

Guide to Database Maintenance: NOLOGGING Option

A feature of Oracle Rdb

By Ian Smith
Oracle Rdb Relational Technology Group
Oracle Corporation

GUIDE TO DATABASE MAINTENANCE AND TUNING: NOLOGGING OPTION.....	3
INTRODUCTION.....	3
WHAT HAPPENS IF I NEED TO RECOVER MY DATABASE USING THE AFTER IMAGE JOURNAL?	5
WHAT IF AFTER IMAGE JOURNALING IS DISABLED?	7
WHAT DOES RMU VERIFY REPORT IF ONE OF THE LOGICAL AREAS IS MARKED INCOMPLETE?	7
WHAT HAPPENS IF A CREATE OR ALTER STATEMENT FAILS WHEN JOURNALING IS DISABLED?	8
WHAT IF AN ERROR OCCURS DURING THE TRANSACTION?.....	8
WHAT HAPPENS IF HOT STANDBY IS ACTIVE ON THE DATABASE?	9
WHAT IF LOGMINER IS ENABLED FOR THE DATABASE?.....	9
IS NOLOGGING APPLIED TO LIST STORAGE MAPS?	9
DOES THE LOGICAL NAME RDM\$CREATE_LAREA_NOLOGGING STILL WORK?.....	10

The Rdb Technical Corner is a regular feature of the Oracle Rdb Web Journal. The examples in this article use SQL language from Oracle Rdb release 7.1 and later versions.

Guide to Database Maintenance: NOLOGGING Option

Introduction

Oracle Rdb automatically records changes made to the database to the recovery unit journal (.RUJ) so that upon an image, process or system failure the uncommitted changes can be rolled back. This journal is also used by the ROLLBACK command to undo any work performed in the current transaction. Additionally Rdb allows database administrators to configure one or more after image journals (.AIJ) to collect all changes to the database so that they may be reapplied to a backup copy of the database. Other tools and facilities such as Hot Standby, LogMiner and Fast Commit also use these AIJ files in various ways. The I/O to these journals is generically known as logging and for the most part is essential to the health and well being of the database system.

The NOLOGGING option can be used to disable most of the journaling to the recovery unit journal and after image journal during certain CREATE and ALTER statements.

Normally when creating a new logical area as part of a **create table**, **create storage map**, **create index**, **alter storage map** or **alter index** statement all updates to these logical areas are written to the recovery unit and after image journals.

This can be a problematic when creating or altering a large index, or reorganizing a storage map for a large table. In these cases, table rows, hash buckets and sorted index nodes are written to the new logical areas and copies must also be written AIJ files. Since the RUJ file is used to return the database to a known initial state it also has a record of the prior, or before image. As the DDL operation continues these records may also be re-written due to a subsequent update – for instance when an internal node is revised. The amount of I/O to these journals may be extensive and the RUJ and AIJ files may grow quite large.

The NOLOGGING option can be used to change the type of information written to the recovery and after image journals so that this I/O can be almost eliminated. The recovery and after image journals will only contain a special logical operation with no associated data for the CREATE or

ALTER statement. The savings during these DDL operations is less journaling I/O and low disk space requirements for these operations on large tables.

Note that the NOLOGGING option is transient and is not a permanent attribute of the table, index or storage area and is only active during the transaction in which the logical area is created. The one exception is the case where an index is created using the **maintenance is enabled deferred** clause. The NOLOGGING option in this case is also deferred until such time that the index partitions are built using the **alter index ... build all partition** or **alter index ... build partition** clauses.

The advantages are the most obvious for a **create index** statement, which by default will read the base table, sort the column data and write index nodes (and duplicate nodes) to the various storage areas, and also journal changes to the recovery and after image journals. The additional I/O to the recovery and after image journals can be replaced by a single logical journal entry for each partition in the index.

Database administrators must be aware of the possible disadvantages of disabling (or minimizing) journaling with NOLOGGING. The trade off is less I/O during the operation versus more slightly more complex recovery procedure. For ROLLBACK or process failure the abbreviated journal entry is sufficient to remove the index entry and also delete the partition (logical area) from the database. However, the after image journals now contain a gap or hole that can not be used by RMU Recover to completely restore the database. This is the reason for the warning messages issued as part of the **create index** statement.

```
$ sql$
attach 'file db$:test_nojournal';
create index T_I on T (a) nologging;
%RDB-W-META_WARN, metadata successfully updated with the reported warning
-RDMS-W-DATACMIT, unjournalled changes made; database may not be recoverable
-RDMS-W-DOFULLBCK, full database backup should be done to ensure future recovery
commit;
```

The changes in recovery strategy are discussed below. Please refer to the *Oracle Rdb SQL Reference Manual* for detail of the syntax NOLOGGING.


```

SQL> -- FAILS
SQL> select * from T where a > 0;
Leaf#01 FFirst T Card=5
  BgrNdx1 T_I [1:0] Fan=17
%RDMS-F-DATATBLCMIT, unjournaled changes made to user-defined object
SQL>
SQL> -- now do a sequential scan
SQL> -- SUCCEEDS
SQL> select b from T;
Get  Retrieval sequentially of relation T
      B
      NULL
      1
      2
3 rows selected
SQL>
SQL> -- now drop the index
SQL> -- SUCCEEDS
SQL> drop index T_I;
Firstn Get  Retrieval by index of relation RDB$INDICES
  Index name RDB$NDX_NDX_NAME_NDX [1:1]      Direct lookup
SQL>
SQL> -- select again (uses sequential scan)
SQL> -- SUCCEEDS
SQL> select * from T where a > 0;
Conjunct  Get  Retrieval sequentially of relation T
      A      B
      1      NULL
      2      1
      2      2
3 rows selected
SQL>
SQL> -- recreate the index
SQL> create index T_I on T (a);
SQL>
SQL> -- select using the new index
SQL> -- SUCCEEDS
SQL> select * from T where a > 0;
Leaf#01 FFirst T Card=5

```

```
BgrNdx1 T_I [1:0] Fan=17
  A      B
  1      NULL
  2      1
  2      2
3 rows selected
SQL>
SQL> commit;
```

Note: If CREATE INDEX or ALTER INDEX refers to a HASHED index then this operation also requires updates to the RDB\$SYSTEM_RECORD on each page of a MIXED format area. When these records are updated the changes are always journaled to the recovery and after image journals. Therefore, some journaling activity will result from these operations.

Therefore, the NOLOGGING action can be safely recovered using this mechanism, however, Oracle recommends performing a full RMU/BACKUP following the use of NOLOGGING to avoid this type of additional recovery strategy.

What if After Image Journaling is disabled?

If after image journaling is not currently in use on the database then the rollback operation can fully recover the database from the recovery unit journal. Therefore, only the DATACMIT warning is issued in this case. This is a warning that an error reported during the transaction must be rolled back to guarantee recovery.

What does RMU Verify report if one of the logical areas is marked incomplete?

RMU Verify will attempt to ready the incomplete logical area. The following example shows that the index T_I is incomplete (the warning message DATATBLCMIT) and the verification of the index is abandoned.

```
$ rmu/verify/all db$:test_nojournal
...
%RMU-I-BGNNDXVER, beginning verification of index T_I
%RMU-W-DATATBLCMIT, unjournaled changes made to user-defined object
```

```
%RMU-E-BDLAREADY, error readying logical area with dbid 48
%RMU-W-NOT_LARDY, area for 48:560:0 not in proper ready mode
%RMU-E-BADDBKFET, Error fetching dbkey 48:560:0
%RMU-W-BTRVFYPRU, B-tree verification pruned at this dbkey
%RMU-I-BTRROODBK, root dbkey of B-tree is 48:560:0
%RMU-I-NDXERRORS,      2 index errors encountered
%RMU-I-ENDNDXVER, completed verification of index T_I
```

The errors BADDBKFET occurs when the root DBKEY of the index that was stored in the Rdb metadata is used to fetch the root node at the start of the index verification.

What happens if a CREATE or ALTER statement fails when journaling is disabled?

In this case the creation of the logical area (which is always journaled) is rolled back. All the data written to that logical area is erased from the database. To erase the data from a MIXED format area requires that each page of the storage area be processed, and so will most likely be slower than similar recovery when journaling is enabled.

When recovery is performed using RMU/RECOVER any rolled back transaction is discarded (i.e. is not applied to the backup database) so no reference to the incomplete logical area will be encountered.

What if an error occurs during the transaction?

If the transaction which performed the CREATE or ALTER statement has already committed, then subsequent transactions will have resumed journaling. This is the normal logging mode for Oracle Rdb and errors will be handled as expected.

However, if the original transaction which performed the CREATE or ALTER is still active and an error occurs while writing to an unjournaled logical area then the logical area is immediately marked as corrupt. Such errors include failures of INSERT or UPDATE due to duplicate key values, constraint is violated, or database locking errors. The transaction should be aborted using the ROLLBACK statement. COMMIT can be used but the affected logical areas should be deleted using a DROP statement.

Oracle recommends that the transaction be committed immediately when the CREATE or ALTER statement has successfully completed and avoid performing additional DML statements such as INSERT, UPDATE and DELETE. Committing promptly will avoid the problems described in this section and will also release locks on rows and other database system resources.

What happens if Hot Standby is active on the database?

The Hot Standby Option requires all operations to be journaled; therefore the NOLOGGING option is ignored for any database enabled for Hot Standby. A warning similar to the following is issued:

```
SQL> create index jobs_index on jobs (WAGE_CLASS) nologging;
%RDB-W-META_WARN, metadata successfully updated with the reported warning
-RDMS-W-IGNNOLOG1, ignoring NOLOGGING - Hot Standby is enabled and logging required
-RDMS-W-DATACMIT, unjournalled changes made; database may not be recoverable
-RDMS-W-DOFULLBCK, full database backup should be done to ensure future recovery
```

What if LogMiner is enabled for the database?

If NOLOGGING is used on a storage map or table and rows are inserted into that table in the same transaction then they will not be visible to RMU/UNLOAD/AFTER_IMAGE. A warning is issued to inform the database administrator that this could be a problem.

```
SQL> create table SAMPLE_TABLE (a int) nologging;
%RDB-W-META_WARN, metadata successfully updated with the reported warning
-RDMS-W-DATACMIT, unjournalled changes made; database may not be recoverable
-RDMS-W-IGNNOLOG3, alert for NOLOGGING - LogMiner requires logging to unload table data
```

Once the COMMIT has completed then all INSERT, DELETE and UPDATE operations to the table will start logging changes to the after image journals.

Is NOLOGGING applied to LIST storage maps?

No unfortunately this is not currently possible. When a list storage map is created or altered any

LIST OF BYTE VARYING data subsequently written will be journaled, regardless of the setting of the **rdms\$create_larea_nologging** logical name or the use of the NOLOGGING clause. The reason for this restriction is that it is not possible for the rollback processing to distinguish between old and new LIST segments that might exist in areas shared with other tables.

Does the logical name RDMS\$CREATE_LAREA_NOLOGGING still work?

Although Oracle recommends using the syntax NOLOGGING with both **alter** and **create** statements it is possible to define this logical name to achieve the same effect. This logical name was introduced in Oracle Rdb V7.0.1.1 and remains the only way to achieve NOLOGGING functionality in Oracle Rdb V7.0.

To disable journaling for **create** and **alter** statements the process logical name **rdms\$create_larea_nologging** can be defined prior to performing the DDL operations. Once attached to the database all journaling to new logical areas will be disabled until the transaction that created them has been committed or rolled back.

The values accepted for this logical name are:

- 0 - meaning that journaling is not disabled -- which is the default if the logical name is not defined,
- 1 - meaning create logical area journaling is disabled. For example, the logical name would be defined as shown in this example.

```
$ define RDMS$CREATE_LAREA_NOLOGGING "1"
```

Please note that nologging only affects newly created logical areas. Once the transaction has been committed logging resumes to those area.

ORACLE

Oracle Rdb
Guide to Database Maintenance: NOLOGGING Option
April 2008

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
www.oracle.com

Oracle Corporation provides the software
that powers the internet.

Oracle is a registered trademark of Oracle Corporation. Various
product and service names referenced herein may be trademarks
of Oracle Corporation. All other product and service names
mentioned may be trademarks of their respective owners.

Copyright © 2008 Oracle Corporation
All rights reserved.