|  |  |  |
| --- | --- | --- |
| **How to identify all the Corrupted Objects in the Database with RMAN (Doc ID 472231.1)** | [To Bottom](https://support.oracle.com/epmos/faces/DocumentDisplay?id=472231.1&_adf.ctrl-state=ahlj0rovk_406&_afrLoop=350721248410209)  [To Bottom](https://support.oracle.com/epmos/faces/DocumentDisplay?id=472231.1&_adf.ctrl-state=ahlj0rovk_406&_afrLoop=350721248410209) |  |



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **In this Document**   |  |  | | --- | --- | |  | [Goal](https://support.oracle.com/epmos/faces/DocumentDisplay?id=472231.1&_adf.ctrl-state=ahlj0rovk_406&_afrLoop=350721248410209#GOAL) | |  | [Solution](https://support.oracle.com/epmos/faces/DocumentDisplay?id=472231.1&_adf.ctrl-state=ahlj0rovk_406&_afrLoop=350721248410209#FIX) | |  |  |  | | --- | --- | |  | [**Step 1: Identify the corrupt blocks**](https://support.oracle.com/epmos/faces/DocumentDisplay?id=472231.1&_adf.ctrl-state=ahlj0rovk_406&_afrLoop=350721248410209#aref_section21) | |  | [Step 2: Identify the corrupt segments](https://support.oracle.com/epmos/faces/DocumentDisplay?id=472231.1&_adf.ctrl-state=ahlj0rovk_406&_afrLoop=350721248410209#aref_section22) | |  |  |  |  |  | | --- | --- | --- | --- | |  | | [Identify corrupt NOLOGGING blocks in 12c](https://support.oracle.com/epmos/faces/DocumentDisplay?id=472231.1&_adf.ctrl-state=ahlj0rovk_406&_afrLoop=350721248410209#aref_section23) | | |  | [References](https://support.oracle.com/epmos/faces/DocumentDisplay?id=472231.1&_adf.ctrl-state=ahlj0rovk_406&_afrLoop=350721248410209#REF) | |       **Applies to:**  Oracle Database - Enterprise Edition - Version 8.1.7.0 to 12.1.0.2 [Release 8.1.7 to 12.1]  Information in this document applies to any platform.  **Goal**  How to identify all the corrupted segments in the database using RMAN.  **Solution**  **Step 1: Identify the corrupt blocks**  Populate the v$database\_block\_corruption view with information of all the corrupted blocks by executing the following command from RMAN:  RMAN> **backup validate check logical database;**  Notes:  This command is not doing a backup but checking the database for corruption.  From 11g and beyond the backup clause can be omitted and use "validate check logical database".  If this command fails due to missing files, the 'SKIP INACCESSIBLE' clause can be used to avoid the failure.  To make it faster,  RMAN can be configured to use PARALLELISM with multiple channels:    RMAN> configure device type disk parallelism 4;  RMAN> **backup validate check logical database;**  OR  RMAN> run {  allocate channel d1 type disk;  allocate channel d2 type disk;  allocate channel d3 type disk;  allocate channel d4 type disk;  **backup validate check logical database;**  }  **Output**  V$DATABASE\_BLOCK\_CORRUPTION is updated with the corrupt blocks.  In 11g RMAN generates a trace file with the details of the corruption description.  Example:  RMAN VALIDATE screen output:  File Status Marked Corrupt Empty Blocks Blocks Examined High SCN  ---- ------ -------------- ------------ --------------- ----------  6    FAILED 0              501          640             1950088    File Name: /oracle/dbs/users.dbf    Block Type Blocks Failing Blocks Processed    ---------- -------------- ----------------    Data       9              9    Index      0              0    Other      0              130  validate found one or more corrupt blocks  **See trace file /oracle/log/diag/rdbms/orcl/orcl/trace/orcl\_ora\_28424.trc for details**  Finished validate at <Date>  The trace file has output with the corruption description. This is an example for two of the corrupt blocks; one Physical corrupt block on file 6 block 9 and one Logical corrupt block on file 6 block 10 respectively:  Corrupt block relative dba: 0x01000009 (file 4, block 9)  Bad check value **found during validation**  Data in bad block:   type: 16 format: 2 rdba: 0x01000009   last change scn: 0x0000.00000000 seq: 0xff flg: 0x04   spare1: 0x0 spare2: 0x0 spare3: 0x0   consistency value in tail: 0x000010ff   check value in block header: 0xb4e0   computed block checksum: 0xa800  Reread of blocknum=9, file=/oracle/dbs/users.dbf found same corrupt data  Block Checking: DBA = 25165834, Block Type = KTB-managed data block  data header at 0x2b2deb49e07c  kdbchk: fsbo(144) wrong, (hsz 78)  Error backing up file 6, block 10: logical corruption  Corrupt blocks are listed in the view V$DATABASE\_BLOCK\_CORRUPTION:  SQL> **select \* from V$DATABASE\_BLOCK\_CORRUPTION;**            FILE#          BLOCK#          BLOCKS CORRUPTION\_CHANGE# CORRUPTIO  --------------- --------------- --------------- ------------------ ---------                6              10               1      8183236781662 LOGICAL                6              42               1                  0 FRACTURED                6              34               2                  0 CHECKSUM                6              50               1      8183236781952 LOGICAL                6              26               4                  0 FRACTURED  5 rows selected.  **Notes:**   * The CHECK LOGICAL option checks for both PHYSICAL and LOGICAL Block corruptions.  Reference [Note 840978.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=472231.1&id=840978.1) for an explanation of these corruption types. * When a logical corruption is found the alert log is updated with:   Error backing up file <file#>, block <block#>: logical corruption  In 11g a trace file is created with the corruption description.   * When a physical corruption is found the alert log is also updated with the corruption description:   Corrupt block relative dba: 0x01000009 (file 4, block 9)  Bad check value **found during validation**  Data in bad block:   type: 16 format: 2 rdba: 0x01000009   last change scn: 0x0000.00000000 seq: 0xff flg: 0x04   spare1: 0x0 spare2: 0x0 spare3: 0x0   consistency value in tail: 0x000010ff   check value in block header: 0xb4e0   computed block checksum: 0xa800  Reread of blocknum=9, file=/oracle/dbs/users.dbf found same corrupt data   * For a single or specific datafiles use "check logical validate datafile 1, 2". * To monitor the progress of the VALIDATE command run the next query:   select sid, serial#, context, sofar, totalwork,round(sofar/totalwork\*100,2) "%\_complete"  from v$session\_longops  where opname like 'RMAN%'    and opname not like '%aggregate%'    and totalwork != 0    and sofar <> totalwork;   * For a NOARCHIVELOG mode database using 10g version and lower, the database must be in MOUNT state; otherwise error ORA-19602 is produced. If it is not possible to close the database, use dbverify instead. This restriction is lifted in 11g. * From 11g and beyond: the validation of a single datafile can be made parallel by using the section clause. RMAN divides the file into sections and processes each file section in parallel. The next example divides the datafile 5 into 1gb sections when multiple channels are configured or allocated and each section is run in parallel (reference The Oracle Database Backup and Recovery User's Guide for more information):   backup validate check logical datafile 5 **SECTION SIZE 1024M**;   * From 11g onwards a range of blocks can be checked within a datafile using the BLOCK TO clause. The next command check blocks from 5 to 20 of datafile 1:   validate check logical datafile 1 **BLOCK** 5 **TO** 20;   * In Oracle8i corruptions found with the RMAN validate command are only reported in the alert log. Oracle8i users must search the alert.log for corruption errors in the time range during which the validate command was started and when it finished. Corruptions found are NOT reported back to the RMAN interface. In Oracle9i and beyond the view  V$DATABASE\_BLOCK\_CORRUPTION can be queried to determine what corruption, if any, was found by RMAN. As in Oracle8i, corruptions found are NOT reported back to the RMAN interface. * The corruption reported in V$DATABASE\_BLOCK\_CORRUPTION is updated with each RMAN backup validate run. To understand what is reported in this view, see the description of the view in the **Oracle® Database Reference** documentation.   **Step 2: Identify the corrupt segments**  The next query can be run to map each block to a segment in the database.  It will map each block from v$database\_block\_corruption to either a segment or if the block is free.  $ sqlplus / as sysdba  set pagesize 2000  set linesize 280  SELECT e.owner, e.segment\_type, e.segment\_name, e.partition\_name, c.file#       , greatest(e.block\_id, c.block#) corr\_start\_block#       , least(e.block\_id+e.blocks-1, c.block#+c.blocks-1) corr\_end\_block#       , least(e.block\_id+e.blocks-1, c.block#+c.blocks-1)         - greatest(e.block\_id, c.block#) + 1 blocks\_corrupted       , corruption\_type description    FROM dba\_extents e, v$database\_block\_corruption c   WHERE e.file\_id = c.file#     AND e.block\_id <= c.block# + c.blocks - 1     AND e.block\_id + e.blocks - 1 >= c.block#  UNION  SELECT s.owner, s.segment\_type, s.segment\_name, s.partition\_name, c.file#       , header\_block corr\_start\_block#       , header\_block corr\_end\_block#       , 1 blocks\_corrupted       , corruption\_type||' Segment Header' description    FROM dba\_segments s, v$database\_block\_corruption c   WHERE s.header\_file = c.file#     AND s.header\_block between c.block# and c.block# + c.blocks - 1  UNION  SELECT null owner, null segment\_type, null segment\_name, null partition\_name, c.file#       , greatest(f.block\_id, c.block#) corr\_start\_block#       , least(f.block\_id+f.blocks-1, c.block#+c.blocks-1) corr\_end\_block#       , least(f.block\_id+f.blocks-1, c.block#+c.blocks-1)         - greatest(f.block\_id, c.block#) + 1 blocks\_corrupted       , 'Free Block' description    FROM dba\_free\_space f, v$database\_block\_corruption c   WHERE f.file\_id = c.file#     AND f.block\_id <= c.block# + c.blocks - 1     AND f.block\_id + f.blocks - 1 >= c.block#  order by file#, corr\_start\_block#;    An output example is:  OWNER SEGMENT\_TYPE       SEGMENT\_NAME PARTITION\_ FILE# CORR\_START\_BLOCK# CORR\_END\_BLOCK# BLOCKS\_CORRUPTED DESCRIPTION  ----- ------------------ ------------ ---------- ----- ----------------- --------------- ---------------- -------------  SCOTT TABLE              EMP                         6                10              10                1  SCOTT TABLE PARTITION    ORDER        ORDER\_JAN      6                26              28                3                                                       6                29              29                1 Free Block  SCOTT TABLE              BONUS                       6                34              34                1                                                       6                35              35                1 Free Block  SCOTT TABLE              DEPT                        6                42              42                1 Segment Header  SCOTT TABLE              INVOICE                     6                50              50                1  Notes:   * If a corrupt block is in a dictionary managed tablespace and if the segment header block is corrupt, the above query may display the same block twice. * If a segment header block is corrupt in an ASSM tablespace, the above query displays the segment header block but subsequent corrupt blocks for the same object may not be displayed.   **Identify corrupt NOLOGGING blocks in 12c**  In 12c the NOLOGGING blocks identified by rman validate are in new view v$nonlogged\_block:  $ sqlplus / as sysdba    set echo on    select systimestamp  from dual;    select FILE#, BLOCK#, BLOCKS, to\_char(NONLOGGED\_START\_CHANGE#, '999999999999999') NONLOGGED\_START\_CHANGE#  from v$nonlogged\_block;    set pagesize 2000  set linesize 250    SELECT e.owner, e.segment\_type, e.segment\_name, e.partition\_name, c.file#  , greatest(e.block\_id, c.block#) corr\_start\_block#  , least(e.block\_id+e.blocks-1, c.block#+c.blocks-1) corr\_end\_block#  , least(e.block\_id+e.blocks-1, c.block#+c.blocks-1)  - greatest(e.block\_id, c.block#) + 1 blocks\_corrupted  , null description  FROM dba\_extents e, v$nonlogged\_block c  WHERE e.file\_id = c.file#  AND e.block\_id <= c.block# + c.blocks - 1  AND e.block\_id + e.blocks - 1 >= c.block#  UNION  SELECT null owner, null segment\_type, null segment\_name, null partition\_name, c.file#  , greatest(f.block\_id, c.block#) corr\_start\_block#  , least(f.block\_id+f.blocks-1, c.block#+c.blocks-1) corr\_end\_block#  , least(f.block\_id+f.blocks-1, c.block#+c.blocks-1)  - greatest(f.block\_id, c.block#) + 1 blocks\_corrupted  , 'Free Block' description  FROM dba\_free\_space f, v$nonlogged\_block c  WHERE f.file\_id = c.file#  AND f.block\_id <= c.block# + c.blocks - 1  AND f.block\_id + f.blocks - 1 >= c.block#  order by file#, corr\_start\_block#;          **References**  [NOTE:836658.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=472231.1&id=836658.1) - Identify the Corruption Extension for Block Corruption, Table/Index Inconsistency, Data Dictionary and Lost Writes  [NOTE:561010.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=472231.1&id=561010.1) - Which Blocks Will RMAN Check For Corruption Or Include In A Backupset?  [NOTE:819533.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=472231.1&id=819533.1) - How to identify the corrupt Object reported by ORA-1578 / RMAN / DBVERIFY  [NOTE:144911.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=472231.1&id=144911.1) - RMAN : Block-Level Media Recovery - Concept & Example  [NOTE:377146.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=472231.1&id=377146.1) - How to Check Archivelogs for Corruption using RMAN  [NOTE:840978.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=472231.1&id=840978.1) - Physical and Logical Block Corruptions. All you wanted to know about it.  [NOTE:338607.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=472231.1&id=338607.1) - How To Check (Validate) If RMAN Backup(s) Are Good  [NOTE:472231.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=472231.1&id=472231.1) - How to identify all the Corrupted Objects in the Database with RMAN  [NOTE:794505.1](https://support.oracle.com/epmos/faces/DocumentDisplay?parent=DOCUMENT&sourceId=472231.1&id=794505.1) - ORA-1578 / ORA-26040 Corrupt blocks by NOLOGGING - Error explanation and solution |