



Early Detection and Correction of Data Block Corruptions Using RMAN



By Krishna Kakatur

Recovery Manager (RMAN) is the best weapon to catch and correct Data Block Corruptions before they bother your database users.



In the middle of the night, you get a phone call from the users who have received an Oracle error when they tried to query supplier_master data.

```
MRP:SALES> select * from supplier_master where supl_id=111111;
select * from supplier_master where supl_id=111111
*
ERROR at line 1:
ORA-01578: ORACLE data block corrupted (file # 11, block # 10)
ORA-01110: data file 11: '/u01/app/Oracle/oradata/MRP/sales_data_01.dbf'
```

You have daily hot backups for the past 30 days, and you quickly try to recover from them. You find that this data block is corrupted in all the backup sets that you have. You understand that the corruption occurred two or three months back. As this record has not been queried for a long time, it was not noticed by the users. At this juncture, what are the options in front of you? More importantly, what COULD you have done to catch the data block corruptions as they crept into your database?

What is Data Block Corruption?

Data Block Corruption is a state of the database in which certain data blocks are corrupted, probably due to memory or I/O problems at the Operating System level, or due to network problems. Information stored in those corrupted blocks is lost and inaccessible. Until we recreate or repair the data by some means, Oracle gives error messages whenever an Oracle session tries to access those blocks. The error messages appear both in the program which accesses the database and in the alert.log file.

When Do Corruptions Get Noticed?

Data block corruptions can go unnoticed for several days, months or years. They are like cancer cells growing in the body. They are unearthed only when the data stored in the corrupted data block is queried. Thus, the seriousness lies not in the amount or frequency of the corruption, but in the obscurity. As a DBA, you should try to catch them as they occur, and recover the data blocks, before they affect the users and before you overwrite backup sets with correct data blocks.

Options for Recovering Data

Assuming RMAN is not used for backups, let us browse through various methods of recovering maximum possible amount of data.

- If the data can be re-created from other source or databases, you are the luckiest person on the earth. However, such databases need not have robust backup mechanisms.
- If you have a logical backup at object level or full export of the database, you can drop and recreate the object. However, you will be losing all the changes that were done after the backup was taken.
- If you don't have any such source, create another table with the same structure and issue INSERT INTO ... SELECT FROM statement from the corrupt table to insert into the new one, ignoring the corrupt block(s). You can probably make use of ROWIDs and issue INSERT statements within a PL/SQL block. An exception block will report the records that could not be retrieved. After this, you have to drop the original table, and re-create it using the data from the newly created table. However, in this case, you will be losing all the data in the corrupt blocks.
- Another popular approach is to make use of the Oracle-supplied DBMS_REPAIR package. Using this approach, you can address corruptions where possible, and also continue to use objects while you attempt to rebuild or repair them. We can also recover data from indexes, if any. However, as Oracle Documentation states, "DBMS_REPAIR is not a magic wand." Depending on the nature of the repair, you might lose data, and logical inconsistencies can be introduced.

As you can see, none of these recovery mechanisms will be able to get all the data from the corrupted blocks. We always have to compromise with some data loss. Using RMAN for backups, coupled with proper alerting scripts, we can recover such data losses due to data block corruptions.

Catching Corruptions Early

One of the finest features of Recovery Manager is to do backup at the data block level. As part of its default backup behavior, it touches each datafile block, performs verification checks, and if it finds any data block corruptions, it logs the address of the corrupt block and the type of corruption in the control file.

Let us take an example of backup with RMAN, without using a catalog database. In this case, RMAN uses control file to store configuration settings. A simple backup script is shown in Listing 1. This shell script expects one parameter, viz., level of backup. Note here that the default feature of RMAN is not to tolerate any data block corruptions in the backup sets. In other words, the backup is aborted when RMAN finds any data block corruption, raising an error.

A sample cron entry, for this backup process, will be something like this:

continued on page 32

```
00 21 * * * rman_backup.ksh 0

LISTING 1: rman_backup.ksh
#!/bin/ksh
# Script Name: rman_backup.ksh

# Set User parameters
# Note: These parameters may be sourced from a parameter file
export ORACLE_SID=MRP
export SYS_PASS=sysspasp
export NOTIFY_MAIL=dba@mycompany.com

# Set script parameters
export datetime=`date +%Y%m%d%H%M%S`
export BKP_PATH=/u01/app/Oracle/backup/$ORACLE_SID
export LOG_FILE=$BKP_PATH/r_$(datetime).log
export BKP_LEV=$1

# Start backup
rman >$LOG_FILE <<EOF
connect target sys/$SYS_PASS
run {
allocate channel d1 type disk format '${BKP_PATH}/r_$(BKP_LEV)_%U.bak';
backup incremental level $(BKP_LEV) tag $(ORACLE_SID)_$(BKP_LEV) database;
release channel d1;
}
EOF

# Check the status of backup operation
export ERR_COUNT=`cat $LOG_FILE | grep "RMAN-00569" | wc -l`
export ORA_COUNT=`cat $LOG_FILE | grep "ORA-" | wc -l`
if [[ ($ERR_COUNT -ne 0) || ($ORA_COUNT -ne 0) ]]; then
echo "Please check log file $LOG_FILE" | mail \
-s "RMAN backup failed for $ORACLE_SID" $NOTIFY_MAIL
echo "RMAN Backup process Failed."
else
echo "RMAN Backup process ran Successfully."
fi
```

Listing 1.

When there is a Data Block Corruption, the backup process aborts, and we can see a message similar to Listing 2 in the log file. The backup script can alert the DBA team, which can now fix the corruption immediately, long before users notice. We will discuss the recovery mechanisms in the next section.

```
LISTING 2: r_20030520213618.log
RMAN-00571: =====
RMAN-00569: ===== ERROR MESSAGE STACK FOLLOWS =====
RMAN-00571: =====
RMAN-03009: failure of backup command on d1 channel at 05/20/2003 21:36:26
ORA-19566: exceeded limit of 0 corrupt blocks for file
/u01/app/Oracle/oradata/MRP/sales_data_01.dbf
```

Listing 2.

If you want to proceed with the backup process even when there are data block corruptions, you may want to set MAXCORRUPT to the maximum allowable number of corrupt blocks, using the command

```
set maxcorrupt for datafile '/u01/app/Oracle/oradata/MRP/sales_data_01.dbf' to 30;
```

You will have to specify each datafile separately.

Now, the backup process completes without any warnings or errors. As the backup process is completed smoothly, we need to have an alert script, to

read from data dictionary views and alert us of any data block corruptions. Listing 3 shows some sample data from these views, which are populated during an RMAN backup process. The view v\$backup_corruption is populated by RMAN's BACKUP command. However, v\$database_block_corruption, which was introduced in 9iR2, is populated by BACKUP, COPY or VALIDATE commands.

```
MRP:SYSTEM> select * from v$backup_corruption;
```

RECID	STAMP	SET_STAMP	SET_COUNT	PIECE#	FILE#
1	494545426	494545343	3	1	11
10	1		0 YES	FRACTURED	
2	494545426	494545343	3	1	11
29	1		0 YES	FRACTURED	
3	494545426	494545343	3	1	11
379	1		0 YES	FRACTURED	

```
MRP:SYSTEM> select * from v$database_block_corruption;
```

FILE#	BLOCK#	BLOCKS	CORRUPTION_CHANGE#	CORRUPTIO
11	10	1		0 FRACTURED
11	29	1		0 FRACTURED
11	379	1		0 FRACTURED

Listing 3.

Listing 4 gives a sample alert.log entry created when data block corruptions are encountered during the COPY command. The second line tells us during what process the corruption was unearthed, i.e., datafile copy or backup or buffered read, etc. If you have any database alert scripts, which read alert.log files, make sure that they catch the phrase "Corrupt block."

```
Corrupt block relative dba: 0x02c0000a (file 11, block 10)
Bad check value found during datafile copy
Data in bad block -
type: 6 format: 2 rdba: 0x02c0000a
last change scn: 0x0000.00044528 seq: 0x1 flg: 0x06
consistency value in tail: 0x45280601
check value in block header: 0x9cbd, computed block checksum: 0x9
spare1: 0x0, spare2: 0x0, spare3: 0x0
***
Reread of blocknum=10, file=/u01/app/Oracle/oradata/MRP/sales_data_01.dbf.
found same corrupt data
```

Listing 4.

Here is an example showing the usage of COPY command. This script tolerates 30 corrupt data block corruptions.

```
run {
set maxcorrupt for datafile '/u01/app/Oracle/oradata/MRP/sales_data_01.dbf' to 30;
copy datafile '/u01/app/Oracle/oradata/MRP/sales_data_01.dbf' to
'/u01/app/Oracle/backup/MRP/sales_data_01.dbf';
}
```

In this case, corruption information will be written to v\$copy_corruption view. Listing 5 shows some sample data from the data dictionary views that are populated during the RMAN backup process, when COPY command is used for backups.

```
MRP:SYSTEM> select * from v$copy_corruption;
```

RECID	STAMP	COPY_RECID	COPY_STAMP	FILE#	BLOCK#
1	494546644	1	494546644	11	10
1		0	YES FRACTURED		
2	494546645	1	494546644	11	29
1		0	YES FRACTURED		
3	494546645	1	494546644	11	379
1		0	YES FRACTURED		

```
MRP:SYSTEM> select * from v$database_block_corruption;
```

FILE#	BLOCK#	BLOCKS	CORRUPTION_CHANGE#	CORRUPTIO
11	10	1	0	FRACTURED
11	29	1	0	FRACTURED
11	379	1	0	FRACTURED

Listing 5.

Recovering the Data Blocks

Surprisingly, the recovery mechanism is simple and can be done online. Shutting down the database or disconnecting the users is unnecessary.

Example 1: Recovering data blocks individually. You may want to recover individual data blocks, if you know the file-id(s) and block-id(s). You may get this information from the log files of the backup process, alert.log file, v\$backup_corruption, v\$copy_corruption, v\$database_block_corruption views, or from users. The last one is not aimed at though.

```
RMAN> BLOCKRECOVER DATAFILE 11 BLOCK 29,379;
```

This command recovers individual data blocks from the latest successful backup set. Listing 6 gives the output of this command. Note that the recovery process took less than a minute.

```
RMAN> BLOCKRECOVER DATAFILE 11 BLOCK 29,379;
```

```
Starting blockrecover at 20-MAY-03
using channel ORA_DISK_1

channel ORA_DISK_1: restoring block(s) from datafilecopy
/u01/app/Oracle/backup/MRP/sales_data_01.dbf

channel ORA_DISK_1: restoring block(s)
channel ORA_DISK_1: specifying block(s) to restore from backup set
restoring blocks of datafile 00011
channel ORA_DISK_1: restored block(s) from backup piece 1
piece handle=/u01/app/Oracle/backup/MRP/r_1_01enhqj1_1_1.bak tag=MRP_1 params=NULL
channel ORA_DISK_1: block restore complete

starting media recovery
media recovery complete

Finished blockrecover at 20-MAY-03
```

Listing 6.

Example 2: Recovering all data block corruptions at once. You may want to recover all the corrupted data blocks at once, if there are lots of

corruptions. In this case, RMAN recovers all the data blocks that are listed in v\$database_block_corruption view.

```
RMAN> BLOCKRECOVER CORRUPTION LIST;
```

Example 3: Recovering data blocks from an old backup set. If, by chance, your latest backup set is lost, you may want to recover from a specified backup set. You can use the command

```
RMAN> BLOCKRECOVER CORRUPTION LIST FROM TAG 'MRP_1';
```

In this case, the data will be recovered from the backup set with the specified tag. If there is more than one backup set with the same tag, the latest one will be picked.

Example 4: Recovering data blocks until specified time. If you want to restore the data until specified time, you may specify:

```
RMAN> BLOCKRECOVER CORRUPTION LIST FROM TAG '2' RESTORE UNTIL '2003-04-25 12:10:15';
```

Similarly, you can use SCN or Log Sequence for recovery

```
RMAN> BLOCKRECOVER DATAFILE 9 BLOCK 13 DATAFILE 2 BLOCK 19 RESTORE UNTIL SCN 100;
```

```
RMAN> BLOCKRECOVER DATAFILE 9 BLOCK 13 DATAFILE 2 BLOCK 19 RESTORE UNTIL SEQUENCE 7024
THREAD 1;
```

Example 5: Recovering from a datafile image copy. In all the above examples, we have recovered from the backup sets. If we had used the COPY command instead of BACKUP command for our backups, here is how we can restore from those image copies.

```
RMAN> BLOCKRECOVER DATAFILE 11 BLOCK 10 FROM DATAFILECOPY;
```

Case Study: Recovering a Data Block in Which Some Data Was Modified After the Last Backup

In this case study, the database has two tables – supplier_master, which is a master table, and product_master, which references the supplier_master table. The database was backed up the previous night, using RMAN. Later, some updates/ inserts are done to the data. Now, the data block was corrupted by some means, in the data file. We recover the data block using BLOCKRECOVER command, which reads from the backup sets of the previous night and apply the redo logs. Note here that transactions were successfully being written to the data file and redo log files until the data block was corrupted. Of course, there would never have been any transactions on the corrupted data block after the point of corruption, because Oracle does not permit any queries or transactions on the corrupted data block. We don't sacrifice the data integrity because all the data was recovered until the last committed transaction in the data block.

Note: Data block corruption can occur either in the data files, redo logs, archive logs, or in the memory. As Oracle performs sanity checks on data blocks when accessed for queries or updates, it immediately aborts further processing and throws out an error message when it encounters a corrupted data block. As such, if there is a data block corruption in the memory, it is

continued on page 34

never written to disk. (Refer Metalink Note: 77587.1). Also, if there is a data block corruption in a data file, it is never read into memory, and never written back to redo logs.

In the absence of block recovery mechanism, we needed to go for a point-in-time recovery of the entire database until the corruption point, on a separate set up. The entire recovery process would have been too much complicated.

Listing 7 gives the order of tasks performed.

```
STEP 1: Create the tables supplier_master and product_master.

MRP:SALES> create table supplier_master (supl_id number(6),
  2  supl_name varchar2(30), primary key (supl_id));

MRP:SALES> create table product_master (prod_id number(6),
  2  prod_name varchar2(30), supl_id number(6),
  3  primary key (prod_id),
  4  foreign key (supl_id) references supplier_master);

STEP 2: Insert some supplier and product data.

MRP:SALES> insert into supplier_master values (111111, 'Fit Machines');
MRP:SALES> insert into supplier_master values (222222, 'Grand Toys');
MRP:SALES> insert into product_master values (555555, 'Fitness Bike', 111111);

STEP 3: Take a full on-line database backup using RMAN.

STEP 4: Update the product information for prod_id=555555 and insert a new record for prod_id=666666.

MRP:SALES> insert into supplier_master values (333333, 'New Fit Machines');
MRP:SALES> update product_master set supl_id=333333 where prod_id=555555;
MRP:SALES> insert into product_master values (666666, 'Christmas Tree', 222222);

STEP 5: Query the block id's for these records.

MRP:SALES> select distinct substr(rowid,7,9) "Block Id"
  2  from supplier_master;

Block Id
-----
AALAAAAAK

MRP:SALES> select distinct substr(rowid,7,9) "Block Id"
  2  from product_master;

Block Id
-----
AALAAAAAa

We find that all the supplier_master records are stored in one database block, and all the product_master records are stored in another single database block.

LISTING 7: A Case Study on Block Recovery

STEP 6: By some means, corrupt the data block in which supplier_master information is stored.

MRP:SALES> select * from supplier_master where supl_id=111111;
select * from supplier_master where supl_id=111111
*
ERROR at line 1:
ORA-01578: ORACLE data block corrupted (file # 11, block # 10)
ORA-01110: data file 11: '/u01/app/Oracle/oradata/MRP/sales_data_01.dbf'

STEP 7: Now, we have a situation where the entire database is intact, except for the one corrupted block. We want to recover all the committed data from this block, without touching the rest of the database. This is where the block recovery helps. Using the blockrecover command of RMAN, we can recover all the committed data from the latest database backup and redo log files, until the last committed transaction in the block.
```

```
STEP 8: Query the latest on-line log file record.

MRP:SYSTEM> select * from v$log where status='CURRENT';

GROUP#    THREAD#  SEQUENCE#    BYTES  MEMBERS ARC STATUS
-----
FIRST_CHANGE# FIRST_TIM
-----
          1          1          8 104857600          1 NO  CURRENT
355500 09-DEC-03

STEP 9: Recover data within the corrupted block using the blockrecover command.

RMAN> blockrecover datafile 11 block 10 restore until sequence=8 thread=1;

Starting blockrecover at 09-DEC-03
using channel ORA_DISK_1

channel ORA_DISK_1: restoring block(s)
channel ORA_DISK_1: specifying block(s) to restore from backup set
restoring blocks of datafile 00011
channel ORA_DISK_1: restored block(s) from backup piece 1
piece handle=/u01/app/Oracle/backup/MRP/r_0_06f8jhkn_1_1.bak tag=MRP_0 params=NULL
channel ORA_DISK_1: block restore complete

starting media recovery
media recovery complete

Finished blockrecover at 09-DEC-03

LISTING 7: A Case Study on Block Recovery

STEP 10: Notice that you can access all the records completely, including the latest updates and additions done to the tables.

MRP:SALES> select * from supplier_master;

SUPL_ID SUPL_NAME
-----
111111 Fit Machines
222222 Grand Toys
333333 New Fit Machines

MRP:SALES> select * from product_master;

PROD_ID PROD_NAME          SUPL_ID
-----
555555 Fitness Bike          333333
666666 Christmas Tree      222222
```

Listing 7.

Conclusion

In addition to a number of benefits not mentioned in this article, Recovery Manager helps us in fixing the data block corruptions, at early stages. This simple backup tool can save us a lot of time, tension and resources. To summarize, make sure that you have the following in your Backup & Recovery process

- Usage of Recovery Manager in your Backup methodology
- Script to alert DBA Team of errors encountered during backup process
- Script to read from v\$backup_corruption, v\$copy_corruption and v\$database_block_corruption views and alert.log file to alert DBA Team, in case your script tolerates data block corruptions.



About the Author

Krishna Kakatur (kakatur2002@yahoo.com), an Oracle 9i Certified DBA and Developer with Atlas Software Technologies (www.atlassoft.com), is currently working as a consultant at Sun Microsystems, Newark, CA. He has more than 10 years of experience using Oracle Tools and Technologies.