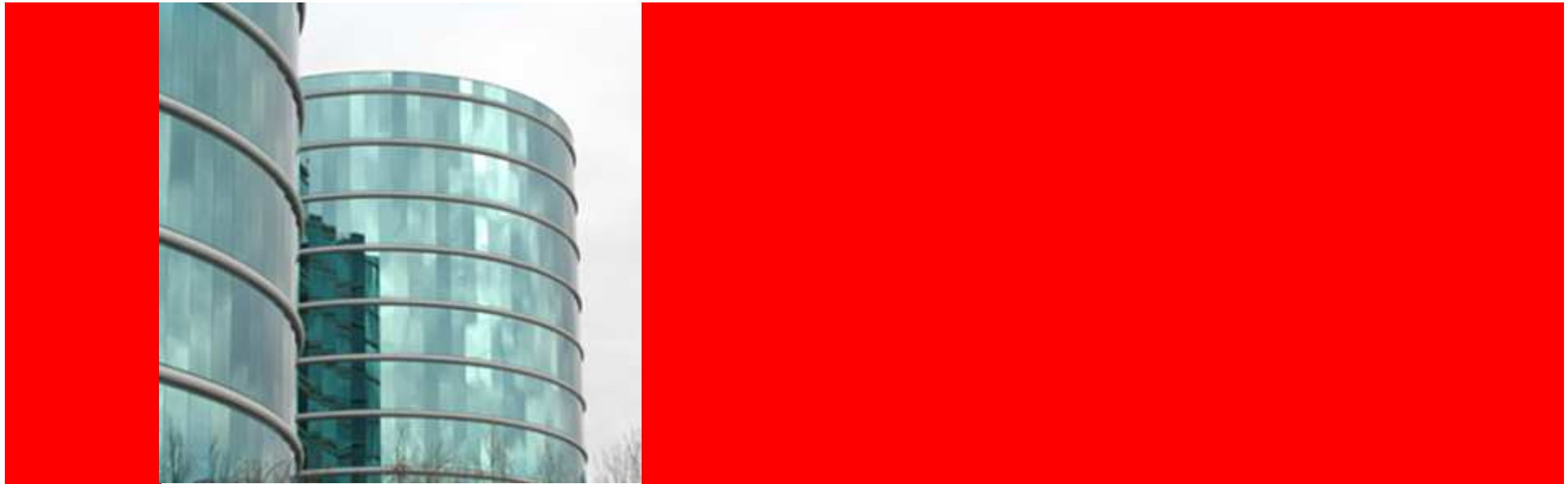




ORACLE®



ORACLE®



Oracle Rdb After and Before Image Journaling

Norman Lastovica
Oracle OpenVMS Development Team

13 October 2009

Agenda

- AIJ
- RUJ





AIJ

- Historical, sequential log of database modification
- Conceptually - continuous record of changes to database
- More or less the “Re-do log”



AIJ

- Contains all database changes made by transaction
- Used to re-do changes upon process or system failure
- Source for Hot Standby and LogMiner features
- Increases database performance
 - by enabling other database optimizations



AIJ Requirements

- Must explicitly:
 - Reserve journal slots
 - Create journal files
 - Enable after-image journaling
 - Perform database backup
- Should place AIJ files on disks separate from other database files or other AIJ files
- Backup AIJ files periodically



Review of Quiet Points

- Special locations within journal backup sequence at which all transactions that have written to journal have committed
- Some quiet points are forced by DBA or Rdb engine requesting quiet points
- Some quiet points are spontaneous
 - Called “micro” quiet points
 - Are surprisingly frequent

After Image Journal Backups





What is Journalled?

1. Most modifications to database root (.RDB)
2. Modifications to storage areas (.RDA)
3. Transaction Rollback/Commit information
4. Detailed transaction start information (LogMiner)
5. Control information
 - a) Headers
 - b) Options
 - c) Transaction Prepare
 - d) AIJ activation
 - e) Checkpoints



What is NOT Journalled?

- Changing database...
 - NUMBER OF USERS
 - NUMBER OF NODES
 - Reserving Journal Files
 - Reserving Storage Area Files
 - Moving Storage Area Files
 - Moving the Root File
 - Some offline RMU/SET operations

These events do not occur within a transaction; are not journalled and can not be rolled back / forward

- Must backup database after these actions



Data in AIJ

- Most AIJ content likely data & index
 - Record add or modify journaled as “Modify”
 - Record delete journaled as “Erase”
- Each entry includes:
 - Physical DBKEY of data record
 - Page Sequence Number
 - Length (zero for delete)
 - Data itself
 - Logical area (sometimes)



What Data is Journalled?

- Data copied from data page or row cache
- Line fragments journalled when pages where fragments reside are flushed
- Lines include data records, index nodes, and metadata records (system relations)



When is Data Journalled?

1. As modified page is flushed to database
 - Transaction commit
 - Checkpoints
 - Buffer pool is full
 - During 'optimistic' batch write flushes
 - Blocking AST for modified page
2. At transaction commit
3. As modified rows are flushed to record cache



Remember That...

- When page flushed to disk, state of row *at that point in time* is journaled
 - Multiple changes to record may be made within transaction
- Deleted row is one with zero length on page
- Row that is added & deleted (perhaps due to constraint failure) is journaled as erase of row when page is flushed
- Index nodes are data too!



Actually Writing to AIJ File

- Multiple AIJ records are collected together & written in single I/O
- AIJ records from other processes on node may be collected together & written at once
- Up to 256 blocks per I/O
- AIJ files are pre-initialized to “-1”s (FF)



AIJ Log Server (ALS)

- Server process that writes AIJ for all processes for a database on node
- Processes notify ALS that data needs to be written to AIJ
- Optimized for AIJ writes
 - Double-buffers I/O
 - Offloads AIJ I/O from user processes & reduces AIJ lock contention (one writer per node)



Recover (aka Roll Forward)

- `RMU /RECOVER`
`RMU /RESTORE /RECOVER`
- For each committed transaction, apply all physical data record modifications
 - Transactions that rolled back are ignored
- Restores database to transaction-consistent state at point in time



Hot Standby

- AIJ writes are sent to standby via network
- Write records to the standby AIJ
- Standby then applies change(s) in an on-going recover operation
- Databases kept physically up-to-date on committed transaction basis



LogMiner

- **RMU /UNLOAD /AFTER_JOURNAL**
- Extract database changes of committed transactions
- Returns modify/add and “pre-delete” record contents
- Use /FORMAT=DUMP for human-readable output

RMU /DUMP /AFTER_JOURNAL

```
2/3          TYPE=N, LENGTH=78, TAD=20-SEP-2009 01:02:07.96, CSM=00
Database Attach Information
PID=000000A7:1, TID=662, Buffer count is 500

2/4          TYPE=B, LENGTH=14, TAD=20-SEP-2009 01:02:07.96, CSM=00
TID=662, TSN=0:0, AIJBL_START_FLG=00, FLUSH=01, SEQUENCE=1

2/5          TYPE=D, LENGTH=59, TAD=20-SEP-2009 01:02:07.96, CSM=00
TID=662, TSN=0:4546, AIJBL_START_FLG=01, FLUSH=01, SEQUENCE=2
MODIFY: PDBK=1:11:30, LDBID=2, PSN=118, FLAGS=00, ABM_PNO=8
REC_LEN=12, LENGTH=13

                0037  0000  line 30 (1:11:30) record type 55
                00 0001  0002  Control information
                        ....  8 bytes of static data
FE0012D687000106  0005  data '....Ö..p`

2/6          TYPE=N, LENGTH=78, TAD=20-SEP-2009 01:02:07.96, CSM=00
Transaction Start Information
PID=000000A7:1, TID=662, TSN=0:4546, START_TAD=20-SEP-2009 01:02:07.60
COMMIT_TAD=20-SEP-2009 01:02:08.26 USERNAME="LASTOVICA  "

2/7          TYPE=C, LENGTH=14, TAD=20-SEP-2009 01:02:07.96, CSM=00
TID=662, TSN=0:4546, AIJBL_START_FLG=00, FLUSH=01, SEQUENCE=4
```



RUJ

- “Un-do Log”
- Used during undo of non-committed changes
- Transaction rollback
 - Explicit
 - Implicit (process or system failure)
- Verb rollback



RUJ Writes

- Data is sent to RUJ prior to line “mark”
- Must be written to disk prior to page modification written to disk
- RUJ contains prior content of every line modification made by transaction
- “Cache” of DBKs modified during transaction or verb
 - Limit writing same data multiple times



RUJ at commit

- RUJ is logically truncated at transaction commit
- Once transaction commits, undo information for the transaction is no longer required



Undo

- Either via process or DBR
 - Process during explicit ROLLBACK or verb rollback
 - DBR during process or node failure recovery
- RUJ file read backwards
- Each operation is “undone” to database by replacing page content with prior data
- Each entry in RUJ file points to prior entry

RMU /DUMP /RECOVERY_JOURNAL

Recovery unit journal file structure is 73.0
Recovery unit journal file creation timestamp is 12-OCT-2009 22:17:05
Database rootfile is DISK\$DATA:[DB.V73]FOO.RDB;3
Journal file was created by process 2824C871:1

This JFA 512	Record sequence number 1
Prior JFA 0	Current TSN is 0:34
Transaction context	
AIJ commit scan stop location is 2:0	
This JFA 548	Record sequence number 1
Prior JFA 512	Previous TSN was 0:0
Stored segment 6:454:1	
This JFA 584	Record sequence number 2
Prior JFA 548	Previous TSN was 0:32
Modified segment 16:578:0 with length of 430 bytes	




Hints

- Distribute RUJ files over multiple volumes
 - Via RDM\$RUJ
- AIJ files should be large enough to reduce switches
- Explicitly backup AIJs prior to database backup
- Keep AIJs on different volumes from database files



For More Information

search.oracle.com



or

oracle.com/rdb



ORACLE IS THE INFORMATION COMPANY