Rdb Continuous LogMiner and the JCC LogMiner Loader

A presentation of MnSCU's use of this technology

Minnesota
STATE COLLEGES
& UNIVERSITIES

Topics

- Overview of MnSCU
- Logmining Uses
- Loading Data
- Logminer Configuration Info

- Bonus: Global Buffer implementation and resulting performance improvements

Overview of MnSCU

Overview of MnSCU

Minnesota
STATE COLLEGES
& UNIVERSITIES

- **M**i**n**nesota **S**tate **C**ollege and **U**niversity System
  - Comprised of 37 Institutions
    - State Universities, Community and Technical Colleges
    - 53 Campuses in 46 Communities
    - Over 16,000 faculty and staff
    - More than 3,600 degree programs
  - Serves over 250,000 Students per year
    - Additional 130,000 Students in non-credit courses

ISRS

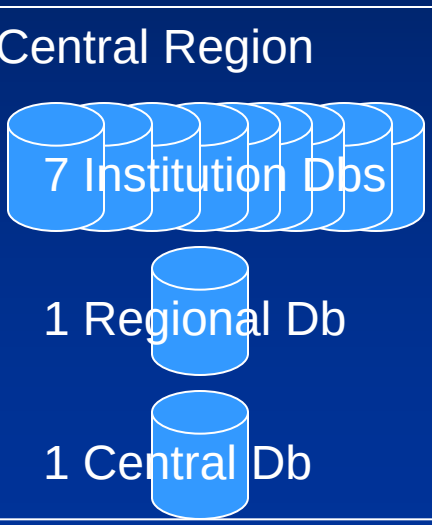
Minnesota
STATE COLLEGES
& UNIVERSITIES

- MnSCU's Primary Application
  - ISRS: Integrated State-wide Record System
- Written in Uniface (4GL), Cobol, C, JAVA
  - 2,000+ 3GL programs; 2,200+ 4GL forms
  - Over 4,000,000 lines of code

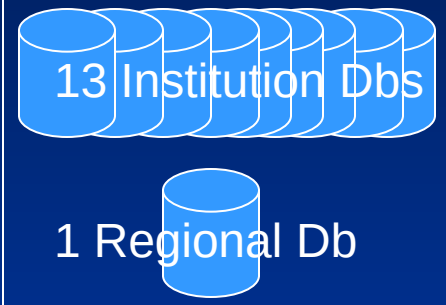
MnSCU's Rdb Topology

North Region

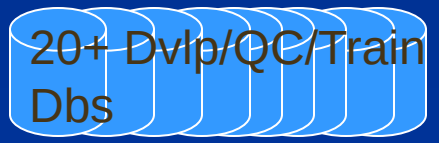
South Region



Central Region



Metro Region



Development

- Each Institution Db has 1283 tables
- Over 1,240,000,000 rows Total
- Over 1 Terra-byte total disk space
- Over 20% Annual Data Growth

Production Users

Minnesota
STATE COLLEGES
& UNIVERSITIES

- Each regional center supports:
  - Between 500 and 1,000 on-line users (during the day)
  - Numerous batch reporting and update jobs daily and over-night
  - 135,000+ Web transactions each day 24x7
    - Registrations, Grades, Charges (fees), On-line Payments, Open Sections Inquiry, etc…
    - 13,331,421 Registrations averaging 0.12 seconds
    - 36,116,726 Others averaging 0.34 seconds

Production System

GS-1280 Physical Configuration:

- 4 hard partitions (one for each region), each configured with 2 sub-partitions:
  - Production sub-partition: 12 CPU, 96 GB memory
  - Replicated sub-partition: 4 CPU, 32 GB memory

- We are using VMS Galaxy to be able to move CPUs between soft partitions if necessary. A total of 8 instances of VMS are running in the GS1280.

- We have disk space on a Storage Area Network (SAN). Distributed as follows:

	EVA5000 (GB)	EVA8000 (GB)	Total (GB)
METRO PROD		1,035	
REPL	352		
CNTRL PROD		891	
REPL	255		
SOUTH PROD		900	
REPL	331		
NORTH PROD		876	
REPL		255	
VMS TOTAL	938	3,957	4,895

LogMining

Minnesota
STATE COLLEGES
& UNIVERSITIES

LogMining Modes

- Static
  - The Rdb LogMiner runs, by default, as a stand-alone process against backup copies of the source database AIJ files
- Continuous
  - The Rdb LogMiner can run against live database AIJs to produce a continuous output stream that captures transactions when they are committed

LogMining Uses

- Hot Standby (replication) replacement
- Mining a single Db to multiple targets
- Mining to Non-Rdb Target(s)
  - XML, File, API, Tuxedo, Orrible
- Mining multiple Dbs to a single target
- Minimizing production Db maintenance downtime

LogMining at MnSCU

Minnesota
STATE COLLEGES
& UNIVERSITIES

The combination of the Rdb Continuous LogMiner and the JCC LogMiner Loader allow us to:

- Distribute centrally controlled data to multiple local databases
- Replicate production databases into multiple partitioned query databases
- Roll up multiple production databases into a single target
- Replicate production data into non-Rdb databases to support development of database-independent application

Source Db Preparation

- The LogMiner input to the Loader is created when the source database is enabled for LogMining
- This is accomplished with an RMU command. The optional parameter ‘continuous’ is used to specify continuous operation:

```
$ rmu/set logminer/enable[/continuous] <database name>
$ rmu/backup/after/quiet <database name> ...
```

- Many of the procedures included with the Loader kit rely on the procedure vms_functions.sql having been applied to the source database:

```
SQL> attach ‘filename <source database>’;
SQL> @jcc_tool_sql:vms_functions.sql
SQL> commit;
```

MnSCU's Oracle Topology

Minnesota
STATE COLLEGES
& UNIVERSITIES

APPL Instance

- 37 Institutions combined in 1 schema (448 tables each)
- 1 VAL Schema (141 Codes Tables)
- Other Schemas in support of specific applications

REPL Instance

- 37 Institutions combined in 1 schema (448 tables each)
- 1 VAL Schema (141 Codes Tables)

WHSE Instance

- Various Warehouse Schemas

The Big Picture

Minnesota STATE COLLEGES & UNIVERSITIES

REGIONALDB
REGIONALDB
REGIONALDB
REGIONALDB

CENTRALDB

CLM w/PK
2 sessions each

CLM w/PK
4 sessions

WAREHOUSE
CC_SUMMARY Combined (all 37)
MNSCUALL Combined (all 37)
Other Data Warehouse Structures
Warehouse Users

CLM w/PK
37 sessions each

APPL
VAL Tables 1 central copy
Applications
ISRS Tables Combined (all 37)
Tables for new Applications

36 Standby Rdb Reporting Dbs
HS
36 Dbs
37 Production Rdb DBs
NWR db

CLM w/PK
And FilterMaps
DBK+RC_ID
37 sessions

NTC
FFC
TRF
BTC

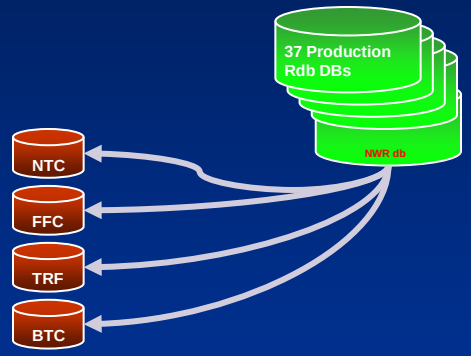
CLM w/DBK and Filter
4 sessions

CLM w/PK
And FilterMaps
DBK+RC_ID
37 sessions

REPL
ISRS Tables Combined (all 37)
VAL Tables 1 central copy
Ad-Hoc Users Read-Only

Mining Single Db to Multiple Targets

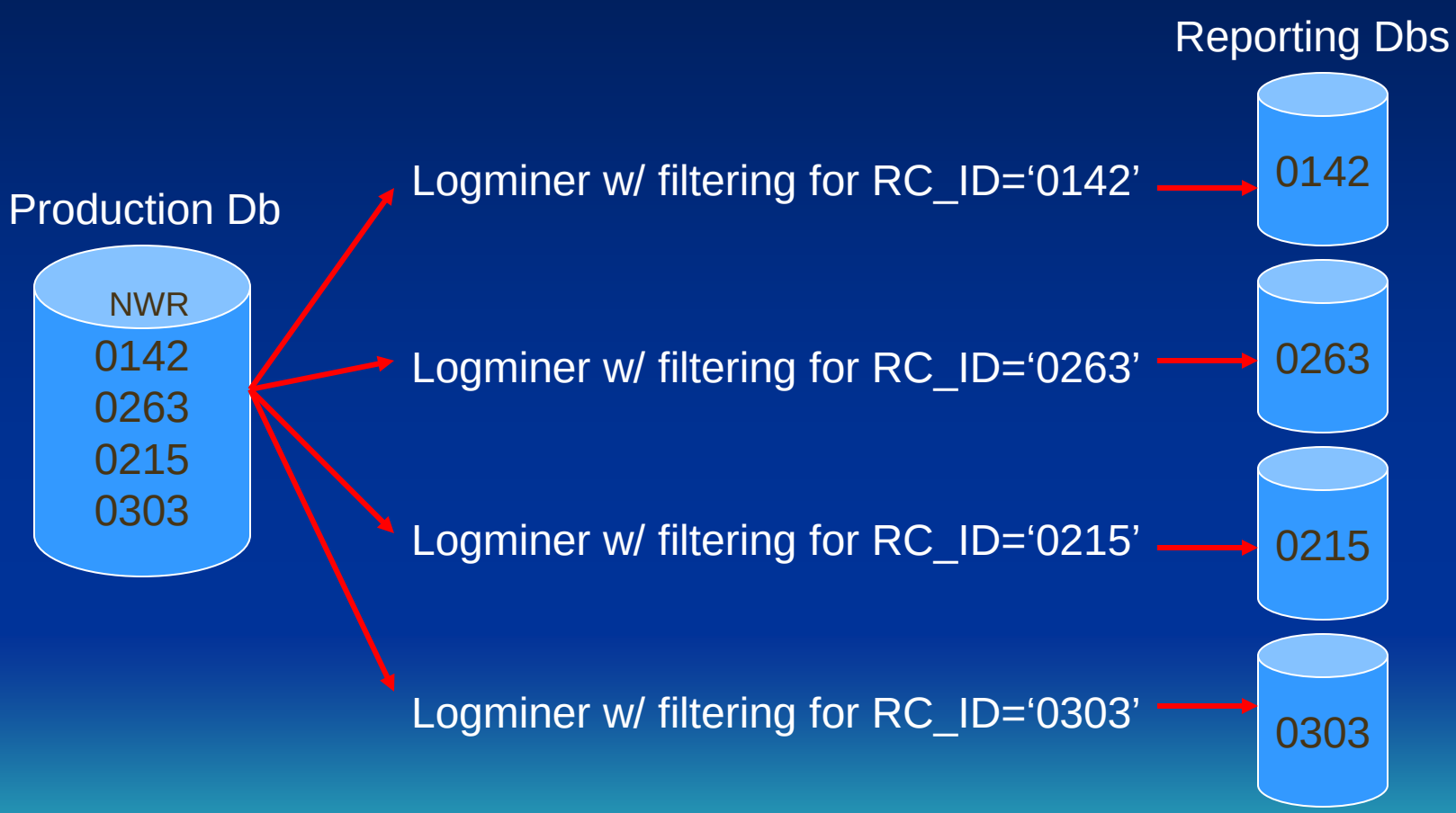
- We've begun combining institutions in some of our production databases
- Introduced a new column called RC_ID to over 700 tables for row-level security
- Row security provided by views
- However, the reporting databases still needed to be separate so users wouldn't have to change hundreds of existing queries to include RC_ID



Keyword: <filter>

LML Tip

Mining to Multiple Targets

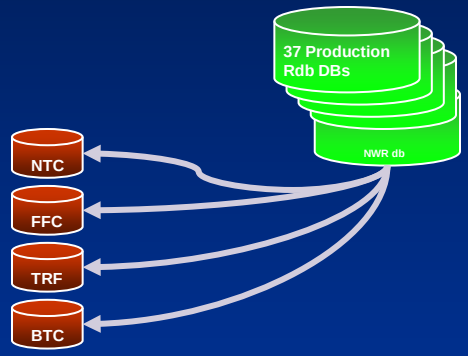


Keyword: filter~include~ST_APP_ADDR~rc_id='0303'

LML Tip

Replication Replacement

- Historically we've used Rdb's Hot Standby feature to maintain reporting Dbs for users (ODBC and some batch reports)
- However, of the 1200+ tables in production ISRS Dbs, we found users only use 260 tables for reporting from the standby Dbs
- Batch reports were found to use another 222 tables
- With Logminer, we can maintain just this subset of tables (482) for reporting purposes

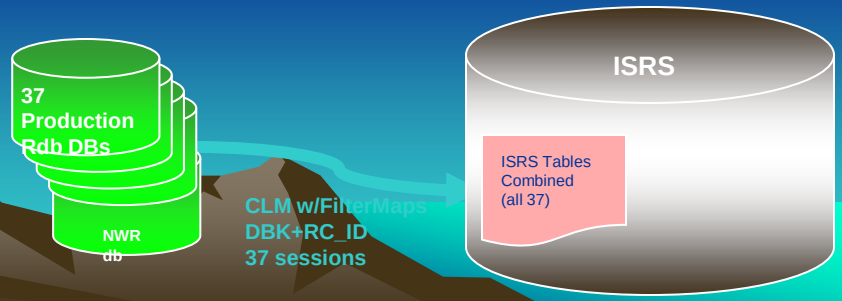


Logical: $ define[/system] JCC_ADD_CLM_SHARED_READ "T"

LML Tip

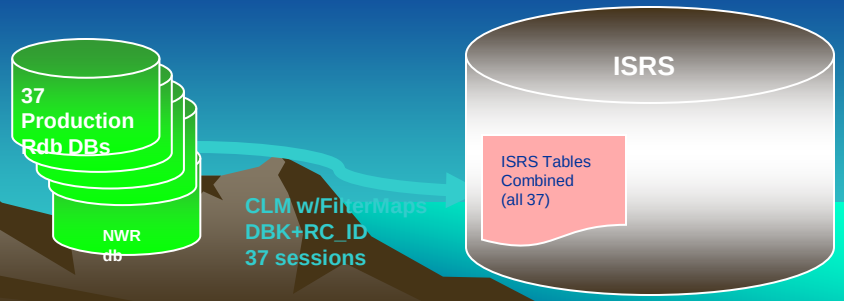
Combining Data

- Since we have 37 separate institutional databases, getting combined data for system-wide reporting was difficult
- With Logminer we can combine data from each of the production ISRS databases into one target, in this case Oracle
- Need PK defined



Non-Rdb Target

- With Logminer we can mine tables from each of our Production ISRS databases into Oracle
- This allows us to test our application against a different DBMS
  - Structure and content same as production
- Data logically separated (with VPD) for each institution's use



Target Db Preparation

- Besides the task of creating the target db and tables itself, there are many ‘details’ to attend to depending upon target db type
- For Oracle targets, Rdb field and table name lengths (and names) can be an issue (and target tablespaces too)
- One thing in common: the HighWater table
  - Used by the loader to keep track of what has been processed
  - AERCP stored here: looks like

    1-28-1941-9959-8798096-8798096

    aij# aij-blk tsn# tsn#

CHAR vs VARCHAR

Minnesota STATE COLLEGES & UNIVERSITIES

- Target data in Oracle can be CHAR or VARCHAR
- CHAR data in target always SPACE filled
  - TRIM on converts values of all SPACES to NULL
  - If PK contains ‘short’ values, logminer will not find match, so will attempt to insert
    - Insert will fail due to dups ☺
- VARCHAR data in target
  - TRIM on removes tailing SPACES, all SPACES become NULL
    - If source data allows SPACE as a valid value, queries in target have to change to allow for NULL (if TRIM is on)
  - TRIM off leaves trailing SPACES or all SPACES intact
    - Joins can become problematic
    - Logminer may not find match depending on # of trailing spaces in short key
- The Logminer Loader currently uses VARCHAR data so comparing to CHAR target data can be problematic
  - The Loader might change in the future to be based on target data type rather than assuming VARCHAR

We found ‘best practice’ seemed to be VARCHAR with TRIM on

LML Tip

Minimizing Production Downtime

Minnesota
STATE COLLEGES
& UNIVERSITIES

- Basic Steps:
  - Do an AIJ backup
  - Create copy of production Db
  - Perform restructuring / maintenance / etc on Db copy
    - This could take many hours
  - Remove users from Production Db
  - Apply AIJ transactions to Db copy using LogMiner
    - This step requires minimal time
  - Switch applications to use Db copy – This is now the new production Db!

Loading Data

Minnesota
STATE COLLEGES
& UNIVERSITIES

Loading Data

Minnesota
STATE COLLEGES
& UNIVERSITIES

- 3 Methods to accomplish this:
  - Direct Load (Oracle SQLLOADER)
    - Could impose data restrictions by using views
    - Can configure commit-interval
    - Cannot load Blob data or Clob data > 4000 bytes
  - LogMiner Pump
    - Use a no-change update transaction on source
    - Allows for data restrictions
    - Commit-interval matches source transaction
    - Consumes AIJ space
  - JCC Data Pump
    - Configurable to do parent/child tables, data restrictions, commit-interval and delay-interval to minimize performance impact
    - Consumes AIJ space
    - Relies on JCC Logminer active session to move the data

Loading Data Example

- Loaded a table with 64,914 rows, 91 bytes each
- Used LogMiner to ‘pump’ the rows via a no-change update from a single transaction
  - This took about 8 minutes; 13,800 AIJ-blocks
  - 1.2 million blocks of sortwork files
  - LML used about 210,000 I/O
- Using the JCC Data Pump (commit interval 1000)
  - Same AIJ and I/O usage / no sortwork files
  - About 6.5 minutes to insert target data
  - About 20% faster than single transaction
  - Also more control and flexibility

Loading Data Example

- SQLLOADER
  - Used RMU to create text .UNL file
  - This took less than 30 seconds, no AIJ blocks, 20,454 blocks for .UNL file
  - Obviously removing the sort and AIJ utilization significantly speeds up the process

Loader Performance

Minnesota
STATE COLLEGES
& UNIVERSITIES

- SCSU_REPISRS session (255 tables)
  - From CLM log:

    19-APR-2006 05:00:08.52 20386121 CLM SCSU_REPISR Total : 53426 records written (50923 modify, 2503 delete)
  - The 3 processes themselves:
    - CTL: 1 min 22.5 CPU secs / 7331 Direct IO 1.8 mil buff
    - CLM: 1 min 3.27 CPU secs / 123,637 Direct IO 94889 buff
    - LML: 2 min 45.7 CPU secs / 125 Direct IO 2.2 mill buff
  - After 23:48 hours of connect time (an entire day), this is about 1.5 Direct IO per second average

Performance

- CSV and T4 stats
- Put example here…

Operations Heartbeat

Minnesota STATE COLLEGES & UNIVERSITIES

- Without Heartbeat a session can become ‘stale’
- AIJ backups can be blocked
  - Example: the last ‘logmined’ table updated was in prior journal, aij-backup would be blocked
- With Heartbeat enabled this does not occur
  - Side-affect is that ‘trailing’ messages are not displayed with heartbeat enabled
  - Only one session per database needs Heartbeat

Logical: $ define JCC_CLML_HEARTBEAT_ENABLE 1

LML Tip

config

- Configuration:
  - PK mining
    - Logminer config determines PK from:
      PK defined on the table -OR-
      The Unique index with fewest columns
    - Except for tables with ALL or NO PK
      - For those DBKey mining must be used
  - Multi-source to single target
  - Automate manual db.ini file changes
  - Optional features employed:
    - VIRTUALCOLUMN
    - MAPTABLE / MAPKEY
      - Allows for defining a PK other than the base table definition, or other target table layouts different from the source table
      - Indicated by db.ini file keyword <nomaptable> (this seems counter-intuitive)
      - Then MAPTABLE configuration overrides db.ini file config for table
  - Example of mining to different target table layout with maptable:
    - TABLE~ST_TERM_DATA~22~Replicate~Nomaptable
    - MAPTABLE~ST_TERM_DATA~ST_TERM_DATA
    - MAPCOLUMN~ST_TERM_DATA~INST_ID
    - MAPCOLUMN~ST_TERM_DATA~CAMPUS_ID
    - MAPCOLUMN~ST_TERM_DATA~TECH_ID
    - MAPCOLUMN~ST_TERM_DATA~SSN
    - MAPCOLUMN~ST_TERM_DATA~RECORD_TYPE
    - MAPCOLUMN~ST_TERM_DATA~LAST_NAME
    - MAPCOLUMN~ST_TERM_DATA~FIRST_NAME
    - MAPCOLUMN~ST_TERM_DATA~BIRTH_DATE

DBK vs PK Mining

Minnesota
STATE COLLEGES
& UNIVERSITIES

- For tables without a PK, DBkey can be used
  - Some tables may be all PK or have no PK defined, for them DBKey is the only option (adding surrogate key or picking arbitrary PK could affect application)
- One drawback: data reloads or export/import operations change all DBkey values
  - Thus requiring reload of all target data dependant upon DBKey

<virtual_column>

LML Tip

Minnesota
STATE COLLEGES
& UNIVERSITIES

- VIRTUALCOLUMN
  - Useful for hard-coding a value:

    VIRTUALCOLUMN~CR_ISRS~JCCLML_CONSTANT,RC_ID~'0215'
  - Necessary for Originating-DBKey:

    VIRTUALCOLUMN~CC_SUMMARY~ORIGINATING_DBKEY

Different Target Layouts

- Creating a PK on the target different from the source
- Excluding columns from source table
- Useful for Combining data from multiple sources
- Use MAPTABLE / MAPKEY feature of Logminer-Loader to create a concatenated PK in target db composed of DBK + some other field

<nomaptable> <maptable> <mapkey>

LML Tip

LogMining Scope at MnSCU

Minnesota STATE COLLEGES & UNIVERSITIES

- Sessions with Oracle Targets:
  - APPL: 448 tables from each of 37 source dbs to 1 Oracle schema (16,576 tables)
    - Allows us to work on converting our application to use Oracle without impacting production or having to do a cold-switch
    - From this Oracle data warehouse structures are built to provide value-added reporting ‘data-marts’
  - REPL: 448 tables from each of 37 source dbs to 1 Oracle schema
    - Allows us to shift our reporting focus to Oracle while continuing to base production on Rdb
  - VAL: 141 tables from 2 source dbs to 1 Oracle schema
    - These sessions allow us to place validation data common to all institutions in one schema
  - CC_SUMMARY: 1 table from each of 37 source dbs
    - Used for warehouse support
  - MNSCUALL: 5 tables from each of 37 source dbs
    - Used for warehouse support

LogMining Scope at MnSCU

Minnesota
STATE COLLEGES
& UNIVERSITIES

- Sessions with Rdb Targets:
  - NWR: 482 tables from 1 source to 4 Rdb targets
    - In these sessions we are separating data from a combined institutional database into separate reporting databases for each institution
  - CENTRLDB: 3 tables from 1 source db to 4 target Rdb dbs
    - In this session we are taking centralized data and placing copies of it on our regional servers (allows us to maintain these 3 tables centrally without changing our application which reads the data locally)
- Total of 17,202 tables being mined by 158 separate continuous LogMiner sessions!

Session Support

- To support so many sessions we've developed a naming convention for sessions
- Includes a specific directory structure
- Built several tools to simplify the task of creating/recreating/reloading tables
- Some of the tools are based on the naming convention
- Currently our tools are all DCL, but better implementations could be made with 3GLs
- List tools? Create table script/load-data-script
- Example: load-oracle-data source-db-spec~table instance~user-pw DBK/RCIDnnnn schema tablespace~index-tablespace work-disk:[dir] debug
  - Tablespace naming convention based on oracle schema-name (user)
- Use Rdb Metadata to define Oracle tables/indexes/PKs/etc

Config mgmt

- Integrated logminer configs with prod db changes
- Volatile source environment – changes need to be grouped/scheduled and coordinated with logmining configs
  - Adding/changing/removing columns
  - Adding/removing constraints

Minnesota
STATE COLLEGES
& UNIVERSITIES

Minnesota
STATE COLLEGES
& UNIVERSITIES

Bonus: Global Buffers

Minnesota
STATE COLLEGES
& UNIVERSITIES

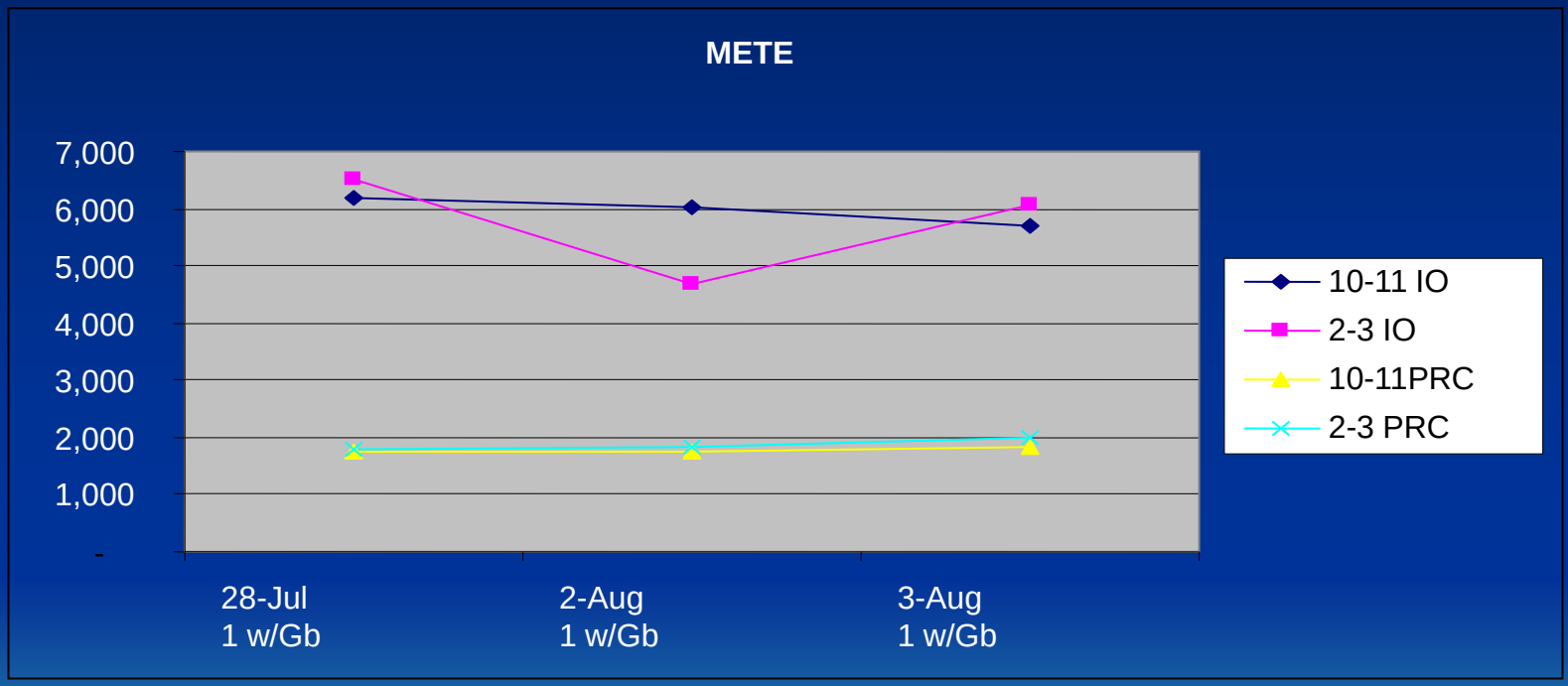
- Despite great performance gains from Row Cache over the past couple of years, we still were anticipating issues for our fall busy period
- We turned on GB on many Dbs
  - On our busiest server, we enabled it on all dbs
  - On other servers, we have about 50% implementation
- Used to run with RDM$BIND_BUFFERS of 220
- Estimated GB at max number of users@200 ea

Global Buffers

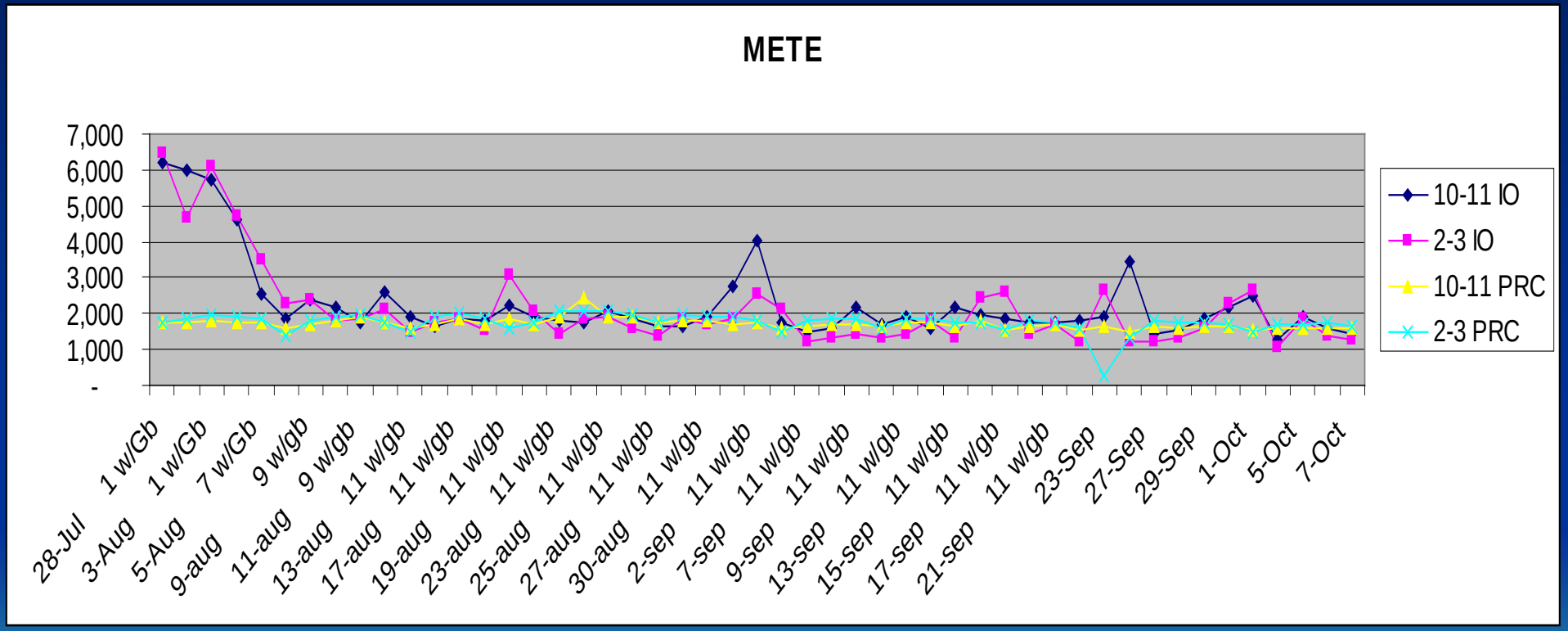
- Prior to implementing GB, our busiest server was running at a constant 6,000-7,000 IO/sec
- Other servers were running around 3,000 but had spikes to 7,000 or more
- Global buffers both lowered overall IO, as well as eliminated spikes
- Cost is in total Locks (Resources)
  - Increase LOCKIDTBL and RESHASHTBL


Oracle Rdb
Technical Forums

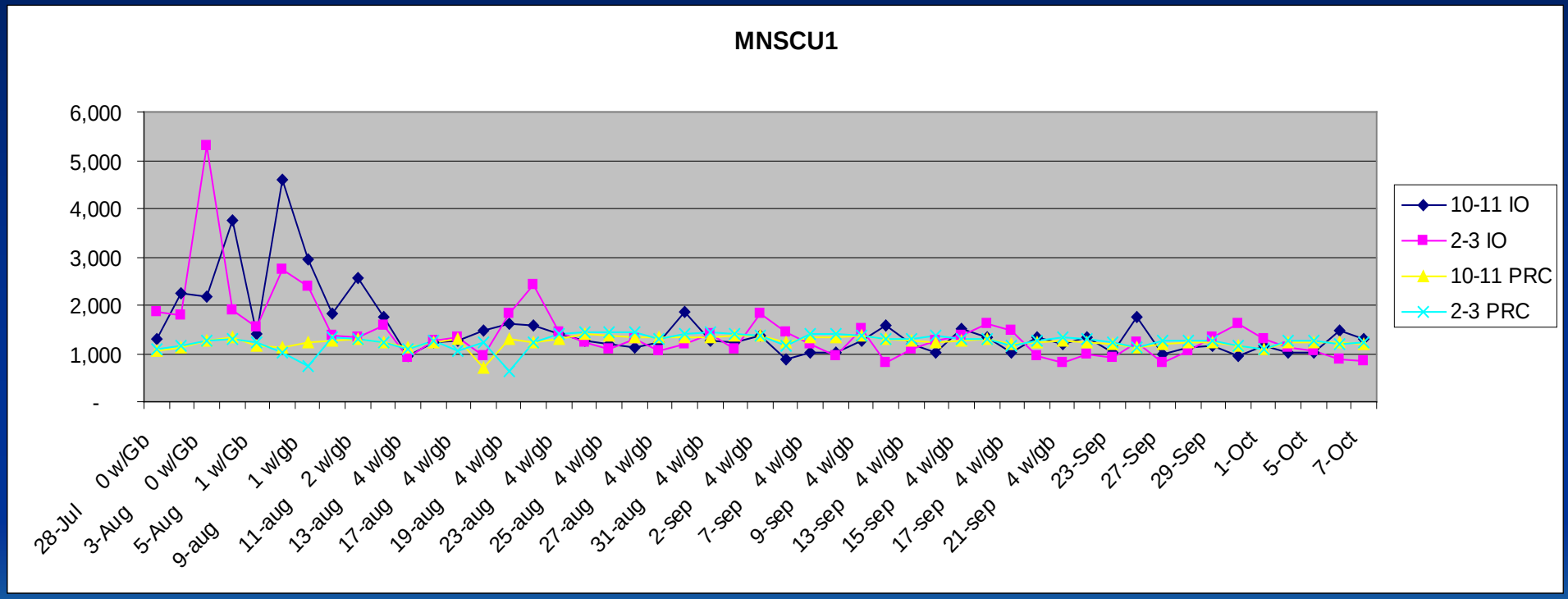
Before GB


METE
7,000
6,000
5,000
4,000
3,000
2,000
1,000
-
28-Jul
1 w/Gb
2-Aug
1 w/Gb
3-Aug
1 w/Gb
10-11 IO
2-3 IO
10-11PRC
2-3 PRC

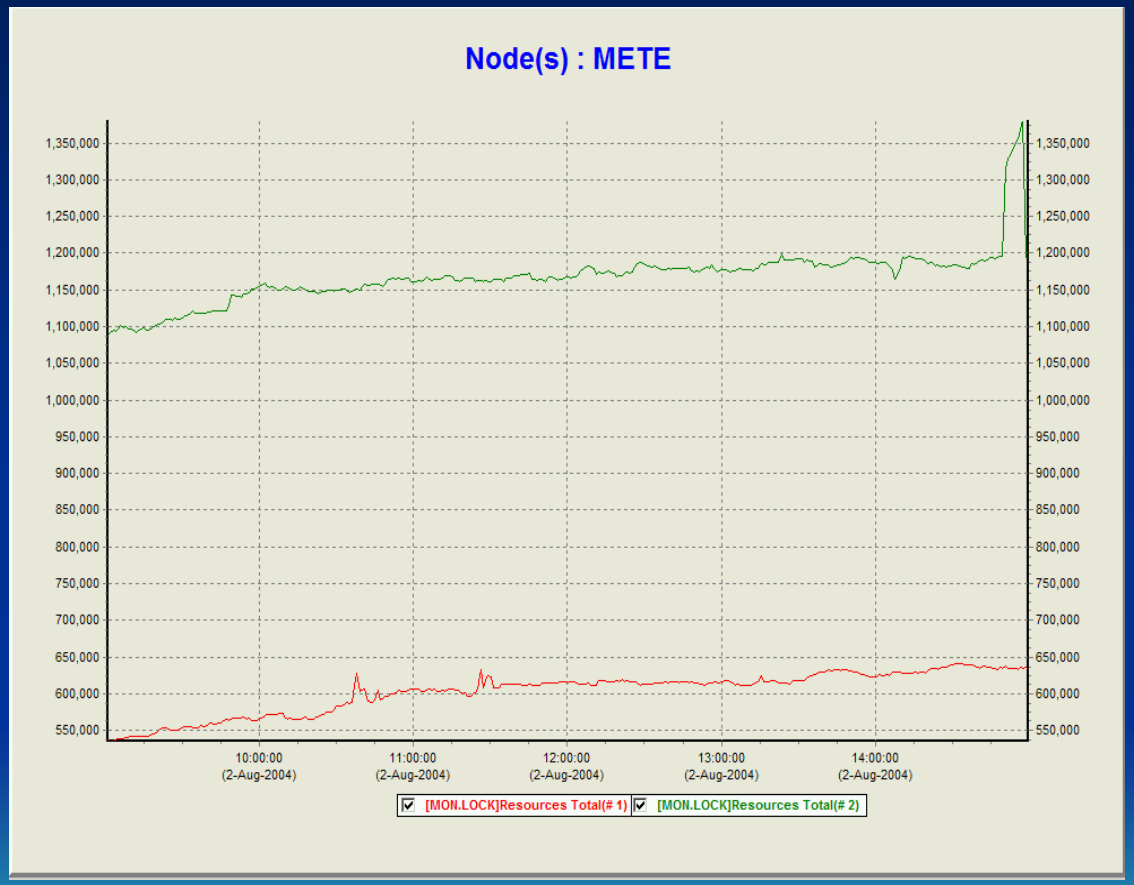
After GB


METE
7,000
6,000
5,000
4,000
3,000
2,000
1,000
-
10-11 IO
2-3 IO
10-11 PRC
2-3 PRC
28-Jul
1 w/Gb
3-Aug
1 w/Gb
5-Aug
7 w/Gb
9-aug
9 w/gb
11-aug
9 w/gb
13-aug
11 w/gb
17-aug
11 w/gb
19-aug
11 w/gb
23-aug
11 w/gb
25-aug
11 w/gb
27-aug
11 w/gb
30-aug
11 w/gb
2-sep
11 w/gb
7-sep
11 w/gb
9-sep
11 w/gb
13-sep
11 w/gb
15-sep
11 w/gb
17-sep
11 w/gb
21-sep
23-Sep
27-Sep
29-Sep
1-Oct
5-Oct
7-Oct

After Gb


MNSCU1
6,000
5,000
4,000
3,000
2,000
1,000
-
28-Jul
3-Aug
5-Aug
9-aug
11-aug
13-aug
17-aug
19-aug
23-aug
25-aug
27-aug
31-aug
2-sep
7-sep
9-sep
13-sep
15-sep
17-sep
21-sep
23-Sep
27-Sep
29-Sep
1-Oct
5-Oct
7-Oct
0 w/Gb
0 w/Gb
1 w/Gb
1 w/gb
2 w/gb
4 w/gb
10-11 IO
2-3 IO
10-11 PRC
2-3 PRC

Resources


Node(s) : METE
1,350,000
1,300,000
1,250,000
1,200,000
1,150,000
1,100,000
1,050,000
1,000,000
950,000
900,000
850,000
800,000
750,000
700,000
650,000
600,000
550,000
10:00:00
(2-Aug-2004)
11:00:00
(2-Aug-2004)
12:00:00
(2-Aug-2004)
13:00:00
(2-Aug-2004)
14:00:00
(2-Aug-2004)
[MON.LOCK]Resources Total(# 1)
[MON.LOCK]Resources Total(# 2)

CPU

- Since we are now using less IO, more CPU is available
- Before:

```
SDA> lck show lck /rep=5/int=10
23-AUG-2004 14:28:48.80 Delta sec: 10.0 Ave Spin: 24005
  Ave Req: 39875 Req/sec: 15656.9 Busy: 62.4%
23-AUG-2004 14:28:58.80 Delta sec: 10.0 Ave Spin: 19121
  Ave Req: 30166 Req/sec: 20289.7 Busy: 61.2%
```

- After:

```
24-AUG-2004 11:44:01.90 Delta sec: 10.0 Ave Spin: 12846
  Ave Req: 13439 Req/sec: 38043.3 Busy: 51.1%
24-AUG-2004 11:44:11.90 Delta sec: 10.0 Ave Spin: 16629
  Ave Req: 15514 Req/sec: 31109.1 Busy: 48.3%
```

Lock Rates

- Reducing these numbers to Lock Operations per 1% of CPU time yields:
  - Before: 299
  - After: 573

For More Information

Minnesota
STATE COLLEGES
& UNIVERSITIES

- Miles.Oustad@CSU.MNSCU.EDU
- (218) 755-4614

Minnesota
STATE COLLEGES
& UNIVERSITIES

QUESTIONS & ANSWERS