

ORACLE®



Oracle 11g on OpenVMS and Rdb 7.3 Feature Highlights

Wolfgang Kobarg-Sachsse, Oracle Rdb Support

Based on presentations from **Gary Huffman**,
Oracle OpenVMS Engineering Group, and **Ian
Smith**, Oracle Rdb Engineering.



ORACLE®



Oracle on OpenVMS Update

An overview

Gary Huffman

Senior Development Manager

Oracle OpenVMS **Engineering** Group



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.

Program Agenda

Review of Current Products

Review Oracle Database 11gR2

Features Available in Oracle Database 11gR2

10g Support Dates

- **Extended Support ended 31-Jul-2013**
- **Terminal PSU for both Alpha and Integrity**
 - 10.2.0.5.12 July 2013
- **Limited Extended Support Added For All Platforms**
 - From Aug 2013 through July 2015 Limited Extended Support is available, Sev 1 fixes only (no Patch Set Update (PSU) or Security Patch Update (SPU) will be produced)
- **Additional Extended Support periods will be available for current Extended Support fees**

Limited Extended Support for OpenVMS Integrity

- **Oracle is waiving all Extended Support fees until July 31, 2015** for customers running Oracle Database 10g Release 2 on OpenVMS Integrity. **After July 31, 2015 a Limited Extended Support service will be available through July 31, 2017** to those customers at then-current charges for Extended Support, with the main limitation being that no Patch Set Updates or Critical Patch Updates will be produced. Please see the [Oracle Software Technical Support Policies](#) document (pages 6-7) for complete details.

Oracle Database 10gR2 Support Dates

Integrity Platforms	
Platform	Additional Extended Support: S1 Fixes Only
HP OpenVMS on Integrity	August 2015 – July 2017
HP OpenVMS on Alpha	August 2013 – July 2015
HP-UX	August 2013 – Dec 2015
Linux Integrity	August 2013 – Dec 2015
Windows Integrity	August 2013 – Dec 2015

Certification Documentation

- My Oracle Support (MOS) Document
 - <https://support.oracle.com>
 - **Doc ID 742060.1**
 - **Note**
 - Please use this document as the current support matrix
 - It is updated frequently
 - This document does not limit Oracle in what will be delivered
 - Primary function is to document high volume releases

Review of Current Products for the Oracle Database on OpenVMS

- 10.2.0.5.0 Released 31-Oct-2012
 - 10.2.0.5.12 PSU available for Alpha & Integrity
- Grid Control Agent
 - 10.2.0.2 Agent for Integrity and Alpha shipping
 - Patch kit 6 is available

Review of Current Products for Oracle Database on OpenVMS

- 10.2.0.5.0 is a Patch Set
 - Suggest install in a new Oracle home
 - Install initial 10.2.0.2.0 release*
 - For Alpha no need to apply any patches
 - For Integrity you should apply the patch 5840282 before applying this patch set
 - No need to install 10.2.0.4.0 before installing 10.2.0.5.0
 - Apply the 10.2.0.5.0 Patch Set
- OpenVMS 8.4 is the minimum OS version for both Alpha and Integrity
- Can be downloaded from My Oracle Support (MOS) – Patch # 8202632

* MOS Note 1071023.1 documents how to request a DVD or access to a downloadable image

Additional Supported Features

- Mixed Architecture RAC – (MAR)
 - Released supporting 10.2.0.4.0
 - 10.2.0.5.0 is supported
 - Both the Alpha and Integrity systems must run the same version of Oracle – matching patches as far as possible
- Data Vault
 - Released with 10.2.0.4.0
 - Updated with 10.2.0.5.0

Oracle MySupport OpenVMS Information

- **OpenVMS: Master Note for Oracle 10.2 on hp OpenVMS**
 - [\[ID 726914.1\]](#)
- **OpenVMS: Oracle Release 10gR2 Mixed Architecture (Multi-Home) CRS/RAC Installation on hp Integrity and Alpha**
 - [\[ID 785970.1\]](#)
- **OpenVMS: Getting started with the 10.2 Grid Control Agent on OpenVMS**
 - [\[ID 739445.1\]](#)
- **Installing AV Agent 10.2.3.2 on HP OpenVMS**
 - [\[ID 1111278.1\]](#)
 - Master Note For Oracle Audit Vault
 - [\[ID 1199033.1\]](#)

Review of Oracle Database 11gR2

Oracle Database 11gR2 on OpenVMS

- Will be a full release - Server/Client Kit
 - Web images
 - No instant client
- Will provide new Oracle features to OpenVMS Integrity
 - Note: An incremental set of features will be available for OpenVMS
 - Not all Oracle features will be available on OpenVMS

OpenVMS 11g Porting Environment (VPE)

- Integrity OpenVMS only planned
- 11gR2 only available for OpenVMS 8.4
 - We use OpenVMS sym-links to create the release
- We are doing labels and builds every two weeks as a default
 - This is including completing core testing
- 40% larger than 10.2.0.5.0

Status October 2014

- November 2012 we resumed 11.2.0.2.0
 - Merged all changes from 10.2.0.5.0 release
- Skipped 11.2.0.3.0
- Base Released 11.2.0.4.0 in August 2013
 - We merged to 11.2.0.4.0 label
 - Starting Q3CY2013
 - Three months of work
 - CY2014
 - Operating System Code (OSD) development continues
 - Short regressions in process
 - Long regressions started

Status October 2014

- 11.2.0.4.0 is final Patch Set for the 11.2 code line
 - PSU planned available until at least Jan 2018 See MOS note 742060.1
 - Plan is to release the current PSU approximately the same time as our 11gR2 release

Status October 2014

- Approximately 1.1 Million entries in the label
 - 216110 source files
 - .c,.h,.cpp,.hpp,.java .hxx .pl .pm
 - 304705 support files and test files
 - Makefile,.mk,.tsc,.log

Status October 2014

- End of October
 - 5828 files have been branched for OpenVMS
 - 2398 source files
 - 1089 support files
 - 2341 various other
 - .txt .com ,sql ,sqlj .xml {installer files}

Porting Issues Encountered

- Compile time issues
 - Uninitialized variables
 - Type-casting miss matches
 - Implicitly declared routines
 - Compiler directives
 - “_start” and “_end”
 - Unsupported #pragma

Porting Issues Encountered

- Java Porting (yes , there are porting issues with Java)
 - System.getenv() on Open VMS only returns specified variable
 - Linux has option to return all of the environment variables
 - Difference in symbol interpretation
 - i.e. ; : # !
 - Creation of external scripts
 - DCL vs Unix shell
 - Device access
 - Unix references RAW Devices
 - Default Block Size (512 vs 1024)

Porting Issues Encountered

- Build Issue With 11.2.0.4 oracle.exe image too large to link static
 - Implemented shared libraries for the oracle.exe image
 - libcorenls11.so
 - libskgxn2.so
 - libskgxp11.so
 - liborashr11.so

Porting Issues Encountered

- Environment issues
 - Base does component builds
 - OpenVMS does full bundle builds
 - Base does all builds within source control system (ADE)
 - OpenVMS does SRCHOME builds
 - Base promotes Derived Objects
 - OpenVMS utilizes Repositories
 - These differences lead to process contentions that OpenVMS encounters

Porting Issues Encountered

- Environment Issues (cont)
 - Affects build and QA process flow
 - VMS Pool and SGA sizes are larger than Linux defaults
 - Test environment relies on Unix style path's to access
 - OpenVMS has to change scripts to enable
 - "T_WORK", "T_COM", "T_SOURCE"
 - Frequently encounter path too long on OpenVMS

Porting Issues Encountered

- Implementation Issues for Oracle run-time
 - Perl is supported by PLSQL External Procedures
 - Involved adding Perl support within the native framework of the QA and install environment
 - Filename conversion
 - Often encounter strings like this
 - <disk>:[oracle_home.]/trc/trace.dmp

Porting Issues Encountered

- Implementation Issues for Oracle run-time (cont)
 - Mixed case file-name support
 - Implemented changes to the Server Background process
 - \$ define/nolog DECC\$ARGV_PARSE_STYLE ENABLE ! preserve argv case
 - \$ define/nolog DECC\$EFS_CASE_PRESERVE ENABLE ! enable ODS-5 names
 - \$ define/nolog DECC\$EFS_CASE_SPECIAL ENABLE ! enable ODS-5 names
 - \$ define/nolog DECC\$EFS_CHARSET ENABLE ! enable ODS-5 names

Features available for Oracle Database 11gR2

Overview of 11gR2

- In Memory Database Cache
- Unstructured Data Types
- Real Applications Testing

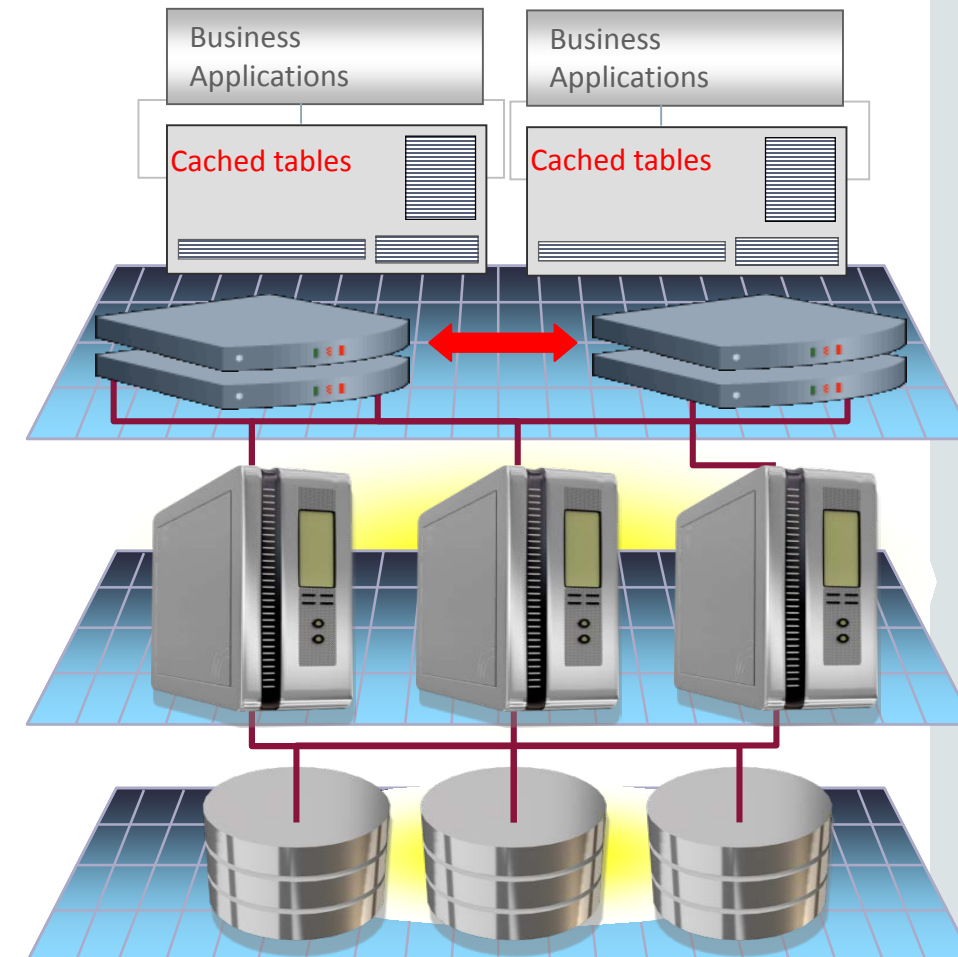
Performance

Improve performance by at least 10x

Oracle In Memory Database Cache

Offload Workload to the Middle Tier

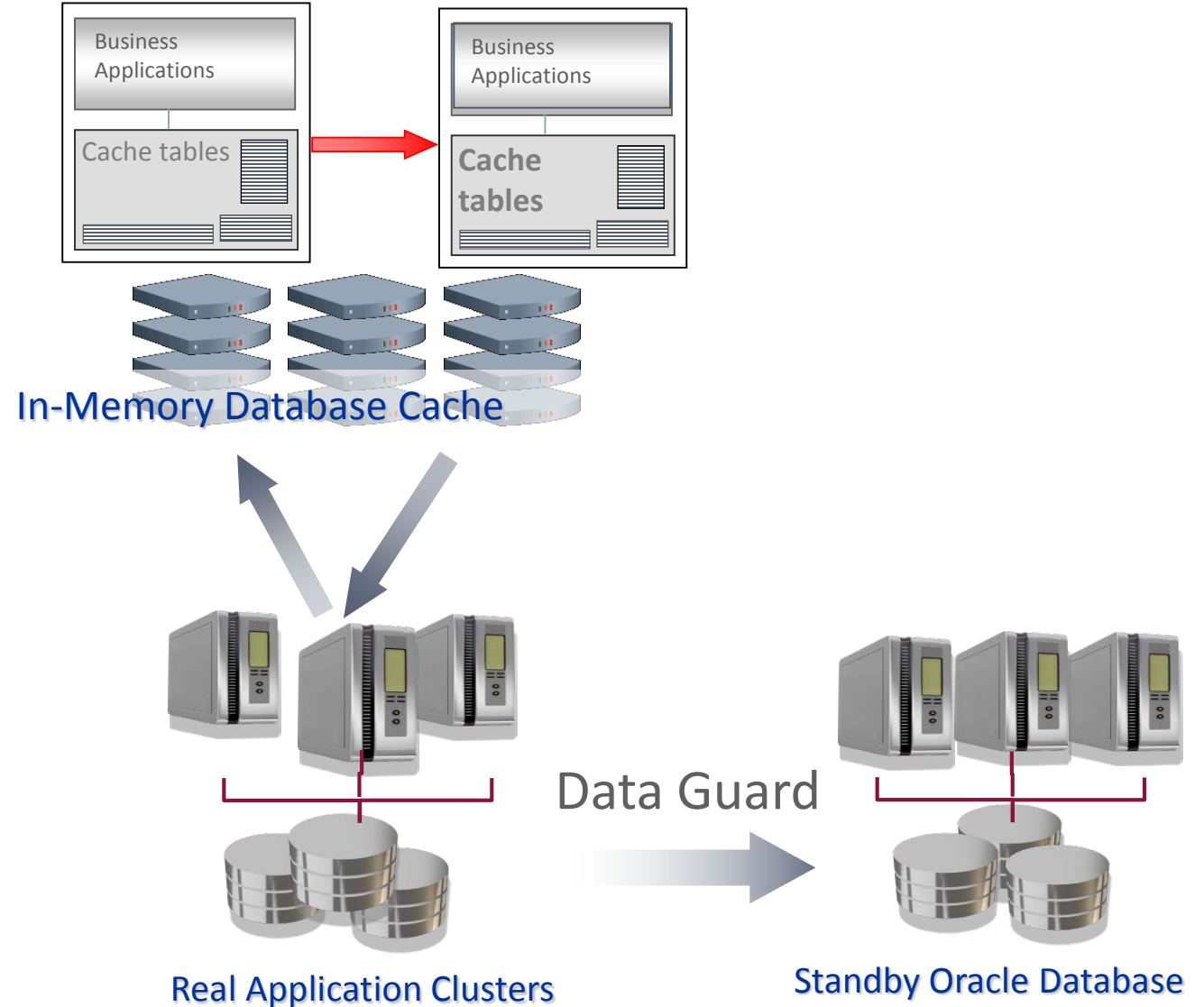
- Data cached in application memory
 - Database tables
 - Subsets of rows & columns
- Standard SQL interface
 - Synchronized with Oracle Database
- Utilizes middle tier resources
- Fast, consistent response times
 - High transaction throughput



Oracle In Memory Database Cache

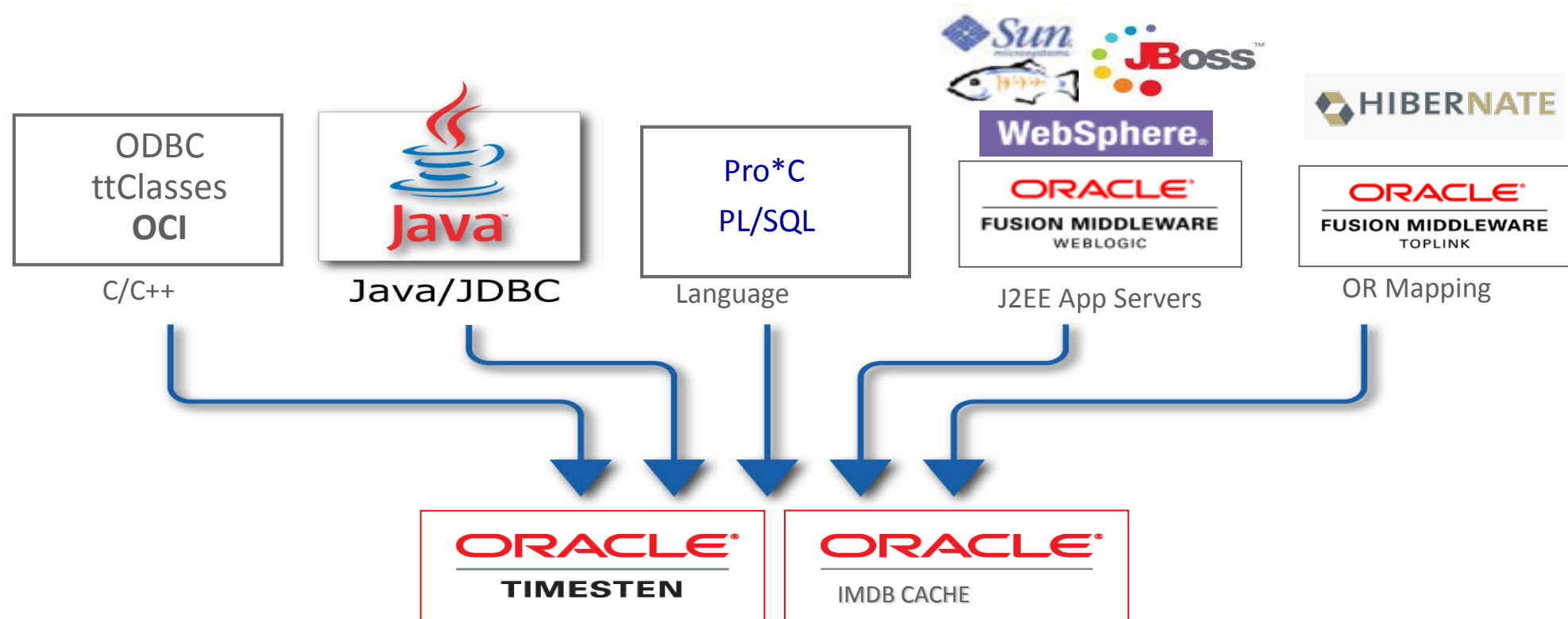
Cross-tier High Availability

- Automatic Client Connection Failover
- Integration with Oracle Clusterware
 - Manages TimesTen / IMDB Cache processes
- Integration with Oracle RAC
 - Automatic recovery from Oracle Database RAC node failures using TAF and FAN
- Support Data Guard synchronous physical standby
 - Failover
 - Switchover
 - Rolling upgrade



Oracle In Memory Database Cache

Application Development

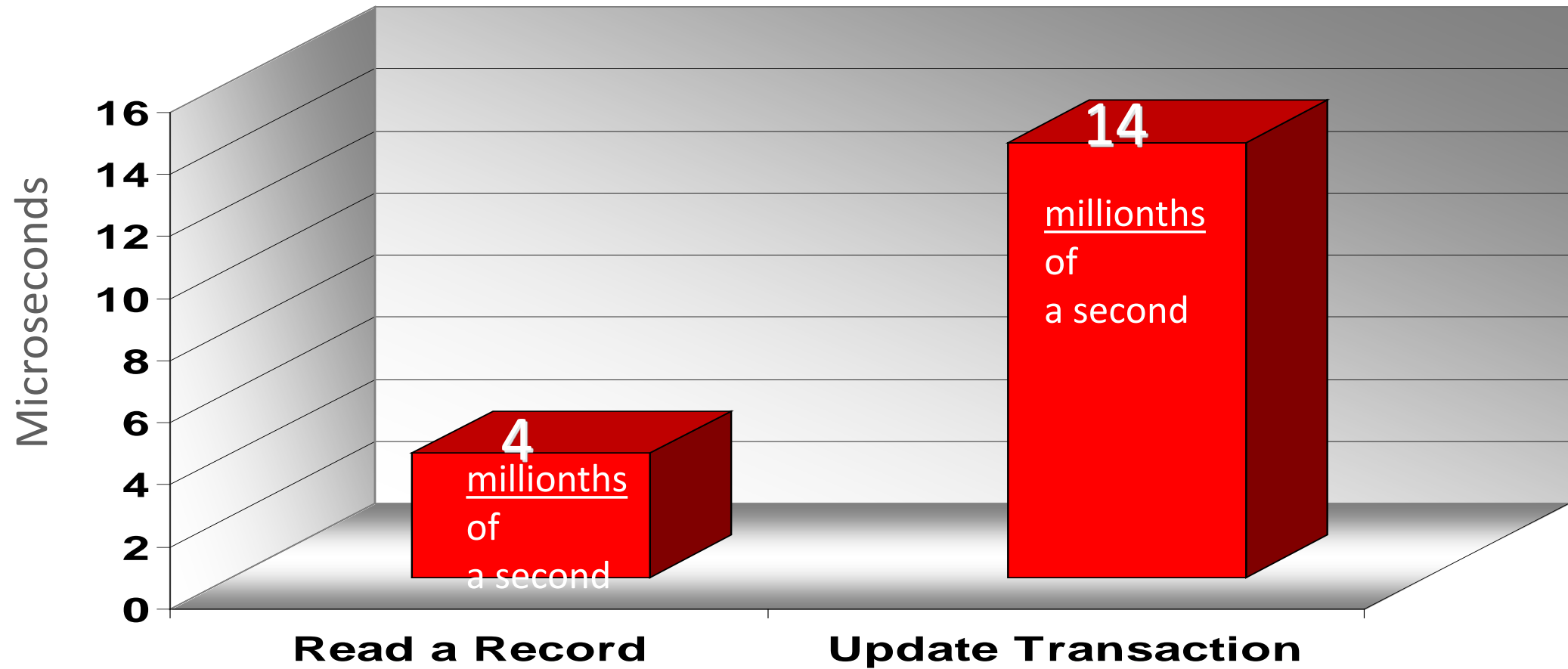


New in 11g release

- PL/SQL, Oracle Call Interface (OCI) and Pro*C Support

Oracle In Memory Database Cache

Lightning Fast Response Time

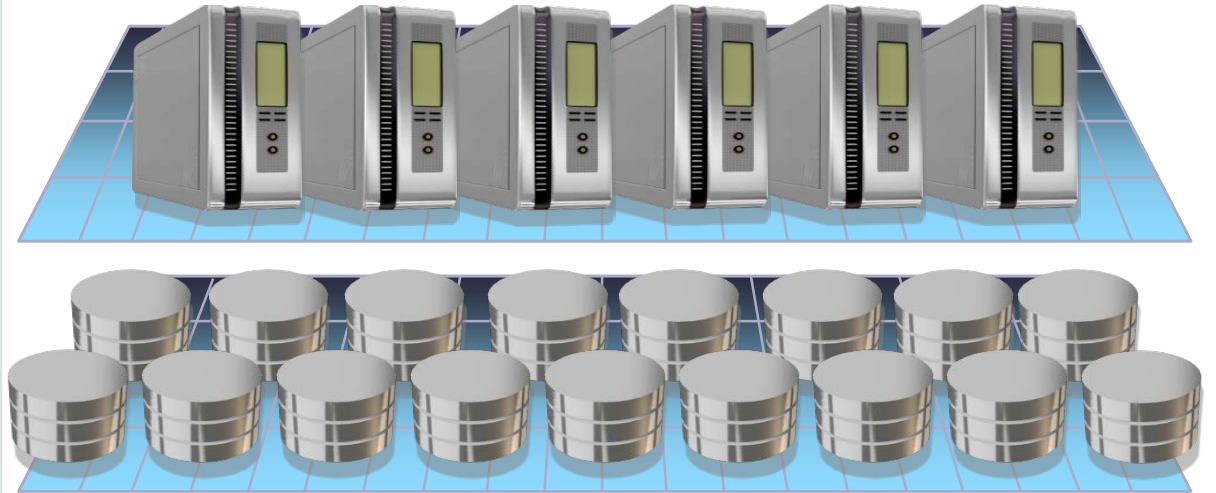


Oracle TimesTen In-Memory Database 11g - Intel Xeon 3.0 Ghz 64-bit Oracle Enterprise Linux

Oracle Database 11g Release 2

In-Memory Parallel Execution

- Data warehouse environments can have large amounts of memory that is not always used
- An algorithm places fragments of an object (partitions) in memory on different nodes
- Compression gets more data in memory
- Parallel servers (aka PQ Slaves) are then executed on the corresponding nodes

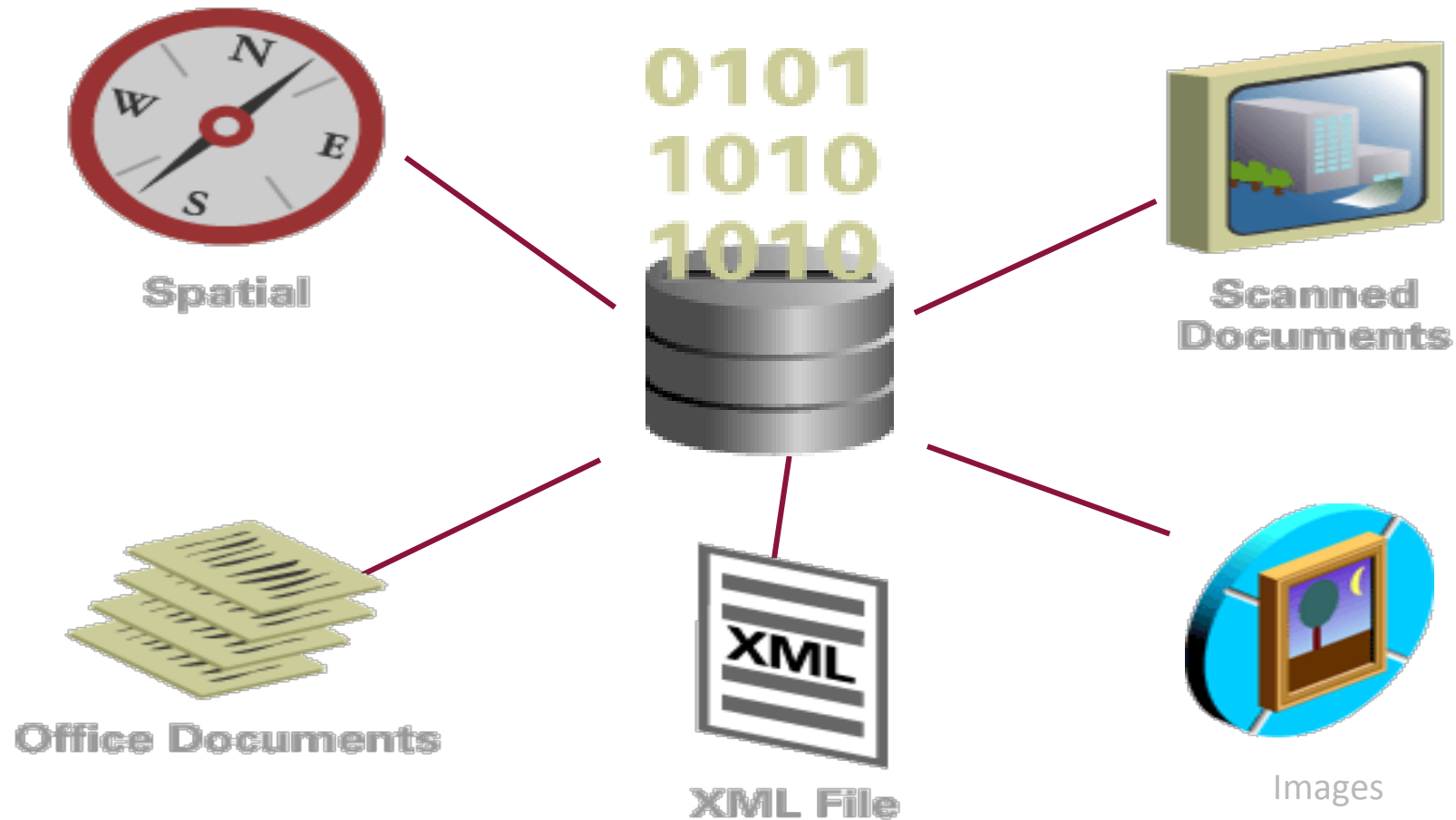


Real Application Clusters

Integrating Unstructured Data

Better business insight into all data types

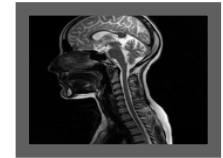
Integrating Unstructured Data



New in Oracle Database 11g



RFID



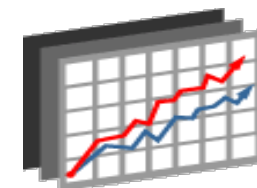
DICOM



3D



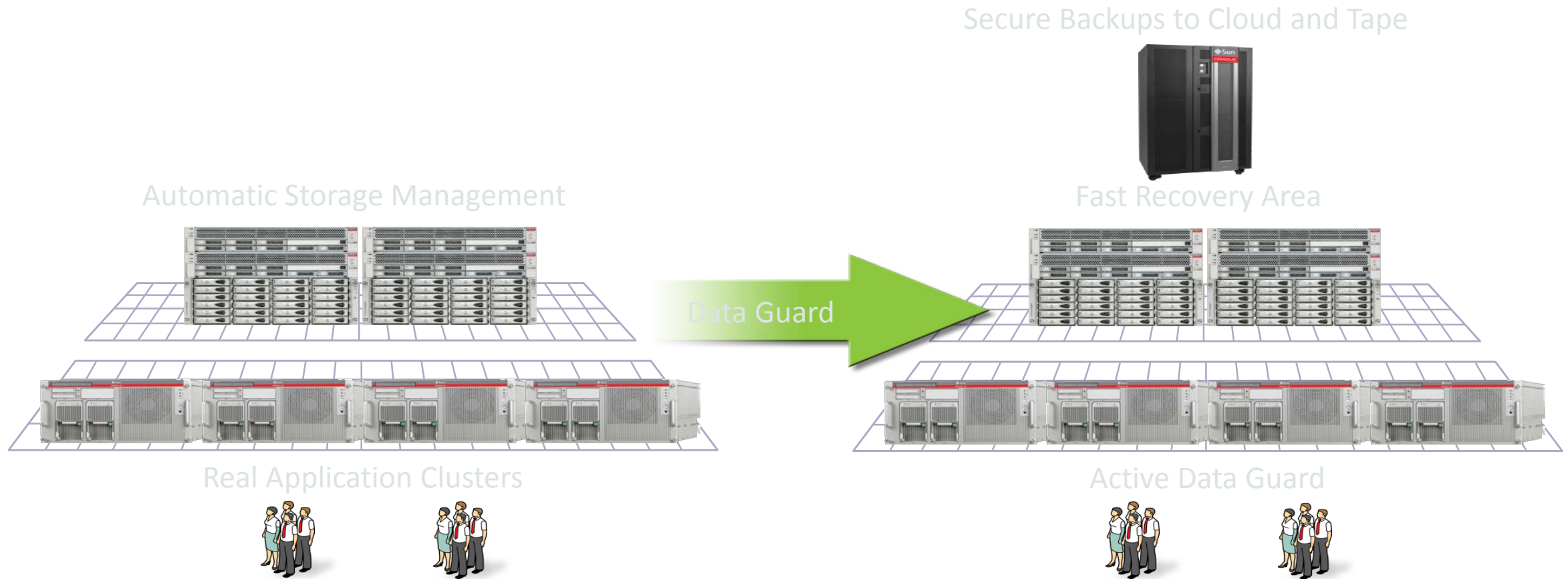
Binary XML



SecureFiles

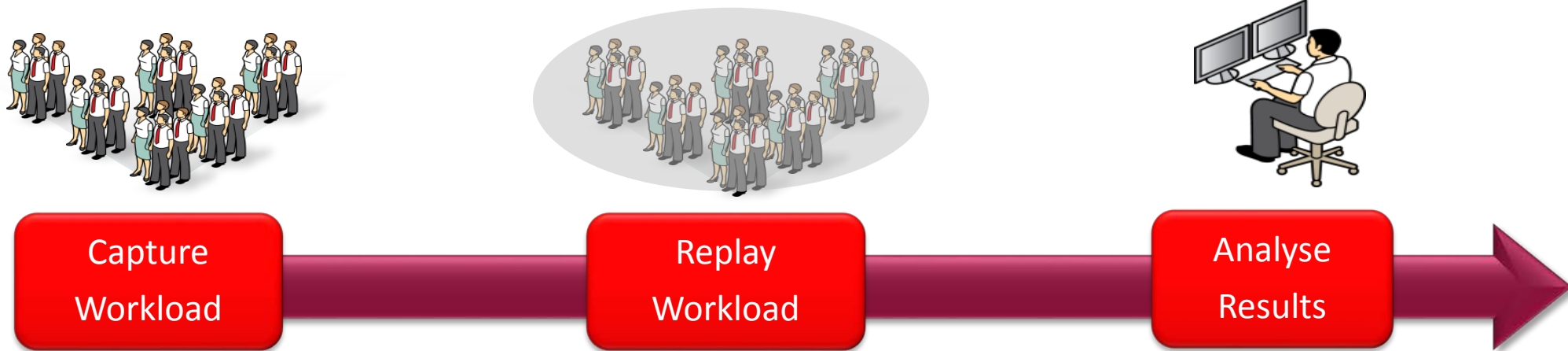
Oracle Maximum Availability Architecture

No idle redundancy



Real Application Testing

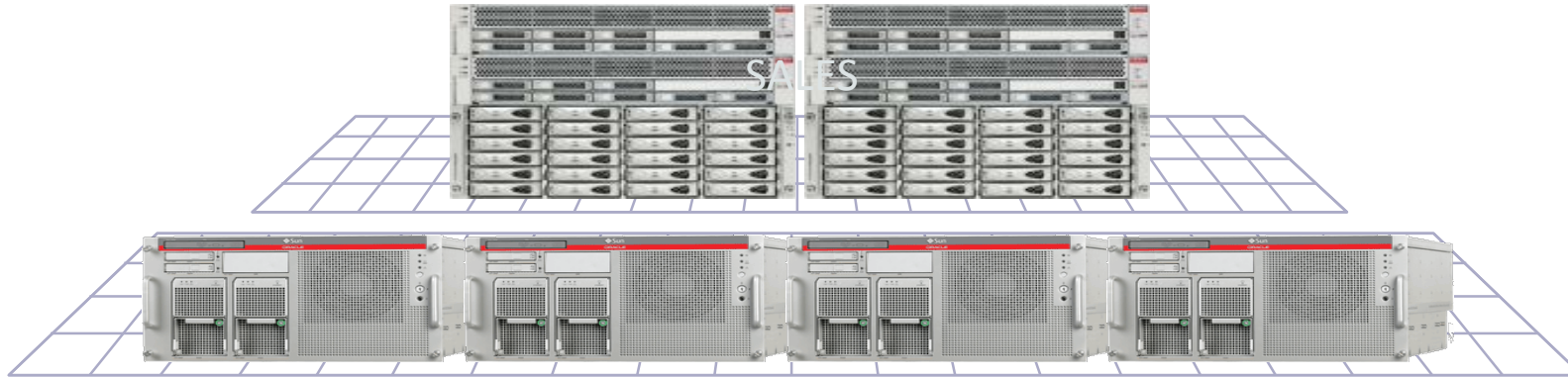
Reducing time and risk of change



- Fully automated workflow with change assurance for:
 - Database and O/S upgrades & migrations
 - Database configuration changes
 - Server and storage changes
- Capture workloads from Oracle9i, 10g and 11g databases

Real Application Clusters

Virtualize servers into a shared platform



- Run all databases for all applications on shared platform
- Highly available and scalable
- No changes required to applications

ORACLE®

What is new in Rdb?

V7.3.1 and later

Ian Smith
Oracle Rdb Product Architect
Oracle Rdb Engineering

Program Agenda

- 1 Development History
- 2 Optimizer Changes
- 3 New SQL language changes
- 4 New SQL precompiler changes
- 5 Q & A

Program Agenda

- 1 Development History
- 2 Optimizer Changes
- 3 New SQL language changes
- 4 New SQL precompiler changes
- 5 Q & A

Program Agenda

- 1 Development History
- 2 **Optimizer Changes**
- 3 New SQL language changes
- 4 New SQL precompiler changes
- 5 Q & A

Program Agenda

- 1 Development History
- 2 Optimizer Changes
- 3 New SQL language changes
- 4 New SQL precompiler changes
- 5 Q & A

Program Agenda

- 1 Development History
- 2 Optimizer Changes
- 3 New SQL language changes
- 4 New SQL precompiler changes
- 5 Q & A

Program Agenda

- 1 Development History
- 2 Optimizer Changes
- 3 New SQL language changes
- 4 New SQL precompiler changes
- 5 Q & A

Development History

State of the Release

History

- Developed in parallel with V7.2
- Some new features were delivered in various Rdb 7.2 releases
 - Built-in functions such as SYSTIMESTAMP
 - Sort improvements
 - Move data to 64 bits address space (P2)
 - Some query rewrite features
- V7.3.1 first feature release of V7.3

Requirements

- Require OpenVMS V8.3 or V8.4 systems
 - Please ensure all recommended OpenVMS patches are installed (see release notes)

Immediate dependencies on OpenVMS

- OpenVMS Patch kit names:
 - (Integrity) VMS84I_SYS-V0500
 - (Alpha) VMS84A_SYS-V0500
- Dependencies:
 - (Integrity) VMS84I_PCSI-V0400 and VMS84I_UPDATE-V0900
 - (Alpha) VMS84A_PCSI-V0400 and VMS84A_UPDATE-V0900
- Really applies to all OpenVMS systems running Rdb

OpenVMS Problem Description

- Process using JDBC hangs connecting to Oracle Rdb DB.
- Problem Description:
 - A multithreaded process using JDBC to connect to an Oracle Rdb database can hang when using V7.3 RDB\$COSIP.
RDB\$COSIP is using SYS\$ACMW for authentication.
- This problem has been fixed.
 - Images Affected:
 - [SYS\$LDR]ACME.EXE

Note

- RDB\$COSIP.EXE is not multi-version
- Will supersede older versions when V7.3 is installed
- Therefore, this problem may affect V7.2 production systems even if V7.3 is not in use but is installed

Current releases

- V7.3.1 in September, 2013
- V7.3.1.1 in January, 2014
- V7.3.1.2 in October, 2014
- Oracle Rdb strongly suggest using V7.3.1.1 or later when converting databases using **RMU Convert** or **RMU Restore** (from an older version)
- All customers are encouraged to upgrade their V7.3.1.* systems to V7.3.1.2 as soon as possible.
 - Known problem with zero cardinality

Optimizer Changes

System Table Changes

- **create database** now implicitly creates system tables with **sorted ranked** indices
- **RMU Convert** also changes system indices to **sorted ranked**
- Benefits:
 - Eliminate duplicate chains
 - Fixed sizes allows better page utilization
 - Improved performance for DDL operations
 - For example, DROP VIEW

Internal use of Bitmapped Scan

- Rdb now enables **bitmapped scan** for all system table references generated by internal tools
 - SQL and RDO queries

Optimize for

- Queries can now specify **optimize for bitmapped scan** on a **select**, **insert ... select**, **update**, **delete** statements
- Compound statements can use **pragma (optimize for bitmapped scan)**
- RMU Unload supports **bitmapped_scan** option for the Optimize qualifier
 - The optimizer will attempt to use bitmapped scan if possible

Query Outlines

Inline query outline syntax

OPTIMIZE OUTLINE clause

- Outline can now be included with the query
- Allows tuning in cases where the outline can not be stored in the database

OPTIMIZE OUTLINE clause

```
select last_name, middle_initial, first_name
from employees
where last_name = 'Toliver' and first_name = 'Alvin'
optimize
  as test1
  outline (mode 0
    as (
      query (subquery (
        EMPLOYEES 0 access path index E1_INDEX
      ) ) )
    compliance optional
    execution options (total time));
~Query Name: "TEST1"
~S: Outline "(unnamed)" used
...
LAST_NAME          MIDDLE_INITIAL    FIRST_NAME
Toliver            A.                Alvin
1 row selected
```

Query Rewrite

Simplifying query during compile

Nullability

- Make use of nullability knowledge to simplify query
- For instance
where employee_id is NULL
can never be **true** if there is a PRIMARY KEY or NOT NULL constraint on the column

SORT simplification

- If value is a constant or parameter that does not vary during SORT then it can be eliminated
- For **order by** can reduce VM usage, or even allow the SORT to be eliminated
- Especially beneficial for **distinct** which often includes string literals

Example; V7.2

```
SQL> select distinct 'Employee: ', employee_id, ' accessed by ', current_user  
cont> from salary_history  
cont> limit to 1 row;
```

Tables:

0 = SALARY_HISTORY

Firstn: 1

Reduce: 'Employee: ', 0.EMPLOYEE_ID, ' accessed by ', CURRENT_USER

Sort: 'Employee: '(a), 0.EMPLOYEE_ID(a), ' accessed by '(a), CURRENT_USER(a)

Index only retrieval of relation 0:SALARY_HISTORY

Index name SH_EMPLOYEE_ID [0:0]

EMPLOYEE_ID

Employee: 00164 accessed by SMITHI

1 row selected

Invariant functions and strings

Example; V7.3

```
SQL> select distinct 'Employee: ', employee_id, ' accessed by ', current_user  
cont> from salary_history  
cont> limit to 1 row;
```

Tables:

0 = SALARY_HISTORY

Firstn: 1

Reduce: 0.EMPLOYEE_ID

Index only retrieval of relation 0:SALARY_HISTORY

Index name SH_EMPLOYEE_ID [0:0]

EMPLOYEE_ID

Employee: 00164 accessed by SMITHI

1 row selected

Simplified and can eliminate sort step

Constant folding

- Evaluate simple expressions in the compiler, rather than generating runtime code
- Detect simple transformations
 - **Value** + 0 transforms to **Value**
 - **Value** * 1 transforms to **Value**
 - **Value** * 0 transforms to 0
 - 0 – **Value** transforms to –**Value**
- Applies to unscaled integer type and floating types
- **Value** must be not nullable

Comparison Simplification

- Some applications (usually generated) include queries like:
where 1 = 1 and ...
- Optimizer now eliminates TRUE predicates ($1 = 1$) during the compile phase

Comparison Simplification

- Some applications (usually generated) include queries like:
where 1 <> 1 or ...
- Optimizer now eliminates FALSE predicates during the compile phase

Comparison Simplification

- Some applications (usually coding error) include queries like:
where project_id = NULL
- Optimizer now eliminates UNKNOWN predicates during the compile phase
- (Note probably meant to be **project_id is NULL**)

AND, OR and NOT propagation of nullability

- As expressions get changed to **true**, **false** or **unknown** we apply logic to the query tree and remove any branch that isn't useful (always **false**)
- May lead to whole table access being eliminated

CASE expression pruning

- Applying logic rules to **case when** Boolean expression allows elimination of never possible branches
- Applies to various functions such as **nullif**, **nvl**, **nvl2**, **coalesce**, **abs**, **decode**, and **sign**
- In some cases remove condition completely

Data types for COUNT

And related operators

COUNT

- In prior versions COUNT was accumulated using an **integer**
- Now Rdb uses a **bigint**
- Automatically converted to smaller types in existing applications
- Dynamic applications must allow for **bigint** type

STDDEV, VARIANCE and AVG

- Each of these function uses the row count to compute their result. This internal count is also a **bigint**

COUNT (value-expr)

- Reimplemented for V7.3.1
- Previously the expression was translated to:
 - **count(*) filter (where value-expr is not null)**
- This meant that SQL null-elimination semantics were never returned
- COUNT() couldn't report "NULL values eliminated in aggregate function" in SQLCA or SQLSTATE because those values were previously excluded
 - Require CDD V7.2.0.4 to support computed by and views with this syntax

COUNT...

- The side effect is that generated internal query (BLR) changed
- The query outline signature is different
- May need to redefine query outlines for these cases

DIVIDE

- DIVIDE always returns a **double precision** result
- In prior versions it might result in a real result (if using **tinyint**, or **smallint**)



SQL Language Changes

BINARY and BINARY VARYING

BINARY

- Fixed length binary string
- Padded with X'00' octets
(during string compare or when a small string assigned to larger string)
- Usage for small images, encrypted values, compressed data, etc.

BINARY VARYING

- Varying length binary strings
- Length included (similar to VARCHAR)
- VARBINARY is a synonym

C/C++ Programmers

- Programmers can use `$SQL_VARBINARY` pseudo type to declare a typedef for binary data
- Then use `.len` and `.body` sub-fields to access length (int) and body (char) data

Supported in SQLDA using new types

- BINARY VARYING (VARBINARY)
 - Value: 909
 - Symbolic name: SQLDA_VARBINARY
- BINARY
 - Value: 913
 - Symbolic name: SQLDA_BINARY

List of Byte Varying Storage Changes


Storage Model

- Also called *segmented strings* (V1.0)
- Chained style
 - Segments use **dbkey** pointer to next data in list
 - Partial chains written as needed (buffer overflow)
 - May require joining chains (extra I/O)
- Pointer style (V4.1)
 - Vector based storage {**length**, **dbkey**}
 - Data segment doesn't include pointers
 - Chain of pointer segments written last
 - Avoids "fixup" I/O

Example V7.2 – 9 data segments

- 'Example 1'
- <empty>
- 'The'
- 'Rain'
- 'In'
- 'Spain'
- 'Falls'
- 'Mainly'
- 'On the Plain'

- Len=9
- Len=0
- Len=3
- Len=4
- Len=2
- Len=5
- Len=5
- Len=6
- Len=12



Store segment even for zero length data

Compact Storage

- We observed that if actual data is tiny then
 - wasted overhead
 - Overhead: record header + TDX/LDX entries on page
 - wasted I/O to fetch
 - Likely page cached but still expend CPU on the fetch
- Examined usage where there are variable length segments
 - Often see zero length segment for paragraph breaks
 - Have seen one application store telemetry readings as list of FLOAT values

Example; dump of page header

```
0003 00000008 0000 page 8, physical area 3
          90832F65 0006 checksum = 90832F65
00AECE93 019C8159 000A time stamp = 18-OCT-2014 19:34:33.02
          0000 00A0 0012 160 free bytes, 0 locked
          000E 0016 14 lines
0020 03CE 0018 line 0: offset 03CE, 32 bytes
0005 03C4 001C line 1: offset 03C4, 5 bytes
0047 037C 0020 line 2: offset 037C, 71 bytes
004E 032E 0024 line 3: offset 032E, 78 bytes
0038 02F6 0028 line 4: offset 02F6, 56 bytes
0005 02EC 002C line 5: offset 02EC, 5 bytes
004C 02A0 0030 line 6: offset 02A0, 76 bytes
004A 0256 0034 line 7: offset 0256, 74 bytes
0028 022E 0038 line 8: offset 022E, 40 bytes
0005 0224 003C line 9: offset 0224, 5 bytes
0005 01D4 0040 line 10: offset 01D4, 80 bytes
0005 0180 0044 line 11: offset 0180, 83 bytes
0005 0176 0048 line 12: offset 0176, 5 bytes
0005 0128 004C line 13: offset 0128, 78 bytes
```

Empty lines

Example; dump of record

```

0000 0128 line 13 (3:8:13) record type 0
00 0001 012A Control information
      .... 73 bytes of static data
20646E61206465706F6C65766544202A 012D data '* Developed and '
726F206E612064656E6961746E69616D 013D data 'maintained an or'
656B64726F6365722064657A696E6167 014D data 'ganized recordke'
646E61206D657473797320676E697065 015D data 'eping system and'
      646572617065727020 016D data ' prepared'

0000 0176 line 12 (3:8:12) record type 0
00 0001 0178 Control information
0000000000 017B padding '.....'

0000 0180 line 11 (3:8:11) record type 0
00 0001 0182 Control information
      .... 73 bytes of static data
6E61206465706F6C65766544202A 012D data 'analytical stren'
726F206E612064656E6961746E69616D 013D data 'gths to isolate '
656B64726F6365722064657A696E6167 014D data 'issues and facil'
646E61206D657473797320676E697065 015D data 'itate their time'
      646572617065727020 016D data 'ly resolution.'
00 01D3 padding '.'
```

Small record also has padding
Need minimum 10 bytes to allow for
fragmented row mechanics

Changes in V7.3

- Now store the data in the pointer segment if less than or equal to 8 bytes
 - (Note: RMU Convert /NOCOMMIT databases do not use new algorithm)
- Zero length segments often appear in source and comments stored in the metadata
- The example list of FLOAT values now 100% stored in the pointer segment
 - (Note: LIST OF BYTE VARYING columns not rewritten by RMU Convert)

Example V7.3 – just 2 data segments

- 'Example 1'
 - <empty>
 - 'The'
 - 'Rain'
 - 'In'
 - 'Spain'
 - 'Falls'
 - 'Mainly'
 - 'On the Plain'
- Len=9
 - (in pointer)
 - (in pointer)
 - (in pointer)
 - (in pointer)
 - (in pointer)
 - (in pointer)
 - Len=12

Declare LOCAL Temporary View

New syntax

LOCAL TEMPORARY VIEW

- Allows a module to have a global view definition without a view being defined in the database
- Based on **declare local temporary table** support
- View definition is loaded from the module when the first routine is called

Local Temporary View

```
SQL> declare local temporary view module.a
cont>  (eid edit string 'XXBXXX'
cont>      comment is 'Employee id'
cont>  ,num_jobs query name 'NUMBER_JOBS'
cont>  ,started query header 'When'/'Started'
cont>  ,current_start
cont>      default value for dtr '1-Jan-1900 00:00:00.00')
cont>  as select employee_id, count(*),
cont>      min (job_start), max (job_start)
cont>      from job_history
cont>      group by employee_id;
SQL>
SQL> select * from module.a where eid <= '00164';

              When
EID          NUM_JOBS   Started      CURRENT_START
00 164             2    5-JUL-1980    21-SEP-1981
1 row selected
SQL>
```

REPLACE Statement

Alternate to INSERT statement

New statement

- Acts like **insert** but will preserve uniqueness for **primary key** using a pre-DELETE action
- If there is no **primary key** it is a simple **insert**
- Triggers are affected
 - Activates BEFORE and AFTER INSERT trigger
 - Activates BEFORE and AFTER DELETE trigger if a primary key value exists
- Valid statement in **create trigger** statement

REPLACE example with PRIMARY KEY

```
SQL> alter table WORK_STATUS
cont>      add constraint PK_WORK_STATUS
cont>      primary key (STATUS_CODE)
cont>      not deferrable;
SQL>
SQL> replace into WORK_STATUS
cont>  values ('0', 'INACTIVE', 'RECORD EXPIRED');
1 row replaced
SQL> select * from work_status;
STATUS_CODE  STATUS_NAME  STATUS_TYPE
1            ACTIVE      FULL TIME
2            ACTIVE      PART TIME
0            INACTIVE    RECORD EXPIRED
3 rows selected
```

Sequences and Identity Changes

ISO/ANSI Standard

Sequence

- Sequences project was based upon Oracle Database syntax
- Feature now part of ISO/ANSI SQL standard
- Rdb accepts both Oracle and ISO/ANSI SQL syntax for **create** and **alter sequence** syntax
- Biggest change is the NO is a separate keyword

```
SQL> create sequence PRODUCT_PK  
cont> no maxvalue  
cont> no cache  
cont> wait  
cont> no cycle  
cont> minvalue 1;  
SQL>
```

Identity

- Based on commonly used syntax from other database systems
 - e.g. IDENTITY (1, 2)
 - Internally built on sequence feature
- Only one identity per table
 - enforced by naming the special sequence with table name
- Now supported by ISO/ANSI SQL Standard
- Clauses are the same as create sequence
 - e.g. IDENTITY (start with 1 increment by 2 cycle **cache**)

Reverse attribute

V7.3.1.2

- New **reverse** keyword for **create sequence** statement and **identity** clause
- Changes the value returned by **.currval** and **.nextval** pseudo columns
- Values are bit reversed
- Allows index keys to be scattered around the index even those base sequence is systematically increasing

SAVEPOINT Support

Sub-transaction units

Savepoint support

- **savepoint** allows for a small inner part of a transaction to be named and managed.
- **release savepoint** is used to discard the controlled changed
- **rollback to savepoint** is used to undo part of the current transaction

SAVEPOINT example

```
set flags 'TRANSACTION';
begin
set transaction read write;

insert into SV_X values (10);

savepoint D;
insert into SV_X values (20);
insert into SV_X values (30);
insert into SV_X values (40);

rollback to savepoint D;

insert into SV_X values (50);
insert into SV_X values (60);

commit;
end;
```

SAVEPOINT example

```
~T Start_transaction (8) on db: 1, db count=1  
~T Savepoint "D" (8.2) on db: 1  
~T Rollback to Savepoint "D" (8.2) on db: 1  
~T Commit_transaction (8) on db: 1  
~T Prepare_transaction (8) on db: 1
```

```
select * from SV_X order by V;  
      V  
      10  
      50  
      60  
3 rows selected
```

Savepoint support

- Each **savepoint** is given a unique name which is specific to a database attach
- If multiple database ALIAS are in use then use the ALIAS name to provide context.
- `SAVEPOINT db1.mysavepoint`

Savepoint support

- Currently Rdb limits a session to just one active **savepoint**
- Extending this to multiple nested savepoints is planned for a future release
- Current limit imposed because of complexity of in memory metadata

WITH Clause

Prefix to SELECT clause

WITH clause

Simplifying queries



WITH clause of SELECT statement

- Define partial queries for use in a query
 - known as subquery factoring
- Currently used by some Oracle clients to collect data. Enables use of those tools with Rdb
 - Rdb does not support recursive queries
- Note: `begin with hold ... end;` is deprecated due to this syntax.
Use `begin pragma (with hold) ... end;` instead.

WITH example

Definition of factor

```
SQL> with emp as (select *
cont>                from employees
cont>                inner join job_history using (employee_id)
cont>                where job_end is null),
cont>      dpt as (select * from departments)
cont> select e.last_name, d.department_name,
cont>        m.last_name as Manager
cont> from emp e
cont>      left outer join dpt d using (department_code)
cont>      inner join emp m on (d.manager_id = m.employee_id)
cont> order by d.manager_id
cont> ;
```

```
  E.LAST_NAME      D.DEPARTMENT_NAME
  Siciliano        Board Manufacturing North
...
  Herbener         Engineering
100 rows selected
SQL>
```

Reference to factor name



SQL Pre-compiler Changes

C++ Compiler Support

CXX compiler

Improved support for C++

- SQL pre-compiler generates .C source files from .SC embedded sources
- Calls routines in generated auxiliary module
- The routine prototypes defined for these routines were not C++ complaint

New CC option

- Can now use /CC=CXX option to change the generated C code
- SQL\$PRE will now invoke the CXX compiler to process the augmented C source
- Changes to various structures and call interface
- Any qualifiers (apart from /SQLOPTIONS) must be acceptable to CXX compiler

Changes in generated code

- Prototypes will include parameter definitions
- Prototypes are enclosed by **extern “C” {...}** to prevent the names being interpreted as C++ routines
- **sql_rdb_headers.h** now provides alternate prototypes
- SQLCODE must be defined as **int** type
- Rdb\$Message_Vector now a **typedef**, not a **struct** type

Notes

- SQL\$PRE is still basically processing C code
- These changes do not include language features from C++
- Sources should use standard C syntax



Questions and Answers

Oracle Beehive Conferencing Client provides a chat area. Please ask questions there too.

Safe Harbor Statement

The preceding 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.

Hardware and Software

Engineered to Work Together

ORACLE®