



LogMiner Oracle9i

What is a Logminer, how can it help me ?

01_001_04

LogMiner Oracle9i

Have you ever wondered who was responsible for changing the salary table to zero out your fellow DBA's pay or entering all the new prices in the online store without using the decimal point? Would it be useful to locate the offending SQL statement and be provided with SQL required to correct it? This is a great tool for undoing a single erroneous transaction without having to go through a database restore.

Every change made to an Oracle database by default generates undo and redo information which is accumulated in Oracle redo log files. Oracle9i LogMiner is an integrated feature of the Oracle9i Database that provides DBA's and auditors with the infrastructure required for relational access to Oracle's redo stream.

The online data dictionary can be extracted into the redo log stream. This enables off-line analysis and provides a snapshot of the data dictionary that matches the database objects in logs created at that time. When mining logs in the same database that generated it, the user can choose to use the online data dictionary for SQL reconstruction and internal identifier to name mapping which would otherwise be a manual process.

The user can group DML statements into transactions with a `COMMITTED_DATA_ONLY` option which directs LogMiner to group DML statements into complete transactions. Only committed transactions will be returned in the commit SCN order. When the `DDL_DICT_TRACKING` option is enabled and LogMiner is run in the context of an open database, it will automatically apply DDL statements executed in the original redo stream to its internal dictionary. This enables correct reconstruction of correct SQL statements on tables whose definition has been altered or were missing when the original dictionary dump was captured. LogMiner automatically versions the metadata kept in the database.

New DDL statements have been added to Oracle's vocabulary to allow for logging of additional column values in case of updates. The extra information can be used either to identify the updated row logically or to capture a before-row image. This allows a DBA or auditor to use the additional column information to construct more complete statements to undo changes or to create SQL statements for applying to a different databases.

A powerful new capability allows for queries that can filter out rows based on actual data values of the columns updated. For instance it is possible to write a query for a postal database that identifies all lottery winners who moved to 90210 after making it big in Redwood Shores. LogMiner improves data availability by providing a window into all changes made to a database. It enables auditing of database changes and reduces the time and effort needed for data recovery.

Installing LogMiner

To install the logminer packages you need to run the scripts: `$ORACLE_HOME/rdbms/admin/dbmslsm.sql` and `$ORACLE_HOME/rdbms/admin/dbmslmd.sql`. The first installs the `DBMS_LOGMNR` package and needs to be run on the database which you will use to analyse the logfiles. The second installs the `DBMS_LOGMNR_D` package which builds the data-dictionary file, and needs to be run on the database from which the redo logs are generated.

Both packages need to run as SYS.

Installing the packages does not grant execution rights to other users (including SYSTEM) so, unless you do so, using logminer is restricted to the SYS user.

Using logminer

To use Oracle logminer, the following procedure should be followed:

- **Build a data-dictionary for logminer**

This allows logminer to refer to columns and tables by their normal names, rather than by internal representations. Logminer can be used without a data-dictionary, but is then of limited use. The data-dictionary is a text file which is built using the DBMS_LOGMNR_D package. It only needs rebuilding if there have been any data-dictionary changes to the tables which are to be monitored. It is possible, and common, to analyse logs generated from one database on another. In this case, the data-dictionary file must, of course, be generated on the database from which the logs were generated.

The dictionary_location parameter must refer to a directory specified in the UTL_FILE_DIR init.ora parameter.

```
EXECUTE DBMS_LOGMNR_D.BUILD(  
  dictionary_filename => 'l_dictionary.ora',  
  dictionary_location => '/oracle/database');
```

This may take several minutes to run.

- **Declare the log_files to be monitored to the DBMS_LOGMNR package.**

The log_files to be monitored must be placed in a directory specified in the UTL_FILE_DIR init.ora parameter.

eg.

```
EXECUTE DBMS_LOGMNR.ADD_LOGFILE(  
  LogFileName => '/oracle/logs/log1.f',  
  Options => dbms_logmnr.NEW);
```

```
EXECUTE DBMS_LOGMNR.ADD_LOGFILE(  
  LogFileName => '/oracle/logs/log2.f',  
  Options => dbms_logmnr.ADDFILE);
```

The "NEW" option (re)initialises the list of logfiles declared to logminer, the "ADDFILE" option adds to the current list.

Note: it is not wise to simply include the directory which contains your archived redo logs into the list of directories specified in the UTL_FILE_DIR parameter, as this would potentially allow any user with access to the UTL_FILE package to overwrite your redo logs.

- **Start logminer**

```
EXECUTE DBMS_LOGMNR.START_LOGMNR(  
  DictFileName => '/oracle/database/l_dictionary.ora');
```

- **Run queries against the logminer views**

```
SELECT sql_redo  
  FROM V$LOGMNR_CONTENTS
```

- **Close logminer**

```
EXECUTE DBMS_LOGMNR.END_LOGMNR();
```

Related documents

All the above has been synthesised from the [Oracle8i Supplied PL/SQL Packages Reference Release 2 \(8.16\)](#) manual, specifically the chapters on [DBMS_LOGMNR](#) and [DBMS_LOGMNR_D](#)