



DOAG Regional-Konferenz München 02/2007

**Gibt es ein Leben nach der
Block Corruption?**

Impressum



◆ **Marco Patzwahl**
MuniQSoft GmbH
Grünwalder Weg 13 a
D-82008 Unterhaching b. München

◆ **Telefon: +49(0)89-67909040**
E-Mail m.patzwahl@muniqsoft.de
Internet: www.muniqsoft.de



- ◆ **Oracle Schulung (SQL, DBA, PL/SQL, Security,...)**
- ◆ **Oracle Consulting & Support**
- ◆ **Oracle Entwicklung & Lizenzvertrieb**

Definition

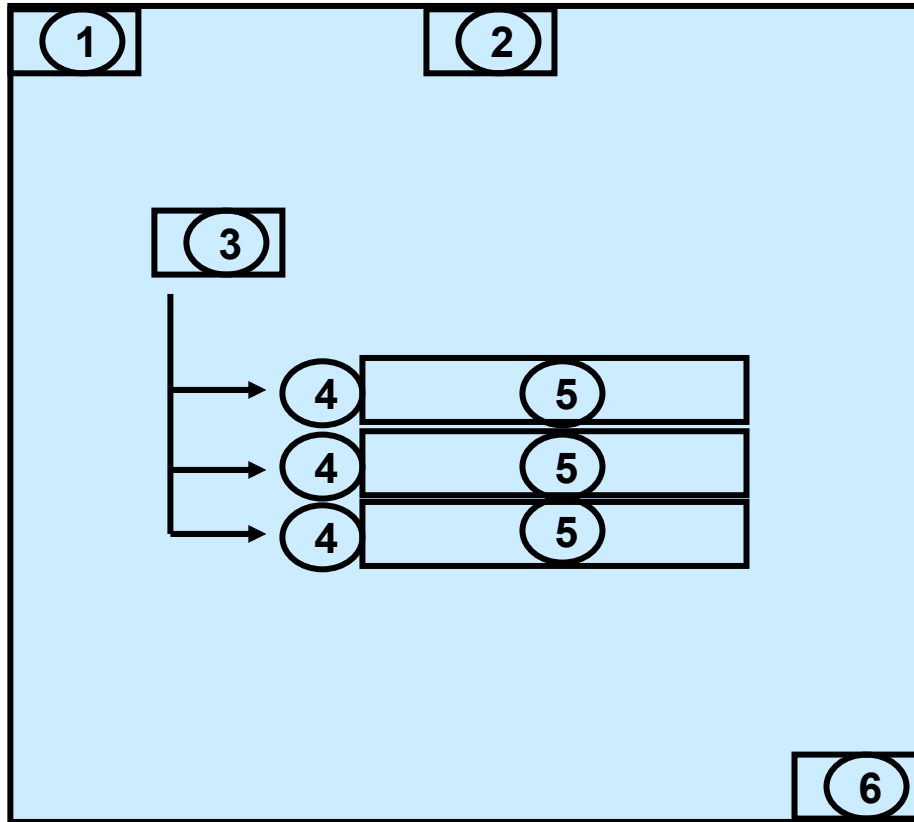
◆ Oracle Block:

- ▶ **Kleinste Speicherverwaltungseinheit für Oracle Tabellen, Indizes, Cluster oder Metadaten**
- ▶ **Übliche Größen: 1K – 32K**

◆ Oracle Block Corruption:

- ▶ **Wenn die Prüfsummen beim Lesen oder Schreiben eines Blocks nicht mehr stimmen, tritt bei Oracle eine Block Corruption auf**
- ▶ **Dieser Defekt lässt sich (derzeit) von Oracle nicht beseitigen**

Aufbau eines Blocks



(1) Blocktyp
(2) Verwaltungskopf (SCN, Initrans)

(3) Row Directory

(4) Row Header

(5) Zeilen

(6) Prüfsumme des Blocks

Block Corruption Fehlerausgabe

- ◆ Folgender Fehler erscheint, wenn ein Daten oder Index-Block nicht mehr gelesen werden kann:
 - ▶ ERROR:
ORA-01578: ORACLE-Datenblock beschädigt
(Datei Nr. 9, Block Nr. 4)
 - ▶ ORA-01110: Datendatei 9:
'D:\ORACLE\ORADATA\ORCL\VORSTANDSGEHALT_01.DBF',

Analyse, welche Datei ist betroffen?

- ◆ Wenn nur die Dateinummer im Fehlertext ausgegeben wurde, erhalten Sie hiermit den Dateinamen:

```
▶ SELECT name FROM v$datafile  
WHERE file#= <corrupt_file_id>;
```

- ◆ Man unterscheidet:

- ▶ Temp-Tablespace (NULL Problem☺)
- ▶ Indextablespace (Mittelschweres Problem)
- ▶ Datentablespace (Je nach Wichtigkeit der Daten)
- ▶ Datei Nr. 1 = System Tablespace,
Datei Nr. 2 = meist SYSAUX Tablespace (10g) (Big Problem)
- ▶ Undo/Rollback Tablespace (Big Problem=> Anzeigenmarkt der Computerwoche lesen, Handwerksberufe werden gesucht☺)

Welches Objekt ist betroffen?

- ◆ Anhand der Blocknummer lässt sich herausfinden, welches Objekt betroffen ist:

```
▶ SELECT
  tablespace_name, segment_type, owner,
  segment_name
FROM dba_extents
WHERE file_id = <corrupt_file_id>
and <corrupt_block_id> between block_id AND
block_id + blocks - 1;
```

Was tun, wenn Tabelle korrupt?

◆ 1. Daten aus einem Backup zurückspielen

◆ 2. Daten aus dem Index lesen:

```
▶ SELECT /*+ INDEX(<table_name>,<ic.index_name>)
*/ rowid, column_name
FROM <owner>.<table_name>
WHERE dbms_rowid.rowid_block_number(rowid)=
<corrupt_block_id>
AND dbms_rowid.ROWID_RELATIVE_FNO(rowid)=
<corrupt_file_id>;
```

◆ 3. Block wegwerfen

◆ 4. BBED (Block Browser and Editor) Oracle Support Tool!

Was tun, wenn Index korrupt?

- ◆ Für den Fall, dass nur ein Index betroffen ist, müssen Sie diesen neu anlegen.
- ◆ Jedoch kann kein REBUILD verwendet werden, da dieser vom defekten Original-Index-Block lesen müsste!
- ◆ Beispiel:
 - ▶ `DROP INDEX scott.pk_emp;`
 - ▶ `CREATE UNIQUE INDEX scott.pk_emp ON
scott.emp (empno)
TABLESPACE myindextbs;`

Recovery für sonstige Objekte

- ◆ **Variante 1: (ab Oracle 9i)**
- ◆ **Sie spielen nur die defekten Blöcke mit dem RMAN zurück:**
- ◆ **Beispiel: Block Nr. 327 in Datenfile 5 reparieren:**
 - ▶ `BLOCKRECOVER DATAFILE 5 BLOCK 327;`
- ◆ **Beispiel: Mehrere Blöcke recovern**
 - ▶ `BLOCKRECOVER
DATAFILE 7 BLOCK 233, 235
DATAFILE 4 BLOCK 101;`
- ◆ **Alle defekten Blöcke reparieren, die in der Liste (V\$BACKUP_CORRUPTION & V\$COPY_CORRUPTION) stehen:**
 - ▶ `BLOCKRECOVER CORRUPTION LIST RESTORE UNTIL TIME
'sysdate-3';`

Recovery für sonstige Objekte

- ◆ Variante 2:

- ◆ Sie spielen ein Backup der kompletten Datei ein und führen ein Recovery aus

- ◆ Beispiel (Syntax ab 9i gültig):

- ▶ RMAN> sql "ALTER DATABASE DATAFILE 6 OFFLINE IMMEDIATE";

- ▶ RMAN> RESTORE DATAFILE 6;

- ▶ RMAN> RECOVER DATAFILE 6;

- ▶ RMAN> sql "ALTER DATABASE DATAFILE 6 ONLINE";

Am defekten Block vorbeilesen

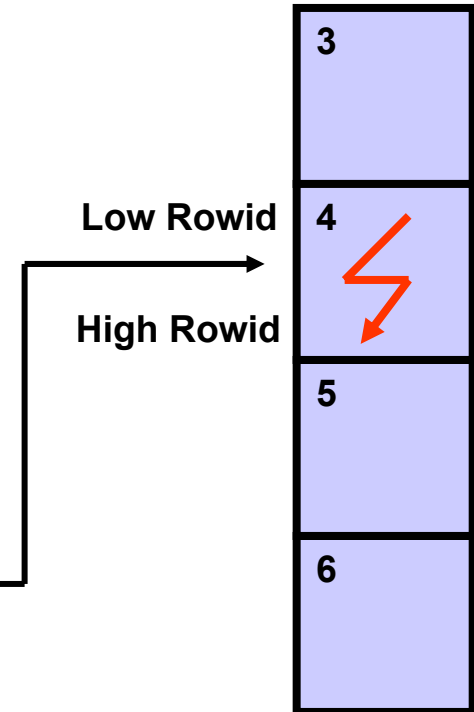
◆ Kleinste Rowid im defekten Block ermitteln:

```
▶ SELECT dbms_rowid.rowid_create  
  (1, <OBJ_ID>, <RelFileID>,  
  <BL_ID>, 0) LOW_ROWID  
FROM DUAL;
```

◆ Beispiel:

```
▶ SELECT dbms_rowid.rowid_create  
  (1, 25119, 9, 4, 0) LOW_ROWID  
FROM DUAL;
```

```
▶ AAAD/XAAGAAAAAMAAA
```



Am defekten Block vorbeilesen (ff)

- ◆ `SELECT dbms_rowid.rowid_create
 (1, <DATA_OBJECT_ID>, <RFN>, <BL>+1, 0) HI_RID
FROM DUAL;`

- ◆ **Beispiel:**
 - ▶ `SELECT dbms_rowid.rowid_create(1,25119,9,5,0) HI_RID
FROM DUAL;`

→ AAAGifAAJAAAAAFAAA

 - ▶ `CREATE TABLE test_neu AS SELECT /*+ ROWID(A) */ *
FROM scott.emp A
WHERE rowid < 'AAD/XAAGAAAAAMAAA';`

 - ▶ `INSERT INTO test_neu
SELECT /*+ ROWID(A) */ * FROM scott.emp A
WHERE rowid >= 'AAAGifAAJAAAAAFAAA';`

"Block Corruption" durch NOLOGGING

- ◆ **Wenn Sie DML Operationen mit der Option NOLOGGING durchgeführt haben, werden die Blockänderungen nicht mehr vollständig in die Redologs übertragen**
- ◆ **Konsequenzen:**
 - ▶ **Physical Standby Datenbanken markieren die betroffenen Blöcke als korrupt**
 - ▶ **Im Falle eines Absturzes der Produktivdatenbank, kann die Tabelle mit Hilfe der Redologs/Archivelogs nicht mehr aufgebaut werden**
- ◆ **Lösung:**
 - ▶ **Kein NOLOGGING verwenden sondern sogar FORCE LOGGING einsetzen, oder neues Backup der betroffenen Dateien erzeugen**

Block Corruption im Verwaltungsapparat

- ◆ Wenn die fehlerhafte Datei die Nummer 1 besitzt, handelt es sich um den **SYSTEM-Tablespace**.
- ◆ Hier kann durch Einspielen der beiden Haupt-Verwaltungsskripte versucht werden, den Fehler zu lösen:
 - ▶ `CONNECT sys/sys as sysdba`
 - ▶ `@?/rdbms/admin/catalog.sql`
 - ▶ `@?/rdbms/admin/catproc.sql`
- ◆ Wenn jedoch die Basis-Tabellen (z.B. `user$`, `obj$`, `tab$`, `idl_ub1$`, ...) betroffen sind, nützt das Einspielen meist nichts

Block Corruption Event

- ◆ Ein Event ermöglicht, defekte Blöcke zu überspringen:
 - ▶ `ALTER SESSION SET EVENTS '10231 TRACE NAME CONTEXT FOREVER, LEVEL 10';`
- ◆ Nun Full Table Scan durchführen bzw. CTAS:
 - ▶ `SELECT * FROM scott.emp;`
 - ▶ `CREATE TABLE scott.emp_neu AS
SELECT * FROM scott.emp;`
- ◆ Wenn der Event für die ges. DB gesetzt werden soll
 - ▶ `ALTER SYSTEM SET EVENTS '10231 TRACE NAME
CONTEXT FOREVER, LEVEL 10';`

Was tun, nach einer Block Corruption?

- ◆ **Defekte Hardware austauschen (Speicher, Prozessor, Controller, Festplatten(-Cache))**
- ◆ **Bei Software-Fehler Oracle Support kontaktieren (z.B. TAR öffnen)**
- ◆ **Bei Korruption des Verwaltungsapparats, DB zeitnah neu aufbauen**
- ◆ **Bei defekten Platten, neue Tablespaces auf neuen Platten anlegen und Objekte verschieben**
- ◆ **Letztes Backup bereitlegen**
- ◆ **Nach weiteren korrupten Blöcken suchen**

Block Corruption erkennen (Übersicht)

◆ Strukturanalysen:

- ▶ `ANALYZE TABLE ... VALIDATE STRUCTURE [ONLINE];`
- ▶ `ANALYZE INDEX ... VALIDATE STRUCTURE [ONLINE];`

◆ Full Exports (mit Full Table Scans auf allen Applikations-Tabellen) auf /dev/null (UNIX) oder null (Windows)

- ▶ `exp userid=system/manager full=y file=NULL
consistent=y direct=y compress=n
log=c:\exp.log`

◆ DB Verify zum Testen von Backups

Block Corruption erkennen (Übersicht)

- ◆ **Diverse Initialisierungsparameter**
- ◆ **RMAN CHECK LOGICAL / VALIDATE**
- ◆ **DBMS_REPAIR Package**

Tabellenanalyse

- ◆ **ANALYZE TABLE [<schema>.]<indexname>
VALIDATE STRUCTURE [CASCADE]
[INTO [<schema.>]<tabellename>]
[ONLINE /*9i/];**
- ◆ **Fehlerhafte Blöcke werden nur in einer Trace-Datei dokumentiert (USER_DUMP_DEST)**
- ◆ **Blöcke werden selbst nicht als korrupt markiert**
- ◆ **Bei der Option *ONLINE* werden keine Statistiken erzeugt, bei *OFFLINE* schon**

Indexanalyse

- ◆ **ANALYZE INDEX [<schema>.]<indexname>
VALIDATE STRUCTURE [CASCADE]
[INTO [<schema.>]<tabellename>]
[ONLINE /*9i/];**
 - ▶ **CASCADE prüft sowohl dass Index- als auch Tabellen-
einträge zueinander passen.**
 - ▶ **ONLINE ermöglicht während der Ausführung auch noch
DML Befehle auf der Tabelle.**
 - ▶ **OFFLINE sperrt das Objekt!**
 - ▶ **Mit der INTO-Klausel können Zeilen einer Partition, die
falsch abgespeichert wurden, in eine Tabelle geschrieben
werden. Diese Tabelle wird mit dem Skript UTLVALID.SQL
angelegt.**

Initialisierungsparameter

◆ Prüfsummen für Blöcke erzeugen

▶ **DB_BLOCK_CHECKSUM = true**

- True ist seit 9.0 Default, Performanceverlust ca. 1%
- Seit 8.1.5 werden auch Prüfsummen der Redolog-Blöcke erzeugt

▶ **DB_BLOCK_CHECKING = true**

- False ist Default (Nur Tablespace SYSTEM wird trotzdem geprüft)
- Performanceverlust ca. 1-10%
- Probleme werden früher erkannt
- Auf Produktivsystemen nur einsetzen, wenn vorher geprüft wurde, dass nicht bereits eine Corruption vorliegt

Undokumentierte Initialisierungsparameter

- ◆ **Achtung! Alle diese Parameter sind UNDOKUMENTIERT, der Einsatz erfolgt auf eigene Gefahr !!!**
- ◆ **Sie sollten vorher mit dem Oracle-Support besprechen, ob es andere Alternativen gibt**
- ◆ **Vorher ist anzuraten, noch ein Full-Backup durchzuführen**

Undokumentierte INIT.ORA Parameter

| | |
|-----------------------|--|
| Parametername: | _allow_resetlogs_corruption |
| Typ: | Boolean |
| Defaultwert: | FALSE |
| Wertebereich: | TRUE, FALSE |
| Bemerkung: | <p>Erlaubt den Start auch ohne funktionierende Redolog-Dateien. Dadurch kann die Datenbank korrupt werden!!! Machen Sie zuvor ein Backup der Controldateien</p> <pre>STARTUP MOUNT RECOVER DATABASE UNTIL CANCEL <CANCEL> ALTER DATABASE OPEN RESETLOGS;</pre> <p>Hier sollte die komplette Datenbank exportiert, gelöscht und dann wieder importiert werden.</p> <p>Kann z.B. verwendet werden, wenn keine Redolog-Dateien mehr vorhanden sind. Es wird die höchste SCN einfach für alle Dateien verwendet => SEHR Gefährlich!!!!!!</p> |

Undokumentierte INIT.ORA Parameter

| | |
|-----------------------|---|
| Parametername: | _allow_read_only_corruption |
| Typ: | Boolean |
| Defaultwert: | FALSE |
| Wertebereich: | TRUE, FALSE |
| Versionsinfo: | |
| Bemerkung: | <p>Erlaubt den READ-ONLY Start auch bei einer DB-Korruption (z.B. Fehlende Redologs). Auch dieser Parameter ist sehr gefährlich.</p> <p>Nach einem Einsatz sollte die komplette Datenbank exportiert, gelöscht und dann wieder importiert werden.</p> |

Undokumentierte INIT.ORA Parameter

| | |
|-----------------------|--|
| Parametername: | _corrupted_rollback_segments |
| Typ: | Boolean |
| Defaultwert: | FALSE |
| Wertebereich: | TRUE,FALSE |
| Versionsinfo: | |
| Bemerkung: | Datenbank lässt sich dann auch mit korrupten Rollback-Segmenten starten. Achtung! Transaktionen sind u.U. nicht vollständig zurückgerollt und damit korrupt |

Undokumentierte INIT.ORA Parameter

| | |
|-----------------------|--|
| Parametername: | _db_always_check_system_ts |
| Typ: | Boolean |
| Defaultwert: | TRUE |
| Wertebereich: | TRUE, FALSE |
| Versionsinfo: | Ab 8.1.7 |
| Bemerkung: | Erlaubt den Start auch bei einer DB-Korruption. Es werden dann keine Blockfehler im SYSTEM Tablespace mehr gemeldet Danach sollte die komplette Datenbank exportiert, gelöscht und dann wieder importiert werden. |

Prüfen der Blockstruktur mit DBVERIFY

- ◆ **DBVERIFY ist ein Utility zum Prüfen der Blockintegrität.**
- ◆ **Als Parameter wird angegeben, welche Daten-Datei geprüft werden soll.**
- ◆ **Es können nur Datenfiles, keine Control- oder Redologfiles geprüft werden.**
- ◆ **Unix erlaubt auch die Prüfung von Online-Dateien, unter Windows sind nur Offline Dateien erlaubt**

Parameter von DBVERIFY

◆ C:\> DBV

- ▶ **FILE=<dateiname>** → Name der zu prüfenden Datei
- ▶ **START=<n>** → Nummer des Startblocks. Default ist der erste im File.
- ▶ **END=<n>** → Nummer des letzten Blocks. Default ist der letzte im File.
- ▶ **BLOCKSIZE=<n>** → Blockgröße in Bytes. Default: 2048!
- ▶ **LOGFILE=<dateiname>** → Name der Log-Datei, in die die Ausgabe geschrieben werden soll. Default: Bildschirm
- ▶ **FEEDBACK=<n>** → Fortschrittsanzeige. Allen Pages wird ein "." ausgegeben. Default= 0 (ausgeschaltet)

Parameter von DBVERIFY (f)

◆ C:\>DBV

- ▶ **HELP=[Y|N]** → Ausgabe der Parameterinfos
- ▶ **PARFILE=<dateiname>** → Parameter für weitere Aufrufe können in ein Parameterfile geschrieben werden.

◆ Beispiele für den Aufruf von DBVERIFY:

- ▶ **DBV FILE=users01.dbf FEEDBACK=100
BLOCKSIZE=8192**
- ▶ **DBV FILE=system01.dbf LOGFILE=dbv_system.log
BLOCKSIZE=4096**

RMAN Blockprüfungen

◆ Beim RMAN Backup werden Blöcke beim Lesen auf Corruption geprüft

▶ Beispiel für die gesamte Datenbank + Archivelogs

```
▶ run {  
  allocate channel ch1 type disk;  
  backup check logical validate database  
  archivelog all;  
}
```

▶ Beispiel für die Datendateien 1-3

```
▶ run {  
  allocate channel ch1 type disk;  
  backup check logical validate datafile 1,2,3;  
}
```

RMAN Backups

- ◆ Wenn RMAN auf eine korrupte Datei stößt, bricht er sofort ab.
- ◆ Sie können einen Schwellenwert (für jede Datei) an defekten Blöcken einstellen, der noch toleriert wird
 - ▶ Beispiel:
 - ▶

```
RMAN> run {  
    SET MAXCORRUPT FOR DATAFILE 1 TO 5;  
    backup check logical datafile 1;  
}
```
- ◆ Jedoch ist zu bedenken, dass hier nur der Fehler ignoriert und nicht gelöst wird.

RMAN Blockprüfungen

- ◆ **In Version 8i werden durch den RMAN erkannte Blockprobleme nur in der Alert.log dokumentiert**
- ◆ **Ab Version 9i schreibt der RMAN die Information zusätzlich in V\$DATABASE_BLOCK_CORRUPTION**

DBMS_REPAIR Package

- ◆ **Zum Erkennen von Datenblockkorruption bei Tabellen und Indizes.**
- ◆ **Freelists können neu erstellt werden.**
- ◆ **Bisher ist nur die Erkennung von Blockfehlern implementiert, die Reparatur soll in einem der nächsten Releases erfolgen.**

DBMS_REPAIR - Schritt 1

◆ Erstellen von Tabellen, die die Fehler aufnehmen:

- ▶ BEGIN
- ▶ `SYS.DBMS_REPAIR.ADMIN_TABLES (`
`TABLE_NAME => 'REPAIR_TAB',`
`TABLE_TYPE => sys.dbms_repair.repair_table,`
`ACTION => sys.dbms_repair.create_action,`
`TABLESPACE => 'USERS');`
- ▶ `SYS.DBMS_REPAIR.ADMIN_TABLES (`
`TABLE_NAME => 'ORPHAN_KEY_TABLE',`
`TABLE_TYPE => sys.dbms_repair.orphan_Table,`
`ACTION => dbms_repair.create_action,`
`TABLESPACE => 'USERS');`
- ▶ END;

DBMS_REPAIR - Schritt 2

◆ Prüfung wie viele defekte Blöcke die Tabelle besitzt:

▶ SET serveroutput on

DECLARE

anz_corrupt number:=0;

BEGIN

SYS.DBMS_REPAIR.CHECK_OBJECT (

SCHEMA_NAME => 'SCOTT',

OBJECT_NAME => 'EMP',

REPAIR_TABLE_NAME => 'REPAIR_TAB',

corrupt_count => anz_corrupt);

DBMS_OUTPUT.PUT_LINE('Anz. korrupte Blöcke: '

||TO_CHAR (anz_corrupt));

END;

DBMS_REPAIR - Schritt 3

◆ Blöcke als fehlerhaft kennzeichnen:

▶ SET serveroutput on

```
DECLARE
```

```
  anz_fix INT:=0;
```

```
BEGIN
```

```
  SYS.DBMS_REPAIR.FIX_CORRUPT_BLOCKS (  
    SCHEMA_NAME => 'SCOTT',  
    OBJECT_NAME => 'EMP',  
    OBJECT_TYPE => sys.dbms_repair.table_object,  
    REPAIR_TABLE_NAME => 'REPAIR_TAB',  
    FIX_COUNT => anz_fix);
```

```
  DBMS_OUTPUT.PUT_LINE('num fix: ' ||  
    to_char(anz_fix));
```

```
END;
```

DBMS_REPAIR (Freelists)

◆ Freelist am Anfang des Extents wird neu erstellt:

▶ BEGIN

```
SYS.DBMS_REPAIR.REBUILD_FREELISTS (  
  SCHEMA_NAME => 'SCOTT',  
  OBJECT_NAME  => 'EMP',  
  OBJECT_TYPE  => dbms_repair.table_object);  
END;  
/
```

DBMS_REPAIR (f)

◆ Automatisches Überspringen von defekten Blöcken einschalten:

```
▶ BEGIN
    SYS.DBMS_REPAIR.SKIP_CORRUPT_BLOCKS (
        SCHEMA_NAME => 'SCOTT',
        OBJECT_NAME => 'EMP',
        OBJECT_TYPE => dbms_repair.table_object,
        FLAGS => dbms_repair.skip_flag);
END;
```

◆ Oder für einzelne Tabellen/Partitionen:

```
▶ exec DBMS_REPAIR.SKIP_CORRUPT_BLOCKS
    ('<schema>', '<table>');

▶ exec DBMS_REPAIR.SKIP_CORRUPT_BLOCKS
    ('<schema>', '<tablename>'. '<partition>');
```

DBMS_REPAIR (ff)

- ◆ Indexzeilen, die auf defekte Datenblöcke zeigen, werden hier angezeigt:

- ▶ SET serveroutput on

```
BEGIN
```

```
  SYS.DBMS_REPAIR.ADMIN_TABLES (  
    TABLE_NAME => 'ORPHAN_KEY_TABLE',  
    TABLE_TYPE => dbms_repair.orphan_table,  
    ACTION => dbms_repair.create_action,  
    TABLESPACE => 'USERS');
```

```
END;
```

```
/
```

Was tun wenn trotzdem nichts zu retten ist?

- ◆ Oracle Support hinzuziehen
- ◆ Wenn Datenbank nicht mehr gestartet werden kann und keine Backups vorhanden sind:
 - ▶ DUL (Data Unloader Utility)
 - Oracle Support bietet dieses Paket für einen Pauschalbetrag an
 - ▶ DUDE/JDUL von Firma Miracle
 - <http://www.ora600.org/>
 - ▶ Flucht-Porsche



Interessante Links

◆ Metalink:

- ▶ <http://metalink.oracle.com>
- ▶ **Note 28814.1 Handling Oracle Block Corruptions in Oracle7/8/8i/9i**
- ▶ **Note 338607.1 How To Check (Validate) If RMAN Backup(s) Are Good**

Impressum

- ◆ **Oracle Schulung (SQL, DBA, PL/SQL, Security,...)**
- ◆ **Oracle Consulting & Support**
- ◆ **Oracle Entwicklung & Lizenzvertrieb**
- ◆ **Marco Patzwahl**
MuniQSoft GmbH
Grünwalder Weg 13 a
D-82008 Unterhaching b. München
- ◆ **Telefon: +49(0)89-67909040**
Fax: +49(0)89-67909050
E-Mail m.patzwahl@muniqsoft.de
Internet: www.muniqsoft.de



Block Internas

◆ Aufbau einer Zeile:

