**http://www.dba-oracle.com/t\_cbo\_using\_logminer.htm**

**Using LogMiner**

*Oracle Database Tips by Donald BurlesonJuly 24, 2015*

Someone has deleted Mr. Ford from Scott's EMP table, and everyone disavows any knowledge of the event. However, you are very suspicious of one of your users, who has a tendency of bending the truth at times. So, you decide to use LogMiner to track down the guilty party.

First, you edit your parameter file entering *utl\_fil\_dir*=C:\ORACLASS\LOGS. LogMiner can now create an operating system file in this path.

SQL> SHOW USER

USER is "SYS"

SQL> SHOW PARAMETER utl

NAME TYPE VALUE

------------ ------ ----------------

utl\_file\_dir string C:\ORACLASS\LOGS

Next, you execute the BUILD procedure in the DBMS\_LOGMNR\_D package to create an operating sys- tem file MINER2.DAT. This file will contain information taken from the specified redo log files.

SQL> CONNECT SYS/CHANGE\_ON\_INSTALL@DBA

Connected.

SQL> BEGIN

2 dbms\_logmnr\_d.build ( 'miner2.dat', 'C:\ORACLASS\LOGS' );

3 END;

4 /

PL/SQL procedure successfully completed.

Next, switch log files and identify the current log file.

SQL> ALTER SYSTEM SWITCH LOGFILE;

System altered.

SQL> column member NEW\_VALUE redo

SQL> SELECT member

2 FROM v$logfile â€” IF REDO LOGS ARE MIRRORED,

ONLY WANT ONE(ROWNUM<2)

3 WHERE group# = ( SELECT group# FROM v$log

WHERE status = 'CURRENT' )

4 AND ROWNUM < 2;

MEMBER

-----------------------------------

C:\ORACLASS\DATA\DISK3\REDO0201.LOG

Next, copy the above redo log member to C:\ORA- CLASS\LOGS\redo.log.

SQL> host xcopy &redo C:\ORACLASS\LOGS\redo.log

Now, indicate to LogMiner that you want information contained in the redo.log file by executing ADD\_LOGFILE.

SQL> BEGIN

2 dbms\_logmnr.add\_logfile('C:\ ORACLASS\ LOGS\redo.log',

3 dbms\_logmnr.NEW );

4 END;

5 /

PL/SQL procedure successfully completed.

Now, tell LogMiner to start. LogMiner takes the data in the operating system file miner2.dat and loads it into the dynamic performance views used by LogMiner. One of these views is *v$logmnr\_contents*.

SQL> BEGIN

2 dbms\_logmnr.start\_logmnr ( dictFileName =>

'c:\ORACLASS\LOGS\miner2.dat' );

4 END;

5 /

PL/SQL procedure successfully completed.

Now you are ready to write a query (Exhibit 8) using *v$logmnr\_contents* to identify the villain in who deleted ford saga. The two columns of interest are SQL\_REDO and SQL\_UNDO. The SQL\_REDO column shows the SQL statement that changes data in a table, and the SQL\_UNDO column shows the SQL statement(s) to reverse or undo the SQL statement that changed the table data. It is more than painfully obvious that user LIAR\_LIAR deleted Ford and also committed his dastardly deed. Notice the undo of the DELETE command is an INSERT command.

SQL> columnsql\_redo format a30 word\_wrapped

SQL> columnsql\_undo format a30 word\_wrapped

SQL> columnusername format a12

SQL> SELECT scn,

username,

sql\_redo,

sql\_undo

FROM V$LOGMNR\_CONTENTS;

SCN USERNAME SQL\_REDO SQL\_UNDO

------ --------- ------------------------------ ------------------------

232009 LIAR\_LIAR set transaction read write;

232009 LIAR\_LIAR delete from SCOTT.EMP where insert into

EMPNO = 7902 and ENAME =SCOTT .EMP(EMPNO,ENAME,JOB,MGR,

'ford' and JOB = 'ANALYST' and HIREDATE,SAL,COMM,DEPTNO)

MGR = 7566 and HIREDATE = values

TO\_DATE('03-DEC-2081 (7902,'ford','ANALYST',

00:00:00', 'DD-MON-YYYY 7566,TO\_DATE('03-DEC-2081

HH24:MI:SS') and SAL = 3000 00:00:00','DD-MON-YYYY

and COMM IS NULL and DEPTNO = HH24:MI:SS'),3000,NULL,

20 and ROWID = 20);

'AAAAsrAADAAAAF6AAM';

232010 LIAR\_LIAR commit;

**Exhibit 8.***Query Using v$logmnr\_contents to Identify the Villain*

To view the INSERT required to place Mr. Ford back into Scott's EMP table, write a query viewing the SQL\_UNDO column of *v$logmnr\_contents*:

SQL> SELECT sql\_undo

FROM V$LOGMNR\_CONTENTS;

insert into

SCOTT.EMP(EMPNO,ENAME,JOB,MGR,

HIREDATE,SAL,COMM,DEPTNO)

values

(7902,'ford','ANALYST',7566,TO

\_DATE('03-DEC-2081 00:00:00',

'DD-MON-YYYY

HH24:MI:SS'),3000,NULL,20);

You can cut and paste the INSERT statement and exe- cute it to place Ford back into Scott's EMP table.

SQL> insert into

SCOTT.EMP(EMPNO,ENAME,JOB,MGR,

HIREDATE,SAL,COMM,DEPTNO)

values

(7902,'ford','ANALYST',7566,

TO\_DATE('03-DEC-2081 00:00:00',' DD-MON-YYYY

HH24:MI:SS'),3000,NULL,20);

1 row created.

SQL> COMMIT;

Commit complete.

Finally, end your LogMiner session.

SQL> BEGIN

SQL> dbms\_logmnr.end\_logmnr;

SQL> END;

SQL> /

PL/SQL procedure successfully completed.

BMC Corporation has had a product named LogMaster for many years. It is much easier to use than Oracle's LogMiner. You merely enter commands at the operating system prompt

**https://oracle-base.com/articles/8i/logminer**

# LogMiner

**LogMiner enables the analysis of the contents of archived redo logs. It can be used to provide a historical view of the database without the need for point-in-time recovery. It can also be used to undo operations, allowing repair of logical corruption.**

* [Create Dictionary File](https://oracle-base.com/articles/8i/logminer#CreateDictionaryFile)
* [Adding Logs](https://oracle-base.com/articles/8i/logminer#AddingLogs)
* [Starting LogMiner](https://oracle-base.com/articles/8i/logminer#StartingLogMiner)
* [Querying Log Information](https://oracle-base.com/articles/8i/logminer#QueryingLogInfo)
* [Stopping LogMiner](https://oracle-base.com/articles/8i/logminer#StoppingLogMiner)

Related articles.

* [LogMiner Enhancements In Oracle9i](https://oracle-base.com/articles/9i/logminer-enhancements-9i)
* [Flashback and LogMiner Enhancements in Oracle Database 11g Release 1](https://oracle-base.com/articles/11g/flashback-and-logminer-enhancements-11gr1)

## **Create Dictionary File**

Without a dictionary file LogMiner displays all tables and columns using their internal object IDs and all values as hex data. The dictionary file is used to translate this data into a more meaningful format. For the dictionary file to be created the following initialization parameter must be set and the instance must be mounted or open.

UTL\_FILE\_DIR=C:\Oracle\Oradata\TSH1\Archive

The dictionary file is created using the BUILD procedure in the DBMS\_LOGMNR\_D package.

BEGIN

DBMS\_LOGMNR\_D.build (

dictionary\_filename => 'TSH1dict.ora',

dictionary\_location => 'C:\Oracle\Oradata\TSH1\Archive');

END;

/

## **Adding Logs**

A list of logs to be analyzed must be added to logminer using the DBMS\_LOGMNR package. The first log in the list is added using the NEW procedure, while subsequent logs are added using the ADD\_LOGFILE procedure.

BEGIN

DBMS\_LOGMNR.add\_logfile (

options => DBMS\_LOGMNR.new,

logfilename => 'C:\Oracle\Oradata\TSH1\Archive\TSH1\T001S00006.ARC');

DBMS\_LOGMNR.add\_logfile (

options => DBMS\_LOGMNR.addfile,

logfilename => 'C:\Oracle\Oradata\TSH1\Archive\TSH1\T001S00007.ARC');

END;

/

## **Starting LogMiner**

At this point LogMiner can be started using the overloaded START\_LOGMNR procedure. The analysis range can be narrowed using time or SCN.

BEGIN

-- Start using all logs

DBMS\_LOGMNR.start\_logmnr (

dictfilename => 'C:\Oracle\Oradata\TSH1\Archive\TSH1dict.ora');

-- Specifiy time range

DBMS\_LOGMNR.start\_logmnr (

dictfilename => 'C:\Oracle\Oradata\TSH1\Archive\TSH1\dict.ora',

starttime => TO\_DATE('01-JAN-2001 00:00:00', 'DD-MON-YYYY HH:MI:SS'),

endtime => TO\_DATE('01-JAN-2001 10:00:00', 'DD-MON-YYYY HH:MI:SS'));

-- Specifiy SCN range

DBMS\_LOGMNR.start\_logmnr (

dictfilename => 'C:\Oracle\Oradata\TSH1\Archive\TSH1\dict.ora',

startscn => 100,

endscn => 150);

END;

/

## **Querying Log Information**

Once LogMiner is started, the contents of the logfiles can be queried using the following views:

* V$LOGMNR\_DICTIONARY - The dictionary file in use.
* V$LOGMNR\_PARAMETERS - Current parameter settings for LogMiner.
* V$LOGMNR\_LOGS - Which redo log files are being analyzed.
* V$LOGMNR\_CONTENTS - The contents of the redo log files being analyzed.

The following query displays the SQL issued along with the undo SQL to reverse it.

SELECT scn, operation, sql\_redo, sql\_undo

FROM v$logmnr\_contents;

The following query displays the number of hits for each object during the analyzed period.

SELECT seg\_owner, seg\_name, count(\*) AS Hits

FROM v$logmnr\_contents

WHERE seg\_name NOT LIKE '%$'

GROUP BY seg\_owner, seg\_name;

## **Stopping LogMiner**

Once the analysis is complete, logminer should be stopped using the END\_LOGMNR procedure.

BEGIN

DBMS\_LOGMNR.end\_logmnr;

END;

/

For more information see:

* [Using LogMiner to Analyze Online and Archived Redo Logs](http://docs.oracle.com/cd/A87860_01/doc/server.817/a76956/archredo.htm#12680)
* [LogMiner Enhancements In Oracle9i](https://oracle-base.com/articles/9i/logminer-enhancements-9i)
* [Flashback and LogMiner Enhancements in Oracle Database 11g Release 1](https://oracle-base.com/articles/11g/flashback-and-logminer-enhancements-11gr1)

**LET OP:**

De log-miner moet GESTART zijn voordat je de V$LOGMNR\_CONTENTS-views kunt benaderen.

Doe je dat niet, krijg je volgende foutmelding:

ORA-01306: dbms\_logmnr.start\_logmnr() must be invoked before selecting from v$logmnr\_contents

01306. 00000 - "dbms\_logmnr.start\_logmnr() must be invoked before selecting from v$logmnr\_contents"

\*Cause: A select was issued from v$logmnr\_contents without first invoking

the dbms\_logmnr.start\_logmnr() procedure.

\*Action: Invoke the dbms\_logmnr.start\_logmnr() procedure before issuing

a select from the v$logmnr\_contents view.