Recent Customer Experiences with Oracle Rdb

Bill Gettys

Oracle New England Development Center

Copyright 2003, Oracle Corporation

Part 1: Data Replication in Oracle Rdb

Why Data Replication is Important

Information access is increasingly important; disks are increasingly cheap

- Ad hoc, reporting access interferes with OLTP
- Access to information needs to be continuous, but
  - Databases must sometimes be restructured
  - Databases must sometimes be isolated
  - Databases and systems sometimes fail
- Information must be protected from disaster
- Oracle Rdb is not always the right database management system

ORACLE

5 Methods

Replication Method	Type	Year
Replication Option for Rdb	Table	~1985
Hot Standby	Journal	~1995
LogMiner/Loader	Journal	2000-02
Application Based	Table	
Shadowing/Mirroring	Disk	~1990

ORACLE

Replication Option for Rdb

- Easy to set up
- Define transfer

```
SQL> CREATE TRANSFER MY_TRANSFER TYPE IS REPLICATION
cont> MOVE TABLES TAB1
cont> TO EXISTING FILENAME DISK:[DIR]TARGET.RDB
cont> LOGFILE IS DISK:[DIR]MYTAB_EXTRACT.LOG;
```

- Define a schedule

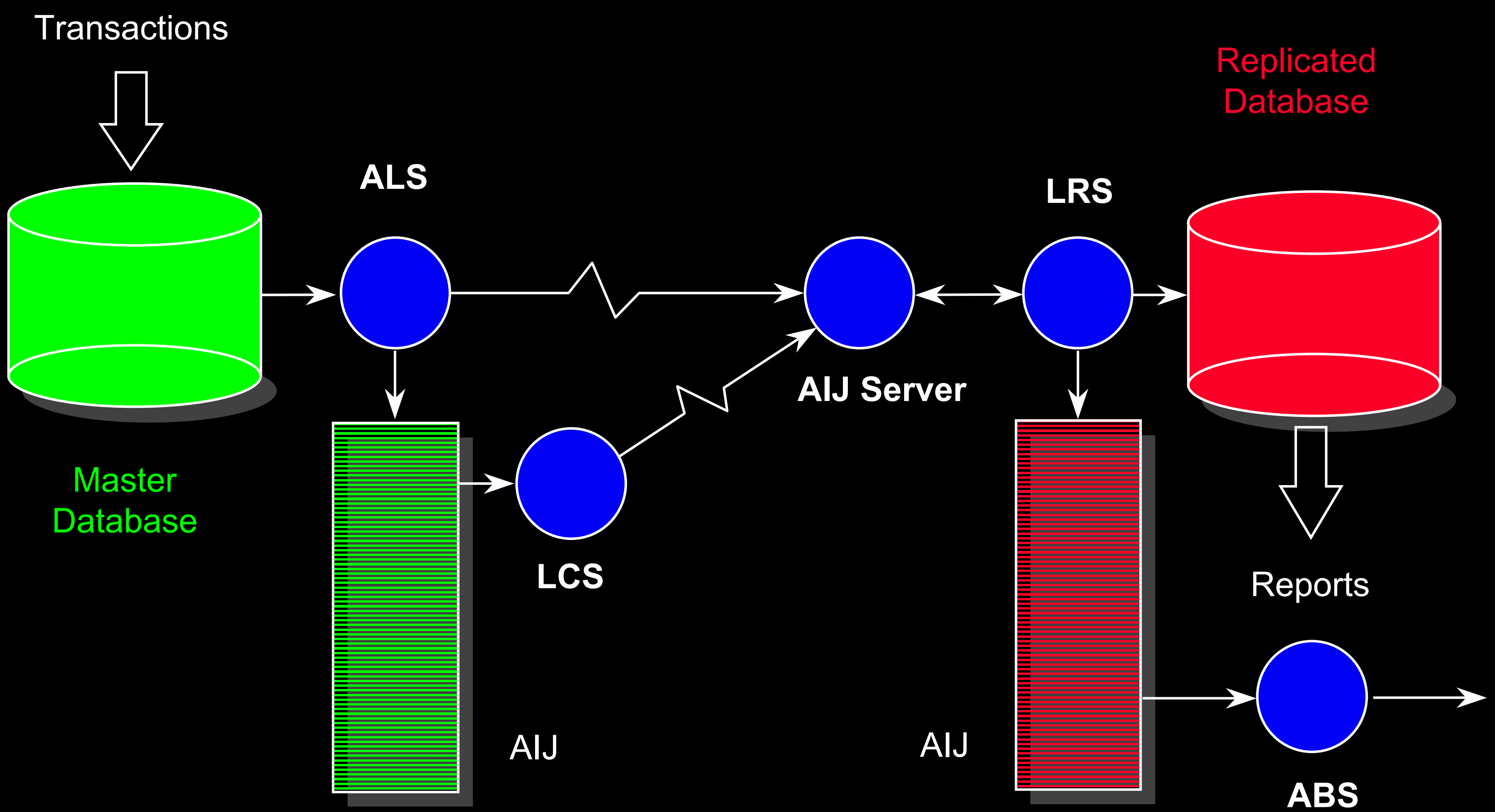
```
CREATE SCHEDULE FOR MY_TRANSFER
START 14-MAR-2003 11:00:00.00
EVERY 1 00:00:00.00
RETRY 3 TIMES RETRY EVERY 0 00:30:00;
```

Replication Option for Rdb

- Extraction and replication supported
- Scheduled rather than event driven
- Extensively used, reliable
- Transactional
- But,
  - Hot Spot: RDB$CHANGES table and index
  - Locking on sorted index
  - Double journaled
  - Possible performance issues on target system

ORACLE

Hot Standby Architecture



Hot Standby

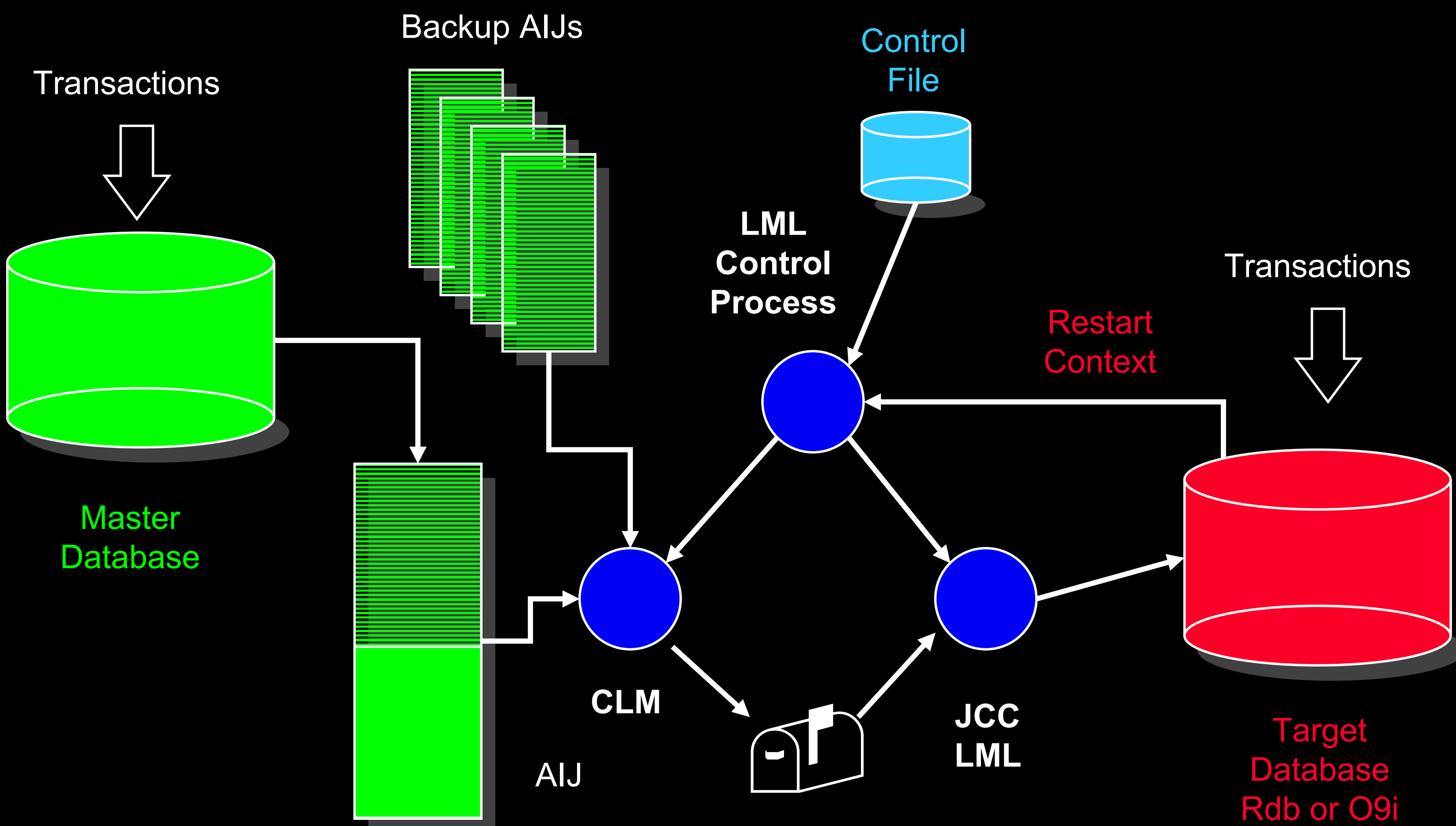
- Excellent performance
  - Near zero cost on master database
  - Standby cost much lower than SQL
    - Physical address based replication
    - Asynchronous IO operations
- Exceedingly low network overhead
- Event, not schedule driven
- Transactional
- Extensively used

Hot Standby (Cont.)

- No real geographic limit
- Excellent recovery from network failure
- Configurable database consistency
- Also maintains standby copy of AIJ
- But,
  - Entire database is replicated
  - Standby database can be read but not written
    - Isolation level is read committed
  - Design center is failover, not reporting
  - Limited to single target database
  - Can’t back up standby database

ORACLE

Continuous LogMiner / JCC LogMiner Loader



CLM/LML

- Transactional
  - One or many source transactions = one target transaction
- Event driven or scheduled (Static LogMiner)
- Excellent performance on source database
  - Uses journals, not tables
  - Takes advantage of hardware disk cache; no database hot spot
- Excellent performance on target database
  - Multiple load threads now supported
- Multiple target databases supported

CLM/LML (Cont.)

- No Geographic Limit
- Low network overhead
- Write your own loader if you like
- Lots of flexibility
  - Logical data model
  - Physical implementation
  - Supports Rdb, Oracle, Tuxedo targets; more possible
  - Read/write access to target possible (be careful)

ORACLE

CLM/LML (Cont.)

- But,
  - More overhead than Hot Standby
  - More complex to set up than ROR
  - Not as extensively used, but…

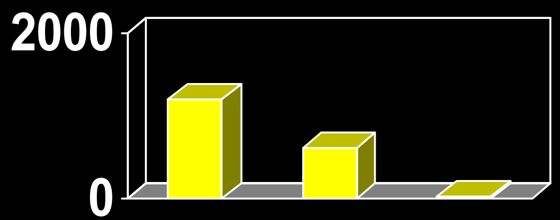
ORACLE

One Customer's Experience with On-line Restructure

- 50GB database
  - 175 tables
  - Largest > 60 million rows; multiple 20+ million row
- Dual-processor ES40, 5GB, SCSI, SW-Raid
- Parallel unload/load streams
- Uses JCC's LogMiner-Loader Technology
  - www.jcc.com

Method	Downtime
Traditional Export/Import	20 hours
Parallel UNLOAD/LOAD	10.5 hours
LogMiner Approach	32 Minutes

Production Downtime in Minutes



ORACLE

Customer Experience 2

- Travel industry reservation system (after September 11 => little capital available)
- Must provide rapid internet access to rate information or the business dies
- Hot standby limitations
  - No Row Cache
  - One standby database
  - No index customization; all tables
- Rate information replicated to two other Rdb databases with < 5 second delay
- Database transaction duration for rate queries <0.02 seconds, independent of reservation system load

ORACLE

Customer Experience 3

- Huge OLTP system
  - 10 Rdb databases
  - 11 million customers
  - 2+ updates per customer per day
  - Most occur in a two hour batch window
- Able to load > 3,200 changes per second in a single Oracle 9i RAC database.

ORACLE

Application Based Replication

- But,
  - You're not seriously interested in writing *and maintaining* all the code required, are you?

ORACLE

Mirroring/Shadowing

- Really easy to implement
- Lots of successful implementations
- But,
  - Limited geographic separation between sites
  - High network bandwidth requirement
  - Really one database, so
    - No protection from software failures

ORACLE

5 Methods

Replication Method	Type	Year
Replication Option for Rdb	Table	~1985
Hot Standby	Journal	~1995
LogMiner/Loader	Journal	2000-02
Application Based	Table	
Shadowing/Mirroring	Disk	~1990

ORACLE

Part 2: Row Cache Benefits with Rdb

Minnesota State Colleges & Universities

Why Row Cache?

- Cache individual records/index nodes
- Avoids page locking
- Can modify records in cache; no database I/O
- VLM → cache many records in memory
- Faster
  - code path for reading
  - checkpointing from cache to disk

ORACLE

…It can make a difference

- Less than 1 I/O per transaction
- Entire sorted indexes locked into memory
- Row modification with no database I/O
- Thousands of modified rows in memory
- Very Large Memory support

Where Row Cache has Stumbled

- Heavy update activity
  - Although cached indexes can often help
- When snapshots are enabled
- Caching many, many rows

ORACLE

Review…
What are Snapshots

- Before RW modifies row, copies current content to “snapshot” storage area for RO
- Allows RO to see consistent, unchanging view of database for duration of transaction
- Space reclaimable as oldest transactions commit

ORACLE

Work in Progress

- Snapshots in Cache
- 64 bit Row Cache

Snapshots & Row Cache

- Initial row cache design didn’t allow snapshots at all
- Phase II added snapshot support with RO & RW

ORACLE

The Problem…

- Too much I/O & locking
  - RW writing to snapshot area
  - RW updating live page with snapshot pointer
  - RO reading snapshot page(s)
- Contention for the snapshot pages
- Contention for the live pages

ORACLE

…A Solution

- Store snapshot copy of row in cache
- Memory write is faster than disk write
- RW can quickly write it
  - No need to write snapshot page
  - No need to update live page
- RO can quickly search for it

Snapshots in Cache

- One visible parameter
  - Number of snapshot rows per cache
- Snapshot chain maintained in cache slots
  - Negative snapshot pointer → slot number in cache
  - Positive snapshot pointer → page number on disk

ORACLE

Cache Sizing Suggestions

- Snapshot cache may be much larger than "regular" part of cache
  - Ratio of live area size to snapshot area size
  - Similar needs
- Long running transactions may cause RW transactions to experience slowness
  - Writing lots of snapshots back to disk

ORACLE

Modified Rows in Memory

- Many modified rows in memory
  - Checkpoints, shutdowns, backups, verifies can take longer → *a lot longer*
- Other changes with prestarted transactions & stale checkpoints helps ease recovery planning
- AIJ is your lifeline - only place data is on disk
  - Hot Standby provides additional protection

ORACLE

Other Considerations

- Limits
  - ~2,100,000,000 pages per snapshot storage area
  - ~2,100,000,000 total slots per cache
- RCS can probably be taught to move snaps from cache to disk proactively
  - May have to look into reducing RCS process priority
- Reduced I/O can (greatly) increase average CPU consumption

ORACLE

Possible Restriction

- For the first production release, objects stored in mixed-format areas won't be eligible for snapshots in cache
  - Sequential scans are problematic

ORACLE

Native 64-bit Row Cache

- Replace existing VLM technique
- Improved performance
- Larger caches viable

ORACLE

32-bit Background

- P0 address space
  - 1GB
  - ...SHARED MEMORY IS PROCESS
- P1 address space
  - 1 GB
  - Mostly DCL & RMS
  - Not used directly by Rdb
- S0/S1 address space
  - 2GB
  - ...SHARED MEMORY IS SYSTEM

Existing VLM Method

- P0 virtual address "window" moved to different physical address locations
- Additional CPU to "turn" window
  - Updates page table entry
  - Invalidate TB
- Kernel-mode code – knows VMS memory management
- Rdb shipped with VLM before VMS

ORACLE

Rdb's Existing Row Cache VLM Limitations

- Some data structures always live in 32-bit space
  - "GRIC" (24 bytes per cache slot)
  - Hash table (~8 bytes per cache slot)
  - Bit vector (one bit per cache slot)
- Limits total number of cache slots
  - 1GB ~= absolute max of ~ 33,000,000 (really less)
- Run-time "window turn" cost

ORACLE

…VMS V7 adds native 64-bit support

- System services allow process manipulation of memory beyond 1GB
- Additional performance options (memory resident, shared page tables, granularity hints)

ORACLE

64-bit & VMS

- P2 address space
  - At least ~4TB
- S2 address space
  - At least ~1TB
  - PFN database
  - Global page table
  - Lock management structures
  - XFC

ORACLE

Row Cache
Moves to 64-Bit Space

- The Cache
  - Cached slots (record & overhead)
  - Hash table
  - Bitmap
- P2 global sections
  - Optional
    - Resident with shared page tables
    - Galaxy resident

ORACLE

64-bit Implementation

- Effectively no…
  - algorithmic changes
  - user-visible changes
- Modify data structures to use 64-bit addresses for row cache shared data
- Return all data to caller via RCWS

What it all Means

- Snapshots in cache
  - Potentially huge reduction in I/O for environments with snapshots enabled
- 64-bit Row Cache
  - Nearly limitless number of records in cache
  - Improved performance over VLM

ORACLE

Introduction

- MnSCU Comprised of 37 Institutions
  - 8 State Universities
  - 29 Community and Technical Colleges
  - Serves over 250,000 Students per year
- Integrated State-wide Record System (ISRS) Application
- Written in Uniface, Cobol, C, JAVA
  - 1951+ 3GL programs, 2181+ 4GL forms
  - 2,460,832+ lines of code

ORACLE

Database Overview

- 4 Distributed Regional Computer Centers
  - Production is GS160
  - OpenVMS 7.2-2 (without fast-path)
  - Hot Standby on 3-4100's clustered to the GS160
- 39 Production ISRS databases (v7.0-63)
  - Each with 1173 tables and 1443 indexes
  - Each with Hot Standby enabled
- 20+ Development, QC, Training, Testing databases
- 6 Regional / Central databases
- Over 500,000,000 rows in production ISRS databases
- Over 550 Gb Production ISRS Db disk space
- Over 1Terra-byte total database disk space

ORACLE

Server Configuration

- GS160
  - Partition 0&1: 2 QBB's – each w/ 4 1001 MHZ CPU's
    - 32 Gb Memory w/ 32 way interleaving
    - 2 HSG80 Dual Redundant Fiber Controllers
      - connected to 16 port Fiber channel SAN switch
      - 8 RAID 3-5 Sets each w/ 36 Gb disk (10,000 RPM)
      - 512 Mb mirrored disk Cache
      - 1.2 ms response
    - 8 HSZ80 Dual Redundant SCSI Controllers
      - 6 RAID 3-5 Sets each w/ 18 Gb disk (10,000 RPM)
      - 512 Mb mirrored disk Cache
      - 1.2 ms response
  - Partition 2: 1 QBB w/ 2 1001 MHZ CPU's (2Gb memory)
    - True64 Web Server
- Hot Standby
  - 3 Alpha 4100's Clustered to the GS160, each with:
    - 4Gb memory
    - 3 466 MHZ CPU's
    - 6 HSZ50 RAID5 Sets each w/ 20 Gb disk (7,200 RPM)

Users

- Each regional server supports between 400 and 800 on-line users (during the day)
- Many batch reporting and update jobs daily and over-night
- 10,000+ Web transactions each day 24x7

Topics of Discussion

- The Problem
- Proof of Concept Testing
- Performance Benchmarks
- Determining what to Cache
- Cache Implementation
- Cache Tuning
- Post Implementation Statistics

ORACLE

Our Problem

- Database Tuning has been nearly ignored for seven years so there were ample opportunities for improvement!
- In July systems managers announced that fall term start-up ‘will bury the machines’
- No time to re-write expensive portions of application
- Had to deliver a solution that would produce large performance gains with no additional resources
- ROW CACHE is the only hope!

ORACLE

The Challenge

- Had to focus on the most expensive portions of the system
- Had list of known expensive 3GL programs, but had nearly no knowledge of Uniface data access patterns
- Spent 2 weeks of intensive tuning – starting with expensive processes and ‘hot’ storage areas
- Obviously did not have time to do extensive tuning

Proof of Concept

- We were certain Row Cache could help a lot
- Needed to ‘try’ it on a small scale as a proof of concept

Concept Testing

- Initial tests on two portions of the application
  - Registration process
  - Full Tuition Calculation
- Preliminary testing showed we could have significant performance improvements by index and query tuning and using Row Cache and Global Buffers

ORACLE

Registration Performance

- Registration Test
  - Many processing steps plus a query from a known expensive view
- Full Tuition Calculation Test
  - Perform a tuition calculation for all students for one term at one institution
- Captured execution times and I/O statistics as benchmark baselines.
- Then implemented Row Cache and Global Buffers

Registration Performance

	Synchronous Reads	Asynchronous Reads
Before RC and GB	19,214	6,158
After RC and GB	3	7

Full Tuition Calc Test

	Production Batch 18-May-2002	After RC and GB
Buffered I/O	3,210	3,504
Direct I/O	69,258,744	1,675,824
CPU Time	4:31:02.88	1:06:33.28
Elapsed Time	12:09:45.57	2:28:12.71

ORACLE

Test Results

- These tests showed us there are great performance gains to be had
- Its relatively easy to tune a single program or set of programs
- Challenge is how to tune an application as large as ISRS (in the timeframe we were allowed)

ORACLE

Benchmarks

- Next captured some System and Database statistics

Benchmarks

- System Stats:
  - Overall Cluster Disk I/O rates
  - Overall Cluster CPU rates
- Database Stats:
  - Synchronous data reads
  - Asynchronous data reads
  - Transactions per second
  - Others – Locks Requested per Trans and Average Trans Duration

ORACLE

System Benchmarks

- Cluster Disk I/O Rates
  - One production system was averaging 4000 to 5000 I/O's per second, with peaks over 5000 prior to row cache
- Cluster CPU Rates
  - Averaging 50% use prior to row cache

ORACLE

Database I/O

Database	Synch I/O	Asynch I/O
MNSCUNRC	113.7	80.9
MNSCUCEM	119.3	103.1
WIN	148.1	54.9
SCSU	222.9	133.7
BEM	160.0	67.5
MHD	196.1	106.2
MAN	138.1	81.2

ORACLE

Database Transactions

Region	Trans/Sec Non-Cached 9/4/2002
MNSCU1	0.16
METE	0.23
STC2	0.23
SATURN	0.30

ORACLE

What to Cache

- Needed to analyze data access patterns
- Used JCC's workload analysis tool to capture SQL and store in a database
- Used RMU file statistics

ORACLE

JCC's Workload Analysis Tool[1]

- Aided in determining SQL generated by Uniface
- Allowed us to determine the most costly queries quickly
- Data gathered from production – real queries by real users

[1]See http://www.jcc.com/

ORACLE

RMU File Stats

mete.metro.mnscu.edu (1) [mete] - QVT/Term

File Edit View Setup Keymaps Font Printer Commands Net Apps Help

```
Node: METE (1/1/16)              Oracle Rdb V7.0-63 Perf. Monitor          3-OCT-2002 10:28:35.71
Rate: 3.00 Seconds                                                   Elapsed: 4 20:31:11.35
Page:  1 of 31           ISRSMNSCUMETDB_ROOTA:[DATA]ISRS_DB.RDB;1                Mode: Online
```

File/Storage.Area.Name........	Sync.Reads	SyncWrites	AsyncReads	AsyncWrits	PgDis
Database Root	1515	10680	0	269847	0
AIJ (After-Image Journal)	5980	10	38	27930	0
RUJ (Recovery-Unit Journal)	5449	1660	0	75524	0
ACE (AIJ Cache Electronic)	0	0	0	0	0
All data/snap files	21616745	90446	15711460	192457	185k
data ISRS_DB	5101998	7663	476067	3546	4959
data PERSON	372380	759	1406814	172	1
data PERSON_INDEX	3196533	1899	41474	143	11910
data ADDRESS	90710	366	23117	280	0
data ADDRESS_INDEX	156020	1154	5939	827	23847
data ISRS_SA	809	137	0	222	0
data ISRS_INDEX	3372	139	1279	170	0
data NON_PERSON	26141	0	48891	0	0
data NON_PERSON_INDEX	268007	0	39152	0	0
data HELP_TEXT	0	0	0	0	0
data HELP_TEXT_INDEX	0	0	0	0	0
data HELP_FIELD	100734	216	114601	88	0
data HELP_FIELD_INDEX	0	0	0	0	0
data EMPL_DEMO	12840	18	5199	1	0
data EMPL_DEMO_INDEX	18695	2	244	0	0
data EMPL_BARG	44313	11	72	0	0
data EMPL_BARG_INDEX	17253	8	66	0	0
data EMPL_SENR	10	0	0	0	0
data EMPL_SENR_INDEX	2	0	0	0	0

```
onfig  xit  ilter  elp  enu  next_page  prev_page  ptions  eset  et_rate  rite  oom
```

Notice the system area tops the list!

Cache Implementation

- Based on analysis of data access patterns, we created 25 caches that could affect up to 337 tables and 440 indexes
- Tuning also included index changes and moving user data out of rdb$system area

ORACLE

Cache Implementation

- Initial Cache sizing is between about 30mb and 60mb of System Memory per database
- Between 500mb and 1500mb of total memory per database
- Used spreadsheet to do sizing and compare results (available from MetaLink)

ORACLE

Sizing Spreadsheet

Microsoft Excel - Row Cache Info

K4 =G4/(1024*1024)

STC2 SCSU db Cache Information

Row	Cache	Type	Rows	Row Size	Est. Mem (mb)	Sys mem	Physical	VLM	Sys mem	Physical	VLM	Total	%diff over estimated
2						-------- dump info (bytes) ---------			--------- dump info (mb) -----------				
4	Rdb$System_Area_Cache	Physical	100,000	700	74.8	2,957,312	72,400,000	1,641,992	2.8	69.0	1.6	73.4	-2%
5	Utf_Detl_Ar_Index_Area_Cache	Physical	100,000	432	47.7	2,957,312	45,600,000	1,641,992	2.8	43.5	1.6	47.9	0%
6	Utf_Detl_Ar	Logical	500,000	320	175.5	14,180,352	172,000,000	1,641,992	13.5	164.0	1.6	179.1	2%
7	Ct_Cou_Index	Physical	10,000	432	6.2	327,680	4,560,000	1,641,992	0.3	4.3	1.6	6.2	0%
8	Yrtr_Cal_Dates_Index_C	Logical	500	960	2.1	32,768	492,000	1,641,992	0.0	0.5	1.6	2.1	-1%
9	Yrtr_Cal_Index_C	Logical	500	960	2.1	32,768	492,000	1,641,992	0.0	0.5	1.6	2.1	-1%
10	Utf_Event_Index_Area_Cache	Physical	200,000	432	93.8	5,898,240	91,200,000	1,641,992	5.6	87.0	1.6	94.2	0%
11	Utf_Detail_Index_Area_Cache	Physical	200,000	432	93.8	5,898,240	91,200,000	1,641,992	5.6	87.0	1.6	94.2	0%
12	Ar_Misc_Data_Index_Area_Cache	Physical	5,000	432	3.9	172,032	2,280,000	1,641,992	0.2	2.2	1.6	3.9	0%
13	St_Index_Area_Cache	Physical	100,000	430	47.5	2,957,312	45,600,000	1,641,992	2.8	43.5	1.6	47.9	1%
14	Rg_Index_Area_Cache	Physical	80,000	432	38.5	2,473,984	36,480,000	1,641,992	2.4	34.8	1.6	38.7	1%
15	Index_Cache	Physical	50,000	1,000	53.4	1,490,944	51,200,000	1,641,992	1.4	48.8	1.6	51.8	-3%
16	Ar_Chg_Generate	Logical	200	51	1.6	24,576	19,200	1,641,992	0.0	0.0	1.6	1.6	0%
17	Need_Anal_Index_Cache	Physical	10,000	1,000	12.0	327,680	10,240,000	1,641,992	0.3	9.8	1.6	11.6	-3%
18	Ct_Cou	Logical	30,000	1,388	44.4	876,544	42,360,000	1,641,992	0.8	40.4	1.6	42.8	-4%
19	Ps_Index_Area_Cache	Physical	5,000	480	4.1	172,032	2,520,000	1,641,992	0.2	2.4	1.6	4.1	0%
20	Misc_Data_Area_Cache	Physical	70,000	100	10.4	1,728,512	7,440,000	1,641,992	1.6	7.1	1.6	10.3	-1%
21	Misc_Index_Area_Cache	Physical	25,000	430	13.1	753,664	11,400,000	1,641,992	0.7	10.9	1.6	13.2	1%
22	rg_cntrl_edit	Logical	200	48	1.6	24,576	14,400	1,641,992	0.0	0.0	1.6	1.6	-1%
23	rg_edit_parms	Logical	1,000	84	1.7	24,576	10,800	1,641,992	0.0	0.0	1.6	1.6	-6%
24	rg_log	Logical	5,000	104	2.2	131,072	512,000	1,641,992	0.1	0.5	1.6	2.2	-3%
25	rg_ovrrd	Logical	6,000	120	2.5	172,032	720,000	1,641,992	0.2	0.7	1.6	2.4	-2%
26	st_term_data	Logical	50,000	408	23.4	1,490,944	21,600,000	1,641,992	1.4	20.6	1.6	23.6	1%
27	val_rg_edit	Logical	100	792	1.7	24,576	81,600	1,641,992	0.0	0.1	1.6	1.7	-1%
28	Index_cache_2	Physical	50,000	1,000	53.4	1,490,944	51,200,000	1,641,992	1.4	48.8	1.6	51.8	-3%
30	TOTALS:		Est. Total Mem:		811.3		Actual Memory used:		44.5	726.3		809.9	0%
31			Est. sys Memory:		44.6								

MNSCUIHC / MNSCUNRC / MNSCUNHC / MNSCUHTD / MNSCUCEM / Generic / **Sheet1**

Ready

ORACLE

Cache Tuning

- Some caches were very successful from the start, while others were under-sized somewhat
- Several dbs have had the caches re-sized to accommodate actual data access

Cache Tuning

Node: MNSCU1 (1/1/1) Oracle Rdb V7.0-63 Perf. Monitor 4-SEP-2002 09:51:24.48
Rate: 3.00 Seconds Row Cache Overview (Unsorted) Elapsed: 24 00:53:51.00
Page: 1 of 2 ISRSMHDDB_ROOTA:[DATA]ISRS_DB.RDB;1 Mode: Online

Cache.Name......................	#Searches	Hit%	Full%	#Inserts	#Wrap	#Slots	Len
RDB$SYSTEM_AREA_CACHE	247251472	99.1	68.6	1659724	21	100000	700
UTF_DETL_AR_INDEX_AREA_CACHE	171363423	99.3	94.4	1028411	10	100000	432
UTF_DETL_AR	47248974	48.4	99.9	20338882	40	500000	320
CT_COU_INDEX	[illegible]	[illegible]	97.9	9836571	[illegible]	[illegible]	432
YRTR_CAL_DATES_INDEX_C	[illegible]	[illegible]	2.4	12	[illegible]	[illegible]	960
YRTR_CAL_INDEX_C	[illegible]	[illegible]	0.8	4	[illegible]	[illegible]	960
UTF_EVENT_INDEX_AREA_CACHE	[illegible]	[illegible]	93.6	2153513	[illegible]	[illegible]	432
UTF_DETAIL_INDEX_AREA_CACHE	57290613	[illegible]	91.3	1379276	[illegible]	200000	432
AR_MISC_DATA_INDEX_AREA_CACHE	3909624	[illegible]	21.2	17266	[illegible]	5000	432
ST_INDEX_AREA_CACHE	66509402	[illegible]	91.0	2128460	[illegible]	100000	432
RG_INDEX_AREA_CACHE	7752297	98.0	62.7	153098	0	80000	432
INDEX_CACHE	22707002	99.2	39.3	174174	9	50000	1000
AR_CHG_GENERATE	7246	43.4	12.0	24	0	200	72
NEED_ANAL_INDEX_CACHE	2462030	95.2	88.6	117144	12	10000	1000
CT_COU	65021925	99.3	83.7	429136	19	30000	1388
PS_INDEX_AREA_CACHE	2459960	99.0	57.0	22607	7	5000	480
MISC_DATA_AREA_CACHE	38997649	31.6	47.5	1857080	19	60000	100

Good Hit %

No Wraps

ORACLE

Cache Performance

```
Node: MNSCU1 (1/1/1)          Oracle Rdb V7.0-63 Perf. Monitor    4-SEP-2002 09:51:24.48
Rate: 3.00 Seconds              Row Cache Overview (Unsorted)    Elapsed: 24 00:53:51.00
Page: 1 of 2                 ISRSMHDDB_ROOTA:[DATA]ISRS_DB.RDB;1               Mode: Online
```

Cache.Name..............	ches		%	#Inserts	#Wrap	#Slots	Len
RDB$SYSTEM_AREA_CACHE	1472		5	1659724	21	100000	700
UTF_DETL_AR_INDEX_AREA_CA	3423		4	1028411	10	100000	432
UTF_DETL_AR	8974		9	20338882	40	500000	320
CT_COU_INDEX	61356		9	9836571	1012	10000	432
YRTR_CAL_DATES_INDEX_C	960250	9	2.4	12	0	500	960
YRTR_CAL_INDEX_C	103813	9 .9	0.8	4	0	500	960
UTF_EVENT_INDEX_AREA_CACHE	1518643232	99.8	93.6	2153513	11	200000	432
UTF_DETAIL_INDEX_AREA_CACHE	57290613	97.5	91.3	1379276	7	200000	432
AR_MISC_DATA_INDEX_AREA_CACHE	3909624	95.8	21.2	17266	0	5000	432
ST_INDEX_AREA_CACHE	66509402	96.7	91.0	2128460	9	100000	432
RG_INDEX_AREA_CACHE	7752297	98.0	62.7	153098	0	80000	432
INDEX_CACHE	22707002	99.2	39.3	174174	9	50000	1000

Check Out This Number

With This Hit %

ORACLE

Post Implementation Statistics

- System Performance
- Database Performance
- Application Performance

System Performance

- Cluster-wide I/O was only about 1500/sec, even during periods last fall with more users than ever
- Cluster CPU Usage is now about 30% - down significantly from before

ORACLE

Database Performance

Database	Synch I/O	Asynch I/O	Synch I/O	Asynch I/O
MNSCUNRC	113.7	80.9	44.5	7.1
MNSCUCEM	119.3	103.1	51.4	13.0
WIN	148.1	54.9	56.1	7.3
SCSU	222.9	133.7	79.6	14.6
BEM	160.0	67.5	57.2	12.8
MHD	196.1	106.2	99.4	38.0
MAN	138.1	81.2	89.8	17.1

Database Performance

Region	Trans/Sec Non-Cached 9/4/2002	Trans/Sec Cached 9/4/2002
MNSCU1	0.16	0.55
METE	0.23	0.50
STC2	0.23	1.40
SATURN	0.30	1.00

ORACLE

Web Function Statistics

Function	Before	After	% Improve	2nd After	2nd %
DRP	2.39	0.42	445	0.27	748
HLD	0.71	0.07	914	0.05	1320
…	…	…	…	…	…
OSI	1.22	0.19	542	0.14	771
SCH	0.87	0.18	383	0.10	770
VCR	1.18	0.05	2260	0.04	2850
…	…	…	…	…	…
Avg	1.04	0.23	975	0.17	1006

Conclusion

- Row Cache has been exceptionally successful for MnSCU
- This has been only ‘Phase 1’ – with more databases getting cached and tuned, our servers performance should continue to improve

ORACLE

Summary

- You can make real differences in performance and efficiency by
  - Replicating data using the most efficient technique for your environment
  - Using row cache to pin critical tables and indexes in memory
- Row Cache enhancements will soon allow
  - Read + write performance improvements for databases with snapshots
  - Much larger caches

ORACLE

For More Information

- http://www.oracle.com/rdb
- http://www.jcc.com/
- http://metalink.oracle.com/
- http://www.openvms.compaq.com/
- bill.gettys@oracle.com
- miles.oustad@csu.mnscu.edu
  +1 (218) 755-4614

QUESTIONS & ANSWERS

ORACLE®