Verified for Oracle8i
REM
PROMPT Percent signs are Wild Cards
PROMPT
ACCEPT own PROMPT 'Enter the user who owns synonym: '
SET PAGES 56 LINES 130 VERIFY OFF FEEDBACK OFF TERM OFF
START title132 "Synonym Report"
SPOOL rep_out/&&db/synonym
COLUMN host      FORMAT a24  HEADING "Connect String"
COLUMN owner     FORMAT a15
COLUMN table     FORMAT a35
COLUMN db_link   FORMAT a6   HEADING Link
COLUMN username  FORMAT a15
SELECT
     a.owner, synonym_name ,
     table_owner ||'.'|| table_name "Table" ,
     b.db_link,username,host
 FROM
     dba_synonyms a,
     dba_db_links b
 WHERE
     a.db_link = b.db_link(+) AND
     a.owner LIKE UPPER('&own');
SPOOL OFF
SET PAGES 22 LINES 80 VERIFY ON FEEDBACK ON TERM ON
CLEAR COLUMNS
CLEAR BREAKS
TTITLE OFF
```
Quellcode 11.20: Berichtsskript für Synonyme

```
REM
REM NAME        : DBLINK_REP.SQL
REM FUNCTION    : GENERATE REPORT OF DATABASE LINKS
REM USE         : FROM SQLPLUS
REM Limitations : None
REM MRA 10/14/01  Verified for Oracle9i
REM
SET PAGES 58 LINES 80 VERIFY OFF TERM OFF
START title80 "Db Links Report"
SPOOL rep_out/&db/dblink_rep
COLUMN host      FORMAT a18  HEADING "Connect|String"
COLUMN owner     FORMAT a8   HEADING "Creator"
COLUMN db_link   FORMAT a19  HEADING "DB Link|Name"
COLUMN username  FORMAT a8   HEADING "Connect|User"
COLUMN created   FORMAT a15  HEADING "Date|Created"
SELECT
     host,owner,db_link,username,
     to_char(created,'dd-mon-yy hh24:mi') created
 FROM
     dba_db_links
 ORDER BY
     owner,
     host;
SPOOL OFF
SET PAGES 22 LINES 80 VERIFY ON FEEDBACK ON TERM ON
CLEAR COLUMNS
TTITLE OFF
PAUSE Press Enter to continue
```
Quellcode 11.21: Berichtsskript für Datenbanklinks

```
REM
REM FUNCTION: create views required for rbk1 and rbk2 reports.
REM
REM
CREATE OR REPLACE VIEW rollback1 AS
SELECT
     d.segment_name, extents, optsize, shrinks,
     aveshrink, aveactive, d.status
 FROM
     v$rollname n,
     v$rollstat s,
     dba_rollback_segs d
 WHERE
     d.segment_id=n.usn(+)
     AND d.segment_id=s.usn(+)
;
CREATE OR REPLACE VIEW rollback2 AS
SELECT
     d.segment_name,extents,xacts,hwmsize,
     rssize,waits,wraps,extends,d.status
 FROM
     v$rollname n,
     v$rollstat s,
     dba_rollback_segs d
 WHERE
     d.segment_id=n.usn(+)
     AND d.segment_id=s.usn(+)
;
```
Quellcode 11.22: SQL-Skripten zum Generieren der Sichten ROLLBACK1 *und* ROLLBACK2

```
REM NAME        : RBK1.SQL
REM FUNCTION    : REPORT ON ROLLBACK SEGMENT STORAGE
REM FUNCTION    : USES THE ROLLBACK1 VIEW
REM USE         : FROM SQLPLUS
REM Limitations : None
REM
COLUMN hwmsize         FORMAT 9999999999 HEADING 'LARGEST TRANS'
COLUMN tablespace_name FORMAT a10        HEADING 'TABLESPACE'
COLUMN segment_name    FORMAT A10        HEADING 'ROLLBACK'
COLUMN optsize         FORMAT 9999999999 HEADING 'OPTL|SIZE'
COLUMN shrinks         FORMAT 9999       HEADING 'SHRINKS'
COLUMN aveshrink       FORMAT 9999999999 HEADING 'AVE|SHRINK'
COLUMN aveactive       FORMAT 9999999999 HEADING 'AVE|TRANS'
COLUMN waits           FORMAT 99999      HEADING 'WAITS'
COLUMN wraps           FORMAT 99999      HEADING 'WRAPS'
COLUMN extends         FORMAT 9999       HEADING 'EXTENDS'
rem
BREAK ON REPORT
COMPUTE AVG OF AVESHRINK ON REPORT
COMPUTE AVG OF AVEACTIVE ON REPORT
COMPUTE AVG OF SHRINKS ON REPORT
COMPUTE AVG OF WAITS ON REPORT
COMPUTE AVG OF WRAPS ON REPORT
COMPUTE AVG OF EXTENDS ON REPORT
COMPUTE AVG OF HWMSIZE ON REPORT
SET FEEDBACK OFF VERIFY OFF LINES 132 PAGES 58
@title132 "ROLLBACK SEGMENT STORAGE"
SPOOL rep_out\&db\rbk1
rem
SELECT
     a.SEGMENT_NAME,a.OPTSIZE,a.SHRINKS,
     a.AVESHRINK,a.AVEACTIVE,b.HWMSIZE,
     b.WAITS,b.WRAPS,b.EXTENDS,A.STATUS
 FROM rollback1 a, rollback2 b
 WHERE A.SEGMENT_NAME=B.SEGMENT_NAME
 ORDER BY segment_name;
SPOOL OFF
CLEAR COLUMNS
TTITLE OFF
SET FEEDBACK ON VERIFY ON LINES 80 PAGES 22
PAUSE Press enter to continue

REM
REM NAME        : RBK2.SQL
REM FUNCTION    : REPORT ON ROLLBACK SEGMENT STATISTICS
REM FUNCTION    : USES THE ROLLBACK2 VIEW
REM USE         : FROM SQLPLUS
REM Limitations : None
REM
COLUMN segment_name FORMAT A8         HEADING 'ROLLBACK'
COLUMN extents      FORMAT 99999      HEADING 'EXTENTS'
COLUMN xacts        FORMAT 9999       HEADING 'TRANS'
COLUMN hwmsize      FORMAT 9999999999 HEADING 'LARGEST TRANS'
COLUMN rssize       FORMAT 9999999999 HEADING 'CUR SIZE'
COLUMN waits        FORMAT 99999      HEADING 'WAITS'
COLUMN wraps        FORMAT 99999      HEADING 'WRAPS'
COLUMN extends      FORMAT 9999       HEADING 'EXTENDS'
rem
SET FEEDBACK OFF VERIFY OFF lines 132 pages 58
BREAK ON REPORT
COMPUTE AVG OF WAITS ON REPORT
COMPUTE AVG OF WRAPS ON REPORT
COMPUTE AVG OF EXTENDS ON REPORT
COMPUTE AVG OF HWMSIZE ON REPORT
rem
```

```
@title132 "ROLLBACK SEGMENT STATISTICS"
SPOOL rep_out\&db\rbk2
rem
SELECT * FROM rollback2 ORDER BY segment_name;
SPOOL OFF
SET LINES 80 PAGES 20 FEEDBACK ON VERIFY ON
TTITLE OFF
CLEAR COLUMNS
PAUSE Press enter to continue

REM
REM NAME        : RBK3.SQL
REM FUNCTION    : REPORT ON ROLLBACK SEGMENT HEALTH
REM FUNCTION    : USES THE ROLLBACK1 and ROLLBACK2 VIEWs
REM USE         : FROM SQLPLUS
REM Limitations : None
REM
COLUMN hwmsize         FORMAT 9999999999 HEADING 'LARGEST TRANS'
COLUMN tablespace_name FORMAT a10        HEADING 'TABLESPACE'
COLUMN segment_name    FORMAT A10        HEADING 'ROLLBACK'
COLUMN optsize         FORMAT 9999999999 HEADING 'OPTL|SIZE'
COLUMN shrinks         FORMAT 9999       HEADING 'SHRINKS'
COLUMN aveshrink       FORMAT 9999999999 HEADING 'AVE|SHRINK'
COLUMN aveactive       FORMAT 9999999999 HEADING 'AVE|TRANS'
COLUMN waits           FORMAT 99999      HEADING 'WAITS'
COLUMN wraps           FORMAT 99999      HEADING 'WRAPS'
COLUMN extends         FORMAT 9999       HEADING 'EXTENDS'
rem
BREAK ON REPORT
COMPUTE AVG OF AVESHRINK ON REPORT
COMPUTE AVG OF AVEACTIVE ON REPORT
COMPUTE AVG OF SHRINKS ON REPORT
COMPUTE AVG OF WAITS ON REPORT
COMPUTE AVG OF WRAPS ON REPORT
COMPUTE AVG OF EXTENDS ON REPORT
COMPUTE AVG OF HWMSIZE ON REPORT
SET FEEDBACK OFF VERIFY OFF LINES 132 PAGES 47
@title132 "ROLLBACK SEGMENT HEALTH"
SPOOL rep_out\&db\rbk3
rem
SELECT c.tablespace_name, a.segment_name, a.optsize, a.shrinks,
     a.aveshrink,a.aveactive,
     b.hwmsize, b.waits, b.wraps, b.extends
 FROM rollback1 a, rollback2 b, dba_rollback_segs c
 WHERE a.segment_name=b.segment_name
     and c.segment_name=a.segment_name
 ORDER BY tablespace_name, segment_name;
SPOOL OFF
CLEAR COLUMNS
TTITLE OFF
SET FEEDBACK ON VERIFY ON LINES 80 PAGES 22
PAUSE Press enter to continue
```
Quellcode 11.23: Skripten zum Erstellen eines Berichts für Rollbacksegmente

```
REM undo_usage.sql
REM Function: reports undo usage for Oracle9i
REM
REM MRA 10/14/01 Initial Creation
REM
COLUMN undo_usage       FORMAT 99,999,999.999 HEADING 'Undo Usage|Blocks/Min'
COLUMN oer_old_errors   FORMAT 99,999,999     HEADING 'Undo|Old Errors'
COLUMN oer_space_errors FORMAT 9,999,999,999  HEADING 'Undo|Space Errors'
SET FEEDBACK OFF
@title80 'Undo Usage'
spool rep_out/&db/undo_usage
SELECT
     sum(undoblks)/sum((end_time-begin_time)*24*60) undo_usage,
     sum(ssolderrcnt) OER_old_errors,
     sum(nospaceerrcnt) OER_space_errors
 FROM
     v$undostat
 WHERE
     undoblks>0
/
spool off
SET FEEDBACK ON
TTITLE OFF
```
Quellcode 11.24: Berichtsskript für die Undo-Auslastung

```
rem Name    : TX_RBS.SQL
rem Purpose : Generate a report of active rollbacks
rem Use     : From SQL*Plus
rem History :
rem Date      Who              What
rem Sept 91   Lan Nguyen       Presented in paper at IOUG
rem           Walter Lindsey
rem 5/15/93   Mike Ault        Added Title80, sets and output
rem 1/04/97   Mike Ault        Verified against 7.3
rem 5/16/99   Mike Ault        Verified against Oracle8i
rem 10/14/01  Mike Ault        Verified against Oracle9i
rem                            reformated added curext, curblk
rem**********************************************************
COLUMN name        FORMAT a10    HEADING "Rollback|Segment"
COLUMN pid         FORMAT 99999  HEADING "Oracle|PID"
COLUMN spid        FORMAT 99999  HEADING "Sys|PID"
COLUMN curext      FORMAT 999999 HEADING "Current|Extent"
COLUMN curblk      FORMAT 999999 HEADING "Current|Block"
COLUMN transaction FORMAT A15    Heading 'Transaction'
COLUMN program     FORMAT a10    HEADING 'Program'
SET PAGES 56 LINES 80 VERIFY OFF FEEDBACK OFF
START title80 "Rollback Segments in Use"
SPOOL rep_out\&db\tx_rbs
SELECT
     r.name, l.Sid, p.spid,
     NVL(p.username, 'no transaction') "Transaction”,
     p.program "Program”,
     s.curext,s.curblk
 FROM
     v$lock l,
     v$process p,
     v$rollname r,
     v$rollstat s
 WHERE
         l.Sid = p.pid (+)
     AND TRUNC(l.id1(+) / 65536) = r.usn
     AND l.type(+) = 'TX'
     AND l.lmode(+) = 6
     AND r.usn=s.usn
     AND p.username is not null
 ORDER BY r.name;
SPOOL OFF
SET PAGES 22 LINES 80 VERIFY ON FEEDBACK ON
CLEAR COLUMNS
TTITLE OFF
```
Quellcode 11.25: Berichtsskript für aktive Rollbacks

```
rem*********************************************************
rem Name        : UNDO.SQL
rem Purpose     : Document rollback usage for a single
rem               transaction
rem Use         : Note: You must alter the UNDO script and add a
rem               call to the transaction at the indicated line
rem Restrictions: The database should be placed in DBA mode
rem               and this transaction should be the only one
rem               running.
rem History     :
rem  Date         Who              What
rem  Sept 91      Lan Nguyen       Presented in paper at IOUG
rem              Walter Lindsey
rem  5/15/93      Mike Ault        Changed to use one table
rem
SET FEEDBACK OFF TERMOUT OFF
COLUMN name FORMAT a40
DEFINE undo_overhead=54
DROP TABLE undo_data;
CREATE TABLE undo_data
     (
     tran_no number, start_writes number, end_writes number
     );
INSERT INTO undo_data
SELECT 1, SUM(writes),0 from v$rollstat;
SET FEEDBACK ON TERMOUT ON
rem
rem    !!! INSERT TRANSACTION HERE !!!
rem
SET FEEDBACK OFF TERMOUT OFF
UPDATE undo_data SET end_writes = SUM(writes) FROM v$rollstat;
 WHERE tran_no=1;
SET FEEDBACK ON TERMOUT ON
SELECT ((end-writes - start_writes) - &undo_overhead)
     "Number of Rollback Bytes Generated"
 FROM undo_data;
SET TERMOUT OFF FEEDBACK OFF
DROP TABLE undo_data;
```

Quellcode 11.26: Skript zur Ermittlung der Byteanzahl beim Rollback einer Transaktion

```
rem
rem Name     : log_stat.sql
rem
rem FUNCTION : Provide a current status for redo logs
rem
rem
COLUMN first_change# FORMAT 99999999 HEADING Change#
COLUMN group#        FORMAT 9,999    HEADING Grp#
COLUMN thread#       FORMAT 999      HEADING Th#
COLUMN sequence#     FORMAT 999,999  HEADING Seq#
COLUMN members       FORMAT 999      HEADING Mem
COLUMN archived      FORMAT a4       HEADING Arc?
COLUMN first_time    FORMAT a21      HEADING 'Switch|Time'
BREAK ON thread#
SET PAGES 60 LINES 131 FEEDBACK OFF
START title132 'Current Redo Log Status'
SPOOL rep_out\&db\log_stat
SELECT thread#,group#,sequence#,bytes,
     members,archived,
     status,first_change#,
     TO_CHAR(first_time, 'DD-MM-YYYY HH24:MI:SS') first_time
 FROM
     sys.v_$log
 ORDER BY
     thread#,
     group#;
SPOOL OFF
PAUSE Press Enter to continue
SET PAGES 22 LINES 80 FEEDBACK ON
CLEAR BREAKS
CLEAR COLUMNS
TTILE OFF
```
Quellcode 11.27: Beispielskript `log_stat.sql`

```
REM
REM NAME        :log_hist.sql
REM PURPOSE     :Provide info on logs for last 24 hours since last
REM PURPOSE     :log switch
REM USE         : From SQLPLUS
REM Limitations : None
REM MRA 10/14/01 Updated for Oracle9i
REM
COLUMN thread#        FORMAT 999    HEADING 'Thrd#'
COLUMN sequence#      FORMAT 99999  HEADING 'Seq#'
COLUMN first_change#                HEADING 'SCN Low#'
COLUMN next_change#                 HEADING 'SCN High#'
COLUMN archive_name   FORMAT a50    HEADING 'Log File'
COLUMN first_time     FORMAT a20    HEADING 'Switch Time'
COLUMN name           FORMAT a30    HEADING 'Archive Log'
SET LINES 132 FEEDBACK OFF VERIFY OFF
START title132 "Log History Report"
SPOOL rep_out\&db\log_hist
REM
SELECT
     X.recid,a.thread#,
     a.sequence#,a.first_change#,
     a.switch_change#,
     TO_CHAR(a.first_time,'DD-MON-YYYY HH24:MI:SS') first_time,
     x.name
 FROM
     v$loghist a, v$archived_log x
 WHERE
     a.first_time>
     (SELECT b.first_time-1
      FROM v$loghist b WHERE b.switch_change# =
         (SELECT MAX(c.switch_change#) FROM v$loghist c)) AND
         x.recid(+)=a.sequence#;
SPOOL OFF
SET LINES 80 VERIFY ON FEEDBACK ON
CLEAR COLUMNS
TTITLE OFF
PAUSE Press Enter to continue
```
Quellcode 11.28: Skript zur Überwachung der Wechsel von Redo-Logs

```
REM
REM NAME        : rdo_stat.sql
REM PURPOSE     : Show REDO latch statistics
REM USE         : from SQLPlus
REM Limitations : Must have access to v$_ views
REM
SET PAGES 56 LINES 78 VERIFY OFF FEEDBACK OFF
START title80 "Redo Latch Statistics"
SPOOL rep_out/&&db/rdo_stat
rem
COLUMN name     FORMAT a30      HEADING Name
COLUMN percent  FORMAT 999.999  HEADING Percent
COLUMN total                    HEADING Total
rem
SELECT
     l2.name,
     immediate_gets+gets Total,
     immediate_gets "Immediates",
     misses+immediate_misses "Total Misses",
     DECODE (100.*(GREATEST(misses+immediate_misses,1)/
     GREATEST(immediate_gets+gets,1)),100,0) Percent
 FROM
     v$latch l1,
     v$latchname l2
 WHERE
     l2.name like '%redo%'
     and l1.latch#=l2.latch# ;
rem
PAUSE Press Enter to continue

rem
rem Name     : Redo_stat.sql
rem
rem Function : Select redo statistics from v$sysstat
rem History  :
rem Who        What               Date
rem ---------  ----------------   -------
rem Mike Ault  Revised from V6    1/04/97
rem Mike Ault  Verified Oracle8   6/15/97
rem
COLUMN name   FORMAT a30          HEADING 'Redo|Statistic|Name'
COLUMN value  FORMAT 999,999,999  HEADING 'Redo|Statistic|Value'
SET PAGES 80 LINES 60 FEEDBACK OFF VERIFY OFF
START title80 'Redo Log Statistics'
SPOOL rep_out/&&db/redo_stat
SELECT
     name,
     value
 FROM
     v$sysstat
 WHERE
     name LIKE '%redo%'
 ORDER BY statistic#;
SPOOL OFF
SET LINES 24 FEEDBACK ON VERIFY ON
TTITLE OFF
CLEAR COLUMNS
CLEAR BREAKS
```

Quellcode 11.29: Berichtsskript für statistische Werte zum Redo

```
rem NAME     : dir_rep.sql
rem FUNCTION : Report on directories known by the database
rem HISTORY  : MRA 6/16/97   Created for Oracle8
rem            MRA 5/16/99   Verified for Oracle8i
rem            MRA 10/14/01  Verified for Oracle9i
rem
COLUMN owner            FORMAT a10  HEADING 'Owner'
COLUMN directory_name   FORMAT a15  HEADING 'Directory'
COLUMN directory_path   FORMAT a45  HEADING 'Full Path'
SET VERIFY OFF PAGES 58 LINES 78 FEEDBACK OFF
START title80 'Database Directories Report'
SPOOL rep_out\&db\dir_rep.lis
SELECT
     owner,directory_name,directory_path
 FROM
     dba_directories
 ORDER BY
     owner;
SPOOL OFF
SET VERIFY ON FEEDBACK ON
TTITLE OFF
CLEAR COLUMNS
```
Quellcode 11.30: Berichtsskript für Datenbankverzeichnisse

```
rem
rem NAME     : lib_rep.sql
rem FUNCTION : Document External Library Entries in Database
rem HISTORY  : MRA 6/16/97   Created
rem            MRA 10/14/01  Updated for Oracle9i
rem
COLUMN owner          FORMAT a8   HEADING 'Library|Owner'
COLUMN library_name   FORMAT a15  HEADING 'Library|Name'
COLUMN file_spec      FORMAT a30  HEADING 'File|Specification'
COLUMN dynamic        FORMAT a7   HEADING 'Dynamic'
COLUMN status         FORMAT a10  HEADING 'Status'
BREAK ON owner
SET FEEDBACK OFF VERIFY OFF LINES 78 PAGES 58
START title80 'Database External Libraries Report'
SPOOL rep_out\&db\lib_rep.lis
SELECT
     owner,library_name,file_spec,dynamic,status
 FROM
     dba_libraries
 ORDER BY
     owner;
SPOOL OFF
SET VERIFY ON FEEDBACK ON
TTITLE OFF
CLEAR COLUMNS
CLEAR BREAKS
```
Quellcode 11.31: Berichtsskript zur Dokumentation externen Bibliotheksspezifikationen

```
rem
rem NAME     : con_file.sql
rem FUNCTION : Document control file location and status
rem HISTORY  : MRA 6/16/97   Creation
rem            MRA 10/14/01  Verified against Oracle9i
rem
COLUMN name    FORMAT a60  HEADING 'Con|File|Location' WORD_WRAPPED
COLUMN status  FORMAT a7   HEADING 'Con|File|Status'
SET LINES 78 FEEDBACK OFF VERIFY OFF
START title80 'Control File Status'
SPOOL rep_out\&db\con_file.lis
SELECT
     name,status
 FROM
     v$controlfile;
SPOOL OFF
SET VERIFY ON FEEDBACK ON
TTITLE OFF
CLEAR COLUMNS
```
Quellcode 11.32: Berichtsskript zur Überwachung von Speicherort und Status für Steuerdateien

```
rem
rem NAME     : con_rec.sql
rem FUNCTION : Provide documentation of control file record stats
rem HISTORY  : MRA 6/16/97   Creation
rem            MRA 10/14/01  Verified for Oracle9i
rem
COLUMN type          FORMAT a18      HEADING 'Record Type'
COLUMN record_size   FORMAT 999999   HEADING 'Record|Size'
COLUMN records_used  FORMAT 999999   HEADING 'Records|Used'
COLUMN first_index   FORMAT 9999999  HEADING 'First|Index'
COLUMN last_index    FORMAT 9999999  HEADING 'Last|Index'
COLUMN last_recid    FORMAT 999999   HEADING 'Last|Record|ID'
SET LINES 80 PAGES 58 FEEDBACK OFF VERIFY OFF
START title80 'Control File Records'
SPOOL rep_out\&db\con_rec.lis
SELECT
     type,record_size,records_total,records_used,first_index,
     last_index,last_recid
 FROM
     v$controlfile_record_section;
SPOOL OFF
CLEAR COLUMNS
SET FEEDBACK ON VERIFY ON
TTITLE OFF
```
Quellcode 11.33: Berichtsskript zur Überwachung der Datensätze in Steuerdateien

```
REM
REM NAME        : init_ora_rct.sql
REM FUNCTION    : Re-create the instance init.ora file
REM USE         : GENERAL
REM Limitations : None
REM History     : MRA 11/7/95    Initial creation
REM               MRA 10/14/01   Updated for Oracle9i
REM
SET NEWPAGE 0 VERIFY OFF
SET ECHO OFF FEEDBACK OFF TERMOUT OFF PAGES 300 LINES 80 HEADING OFF
COLUMN name  FORMAT a80 WORD_WRAPPED
COLUMN dbname NEW_VALUE db NOPRINT
SELECT name dbname FROM v$database;
DEFINE OUTPUT = 'rep_out\&db\init.ora'
SPOOL &OUTPUT
SELECT '# Init.ora file FROM v$system_parameter' name FROM dual
UNION
SELECT '# generated on:'||sysdate name FROM dual
UNION
SELECT '# script by MRA 10/14/01 TUSC' name FROM dual
UNION
SELECT '#' name FROM dual
UNION
SELECT name||' = '||value name FROM v$system_parameter
 WHERE value IS NOT NULL and isdefault='FALSE';
SPOOL OFF
CLEAR COLUMNS
SET NEWPAGE 0 VERIFY OFF
SET TERMOUT ON PAGES 22 LINES 80 HEADING ON
SET TERMOUT ON
UNDEF OUTPUT
PAUSE Press Enter to continue
```
Quellcode 11.34: Skript zur Wiederherstellung der Datei `init<SID>.ora`

```
rem NAME     : waiters.sql
rem FUNCTION : Report on sessions waiting for locks
rem HISTORY  : MRA 1/12/96    Creation
rem            MRA 10/14/01   Updated for Oracle9i
rem
COLUMN busername       FORMAT a10     HEADING 'Holding|User'
COLUMN wusername       FORMAT a10     HEADING 'Waiting|User'
COLUMN bsession_id                    HEADING 'Holding|SID'
COLUMN wsession_id                    HEADING 'Waiting|SID'
COLUMN mode_held       FORMAT a10     HEADING 'Mode|Held'
COLUMN mode_requested  FORMAT 999999  HEADING 'Mode|Requested'
COLUMN lock_id1        FORMAT 999999  HEADING 'Lock|ID1'
COLUMN lock_id2        FORMAT a15     HEADING 'Lock|ID2'
COLUMN type                           HEADING 'Lock|Type'
SET LINES 132 PAGES 59 FEEDBACK OFF ECHO OFF
START title132 'Processes Waiting on Locks Report'
SPOOL rep_out/&db/waiters
SELECT
     holding_session bsession_id,
     waiting_session wsession_id,
     b.username busername,
     a.username wusername,
     c.lock_type type,
     mode_held, mode_requested,
     lock_id1, lock_id2
 FROM
     sys.v_$session b,
     sys.dba_waiters c,
     sys.v_$session a
 WHERE
     c.holding_session=b.sid and
     c.waiting_session=a.sid
/
SPOOL OFF
PAUSE press Enter to continue
CLEAR COLUMNS
SET LINES 80 PAGES 22 FEEDBACK ON
TTITLE OFF
```

Quellcode 11.35: Berichtsskript für Sitzungen, die auf die Freigabe von Sperren warten

```
rem NAME     : blockers.sql
rem FUNCTION : Show all processes causing a dead lock
rem HISTORY  : MRA 1/15/96   Created
rem            MRA 5/21/99   dba_locks becomes dba_lock in 8.1.5
rem            MRA 10/14/01  Verified for oracle9i
rem
COLUMN username        FORMAT a10  HEADING 'Holding|User'
COLUMN session_id                  HEADING 'SID'
COLUMN mode_held       FORMAT a10  HEADING 'Mode|Held'
COLUMN mode_requested  FORMAT a10  HEADING 'Mode|Requested'
COLUMN lock_id1        FORMAT a10  HEADING 'Lock|ID1'
COLUMN lock_id2        FORMAT a10  HEADING 'Lock|ID2'
COLUMN type                        HEADING 'Lock|Type'
SET LINES 132 PAGES 59 FEEDBACK OFF ECHO OFF
START title132 'Sessions Blocking Other Sessions Report'
SPOOL rep_out\&db\blockers
SELECT
     a.session_id, username,type,mode_held,mode_requested,
     lock_id1,lock_id2
 FROM
     sys.v_$session b,
     sys.dba_blockers c,
     sys.dba_lock a
 WHERE
     c.holding_session=a.session_id AND
     c.holding_session=b.sid
/
SPOOL OFF
PAUSE press Enter to continue
CLEAR COLUMNS
SET LINES 80 PAGES 22 FEEDBACK ON
```
Quellcode 11.36: Berichtsskript für Sitzungen, die Blockaden verursachen

```
rem Name     : ddl_lock.sql
rem Function : Document DDL Locks currently in use
rem History  : MRA 1/15/97   Creation
rem            MRA 5/21/99   Reformat, verify for 8i
rem
COLUMN owner           FORMAT a7    HEADING 'User'
COLUMN session_id      FORMAT 9999  HEADING 'SID'
COLUMN mode_held       FORMAT a7    HEADING 'Lock|Mode|Held'
COLUMN mode_requested  FORMAT a7    HEADING 'Lock|Mode|Request'
COLUMN type            FORMAT a20   HEADING 'Type|Object'
COLUMN name            FORMAT a21   HEADING 'Object|Name'
SET FEEDBACK OFF ECHO OFF PAGES 48 LINES 79
START title80 'Report on All DDL Locks Held'
SPOOL rep_out\&db\ddl_lock
SELECT
     NVL(owner,'SYS') owner, session_id,name,type,
     mode_held, mode_requested
 FROM
     sys.dba_ddl_locks
 ORDER BY 1,2,3
/
SPOOL OFF
PAUSE press Enter/return to continue
CLEAR COLUMNS
SET FEEDBACK ON PAGES 22 LINES 80
TTITLE OFF
```
Quellcode 11.37: Berichtsskript für DDL-Sperren

```
rem NAME     : int_lock.sql
rem FUNCTION : Document current internal locks
rem HISTORY  : MRA 1/15/96 Creation
rem
COLUMN username        FORMAT a10  HEADING 'Lock|Holder'
COLUMN session_id                  HEADING 'User|SID'
COLUMN lock_type       FORMAT a27  HEADING 'Lock Type'
COLUMN mode_held       FORMAT a10  HEADING 'Mode|Held'
COLUMN mode_requested  FORMAT a10  HEADING 'Mode|Requested'
COLUMN lock_id1        FORMAT a30  HEADING 'Lock/Cursor|ID1'
COLUMN lock_id2        FORMAT a10  HEADING 'Lock|ID2'
PROMPT 'ALL is all types or modes'
ACCEPT lock PROMPT 'Enter Desired Lock Type: '
ACCEPT mode PROMPT 'Enter Lock Mode: '
SET LINES 132 PAGES 59 FEEDBACK OFF ECHO OFF VERIFY OFF
BREAK ON username
START title132 'Report on Internal Locks Mode: &mode Type: &lock'
SPOOL rep_out\&db\int_locks
SELECT
     NVL(b.username,'SYS') username,
     session_id,lock_type,mode_held,
     mode_requested,lock_id1,lock_id2
 FROM
     sys.dba_lock_internal a, sys.v_$session b
 WHERE
     UPPER(mode_held) like UPPER('%&mode%') OR
     UPPER('&mode')='ALL' AND
     UPPER(lock_type) like UPPER('%&lock%') OR
     UPPER(mode_held) like UPPER('%&mode%') OR
     UPPER('&mode')='ALL' AND
     UPPER('&lock')='ALL' AND
     a.session_id=b.sid
 ORDER BY 1,2
/
SPOOL OFF
PAUSE press Enter to continue
SET LINES 80 PAGES 22 FEEDBACK ON VERIFY ON
CLEAR COLUMNS
CLEAR BREAKS
UNDEF LOCK
UNDEF MODE
```

Quellcode 11.38: Berichtsskript zur Dokumentation von aktuellen internen Sperren

```
rem
rem FUNCTION : Generate a report on session events by user
rem
rem NAME     : events.sql
rem HISTORY  : MRA 6/15/97   Created
rem            MRA 5/22/99   Verified on 8i
rem
COLUMN sid             HEADING Sid
COLUMN event           HEADING Event          FORMAT a40
COLUMN total_waits     HEADING Total|Waits
COLUMN total_timeouts  HEADING Total|Timeouts
COLUMN time_waited     HEADING Time|Waited
COLUMN average_wait    HEADING Average|Wait
COLUMN username        HEADING User
BREAK ON username
START title132 "Session Events By User"
SPOOL rep_out\&db\events
SET LINES 132 PAGES 59 VERIFY OFF FEEDBACK OFF
SELECT
     username, event,total_waits,total_timeouts,
     time_waited,average_wait
 FROM
     sys.v_$session_event a,
     sys.v_$session b
 WHERE
     a.sid= b.sid
 ORDER BY 1;
SPOOL OFF
PAUSE Press Enter to continue
CLEAR COLUMNS
CLEAR BREAKS
SET LINES 80 PAGES 22 VERIFY ON FEEDBACK ON
TTITLE OFF
```

Quellcode 11.39: Skript zur Erzeugung eines Ereignisberichts

```
rem Name     : workspace_status.sql
rem Function : Show status of workspaces in database
rem
rem History  : MRA 10/14/2001  Initial Creation
rem
COLUMN WORKSPACE         FORMAT a10  HEADING 'Workspace'
COLUMN owner             FORMAT a10  HEADING 'Owner'
COLUMN freeze_status     FORMAT a8   HEADING 'Freeze|Status'
COLUMN resolve_status    FORMAT a8   HEADING 'Resolve|Status'
COLUMN parent_workspace  FORMAT a10  HEADING 'Parent|Workspace'
COLUMN freeze_mode       FORMAT a8   HEADING 'Freeze|Mode'
start title80 'Workspace Status'
spool rep_out\&db\workspace_status
SELECT
     workspace,
     NVL(parent_workspace,'NONE') parent_workspace,
     owner,
     freeze_status,
     NVL(freeze_mode,'NONE') freeze_mode,
     resolve_status
 FROM
     dba_workspaces
/
spool off
ttitle off
```

Quellcode 11.40: Berichtsskript für den Status eines Arbeitsbereichs

Abbildung 11.1:

```
rem Name     : workspace_status.sql
rem Function : Show status of workspaces in database
rem
rem History  : MRA 10/14/2001  Initial Creation
rem
COLUMN WORKSPACE         FORMAT a10  HEADING 'Workspace'
COLUMN owner             FORMAT a10  HEADING 'Owner'
COLUMN freeze_status     FORMAT a8   HEADING 'Freeze|Status'
COLUMN resolve_status    FORMAT a8   HEADING 'Resolve|Status'
COLUMN parent_workspace  FORMAT a10  HEADING 'Parent|Workspace'
COLUMN freeze_mode       FORMAT a8   HEADING 'Freeze|Mode'
start title80 'Workspace Status'
spool rep_out\&db\workspace_status
SELECT
     workspace,
     NVL(parent_workspace,'NONE') parent_workspace,
     owner,
     freeze_status,
     NVL(freeze_mode,'NONE') freeze_mode,
     resolve_status
 FROM
     dba_workspaces
/
spool off
ttitle off
```

Quellcode 11.41: Berichtsskript für den Status eines Arbeitsbereichs

```
rem Name    : inv_obj.sql
rem Purpose : Show all invalid objects in database
rem Mike Ault 7/2/96
rem Mike Ault 10/14/01  verified for Oracle9i
rem
COLUMN object_name  FORMAT A20  HEADING 'Object|Name'
COLUMN owner        FORMAT a10  HEADING 'Object|Owner'
COLUMN last_time    FORMAT a18  HEADING 'Last Change|Date'
COLUMN object_type  FORMAT a15  Heading 'Object|Type'
SET LINES 80 FEEDBACK OFF PAGES 0 VERIFY OFF
START title80 'Invalid Database Objects'
SPOOL rep_out/&db/inv_obj
SELECT
     owner,
     object_name,
     object_type,
     TO_CHAR(last_ddl_time,'DD-MON-YY hh:mi:ss') Last_time
 FROM
     dba_objects
 WHERE
     status='INVALID'
/
PAUSE Press Enter to continue
SET LINES 80 FEEDBACK ON PAGES 22 VERIFY ON
CLEAR COLUMNS
TTITLE OFF
```
Quellcode 11.42: Berichtsskript für den Objektstatus

```
rem Name     : com_proc.sql
rem Function : Create a compile list for invalid procedures
rem
rem MRA 5/1/96
rem
DEFINE cr='chr(10)'
SET HEADING OFF PAGES 0 ECHO OFF TERMOUT OFF FEEDBACK OFF VERIFY OFF
SPOOL recompile.sql
SELECT 'ALTER '||object_type||' '||object_name||' COMPILE;'||&&cr||
'SHOW ERROR'
FROM dba_objects WHERE status='INVALID'
/
SPOOL OFF
SET HEADING ON TERMOUT ON FEEDBACK ON VERIFY ON
UNDEF cr
```
Quellcode 11.43: Dynamisches SQL-Skript zur Neukompilierung ungültiger Objekte.

```
REM Name: sqldrd.sql
REM Function: return the sql statements from the shared area with
REM Function: highest disk reads
REM History: Presented in paper 35 at IOUG-A 1997, converted for
REM use 6/24/97 MRA
REM
DEFINE access_level = 10000 (NUMBER)
COLUMN parsing_user_id FORMAT 9999999     HEADING 'User Id'
COLUMN executions      FORMAT 9999        HEADING 'Exec'
COLUMN sorts           FORMAT 99999       HEADING 'Sorts'
COLUMN command_type    FORMAT 99999       HEADING 'CmdT'
COLUMN disk_reads      FORMAT 999,999,999 HEADING 'Block Reads'
COLUMN sql_text        FORMAT a40         HEADING 'Statement' WORD_WRAPPED
SET LINES 130 VERIFY OFF FEEDBACK OFF
START title132 'SQL Statements With High Reads'
SPOOL rep_out/&db/sqldrd.lis
SELECT * FROM (SELECT
     parsing_user_id, executions,sorts,command_type,
     disk_reads,sql_text
 FROM
     v$sqlarea
 WHERE
     disk_reads > &&access_level
 ORDER BY
     disk_reads) WHERE rownum<26;
SPOOL OFF
SET LINES 80 VERIFY ON FEEDBACK ON
```

Quellcode 12.1: SQL-Code zur Ausgabe der vorderen SQL-Anweisungen bei Sortierung nach Lesezugriffen

```
col sql_text format a40 word_wrapped
col username format a15
col sid format 999999
col system_date format a20 heading 'System|Date'
set lines 132 pages 50
@title132 'Sorters Report'
spool rep_out\&db\sorters
select to_char(system_date,'dd-mon-yyyy hh24:mi') system_date,
sid,username,extents,blocks,sql_text from sorters
/
spool off

The view used in the above script looks like:

rem Code for view: SORTERS
CREATE OR REPLACE VIEW sorters as
select
SYSDATE system_date , s.sid, s.username
, b.extents, b.blocks, c.sql_text
from v$session s
, v$sort_usage b, v$sqlarea c
where s.saddr = b.session_addr
and s.sql_address = c.address;
```
Quellcode 12.2: Beispielskript zur Ermittlung von Angaben zu Sortiervorgängen

```
REM
REM NAME         : DD_CACHE.SQL
REM FUNCTION     : GENERATE REPORT ON DATA DICTIONARY CACHE
REM                CONDITION
REM USE          : FROM SQLPLUS
REM Limitations  : None
REM Revisions:
REM Date          Modified By     Reason For change
REM 21-AUG-1991     MIKE AULT     INITIAL CREATE
REM 27-NOV-1991     MIKE AULT     ADD % CALCULATION TO REPORT
REM 28-OCT-1992     MIKE AULT     ADD CALL TO TITLE PROCEDURE
REM 21-Jun-1997     MIKE AULT     Updated to ORACLE8
REM 07-nov-2001     MIKE AULT     Tested on 9i, reformatted
REM SET FLUSH OFF
REM SET TERM OFF
SET HEAD ON
SET PAGESIZE 59
SET LINESIZE 79
COLUMN parameter  FORMAT A20
COLUMN type       FORMAT a11
COLUMN percent    FORMAT 999.99   HEADING "%";
COLUMN gets       FORMAT 999,999  HEADING 'Gets'
COLUMN getmisses  FORMAT 999,999  heading 'Get|Misses'
COLUMN count      FORMAT 999,999  heading 'Count'
COLUMN usage      FORMAT 999,999  HEADING 'Usage'
ttitle "DATA DICTIONARY CACHE STATISTICS"
SPOOL rep_out/ddcache.lis
SELECT
     parameter,
     type,
     gets,
     getmisses,
     ( getmisses / gets * 100) percent,
     count,
     usage
 FROM
     v$rowcache
 WHERE
     gets > 100 AND
     getmisses > 0
 ORDER BY parameter;
SPOOL OFF
```
Quellcode 13.1: Bericht über den Cache des Data Dictionarys

```
REM Script to report on shared pool usage
REM
column shared_pool_used  format 9,999.99
column shared_pool_size  format 9,999.99
column shared_pool_avail format 9,999.99
column shared_pool_pct   format 999.99
@title80 'Shared Pool Summary'
spool rep_out\&db\shared_pool
select  sum(a.bytes)/(1024*1024) shared_pool_used,
        max(b.value)/(1024*1024) shared_pool_size,
       (max(b.value)/(1024*1024))-(sum(a.bytes)/(1024*1024))
shared_pool_avail,
       (sum(a.bytes)/max(b.value))*100 shared_pool_pct
 from v$sgastat a, v$parameter b
 where a.pool = 'shared pool' and a.name != 'free memory'
       and b.name = 'shared_pool_size'
spool off
ttitle off
```

Quellcode 13.2: Berichtsskript zur Nutzung des gemeinsam nutzbaren Pools

```
rem FUNCTION: Creates summary of v_$sqlarea and dba_users for use
rem           in sqlmem.sql and sqlsummary.sql reports
rem
rem
create or replace view sql_summary as
select username, sharable_mem, persistent_mem, runtime_mem
 from  sys.v_$sqlarea a, dba_users b
 where a.parsing_user_id = b.user_id;
```

Quellcode 13.3: Skript zur Erstellung der Sicht für den SQL-Überblick

```
rem FUNCTION: Generate a summary of SQL Area Memory Usage
rem FUNCTION: uses the sqlsummary view.
rem           showing user SQL memory usage
rem sqlsum.sql
rem
column areas                           heading Used|Areas
column sharable    format 999,999,999  heading Shared|Bytes
column persistent  format 999,999,999  heading Persistent|Bytes
column runtime     format 999,999,999  heading Runtime|Bytes
column username    format a15          heading "User"
column mem_sum     format 999,999,999  heading Mem|Sum
start title80 "Users SQL Area Memory Use"
spool rep_out\&db\sqlsum
set pages 59 lines 80
break on report
compute sum of sharable on report
compute sum of persistent on report
compute sum of runtime on report
compute sum of mem_sum on report
select username,
     sum(sharable_mem) Sharable,
     sum( persistent_mem) Persistent,
     sum( runtime_mem) Runtime ,
     count(*) Areas, sum(sharable_mem+persistent_mem+runtime_mem) Mem_sum
 from sql_summary
 group by username
 order by 2;
spool off
pause Press enter to continue
clear columns
clear breaks
set pages 22 lines 80
ttitle off
```
Quellcode 13.4: Skript zur Erstellung des Berichts mit dem SQL-Überblick

```
rem FUNCTION: Generate a report of SQL Area Memory Usage
rem           showing SQL Text and memory catagories
rem sqlmem.sql
rem
column sql_text      format a60  heading Text word_wrapped
column sharable_mem              heading Shared|Bytes
column persistent_mem            heading Persistent|Bytes
column loads                     heading Loads
column users         format a15  heading "User"
column executions                heading "Executions"
column users_executing           heading "Used By"
start title132 "Users SQL Area Memory Use"
spool rep_out\&db\sqlmem
set long 2000 pages 59 lines 132
break on users
compute sum of sharable_mem on users
compute sum of persistent_mem on users
compute sum of runtime_mem on users
select username users, sql_text, Executions, loads, users_executing,
     sharable_mem, persistent_mem
  from sys.v_$sqlarea a, dba_users b
 where a.parsing_user_id = b.user_id
       and b.username like upper('%&user_name%')
 order by 3 desc,1;
spool off
pause Press enter to continue
clear columns
clear computes
clear breaks
set pages 22 lines 80
```
Quellcode 13.5: Beispielbericht über die SQL-Bereiche

```
REM View to sort SQL into GOOD and GARBAGE
REM
CREATE OR REPLACE VIEW sql_garbage AS
SELECT b.username users,
     SUM(a.sharable_mem+a.persistent_mem) Garbage,
     TO_NUMBER(null) good
 FROM sys.v_$sqlarea a, dba_users b
 WHERE (a.parsing_user_id = b.user_id and a.executions<=1)
 GROUP BY b.username
UNION
SELECT DISTINCT b.username users,
     TO_NUMBER(null) garbage,
     SUM(c.sharable_mem+c.persistent_mem) Good
 FROM dba_users b, sys.v_$sqlarea c
 WHERE (b.user_id = c.parsing_user_id and c.executions>1)
 GROUP BY b.username;
```

Quellcode 13.6: Sicht für die Nützlichkeit von SQL-Codes

```
column garbage format a14 heading 'Non-Shared SQL'
column good format a14 heading 'Shared SQL'
column good_percent format a14 heading 'Percent Shared'
column users format a14 heading users
column nopr noprint
set feedback off
@title80 'Shared Pool Utilization'
spool rep_out\&db\sql_garbage
select 1 nopr, a.users users,
     to_char(a.garbage,'9,999,999,999') garbage,
     to_char(b.good,'9,999,999,999') good,
     to_char((b.good/(b.good+a.garbage))*100,'9,999,999.999')
       good_percent
 from sql_garbage a, sql_garbage b
 where a.users=b.users
     and a.garbage is not null and b.good is not null
union
select 2 nopr, '-------------' users,
     '--------------' garbage,
     '--------------' good,
     '--------------' good_percent
 from dual
union
select 3 nopr, to_char(count(a.users)) users,
     to_char(sum(a.garbage),'9,999,999,999') garbage,
     to_char(sum(b.good),'9,999,999,999') good,
     to_char(((sum(b.good)/(sum(b.good)+sum(a.garbage)))*100),
       '9,999,999.999') good_percent
 from   sql_garbage a, sql_garbage b
 where  a.users=b.users
     and a.garbage is not null and b.good is not null
 order by 1,3 desc
/
spool off
```

Quellcode 13.7: Beispielskript für einen Bericht über die Nützlichkeit von SQL-Code

```
set lines 140 pages 55
col num_of_times heading 'Number|Of|Repeats'
col text heading 'SubString - &&chars Characters'
col username format a10 heading 'User'
@title132 'Similar SQL'
spool rep_out\&db\similar_sql
select b.username,substr(a.sql_text,1,&&chars) text, count(a.sql_text)
num_of_times
 from v$sqlarea a, dba_users b
 where a.parsing_user_id=b.user_id
 group by b.username,substr(a.sql_text,1,&&chars)
 having count(a.sql_text)>1
 order by 3 desc
/
spool off
```

Quellcode 13.8: Bericht über ähnliche SQL-Anweisungen

```
rem FUNCTION: Report Stored Object Statistics
rem
column owner         format a11      heading Schema
column name          format a30      heading Object|Name
column namespace                     heading Name|Space
column type                          heading Object|Type
column kept          format a4       heading Kept
column sharable_mem  format 999,999  heading Shared|Memory
column executions    format 999,999  heading Executes
set lines 132 pages 47 feedback off
@title132 'Oracle Objects Report'
break on owner on namespace on type
spool rep_out/&db/o_stat
select OWNER, NAMESPACE, TYPE, NAME, SHARABLE_MEM, LOADS,
     EXECUTIONS, LOCKS, KEPT
 from   v$db_object_cache
 where type not in (
     'NOT LOADED','NON-EXISTENT','VIEW','TABLE','SEQUENCE','PACKAGE BODY')
     and executions>0 and loads>1 and kept='NO'
 order by owner,namespace,type,executions desc;
spool off
set lines 80 pages 22 feedback on
clear columns
clear breaks
ttitle off
```

Quellcode 13.9: Berichtsskript für die statistischen Werte von gespeicherten Objekten

```
REM Name: sqldrd.sql
REM Function: return the sql statements from the shared area with
REM Function: highest disk reads
REM History: Presented in paper 35 at IOUG-A 1997, converted for
REM use 6/24/97 MRA
REM
DEFINE access_level = 1000 (NUMBER)
COLUMN parsing_user_id FORMAT 9999999     HEADING 'User Id'
COLUMN executions      FORMAT 9999        HEADING 'Exec'
COLUMN sorts           FORMAT 99999       HEADING 'Sorts'
COLUMN command_type    FORMAT 99999       HEADING 'CmdT'
COLUMN disk_reads      FORMAT 999,999,999 HEADING 'Block Reads'
COLUMN sql_text        FORMAT a40    HEADING 'Statement' WORD_WRAPPED
SET LINES 130 VERIFY OFF FEEDBACK OFF
START title132 'SQL Statements With High Reads'
SPOOL rep_out/&db/sqldrd.lis
SELECT
     parsing_user_id, executions,
     sorts, command_type,
     disk_reads, sql_text
 FROM
     v$sqlarea
 WHERE
     disk_reads > &&access_level
 ORDER BY
     disk_reads;
SPOOL OFF
SET LINES 80 VERIFY ON FEEDBACK ON
```
Quellcode 13.10: Bericht über Zusammenhänge zwischen SQL-Code und Festplattenlesezugriffen

```
rem Title: libcache.sql
rem FUNCTION: Generate a library cache report
column namespace                      heading "Library Object"
column gets          format 9,999,999 heading "Gets"
column gethitratio   format 999.99    heading "Get Hit%"
column pins          format 9,999,999 heading "Pins"
column pinhitratio   format 999.99    heading "Pin Hit%"
column reloads       format 99,999    heading "Reloads"
column invalidations format 99,999    heading  "Invalid"
column db            format a10
set pages 58 lines 80
start title80 "Library Caches Report"
define output = rep_out\&db\lib_cache
spool &output
select  namespace, gets, gethitratio*100 gethitratio,
     pins, pinhitratio*100 pinhitratio, RELOADS,
     INVALIDATIONS
 from v$librarycache;
spool off
pause Press enter to continue
set pages 22 lines 80
ttitle off
undef output
```
Quellcode 13.11: Bericht über die Bibliothekscaches

```
rem
rem title:    ddcache.sql
rem FUNCTION: report on the v$rowcache table
rem HISTORY:  created sept 1995 MRA
rem
start title80 "DD Cache Hit Ratio"
spool rep_out\&db\ddcache
SELECT (SUM(getmisses)/SUM(gets))*100 RATIO
 FROM v$rowcache;
spool off
pause Press enter to continue
ttitle off
```

Quellcode 13.12: Überwachen des Trefferverhältnisses für das Data Dictionary

```
Rem db_cache_ad.sql
Rem from Oracle9i tuning
Rem Mike Ault Initial creation
Rem
column size_est format 999,999,999,999 heading 'Cache Size (m)'
column buf_est  format 999,999,999     heading 'Buffers'
column estd_rf  format 999.90          heading 'Estd Phys|Read Factor'
column estd_pr  format 999,999,999     heading 'Estd Phys| Reads'
SET LINES 80 PAGES 55
@title80 'DB Cache Advisor Report'
SPOOL rep_out/&db/db_cache_ad
SELECT
     size_for_estimate size_est,
     buffers_for_estimate buf_est,
     estd_physical_read_factor est_rf,
     estd_physical_reads est_pr
 FROM V$DB_CACHE_ADVICE
 WHERE name = 'DEFAULT'
     AND block_size = (SELECT value FROM V$PARAMETER
                       WHERE name = 'db_block_size')
     AND advice_status = 'ON';
SPOOL OFF
SET PAGES 22
TTITLE OFF
```

Quellcode 13.13: Bericht über DB_CACHE_ADVICE

```
rem block_usage.sql
rem
rem Mike AUlt - TUSC
rem
@title80 'Block Usage Inside SGA Block Buffers'
spool rep_out\&db\block_usage
SELECT decode(c.name,null,’UNUSED’,c.name) ts_name,
     a.file# file_number,
     COUNT(a.block#) Blocks,
     COUNT (DISTINCT a.file# || a.block#) Distinct_blocks
 FROM V$BH a, file$ b, ts$ c
 WHERE a.file#=b.file#(+)
     AND b.ts#=c.ts#(+)
 GROUP BY a.file#,decode(c.name,null,'UNUSED',c.name)
/
spool off
```
Quellcode 13.14: Skript zur Blockbenutzung

```
rem vbh_status.sql
rem
rem Mike Ault -- Tusc
rem
@title80 'Status of DB Block Buffers'
spool rep_out\&db\vbh_status
select status,count(*) number_buffers
 from v$bh
 group by status;
spool off
ttitle off
```

Quellcode 13.15: Skript für den `V$BH`*-Status*

```
rem dbwr_stat.sql
rem Mike Ault - TUSC 11/09/01 Created
rem
col name  format a46           heading 'DBWR Statistic'
col value format 9,999,999,999 heading 'Statistic Value'
set pages 40
@title80 'DBWR Statistic Report'
spool rep_out\&db\dbwr_stat
select a.name,a.value
 from (select name, value from v$sysstat
        where name not like '%redo%' and name not like '%remote%') a
 where (a.name like 'DBWR%' or a.name like '%buffer%'
     or a.name like '%write%' or a.name like '%summed%)
union
select class name, count value
 from v$waitstat
 where class='data block''
union
select name||' '||to_char(block_size/1024)||'K hit ratio',
     round(((1 - (physical_reads / (db_block_gets + consistent_gets))) * 100),3)
value
 from V$buffer_pool_statistics
union
select name||' '||to_char(block_size/1024)||'K free buffer wait',free_buffer_wait
value
 from V$buffer_pool_statistics
union
select name||' '||to_char(block_size/1024)||'K buffer busy wait',buffer_busy_wait
value
 from V$buffer_pool_statistics
union
select name||' '||to_char(block_size/1024)||'K write complete
wait',write_complete_wait value
 from V$buffer_pool_statistics
/
spool off
set pages 22
ttitle off
```

Quellcode 13.16: Bericht zum Abrufen der statistischen Werte für den DBWR-Prozess

```
REM
REM NAME        :FILE_EFF.SQL
REM PURPOSE     :GENERATE FILE IO EFFICIENCIES REPORT
REM USE         :FROM STATUS_REPORTS.COM
REM Limitations :MUST BE RUN FROM ORACLE DBA ACCOUNT
REM Revisions:
REM Date          Modified By    Reason For change
REM 10-JUL-1992     M. AULT      INITIAL CREATE
REM 07-JUN-1993     M.AULT       Added reads to writes, reformatted
REM 23-Jun-1997     M.Ault       kcffio went away, rewrote to use
REM                              existing views/tables
SET PAGES 58 NEWPAGE 0
SET LINES 131
COLUMN eff   FORMAT A6         HEADING '% Eff'
COLUMN rw    FORMAT 9,999,999  HEADING 'Phys Block|read/writes'
COLUMN ts    FORMAT A22        HEADING 'Tablespace Name'
COLUMN name  FORMAT A40        HEADING 'File Name'
START title132 "FILE IO EFFICIENCY"
BREAK ON ts
DEFINE OUTPUT = 'rep_out/&db/file_io.lis'
SPOOL &OUTPUT
SELECT
     f.tablespace_name ts,
     f.file_name name,
     v.phyreads+v.phywrts rw,
     TO_CHAR(DECODE(v.phyblkrd,0,null,
     ROUND(100*(v.phyrds+v.phywrts)/(v.phyblkrd+v.phyblkwrt),2))) eff
 FROM dba_data_files f, v$filestat v
 WHERE f.file_id=v.file#
 ORDER BY 1,file#;
SPOOL OFF
PAUSE Press return to continue
```
Quellcode 13.17: Bericht über die Dateieffizienz

```
REM
REM NAME     : DO_CALSTAT.SQL
REM FUNCTION :Generate calculated statisitics report using
REM FUNCTION :just_statistics procedure
REM USE     :FROM STATUS.SQL or SQLPLUS
REM Limitations      :
REM Revisions:
REM Date          Modified By     Reason For change
REM 05-MAY-1992     Mike Ault     Initial Creation
REM 23-JUN-1997     Mike Ault     Updated to V8
REM
SET PAGES 58  NEWPAGE 0
EXECUTE dba_utilities.running_stats(TRUE);
START title80 "CALCULATED STATISTICS REPORT"
DEFINE output = rep_out\&db\cal_stat.lis
SPOOL &output
SELECT * FROM dba_temp;
SPOOL OFF
```

Quellcode 13.18: Aufrufskript für `RUNNING_STATS`

```
REM
REM FUNCTION: Generate a summary of Disk Sort Area Usage
REM
REM disksort.sql
REM
COLUMN value NEW_VALUE bs NOPRINT
SELECT value FROM v$parameter WHERE name='db block size';
START title80 "Instance Disk Area Average Sizes"
SPOOL rep_out\&&db\disk_sort
SELECT
     Tablespace_name,
     COUNT(*) areas,   (SUM(total_blocks)/COUNT(*))*&&bs avg_sort_bytes
 FROM v$sort_segment
 GROUP BY tablespace_name;
SPOOL OFF
```
Quellcode 13.19: Beispiel für Sortierungsbericht

```
rem
rem Name: mts_disp.sql
rem Funktion: Bericht für Prozentsatz beschäftigter Dispatcher erstellen
rem History: MRA 10/11/96 erstellt
rem          MRA 11/24/01 geprüft und formatiert für Oracle9i
rem
COL protocol FORMAT a60 HEADING 'Dispatcher|Protocol'
COL busy     FORMAT 999.99 HEADING 'Percent|Busy'
rem
SET FEEDBACK OFF VERIFY OFF LINES 78 PAGES 58
START title80 'Dispatcher-Status'
SPOOL rep_out\&db\mts_disp
rem
SELECT network protocol,
      ((SUM(busy)/(SUM(busy)+SUM(idle)))*100) busy
FROM v$dispatcher
GROUP BY network;
rem
SPOOL OFF
SET FEEDBACK ON VERIFY ON LINES 22
TTITLE OFF
```
Quellcode 14.1: Beispielbericht für den Prozentsatz beschäftigter Dispatcher

```
rem
rem Name: mts_wait.sql
rem Funktion: Wartezeitbericht für die Dispatcher erstellen
rem History: MRA 10/11/96 erstellt
rem          MRA 11/25/01 geprüft in Oracle9i
rem
COLUMN network FORMAT a40 HEADING 'Dispatcher-|Protokoll'
COLUMN aw      FORMAT a32 HEADING 'Durchschnittl. Wartezeit %'
SET FEEDBACK OFF VERIFY OFF LINES 78 PAGES 55
START title80 'Dispatcher-Wartezeiten'
SPOOL rep_out\&db\mts_wait
SELECT
   NETWORK,
   DECODE (SUM(totalq),0,’no responses',
           SUM(wait)/SUM(totalq)*100||' sec wait/response') aw
FROM v$queue q, v$dispatcher d
WHERE q.type=’DISPATCHER' AND
      q.paddr = d.paddr
GROUP BY network;
SPOOL OFF
SET FEEDBACK ON VERIFY ON PAGES 80 LINES 22
TTITLE OFF
```

Quellcode 14.2: Bericht über die durchschnittliche Wartezeit

```
rem
rem Name: mts_serv.sql
rem Funktion: Bericht über Prozentsatz beschäftigter gemeinsamer Server
rem History: MRA 11/24/01 geprüft und formatiert für Oracle9i
rem
COL name   FORMAT a4     HEADING 'Name'
COL busy   FORMAT 999.99 HEADING 'Prozentsatz'
COL status FORMAT a13    HEADING 'Serverstatus'
COL messages             HEADING 'Meldungen'
COL bytes                HEADING 'Byte'
COL requests             HEADING 'Anforderungen'
rem
SET FEEDBACK OFF VERIFY OFF LINES 78 PAGES 58
START title80 'Serverstatus'
SPOOL rep_out\&db\mts_disp
rem
SELECT name,
      ((SUM(busy)/(SUM(busy)+SUM(idle)))*100) busy,
      MAX(status) status, MAX(messages) messages,
      MAX(bytes) bytes, MAX(requests) requests
FROM v$shared_server
GROUP BY name;
rem
SPOOL OFF
SET FEEDBACK ON VERIFY ON LINES 22
TTITLE OFF
```
Quellcode 14.3: Skript zur Überwachung gemeinsam nutzbarer Serverprozesse

```
#***************************************************************
# Name        : hot_backup
# Purpose     : Perform a hot backup of an Oracle Database
# Use         : sh hot_backup
# Limitations : Creates a read-consistent image, but doesn't back
#               up in-process transactions
#
# Revision History:
# Date          Who              What
# ---------     -----------      -------------------------------
# June 1993     K. Loney         Featured in Oracle Mag. Article
# 29-Jun-93     M. Ault          Modified, commented
# 02-Aug-93     M. Ault          Converted to UNIX script
# 03-Aug-93     M. Phillips      Added error detection
#*****************************************************************
#
ERROR="FALSE"
LOGFILE="$ORACLE_HOME/adhoc/scripts/hot_back_log"
while [ "$error"=FALSE ]
do
svrmgrl << ending1
     connect internal
     alter tablespace system begin backup;
     exit
ending1
     if ( tar cfv /oracle/backup /data/ORA_SYSTEM_1.DBF )
     then
          :
     else
          ERROR="TRUE";
          echo "Tar backup failed for ora_system1.dbf" >$LOGFILE
     fi
svrmgrl << ending2
connect internal
     alter tablespace system end backup;
     exit
ending2

dup_it="tar rv /oracle/backup"
svrmgrl << ending3
connect internal
     alter tablespace user_tables begin backup;
     exit
ending3
if ( $dup_it /data/ora_user_tables_1.dbf )
then
     :
else
     ERROR="TRUE";echo "Tar backup failed for ora_user_tables_1.dbf">>$LOGFILE
fi #we must still end backup for tablespaces
svrmgrl << ending4
     connect internal
     alter tablespace user_tables end backup;
     exit
ending4
# force write of all archive logs
svrmgrl << ending5
     connect internal
     alter system switch logfile;
     archive log all;
     exit
ending5
if ( cp /usr/oracle/oracle7/db_example.archives/*.arc *.oldarc )
then
     :
```

```
else
     ERROR="TRUE";echo "Copy of archive logs failed">>$LOGFILE
fi
# Now backup a control file
svrmgrl << ending6
     connect internal
     alter database example
     backup controlfile to
     '/usr/oracle/oracle7/db_example/ora_control.bac
     reuse;
     exit
ending6
if ( $dup_it /usr/oracle/oracle7/db_example/ora_control.bac )
then
     :
else
     ERROR="TRUE";echo “Tar backup failed for control file”>>$LOGFILE
fi
# now backup all archive logs
if ( $dup_it /usr/oracle/oracle7/db_example.archives/*.oldarc )
then
     :
else
     ERROR="TRUE";echo “Tar backup failed for archive files">>$LOGFILE
fi
# Now delete logs
if ( rm /usr/m_oracle/oracle7/db_examples.archives/*.oldarc;* )
then
     ERROR="TRUE"
else
     ERROR="TRUE";echo “Delete of archive files failed”>>$LOGFILE
fi
done
exit
done
```
Quellcode 15.1: Beispielskript für ein Online-Backup unter Unix mit der Korne-Shell

```
REM Script to create a hot backup script on UNIX
REM Created 6/23/98 MRA
REM
create table bu_temp (line_no number,line_txt varchar2(2000))
storage (initial 1m next 1m pctincrease 0);
truncate table bu_temp;
set verify off embedded off lines 1000 termout off long 1000
define dest_dir=&1;
declare
--
-- Declare cursors
--
-- Cursor to get all tablespace names
--
cursor get_tbsp is
select tablespace_name from dba_tablespaces;
--
-- cursor to create BEGIN BACKUP command
--
cursor bbu_com (tbsp varchar2) is
select
'alter tablespace '||tablespace_name||' begin backup;'
from dba_tablespaces where tablespace_name=tbsp;
--
-- Cursor to create HOST backup commands
--
cursor tar1_com (tbsp varchar2) is
select '! /bin/tar cvf - '||file_name
from dba_data_files where tablespace_name=tbsp
and file_id=(select min(file_id)from dba_data_files
where tablespace_name=tbsp);
--
cursor tar2_com (tbsp varchar2) is
select
file_name
from dba_data_files where tablespace_name=tbsp
and file_id>(select min(file_id) from dba_data_files
where tablespace_name=tbsp);
--
cursor tar3_com (tbsp varchar2) is
select '! /bin/tar cvf - '||file_name
from dba_data_files where tablespace_name=tbsp
and file_id=(select min(file_id)from dba_data_files
where tablespace_name=tbsp);
--
cursor comp_com (tbsp varchar2) is
select
'|compress -c
>&&dest_dir/'||tablespace_name||'_'||to_char(sysdate,'dd_mon_yy')||'.Z'||chr(
10)
from dba_tablespaces where tablespace_name=tbsp;
--
-- Cursor to create END BACKUP command
--
cursor ebu_com (tbsp varchar2) is
select
'alter tablespace '||tablespace_name||' end backup;' from
dba_tablespaces
where tablespace_name=tbsp;
--
-- Cursor to create redo log HOST backup commands
--
cursor tar1_rdo is
select '! /bin/tar cvf - '
from dual;
```

```
--
cursor tar2_rdo is
select
member||' '
from v$logfile;
--
cursor comp_rdo is
select
'|compress -c
>&&dest_dir/redo_logs_'||to_char(sysdate,'dd_mon_yy')||'.Z'||chr(10)
from dual;
--
-- Temporary variable declarations
--
tbsp_name varchar2(64);
line_num number:=0;
line_text varchar2(2000);
fetch_text varchar2(2000);
min_value number;
first_tbsp boolean;
temp_var varchar2(128);
--
-- Begin build of commands into temporary table
--
begin
--
-- first, create script header
--
line_num := line_num+1;
select 'REM Online Backup Script for '||name||' instance'
into line_text from v$database;
insert into bu_temp values (line_num,line_text);
line_num := line_num+1;
select 'REM Script uses UNIX tar format backup commands'
into line_text from dual;
insert into bu_temp values (line_num,line_text);
line_num := line_num+1;
select 'REM created on '||to_char(sysdate, 'dd-mon-yyyy hh24:mi')||' by user
'||user
into line_text from dual;
insert into bu_temp values (line_num,line_text);
line_num := line_num+1;
select 'REM developed by Mike Ault - TUSC 2-May-2001'
into line_text from dual;
insert into bu_temp values (line_num,line_text);
line_num := line_num+1;
select 'REM Script expects to be fed backup directory location on execution.'
into line_text from dual;
insert into bu_temp values (line_num,line_text);
line_num := line_num+1;
select 'REM Script should be re-run anytime physical structure of database
altered.'
into line_text from dual;
insert into bu_temp values (line_num,line_text);
line_num := line_num+1;
select 'REM '
into line_text from dual;
insert into bu_temp values (line_num,line_text);
line_num := line_num+1;
select 'spool &&dest_dir/log/hot_bu'||to_char(sysdate,'dd_mon_yy')||'.log'
into line_text from dual;
insert into bu_temp values (line_num,line_text);
line_num := line_num+1;
--
-- Now get tablespace names and loop through until all are handled
--
commit;
```

```
open get_tbsp;
first_tbsp:=TRUE;
loop
--
-- Get name
--
     fetch get_tbsp into tbsp_name;
     exit when get_tbsp%NOTFOUND;
--
-- Add comments to script showing which tablespace
--
     select 'REM' into line_text from dual;
     insert into bu_temp values (line_num,line_text);
     line_num:=line_num+1;
     select 'REM Backup for tablespace '||tbsp_name into line_text from dual;
     insert into bu_temp values (line_num,line_text);
     line_num:=line_num+1;
     select 'REM' into line_text from dual;
     insert into bu_temp values (line_num,line_text);
     line_num:=line_num+1;
--
-- Get begin backup command built for this tablespace
--
     open bbu_com (tbsp_name);
     fetch bbu_com into line_text;
     insert into bu_temp values (line_num,line_text);
     line_num:=line_num+1;
     close bbu_com;
--
-- The actual backup commands are per datafile, open cursor and loop
--
     open tar1_com (tbsp_name);
     open tar2_com (tbsp_name);
     open tar3_com (tbsp_name);
     open comp_com (tbsp_name);
     min_value:=1;
     line_text:=NULL;
     loop
          if min_value=1
          then
            if first_tbsp THEN
              fetch tar1_com into fetch_text;
              select trim(fetch_text) into line_text from dual;
            else
              fetch tar3_com into fetch_text;
              select trim(fetch_text) into line_text from dual;
          end if;
          else
            fetch tar2_com into fetch_text;
            exit when tar2_com%NOTFOUND;
            select trim(line_text)||' '||trim(fetch_text) into line_text from
dual;
          end if;
          first_tbsp:=FALSE;
          min_value:=min_value+1;
     end loop;
     fetch comp_com into fetch_text;
     select trim(line_text)||' '||trim(fetch_text) into line_text from dual;
     insert into bu_temp values (line_num,line_text);
     line_num:=line_num+1;
     close tar1_com;
     close tar2_com;
     close tar3_com;
     close comp_com;
--
-- Build end backup command for this tablespace
--
```

```
    open ebu_com(tbsp_name);
    fetch ebu_com into line_text;
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
    close ebu_com;
  end loop;
    close get_tbsp;
  --
  -- Backup redo logs, normally you won't recover redo logs you
  -- will use your current redo logs so current SCN information not lost
  -- commands just here for completeness
  --
    select 'REM' into line_text from dual;
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
    select 'REM Backup for redo logs' into line_text from dual;
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
    select 'REM Normally you will not recover redo logs' into line_text from dual;
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
    select 'REM' into line_text from dual;
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
  --
  -- Create host backup commands for all redo logs
  --
    open tar1_rdo;
    open tar2_rdo;
    open comp_rdo;
    min_value:=1;
    loop
      if min_value=1
      then
        fetch tar1_rdo into fetch_text;
        select trim(fetch_text) into line_text from dual;
      else
        fetch tar2_rdo into fetch_text;
        select trim(line_text)||' '||trim(fetch_text) into line_text from dual;
        exit when tar2_rdo%NOTFOUND;
      end if;
      min_value:=min_value+1;
    end loop;
    fetch comp_rdo into fetch_text;
    select trim(line_text)||' '||trim(fetch_text) into line_text from dual;
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
    close tar1_rdo;
    close tar2_rdo;
    close comp_rdo;
  --
  -- Now get all archive logs, performing a switch to be sure all
  -- required archives are written out
  --
    select 'REM' into line_text from dual;
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
    select 'REM Backup for archive logs' into line_text from dual;
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
    select 'REM' into line_text from dual;
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
    select 'alter system switch logfile;' into line_text from dual;
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
    select 'alter system archive log all;' into line_text from dual;
  insert into bu_temp values (line_num,line_text);
  line_num:=line_num+1;
--
-- The next command builds the actual backup command based on the
-- value of the log_archive_dest initialization parameter, it looks for the
-- last right square bracket in the name and just uses that section with
-- a wildcard
--
  temp_var:=null;
  select substr (value,1,instr(value,'/',-1,1)) into temp_var
  from v$parameter where name='log_archive_dest';
  if temp_var is not null
  then
    select '! compress '||substr (value,1,instr(value,'/',-1,1))||'/*'
    into line_text from v$parameter where name='log_archive_dest';
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
    select '! tar cvf - '||substr (value,1,instr(value,'/',-1,1))||'/*.Z'||
    '|compress -c >&&dest_dir/'||
    substr (value,instr(value,'/',-
1,1)+1,length(value))||'_'||to_char(sysdate,'dd_mon_yy')||'.Z'
    into line_text from v$parameter where name='log_archive_dest';
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
  else
    select 'REM no log_archive_dest specified' into line_text from dual;
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
  end if;
  temp_var:=null;
  select substr (value,10,instr(value,'/',-1,1)) into temp_var
  from v$parameter where name='log_archive_dest_1';
  if temp_var is not null
  then
    select '! compress '||substr (value,10,instr(value,'/',-1,1))||'/*'
    into line_text from v$parameter where name='log_archive_dest_1';
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
    select '! tar cvf - '||substr (value,10,instr(value,'/',-1,1))||'/*.Z'||
    '|compress -c >&&dest_dir/'||
    substr (value,instr(value,'/',-
1,1)+1,length(value))||'_'||to_char(sysdate,'dd_mon_yy')||'.Z'
    into line_text from v$parameter where name='log_archive_dest_1';
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
  else
    select 'REM no log_archive_dest_1 specified' into line_text from dual;
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
  end if;
--
-- Next, backup a control file just to be sure
-- we have a good one available that is current with this backup
--
  select 'alter database backup controlfile to
'||chr(39)||'&&dest_dir'||'/ora_cnbkp_'||to_char(sysdate,'dd_mon_yy')||'.bac'
||chr(39)||';'
  into line_text from dual;
  insert into bu_temp values (line_num,line_text);
  line_num:=line_num+1;
  select 'spool off'||chr(10) into line_text from dual;
  insert into bu_temp values (line_num,line_text);
  line_num:=line_num+1;
  commit;
end;
/
rem
```

```
rem Now generate output based on bu_temp table contents
rem
set verify off feedback off heading off termout off pages 0
set embedded on lines 1000
column line_no noprint
column dbname new_value db noprint
select value dbname from v$parameter where name='db_name';
spool rep_out/&db/thot_bu.sql
select * from bu_temp order by line_no;
spool off
rem directory syntax for UNIX
rem
! sed '1,$ s/ *$//g' rep_out/&db/thot_bu.sql>rep_out/&db/hot_bu.sql
rem
drop table bu_temp;
set verify on feedback on heading on termout on pages 22
set embedded off lines 80
clear columns
undef dest_dir
```

Quellcode 15.2: Beispielskript zur Erzeugung eines Skripts für Online-Backups unter Unix

```
REM Online Backup Script for AULTDB1 instance
REM Script uses UNIX tar format backup commands
REM created on 27-nov-2001 11:21 by user SYSTEM
REM developed by Mike Ault - TUSC 2-May-2001
REM Script expects to be fed backup directory location on execution.
REM Script should be re-run anytime physical structure of database altered.
REM
spool /opt/backup/aultdb1/log/hot_bu27_nov_01.log
REM
REM Backup for tablespace SYSTEM
REM
alter tablespace SYSTEM begin backup;
! /bin/tar cvf - /ora1/ORACLE/ORADATA/AULTDB1/SYSTEM01.DBF |compress -c
>/opt/backup/aultdb1/SYSTEM_27_nov_01.Z
alter tablespace SYSTEM end backup;
REM
REM Backup for tablespace RBS
REM
alter tablespace RBS begin backup;
! /bin/tar cvf - /ora2/ORACLE/ORADATA/AULTDB1/RBS01.DBF |compress -c
>/opt/backup/aultdb1/RBS_27_nov_01.Z
alter tablespace RBS end backup;
REM
REM Backup for tablespace USERS
REM
alter tablespace USERS begin backup;
! /bin/tar cvf - /ora3/ORACLE/ORADATA/AULTDB1/USERS01.DBF |compress -c
>/opt/backup/aultdb1/USERS_27_nov_01.Z
alter tablespace USERS end backup;
REM
REM Backup for tablespace TEMP
REM
alter tablespace TEMP begin backup;
! /bin/tar cvf - /ora4/ORACLE/ORADATA/AULTDB1/TEMP01.DBF |compress -c
>/opt/backup/aultdb1/TEMP_27_nov_01.Z
alter tablespace TEMP end backup;
REM
REM Backup for tablespace TOOLS
REM
alter tablespace TOOLS begin backup;
! /bin/tar cvf - /ora5/ORACLE/ORADATA/AULTDB1/TOOLS01.DBF |compress -c
>/opt/backup/aultdb1/TOOLS_27_nov_01.Z
alter tablespace TOOLS end backup;
REM
REM Backup for tablespace INDX
REM
alter tablespace INDX begin backup;
! /bin/tar cvf - /ora5/ORACLE/ORADATA/AULTDB1/INDX01.DBF |compress -c
>/opt/backup/aultdb1/INDX_27_nov_01.Z
alter tablespace INDX end backup;
REM
REM Backup for tablespace DRSYS
REM
alter tablespace DRSYS begin backup;
! /bin/tar cvf - /ora1/ORACLE/ORADATA/AULTDB1/DR01.DBF |compress -c
>/opt/backup/aultdb1/DRSYS_27_nov_01.Z
alter tablespace DRSYS end backup;
REM
REM Backup for tablespace PERFSTAT
REM
alter tablespace PERFSTAT begin backup;
! /bin/tar cvf - /ora1/ORACLE/ORADATA/AULTDB1/PERFSTAT.DBF |compress -c
>/opt/backup/aultdb1/PERFSTAT_27_nov_01.Z
alter tablespace PERFSTAT end backup;
REM
```

```
REM Backup for tablespace TEST_2K
REM
alter tablespace TEST_2K begin backup;
! /bin/tar cvf - /ora2/ORACLE/ORADATA/AULTDB1/TEST_2K.DBF |compress -c
>/opt/backup/aultdb1/TEST_2K_27_nov_01.Z
alter tablespace TEST_2K end backup;
REM
REM Backup for redo logs
REM Normally you will not recover redo logs
REM
! /bin/tar cvf - /ora6/ORACLE/ORADATA/AULTDB1/REDO011.LOG
/ora6/ORACLE/ORADATA/AULTDB1/REDO032.LOG
/ora7/ORACLE/ORADATA/AULTDB1/REDO021.LOG
/ora7/ORACLE/ORADATA/AULTDB1/REDO012.LOG
/ora8/ORACLE/ORADATA/AULTDB1/REDO031.LOG
/ora8/ORACLE/ORADATA/AULTDB1/REDO022.LOG |compress -c
>/opt/backup/aultdb1/redo_logs_27_nov_01.Z
REM
REM Backup for archive logs
REM
alter system switch logfile;
alter system archive log all;
host compress /ora9/ORACLE/ORADATA/AULTDB1/ARCHIVE/*
host tar cvrf - *.Z|compress>/tape1/_25_may_99.Z
alter database backup controlfile to
'/opt/backup/aultdb1/ora_cnbkp_27_nov_01.bac';
spool off
```
Quellcode 15.3: Beispielausgabe des generierten Skripts für Online-Backups

```
REM Script to create a hot backup recovery script on NT using ocopy
REM Created 6/23/98 MRA
REM
create table bu_temp (line_no number,line_txt varchar2(2000));
truncate table bu_temp;
set verify off embedded off esc ^
REM &&ora_home &&dest_dir
column dup new_value dup_it noprint
select ''||chr(39)||'&&ora_home'||'\ocopy '||chr(39)||'' dup
from dual;

declare
--
-- Declare cursors
--
-- Cursor to get all tablespace names
--
cursor get_tbsp is
select tablespace_name from dba_tablespaces;
--
-- Cursor to create recovery commands
--
cursor rec_com (tbsp varchar2) is
select
&&dup_it||' '||'&&dest_dir'||'\datafiles\'||tbsp||file_id||'.bck '||file_name
from dba_data_files where tablespace_name=tbsp;
--
-- Cursor to create redo log recovery commands
--
cursor rec_rdo (num number) is
select
&&dup_it||
'
'||'&&dest_dir'||'\logs'||substr(member,instr(member,'\LOG',2,1),instr(member
,'.',1,1))||' '||
member
from v$logfile order by group#;
--
-- Temporary variable declarations
--
tbsp_name varchar2(64);
line_num number:=0;
line_text varchar2(2000);
num number:=0;
--
-- Begin build of commands into temporary table
--
begin
--
-- first, create script header
--
line_num := line_num+1;
select 'REM Recovery Script for '||name||' instance'
into line_text from v$database;
insert into bu_temp values (line_num,line_text);
line_num := line_num+1;
select 'REM Script uses ocopy - NT format backup commands'
into line_text from dual;
insert into bu_temp values (line_num,line_text);
line_num := line_num+1;
select 'REM created on '||to_char(sysdate, 'dd-mon-yyyy hh24:mi')||' by user
'||user
into line_text from dual;
insert into bu_temp values (line_num,line_text);
line_num := line_num+1;
```

```
select 'REM developed for RevealNet by Mike Ault - DMR Consulting 15-Dec-
1998'
into line_text from dual;
insert into bu_temp values (line_num,line_text);
line_num := line_num+1;
select 'REM '
into line_text from dual;
insert into bu_temp values (line_num,line_text);
line_num := line_num+1;
select 'REM Script should be re-run anytime physical structure of database
altered.'
into line_text from dual;
insert into bu_temp values (line_num,line_text);
line_num := line_num+1;
select 'REM '
into line_text from dual;
insert into bu_temp values (line_num,line_text);
line_num := line_num+1;
--
-- Now get tablespace names and loop through until all are handled
--
open get_tbsp;
loop
--
-- Get name
--
     fetch get_tbsp into tbsp_name;
     exit when get_tbsp%NOTFOUND;
--
-- Add comments to script showing which tablespace
--
     select 'REM’ into line_text from dual;
     insert into bu_temp values (line_num,line_text);
     line_num:=line_num+1;
     select 'REM Recovery for tablespace '||tbsp_name into line_text from dual;
     insert into bu_temp values (line_num,line_text);
     line_num:=line_num+1;
     select 'REM' into line_text from dual;
     insert into bu_temp values (line_num,line_text);
     line_num:=line_num+1;
--
-- The actual recovery commands are per datafile, open cursor and loop
--
     open rec_com (tbsp_name);
     loop
          fetch rec_com into line_text;
          exit when rec_com%NOTFOUND;
          line_num:=line_num+1;
          insert into bu_temp values (line_num,line_text);
     end loop;
     close rec_com;
end loop;
close get_tbsp;
--
-- Recover redo logs, normally you won't recover redo logs you
-- will use your current redo logs so current SCN information not lost
-- commands just here for completeness uncomment commands below to
-- enable redo log recovery (not advised)
--
  select 'REM' into line_text from dual;
  insert into bu_temp values (line_num,line_text);
  line_num:=line_num+1;
  select 'REM Recovery for redo logs' into line_text from dual;
  insert into bu_temp values (line_num,line_text);
  line_num:=line_num+1;
  select 'REM Normally you will not recover redo logs' into line_text from dual;
  insert into bu_temp values (line_num,line_text);
```

```
    line_num:=line_num+1;
    select 'REM' into line_text from dual;
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
  --
  -- Create host backup commands for all redo logs
  --
    /*open rec_rdo(num);
    loop
        fetch rec_rdo into line_text;
        exit when rec_rdo%NOTFOUND;
        num:=num+1;
        line_num:=line_num+1;
        insert into bu_temp values (line_num,line_text);
    end loop;
    close rec_rdo;*/
  --
  -- Now recover all archive logs
  --
    line_num:=line_num+1;
    select 'REM' into line_text from dual;
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
    select 'REM Recovery for archive logs' into line_text from dual;
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
    select 'REM' into line_text from dual;
    insert into bu_temp values (line_num,line_text);
    line_num:=line_num+1;
  --
  -- The next command builds the actual recovery command based on the
  -- value of the log_archive_dest initialization parameter, it looks for the
  -- last right square bracket in the name and just uses that section with
  -- a wildcard
  --
    select &&dup_it||' '||'&&dest_dir'||'\archives\*.* '||value||'\*.*'
    into line_text from v$parameter where name='log_archive_dest';
    line_num:=line_num+1;
    insert into bu_temp values (line_num,line_text);
  end;
  /
  rem
  rem Now generate output based on bu_temp table contents
  rem
  set verify off feedback off heading off termout off pages 0
  set embedded on lines 132
  column db_name new_value db noprint
  column line_no noprint
  select name db_name from v$database;
  spool rep_out\&&db\rec_db.bat
  select * from bu_temp order by line_no;
  spool off
  rem
  rem get rid of bu_temp table
  rem
  drop table bu_temp;
  set verify on feedback on heading on termout on pages 22
  set embedded off lines 80 esc \
  clear columns
  undef ora_home
  undef dest_dir
  exit
```
Quellcode 15.4: Beispielskript zur Erzeugung eines Recovery-Skripts unter NT