Oracle Database In-Memory

A Practical Solution

ORACLE DATABASE 12c

Sreekanth Chintala
Oracle Enterprise Architect

Dan Huls
Sr. Technical Director, AT&T WiFi

CON3087 Moscone South—307

Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

ORACLE®

Agenda

Oracle Database In-Memory Goals

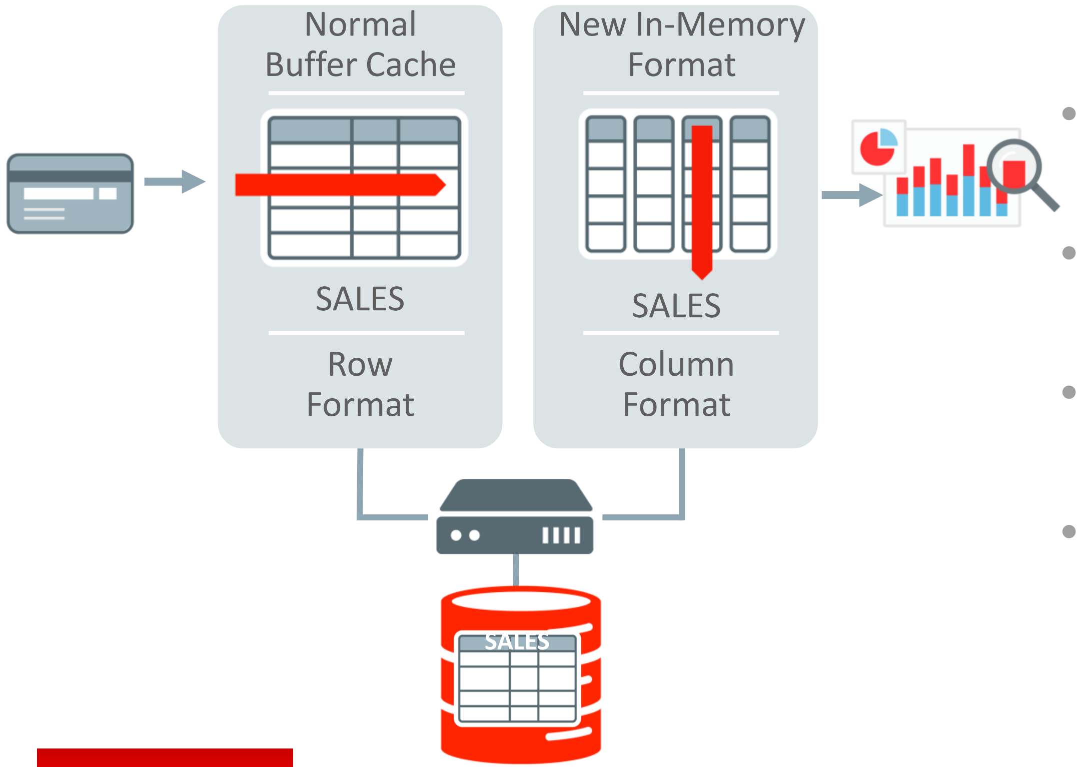
Real Time Analytics

Accelerate Mixed Workload OLTP

No Changes to Applications

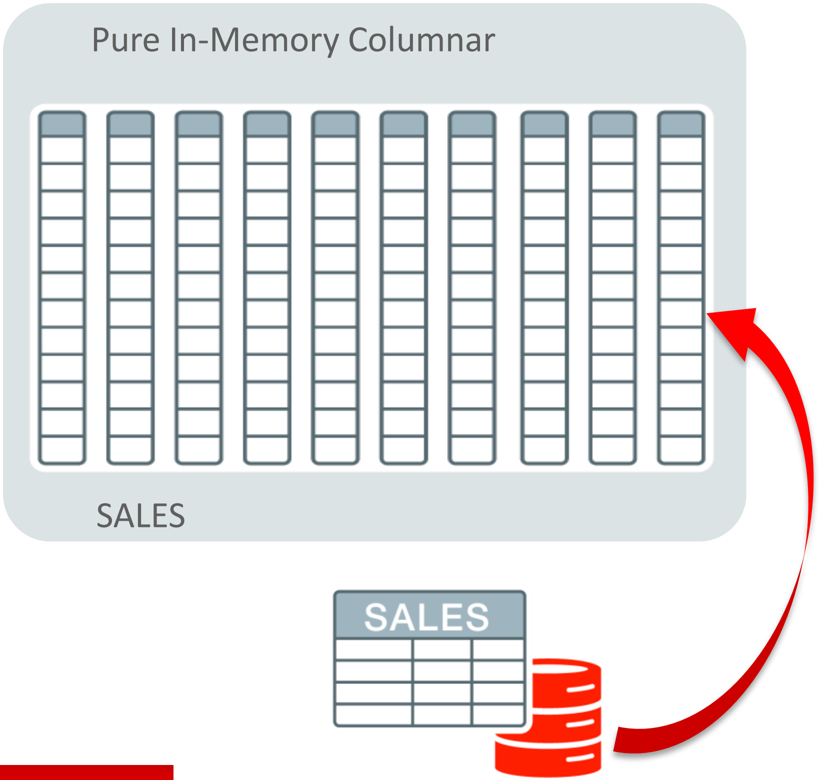
Trivial to Implement

Breakthrough: Dual Format Database



- **BOTH** row and column formats for same table
- Simultaneously active and transactionally consistent
- Analytics & reporting use new in-memory Column format
- OLTP uses proven row format

Oracle In-Memory Columnar Technology



- Pure in-memory column format
  - Not persistent, and no logging
  - Quick to change data: fast OLTP
- Enabled at table or partition
  - Only active data in-memory
- 2x to 20x compression typical
- Available on all hardware platforms

Agenda

Top 5 Reasons to Consider In Memory

#1 Supports both OLTP and Data Warehouse

#2 Maximum Availability is Built In

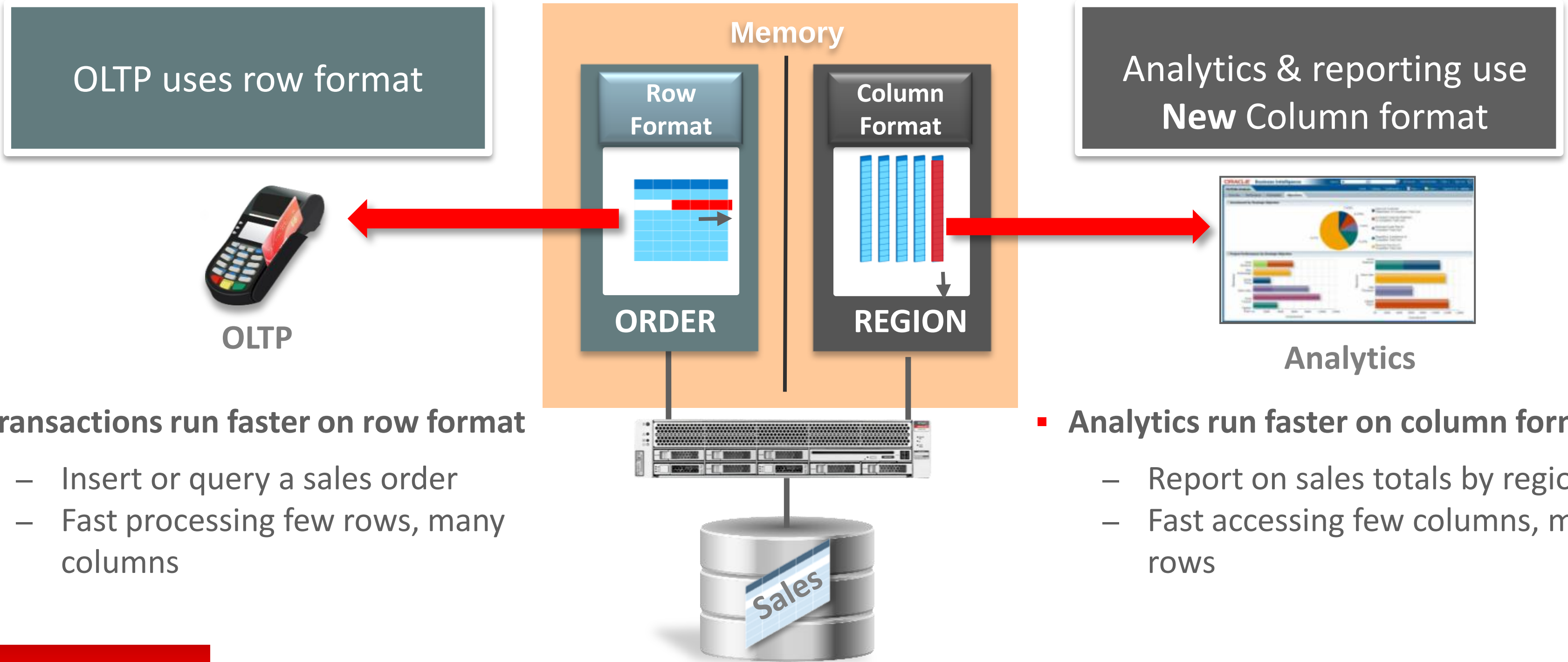
#3 Database Size Isn't Limited by Memory

#4 All Applications are Transparently Compatible

#5 No Additional Personnel/Training Needed

#1 Supports both OLTP and Data Warehouse

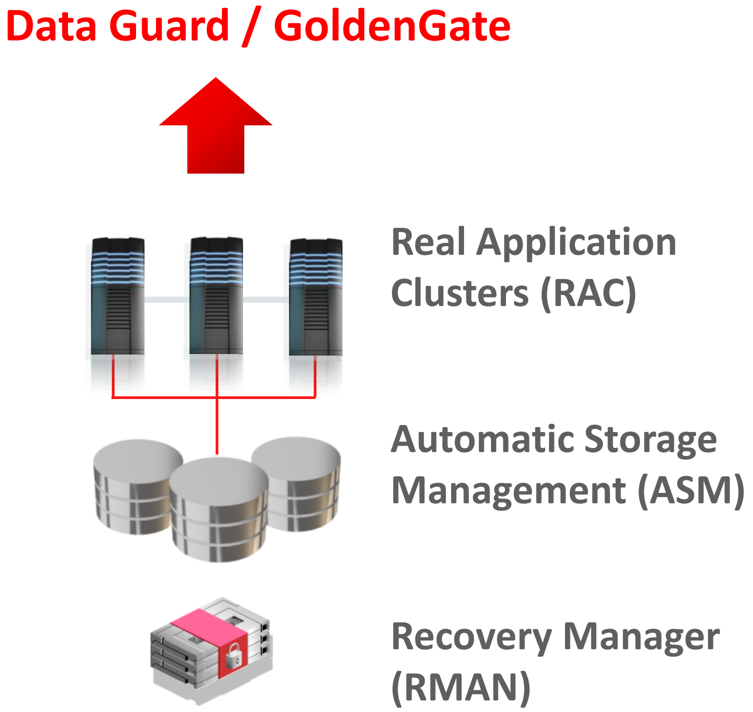
Until Now Must Choose One Format and Suffer Tradeoffs



- **Transactions run faster on row format**
  - Insert or query a sales order
  - Fast processing few rows, many columns
- **Analytics run faster on column format**
  - Report on sales totals by region
  - Fast accessing few columns, many rows

#2 Maximum Availability is Built In

- Pure In-Memory format does not change Oracle's storage format, logging, backup, recovery, etc.
- All Oracle's mature availability technologies work transparently
- Protection from all failures
  – Node, site, corruption, error, etc.

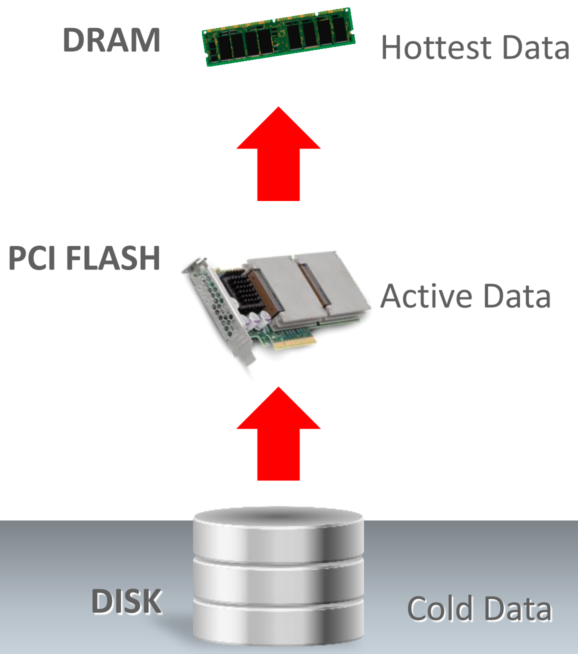

Data Guard / GoldenGate
Real Application Clusters (RAC)
Automatic Storage Management (ASM)
Recovery Manager (RMAN)

ORACLE

#3 Database Size Isn't Limited by Memory

- Specify tables, partitions or columns in-memory, not whole databases
- Data is transparently accessed across storage tiers
- Engineered Systems automatic tiered storage

Capacity	of Disk
IOs	of Flash
Speed	of DRAM



#5 No New Personnel Needed

- Use your existing Oracle teams – application developers and DBAs
- HANA requires special training and skills

3 Easy Steps to Oracle In-Memory

❶ Specify memory capacity

```
> inmemory_size = XXXX GB
```

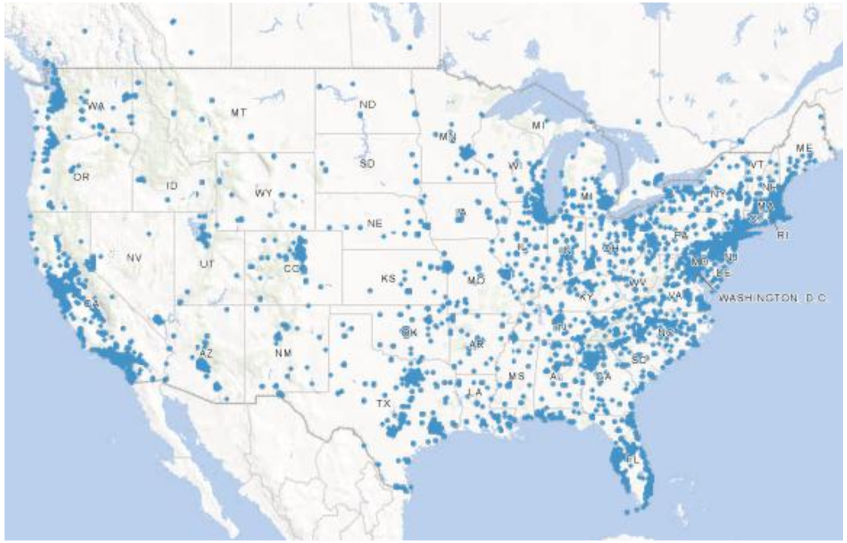
❷ Configure tables or partitions to be in memory

```
> alter table|partition … inmemory;
```

❸ Later drop analytic indexes to speed up OLTP

Agenda

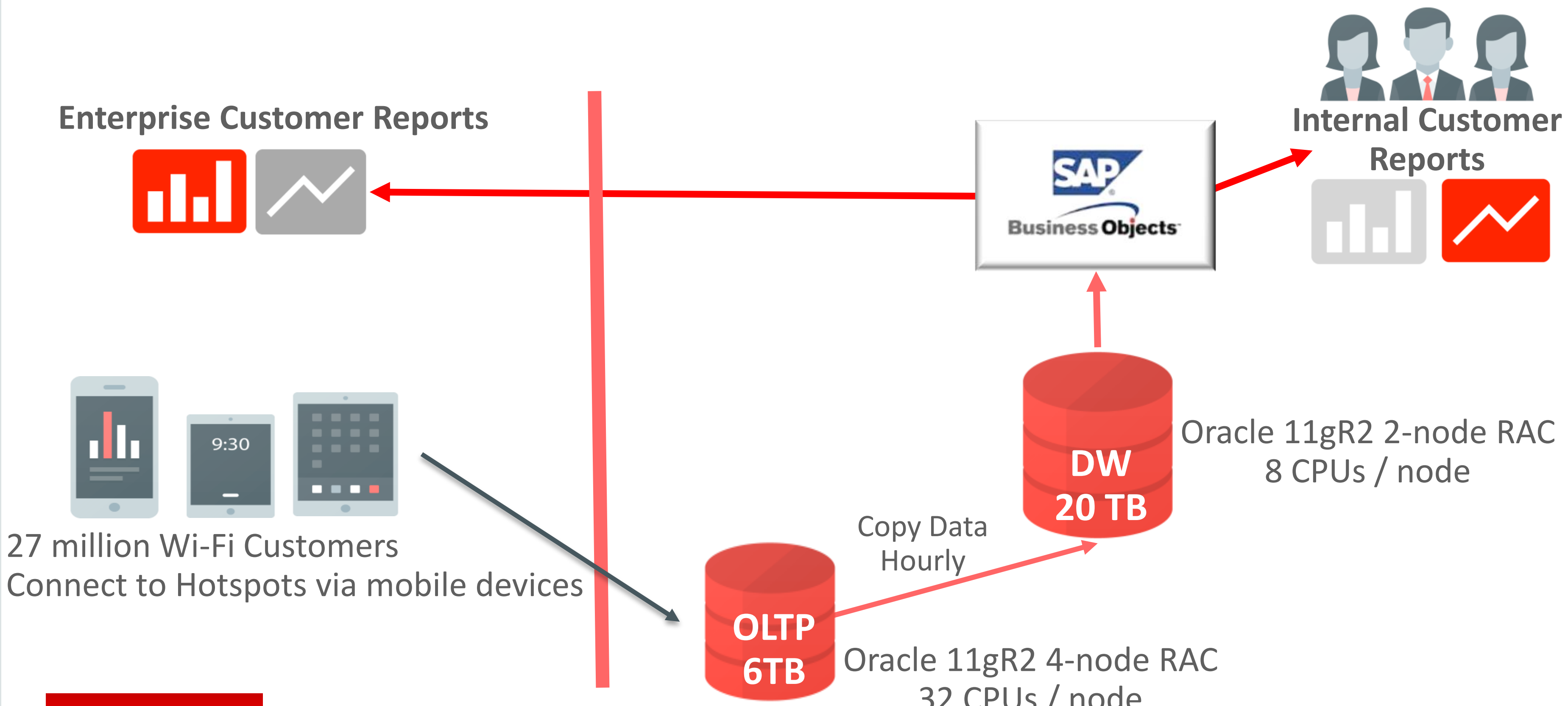
Early Adopters – DW Use Case



- Provides Wi-Fi hotspots in over 45,000 locations across the United States

- Venues include hotels, airports, sports arenas, retail stores, fast food chains, etc.

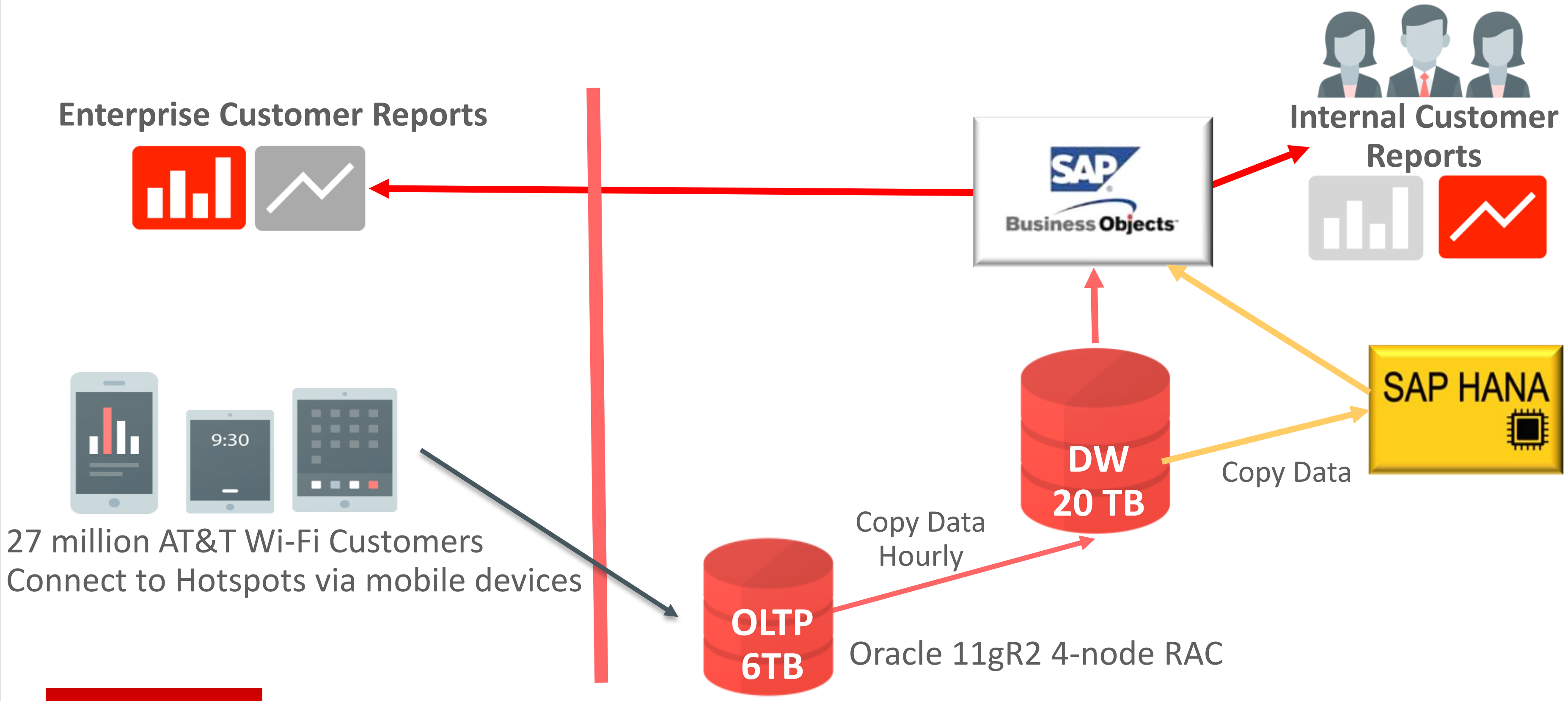
Original Configuration



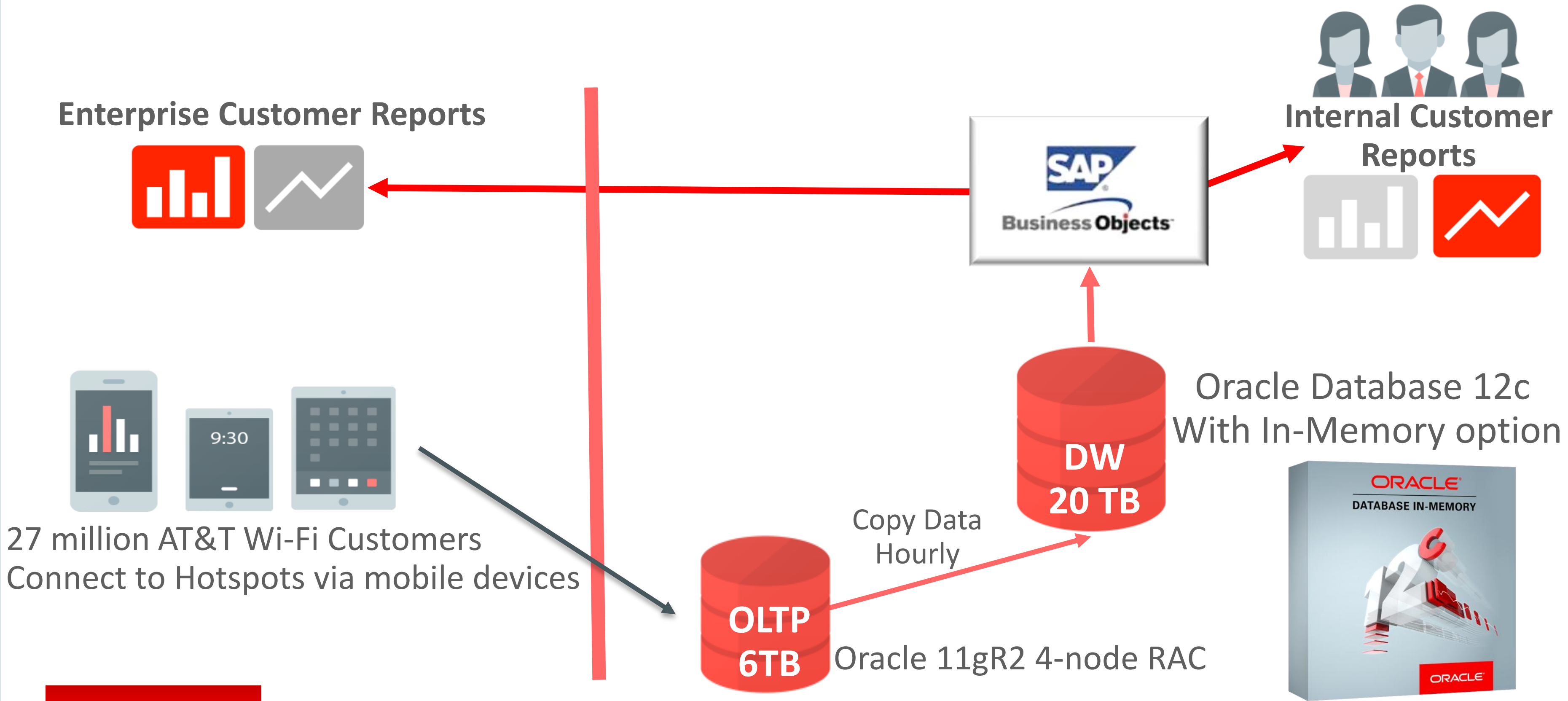
Business Concerns

- Enterprise customers access Wi-Fi reports daily through Business Objects
- Unable to meet enterprise customers' expectations
  - Missed performance SLAs of critical reports
  - 15 minute timeout resulted in customers not being able to access some reports
  - Hourly data load job impacted by increased number of indexes
  - Need faster system to handle new business reporting requirements
- Poor performance resulted in:
  - Delayed rollout of new reports and near real time reports
  - Impact on customer satisfaction, with possible impact to revenue generation capabilities

Proposed Solution 1


Enterprise Customer Reports
Internal Customer Reports
SAP Business Objects
SAP HANA
DW 20 TB
Copy Data
Copy Data Hourly
OLTP 6TB
Oracle 11gR2 4-node RAC
9:30
27 million AT&T Wi-Fi Customers Connect to Hotspots via mobile devices

Proposed Solution 2



Challenges with Proposed Solution 1

SAP HANA

- Requires new HW
- Relatively expensive S/W licenses
- Makes it very expensive to scale, limiting the number of reports they can offer
- There is very limited High Availability and no transparent failover options
- Need to hire highly specialized staff
- Another copy of data
- Extra level of effort in order to develop and manage these new processes.

Cost
Time to market
Resource Requirements

ORACLE

Cost of Acquisition

Step 1: Purchase HANA Licenses and HANA Appliances

- License cost depends on customer's ULA, but can exceed $1,000,000
- All data must be in memory – HANA cannot tier less active data to flash or disk

Step 2: Hire/Train a New Team of DBAs and Developers

- HANA consultants @ $150,000 per year, if available

Step 3: Migrate and Transform Databases into HANA Format

Step 4: Rewrite Custom Apps

Step 5: Introduce a New Operational Regimen

This is in Addition to Your Current RDBMS Organization

HANA S/W & H/W + Re-coding + Data Migration + New Team = $$$

ORACLE

Comparison with HANA

SAP HANA

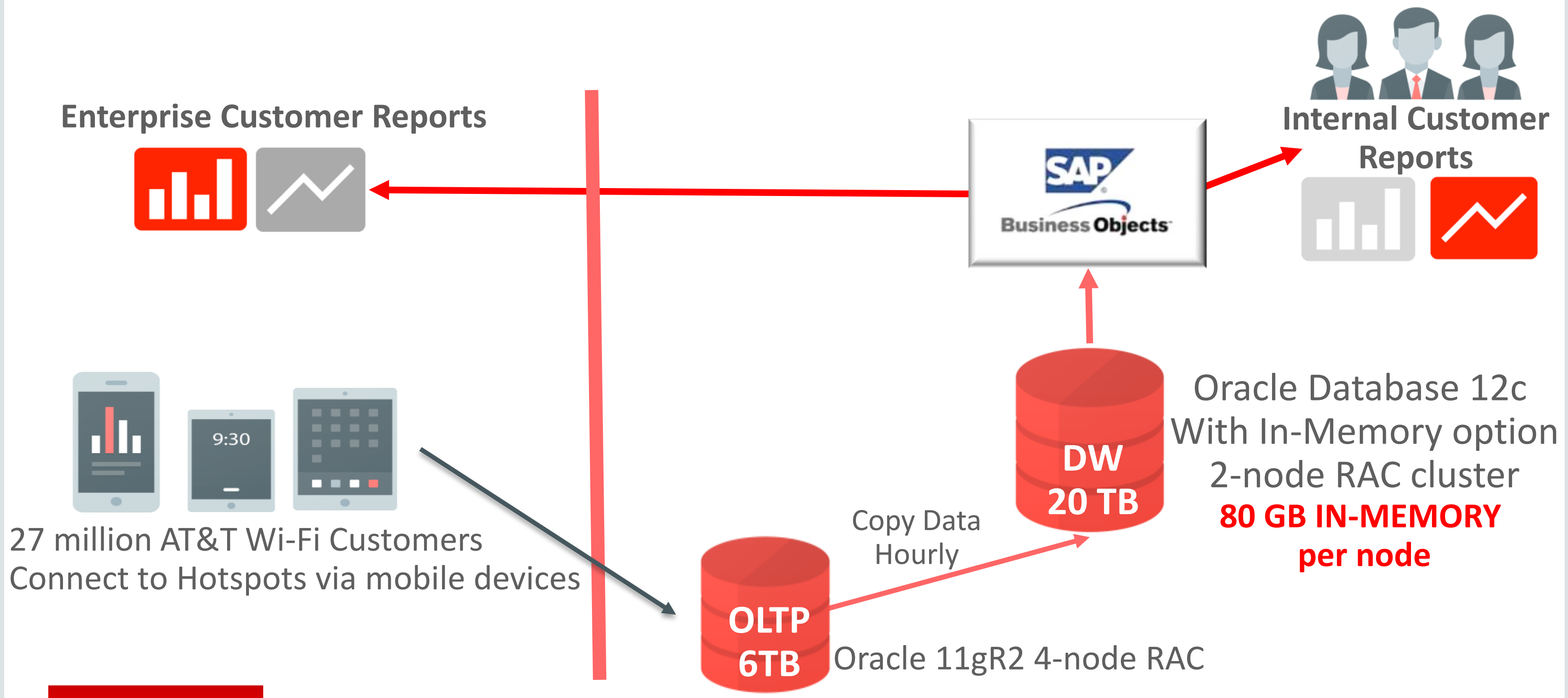
- High cost of acquisition
  - New hardware & software costs
- Application redesign
- Redesign of aggregations
- Training: Application & DBA Teams
- Time to Market: 6 months – 1 year
- Dedicated resources to maintain & manage: Min 6
- One more copy of data

- Incremental license cost & memory upgrade (fraction of money)
- No application changes
- No changes to aggregations
- Minimal training to DBAs
- 2 months to production deployment

Business Value
Licensing of SAP HANA alone was cost prohibitive compared to Oracle In-Memory

ORACLE

Final Configuration



Implementation Steps

- Database Upgrades
  - Upgrade to 12.1.0.2
  - Add additional memory
  - Enable In Memory option
- Populate Tables
  - Populate in-memory with tables, partitions and materialized views
  - Make some indexes invisible
- Oracle Enterprise Manager Upgrades
  - Upgrade to OEM 12c Release 4 to access additional features for In Memory monitoring and management

ORACLE

Details on Database Configuration

- 40 tables populated into In-Memory
  - All dimension tables
  - Rolling-window of partitions from 2 large fact tables
  - Scheduled job controls the automatic movement of partitions in & out of column store

- Data compressed using MEMCOMPRESS FOR QUERY
  - Compression ratio 76X
  - Data is predominately read-only

Benefits

- Production as of Oct 2014
- No application changes required
- Over 100X improvement on SLA for Business Objects reports
  - Reports that took 20 minutes now return in 10 seconds
- ETL processes improved by 50% due to reduction in IO

Agenda

ORACLE

Lessons Learned

- Hints
  - NO_INDEX: Force optimizer to ignore indexes (Which you don’t want hidden)
  - VECTOR_TRANSFORM: Force In Memory aggregation using KEY VECTOR & VECTOR GROUP BY operations

- Parameters

compatible = 12.1.0.2.0	inmemory_max_populate_servers = 4
inmemory_size = 85899345920	inmemory_trickle_repopulate_servers_percent = 1
optimizer_features_enable = 12.1.0.2	parallel_degree_limit = 8
parallel_degree_policy = AUTO	parallel_force_local = TRUE
sga_max_size = 111669149696	sga_target = 111669149696

Lessons Learned

- What happens when a query touches a table with some partitions in memory and some not?

```
SELECT      s.product_id, SUM(s.cost), SUM(s.price)
FROM        sales s
WHERE       product_id < 500
GROUP BY s.product_id;
```

- Only 1 partition of the sales table is in the In-Memory column store
- Table Expansion (Query Transformation Feature)

  The optimizer can transform a query into a UNION ALL statement, with some subqueries accessing indexed partitions and other subqueries accessing unindexed (In-Memory) partitions

  Useful when needing to place only specific partitions of large tables In-Memory

Lessons Learned

- Table Expansion
  - The optimizer can transform a query into a UNION ALL statement, accessing partitions on disk via an index and partitions in-memory via a full scan

```
--------------------------------------------------------------------
| Id  | Operation                        | Name         | Pstart| Pstop |
--------------------------------------------------------------------
|   0 | SELECT STATEMENT                 |              |       |       |
|   1 |  SORT AGGREGATE                  |              |       |       |
|   2 |   VIEW                           | VW_TE_2      |       |       |
|   3 |    UNION-ALL                     |              |       |       |
|   4 |     PARTITION RANGE SINGLE       |              |     1 |     1 |
|*  5 |      TABLE ACCESS INMEMORY FULL| SALES        |     1 |     1 |
|   6 |     PARTITION RANGE SINGLE       |              |     2 |     2 |
|*  7 |      INDEX RANGE SCAN            | IDX_SALES    |     2 |     2 |
--------------------------------------------------------------------
```

5 Things To Analyze Before implementing HANA

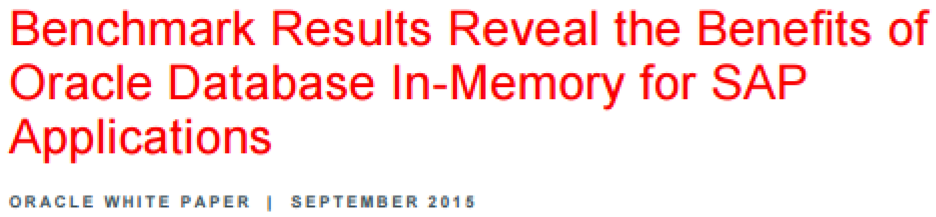
#1 Cost of Implementation

#2 Complexity

#3 Database Size

#4 Compatibility

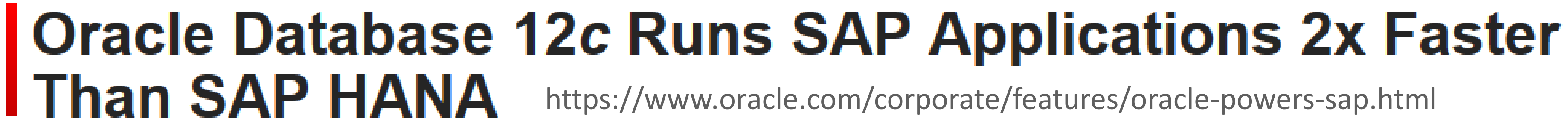
#5 Scalability & Availability



Benchmark Results Reveal the Benefits of Oracle Database In-Memory for SAP Applications

ORACLE WHITE PAPER | SEPTEMBER 2015

http://www.oracle.com/technetwork/database/in-memory/overview/benefits-of-dbim-for-sap-apps-2672504.html



Oracle Database 12*c* Runs SAP Applications 2x Faster Than SAP HANA

https://www.oracle.com/corporate/features/oracle-powers-sap.html

IDC
Analyze the Future

WHITE PAPER

Memory-Optimized Transactions and Analytics in One Platform: Achieving Business Agility with Oracle Database

Daniel Huls, Senior Technical Director for AT&T WiFi, said the company considered other in-memory columnar databases but decided to go with the Oracle Database In-Memory Option largely because the staff was already familiar with Oracle Database and the product required no changes to the applications or migration of the data.

Huls said that this initial implementation was basically effortless

Huls said that after implementing the In-Memory Option, query time dropped from an average of 400 seconds to 10 seconds.

Additional Resources

Join the Conversation

https://twitter.com/db_inmemory

https://blogs.oracle.com/in-memory/

https://www.facebook.com/OracleDatabase

http://www.oracle.com/goto/dbim.html

Comparison Between Database In-Memory & HANA

- BW-EML Benchmark Details for Database In-Memory
- Comparison of Database In-Memory with SAP HANA

Related White Papers

- Oracle Database In-Memory White Paper
- Oracle Database In-Memory Aggregation Paper
- When to use Oracle Database In-Memory
- Oracle Database In-Memory Advisor

Related Videos

- In-Memory YouTube Channel
- Database Industry Experts Discuss Oracle Database In-Memory (11:10)
- Software on Silicon

Any Additional Questions

- Oracle Database In-Memory Blog
- My email: **sreekanth.s.chintala@oracle.com**

ORACLE

Program Agenda

Testing Methodology

Step 1: Create a Baseline

- Run the workload without the IM column store
  - Set INMEMORY_SIZE = 0
  - Enable capture of SQL Plan Baselines

    ```
    OPTIMIZER_CAPTURE_SQL_PLAN_BASELINES = true
    ```
  - Capture the workload in a 12c environment
- This will be the baseline to judge how much improvement Database In-Memory offers
  - Verify the performance against the existing system if it exists
  - **Do not continue until satisfied with the performance**
    - Database In-Memory cannot fix underlying performance issues

Testing Methodology

Step 2: Populate Tables In-Memory

- Allocate the IM column store
  - Set the INMEMORY_SIZE parameter and recycle the database
- Populate identified tables into the IM column store
- Verify that all objects have been populated
  - BYTES_NOT_POPULATED field of the v$im_segments view

```
SQL> select segment_name, populate_status, inmemory_priority, inmemory_size,
bytes_not_populated from v$im_segments;

SEGMENT_NAME POPULATE_STATUS INMEM_PRIORITY INME_SIZE    BYTES_NOT_POPULATED
------------ --------------- -------------- ------------ -------------------
ACCOUNTS     STARTED         HIGH                 196606     2434886912
SALES        COMPLETED       CRITICAL          135790592     0
```

Testing Methodology

Step 3: Run workload with In-Memory enabled

- Run the workload
- Optional: Identify any new, unaccepted baselines

```
SELECT sql_handle, plan_name, enabled, accepted,
   parsing_schema_name schema,sql_text
FROM dba_sql_plan_baselines ORDER BY 1,2;
```

- Plan changes should reflect the use of In-Memory

Testing Methodology

Step 4: Evolve SQL Plan Baselines

- Evolve SQL Plan Baselines
  - This will allow the optimizer to use plans that perform better than the current baseline plans
  - This is a key step!
  - Prevents regressions and allows the use of the best plans
- Use the evolve task functions of the dbms_spm package
  - CREATE_EVOLVE_TASK
  - EXECUTE_EVOLVE_TASK
  - REPORT_EVOLVE_TASK
  - IMPLEMENT_EVOLVE_TASK

Testing Methodology

Step 5: Final workload execution

- Run the workload one more time
- Compare the elapsed times to the baseline performance
- These final results should provide the best performance

Application Architecture

A word about indexes

- Don't drop all the indexes, you may still need them
- Definitely keep primary key, foreign key and unique indexes, they enforce referential integrity
  - Insure that there are corresponding constraint definitions
- You may also need other existing indexes for OLTP type access
- Target reporting indexes that won't be needed with In-Memory
  - Start by making these indexes invisible
  - After running the workload and verifying that the indexes are not needed, then drop
  - Making indexes invisible is much easier than dropping them and having to re-create

Performance History

Make sure AWR is available

- AWR is valuable to measure system level workload during testing
- AWR is valuable to see how the database is setup
  - Initialization parameters
  - Memory allocation
  - Resource utilization
  - And don't forget the Advisors section
- Consider increasing the default AWR retention (7 days)

Proper Preparation

Database Parameters

- Start with the defaults!
- Specifically look to see if any of these are set (and unset them):
  - OPTIMIZER_DYNAMIC_SAMPLING
  - OPITIMIZER_INDEX_CACHING
  - OPTIMIZER_INDEX_COST_ADJ
  - OPTIMIZER_FEATURES_ENABLE
- Important In-Memory parameters:
  - INMEMORY_SIZE – Controls how large the IM Column store will be
  - PARALLEL_DEGREE_POLICY – Enable AutoDOP, required for RAC environments

Proper Preparation

Other In-Memory Database Parameters

- Important Optimizer parameters that will affect In-Memory performance:
  - OPTIMIZER_USE_SQL_PLAN_BASELINES
  - OPITIMIZER_CAPTURE_SQL_PLAN_BASELINES
  - OPTIMIZER_INMEMORY_AWARE
  - OPTIMIZER_ADAPTIVE_FEATURES
  - OPTIMIZER_DYNAMIC_SAMPLING

ORACLE

More Information

- White paper on SQL Plan Management
  - http://www.oracle.com/technetwork/database/bi-datawarehousing/twp-sql-plan-mgmt-12c-1963237.pdf
- White paper on what to expect from the Optimizer
  - http://www.oracle.com/technetwork/database/bi-datawarehousing/twp-optimizer-with-oracledb-12c-1963236.pdf
- Optimizer blog at
  - http://blogs.oracle.com/optimizer/
- Database In-Memory blog at
  - http://blogs.oracle.com/In-Memory/