Rich’s Overview…

- Advisor to Rolta International Board
- Former President of TUSC
  - Inc. 500 Company (Fastest Growing 500 Private Companies)
  - 10 Offices in the United States (U.S.); Based in Chicago
  - Oracle Advantage Partner in Tech & Applications
- Former President Rolta TUSC & President Rolta EICT International
- Author (3 Oracle Best Sellers – #1 Oracle Tuning Book for a Decade):
  - Oracle Performing Tips & Techniques (Covers Oracle7 & 8i)
  - Oracle9i Performance Tips & Techniques
  - Oracle Database 10g Performance Tips & Techniques
- Former President of the International Oracle Users Group
- Current President of the Midwest Oracle Users Group
- Chicago Entrepreneur Hall of Fame - 1998
- E&Y Entrepreneur of the Year & National Hall of Fame - 2001
- MOUG Top Speaker Twelve Times
- National Trio Achiever award - 2006
- Oracle Certified Master & Oracle Ace Director
- Purdue Outstanding Electrical & Computer and Engineer - 2007
Technology is Everywhere…

ORACLE CLOUD

Engineered For Heroes
oracle.com/avengers

Iron Man 3
In Theaters May 3

ORACLE

CLEAR CHANNEL

SHOE SHINE AND REPAIR

Restroom
How Much Data …

- 2004 monthly internet traffic >1E; 2010 it was 21E/month.
- In 2012, **2.5E data created every day** (about 1Z=1000E per year)
- June 2012 – Facebook has **100P Hadoop cluster**
- Facebook: **500T** processed daily – (210T/hr DWHSE scanned)
- A Single Jet Engine – **20T/hour** – same rate as Facebook!
- Gmail has **450 million users**
- Wal-Mart – 1 million customer transactions/hour (2.5P DB)
- Large Hadron Collider produced **13P in one year**
- Business data **doubles every 1.2 years**
- 19% of $1B companies have **>1P of data** (31% in 2013)
- 2011 – First **Exabyte tape library** from Oracle
- Decoding Human Genome took 10 yrs; Now takes a week!
Audience Knowledge / Versions

- Oracle7 Experience
- Oracle8i Experience
- Oracle9i Experience
- Oracle10g Experience
- Oracle Database 11g R2 Experience
- Oracle Database 12c Experience

Goals
- Present NEW features in an EASY way
- Focus on a few nice features of Oracle12c

Non-Goals
- Learn ALL aspects of Oracle12c
- Learn Tips that will make you an expert
Overview – 12c

- Know the Oracle!
- Start Me Up – Using Memory Target, The Buffer Cache & The Result Cache
- Invisible Columns (12c) & virtual columns (11g)
- Multiple indexes on the same Column (12c) & Invisible Indexes (11g)
- Adaptive Execution Plans (12c) & Adaptive Cursor Sharing & Bind Peeking (11g)
- Runaway query Management (12c)
- Change Table Compression at import Time (12c) & (Partition Compression – 11g)
- Create Views as Tables (12c)
- Online Move Partition (12c) & Interval Partitioning (11g)
- Partial Indexes for Partitioned Table (12c)
- Pluggable Databases (12c)
- Enhanced DDL Online (12c)
- Exadata and Big Data (In-Database MapReduce in 12c)
- Consolidated Database Replays & Better Reporting (12c)
- Automatic Diagnostics Repository (12c)
- Security Enhancements (12c)
- Other 12c New Features
Know the Oracle

“I admire risk takers. I like leaders – people who do things before they become fashionable or popular. I find that kind of integrity inspirational.”

LAWRENCE J. ELLISON | Chairman & Chief Executive Officer, 2003
Oracle Firsts – Innovation!

1979 First commercial SQL relational database management system
1983 First 32-bit mode RDBMS
1984 First database with read consistency
1987 First client-server database
1994 First commercial and multilevel secure database evaluations
1995 First 64-bit mode RDBMS
1996 First to break the 30,000 TPC-C barrier
1997 First Web database
1998 First Database - Native Java Support; Breaks 100,000 TPC-C
1998 First Commercial RDBMS ported to Linux
2000 First database with XML
2001 First RDBMS with Real Application Clusters & First middle-tier database cache
2004 First True Grid Database
2005 First FREE Oracle Database (10g Express Edition)
2006 First Oracle Support for LINUX Offering
2007 Oracle 11g Released!
2008 Exadata V1 Server Announced (Oracle buys BEA)
2009 Oracle buys Sun – Java; MySQL; Solaris; Hardware; OpenOffice
2010 Oracle announces MySQL Cluster 7.1, Exadata, Exalogic, America’s Cup Win
2011 X2-2 Exadata, ODA, Exalytics, SuperCluster, Big Data, Cloud, Social Network
2012 X3-2 Exadata, Expanded Cloud Offerings, Solaris 11.1
2013 Oracle12c Released! Oracle X3-8 Exadata, Acquisitions (Acme Packet...etc.)!
Testing the **Future** Version

**Version 12.1.0.0.1 of the Database**

**Version 11.2.0.1.0 of the Database** for 11g R2 Examples
# Oracle Database 12c Release 1: Upgrade Paths

## Direct Upgrade Path

<table>
<thead>
<tr>
<th>Source Database</th>
<th>Target Database</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.2.0.5 (or higher)</td>
<td>12.1.x</td>
</tr>
<tr>
<td>11.1.0.7 (or higher)</td>
<td>12.1.x</td>
</tr>
<tr>
<td>11.2.0.2 (or higher)</td>
<td>12.1.x</td>
</tr>
</tbody>
</table>

## In-Direct Upgrade Path

<table>
<thead>
<tr>
<th>Source Database</th>
<th>Upgrade Path for Target Database</th>
<th>Target Database</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.3.3.0.0 (or lower)</td>
<td>7.3.4.x --&gt; 9.2.0.8</td>
<td>11.2.x</td>
</tr>
<tr>
<td>8.0.5.0.0 (or lower)</td>
<td>8.0.6.x --&gt; 9.2.0.8</td>
<td>11.2.x</td>
</tr>
<tr>
<td>8.1.7.0.0 (or lower)</td>
<td>8.1.7.4 --&gt; 9.2.0.8</td>
<td>11.2.x</td>
</tr>
<tr>
<td>9.0.1.3.0 (or lower)</td>
<td>9.0.1.4 --&gt; 9.2.0.8</td>
<td>11.2.x</td>
</tr>
</tbody>
</table>
Database Upgrade Assistant (DBUA)

- More automation during the upgrade process
- Additional validation steps (also for on-line)
- Post upgrade more automated as well
- Better status as to specific component success
- Post upgrade fix-it scripts to help automate future needs
- Parallel upgrade takes advantage of multiple CPU cores
- Express Edition Upgrade to others (since 11g)
Database Upgrade Assistant (DBUA)

- DBUA checks before the upgrade:
  - Invalid user accounts or roles
  - Invalid data types or invalid objects
  - De-supported character sets
  - **Adequate resources** (rollback segments, tablespaces, and free disk space)
  - Missing SQL scripts needed for the upgrade
  - Listener running (if Oracle Enterprise Manager Database Control upgrade or configuration is requested)
  - Oracle Database software linked with Database Vault option. If Database Vault is enabled, Disable Database Vault before upgrade.
$ sqlplus ***/*** 

Copyright (c) 1982, 2007, Oracle. All rights reserved.

Connected to:
Oracle Database 12c Enterprise Edition Release 12.1.0.0.1 - 64bit Beta
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> sho sga

Total System Global Area 626327552 bytes
Fixed Size 2276008 bytes
Variable Size 524289368 bytes
Database Buffers 92274688 bytes
Redo Buffers 7487488 bytes
SQL>
MEMORY_TARGET
&
Automatic Memory Management
Automatic Memory Management (AMM)
MEMORY_TARGET in 11g

• First there was some Automatic Memory Mgmt - 9i
  – SGA_MAX_SIZE introduced in 9i – Dynamic Memory
  – No more Buffers – DB_CACHE_SIZE
  – Granule sizes introduced - _ksm_granule_size

• Then came SGA_TARGET – 10g
  – Oracle Applications recommends setting this for SGA
  – Set minimums for key values (Data Cache / Shared Pool)

• Now there is MEMORY_TARGET – 11g
  – SGA + PGA all in one setting; Still set minimums
Database Smart Flash Cache (Solaris/Oracle Linux) – L2 cache set 2-10x SGA:

- `DB_FLASH_CACHE_FILE = /dev/sda, /dev/sdb, /dev/sdc`
- `DB_FLASH_CACHE_SIZE = 32G, 32G, 64G`
<table>
<thead>
<tr>
<th>Component</th>
<th>Initialization Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed SGA</td>
<td>None</td>
</tr>
<tr>
<td>Shared Pool</td>
<td>SHARED_POOL_SIZE</td>
</tr>
<tr>
<td>Large Pool</td>
<td>LARGE_POOL_SIZE</td>
</tr>
<tr>
<td>Java Pool</td>
<td>JAVA_POOL_SIZE</td>
</tr>
<tr>
<td>Buffer Cache</td>
<td>DB_CACHE_SIZE</td>
</tr>
<tr>
<td>Streams Pool</td>
<td>STREAMS_POOL_SIZE</td>
</tr>
</tbody>
</table>
### Manually Sized SGA (use SGA_TARGET)

**PGA_AGGREGATE_LIMIT (New 12c)**

<table>
<thead>
<tr>
<th>Component</th>
<th>Initialization Parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log buffer</td>
<td>LOG_BUFFER (pfile only since 10g)</td>
</tr>
<tr>
<td>Keep Pool</td>
<td>DB_KEEP_CACHE_SIZE</td>
</tr>
<tr>
<td>Recycle Pool</td>
<td>DB_RECYCLE_CACHE_SIZE</td>
</tr>
<tr>
<td>Block caches</td>
<td>DB_nK_CACHE_SIZE</td>
</tr>
</tbody>
</table>

**Program Global Area (now in MEMORY_TARGET):**

- Aggregate PGA: **PGA_AGGREGATE_TARGET (11g)**
- New PGA Limit: **PGA_AGGREGATE_LIMIT (12c)**
Moving from SGA_TARGET to:
MEMORY_TARGET (set minimums)

ALTER SYSTEM SET SGA_TARGET=200M;
ALTER SYSTEM SET PGA_AGGREGATE_TARGET=100M;
ALTER SYSTEM SET PGA_AGGREGATE_LIMIT=140M;

SQL> sho parameter target

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>memory_max_target</td>
<td>big integer</td>
<td>360M</td>
</tr>
<tr>
<td>memory_target</td>
<td>big integer</td>
<td>360M</td>
</tr>
<tr>
<td>pga_aggregate_target</td>
<td>big integer</td>
<td>100M</td>
</tr>
<tr>
<td>sga_target</td>
<td>big integer</td>
<td>200M</td>
</tr>
</tbody>
</table>
The System Global Area (SGA) is a group of shared memory structures that contains data and control information for one Oracle database. The SGA is allocated in memory when an Oracle database instance is started. The Allocation History chart shows the history of the components of the SGA.

Current Allocation

Automatic Shared Memory Management Enabled

Total SGA Size (MB): 636

<table>
<thead>
<tr>
<th>SGA Component</th>
<th>Current Allocation (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shared Pool</td>
<td>352</td>
</tr>
<tr>
<td>Buffer Cache</td>
<td>260</td>
</tr>
<tr>
<td>Large Pool</td>
<td>4</td>
</tr>
</tbody>
</table>
Use **DBMS_XPLAN** to query the execution plan
- Automatically queries the **last plan** in **PLAN_TABLE**
- uses a TABLE() function with another pipelined function
- Operation text truncation might be a problem
- Will give additional information after plan
  - Highlight filter vs join conditions, if plan table is current
  - Displays warning message of old version plan table is being used
- In 11g, a procedure for SQL Plan Baselines (we’ll cover these later).

```sql
DBMS_XPLAN.DISPLAY_SQL_PLAN_BASELINE (  
    sql_handle IN VARCHAR2 := NULL,  
    plan_name IN VARCHAR2 := NULL,  
    format IN VARCHAR2 := 'TYPICAL')  
  <'BASIC','ALL'>
RETURN dbms_xplan_type_table;
```
DBMS_XPLAN Example:

Select *  
from table (dbms_xplan.display);

PLAN_TABLE_OUTPUT

<table>
<thead>
<tr>
<th>Id</th>
<th>Operation</th>
<th>Name</th>
<th>Rows</th>
<th>Bytes</th>
<th>Cost</th>
<th>Pstart</th>
<th>Pstop</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>UPDATE STATEMENT</td>
<td></td>
<td>328</td>
<td>2296</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>UPDATE</td>
<td>JOURNAL_LINE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>PARTITION RANGE ALL</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>TABLE ACCESS FULL</td>
<td>JOURNAL_LINE</td>
<td>328</td>
<td>2296</td>
<td>2</td>
<td>1</td>
<td>4</td>
</tr>
</tbody>
</table>

Note: cpu costing is off, 'PLAN_TABLE' is old version
11 rows selected
Select *
from table (dbms_xplan.display);

<table>
<thead>
<tr>
<th>Id</th>
<th>Operation</th>
<th>Name</th>
<th>Rows</th>
<th>Bytes</th>
<th>Cost (%CPU)</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SELECT STATEMENT</td>
<td></td>
<td>14</td>
<td>728</td>
<td>2</td>
<td>00:00:01</td>
</tr>
<tr>
<td>1</td>
<td>TABLE ACCESS FULL</td>
<td>EMP_RICH</td>
<td>14</td>
<td>728</td>
<td>2</td>
<td>00:00:01</td>
</tr>
</tbody>
</table>

Note
-----

- **dynamic sampling used** for this statement (level=2)
The Virtual Column
The Virtual Column

• The value of the virtual column is a derived expression.
  – Can be derived from columns of the same table or from constants
  – Can include SQL or user-defined PL/SQL functions

• Virtual column DATA is NOT PHYSICALLY STORED.

• You CAN NOT explicitly write to a virtual column

• You CAN create a PHYSICAL index (result is function-based index) or partition on a virtual column <unlike a computed column in SQL Server or other databases>

• If you UPDATE columns of a virtual column and it has an index, then it will be computed on the UPDATE vs. on the SELECT (very important from a tuning standpoint).

• Index Organized and External Tables can NOT have virtual columns.
create table emp_rich
  (empno number(4),
   sal   number(7,2),
   yearly_sal generated always as (sal*12),
   deptno number(2));

Table created.

insert into emp_rich(empno, sal, deptno)
  select empno, sal, deptno from scott.emp;

14 rows created.
select * from emp_rich;

<table>
<thead>
<tr>
<th>EMPNO</th>
<th>SAL</th>
<th>YEARLY_SAL</th>
<th>DEPTNO</th>
</tr>
</thead>
<tbody>
<tr>
<td>7369</td>
<td>800</td>
<td>9600</td>
<td>20</td>
</tr>
<tr>
<td>7499</td>
<td>1600</td>
<td>19200</td>
<td>30</td>
</tr>
<tr>
<td>7521</td>
<td>1250</td>
<td>15000</td>
<td>30</td>
</tr>
<tr>
<td>7566</td>
<td>2975</td>
<td>35700</td>
<td>20</td>
</tr>
<tr>
<td>7654</td>
<td>1250</td>
<td>15000</td>
<td>30</td>
</tr>
<tr>
<td>7698</td>
<td>2850</td>
<td>34200</td>
<td>30</td>
</tr>
</tbody>
</table>

...
Invisible Columns
Invisible Columns

- The new 12c feature allows you to *hide* columns.
- If a user or developer selects **ALL** columns from a table (i.e. select *...*) the invisible columns will **NOT** be displayed.
- If a user specifically selects the invisible column (i.e. select salary,...) the column **WILL** be displayed in the output (you have to *know* it’s there).
- You can set column(s) to be visible/invisible with an alter table:

```
SQL> ALTER TABLE EMPLOYEE MODIFY (SSN INVISIBLE);
Table altered.
```
Invisible Columns
Example – Simple EMP SELECT

```
SELECT *
FROM EMP
WHERE SAL > ANY
    (SELECT SAL
     FROM EMP
     WHERE DEPTNO=30)
AND DEPTNO=10
ORDER BY SAL DESC;
```

<table>
<thead>
<tr>
<th>EMPNO</th>
<th>ENAME</th>
<th>JOB</th>
<th>MGR</th>
<th>HIREDATE</th>
<th>DEPTNO</th>
<th>SAL</th>
<th>COMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>7839</td>
<td>KING</td>
<td>PRESIDENT</td>
<td>7839</td>
<td>17-NOV-81</td>
<td>10</td>
<td>5000</td>
<td></td>
</tr>
<tr>
<td>7782</td>
<td>CLARK</td>
<td>MANAGER</td>
<td>7839</td>
<td>09-JUN-81</td>
<td>10</td>
<td>2450</td>
<td></td>
</tr>
<tr>
<td>7934</td>
<td>MILLER</td>
<td>CLERK</td>
<td>7782</td>
<td>23-JAN-82</td>
<td>10</td>
<td>1300</td>
<td></td>
</tr>
</tbody>
</table>

Both SAL & COMM columns displayed above!
alter table emp modify (sal invisible, comm invisible);
Table altered.

SELECT *
FROM EMP
WHERE SAL > ANY
  (SELECT SAL
   FROM EMP
   WHERE DEPTNO=30)
AND DEPTNO=10
ORDER BY SAL DESC;

<table>
<thead>
<tr>
<th>EMPNO</th>
<th>ENAME</th>
<th>JOB</th>
<th>MGR</th>
<th>HIREDATE</th>
<th>DEPTNO</th>
</tr>
</thead>
<tbody>
<tr>
<td>7839</td>
<td>KING</td>
<td>PRESIDENT</td>
<td></td>
<td>17-NOV-81</td>
<td>10</td>
</tr>
<tr>
<td>7782</td>
<td>CLARK</td>
<td>MANAGER</td>
<td>7839</td>
<td>09-JUN-81</td>
<td>10</td>
</tr>
<tr>
<td>7934</td>
<td>MILLER</td>
<td>CLERK</td>
<td>7782</td>
<td>23-JAN-82</td>
<td>10</td>
</tr>
</tbody>
</table>

No SAL or COMM columns displayed above!
Invisible Columns

Example – sal invisible but selected

```
SELECT SAL, JOB, ENAME, DEPTNO
FROM EMP
WHERE SAL > ANY
  (SELECT SAL
       FROM EMP
       WHERE DEPTNO=30)
AND DEPTNO=10
ORDER BY SAL DESC;
```

<table>
<thead>
<tr>
<th>SAL</th>
<th>JOB</th>
<th>ENAME</th>
<th>DEPTNO</th>
</tr>
</thead>
<tbody>
<tr>
<td>5000</td>
<td>PRESIDENT</td>
<td>KING</td>
<td>10</td>
</tr>
<tr>
<td>2450</td>
<td>MANAGER</td>
<td>CLARK</td>
<td>10</td>
</tr>
<tr>
<td>1300</td>
<td>CLERK</td>
<td>MILLER</td>
<td>10</td>
</tr>
</tbody>
</table>

SAL column IS displayed since I specifically SELECTED it.
Invisible Columns
Example – sal/comm to visible

To turn it back to being visible):

ALTER TABLE EMP MODIFY (SAL VISIBLE, COMM VISIBLE);
Table Altered.

Note: This is not for heavy security; there are other ways to achieve that:

• You can use column level security using Oracle's VPD (Virtual Private Database) to create a policy function and apply the policy function to our table, so that it does NOT display certain rows for a given deptno, BUT ONLY when the salary and/or COMM columns are selected. So all rows displayed when I DON'T choose SAL and/or COMM and all rows EXCEPT deptno 10 when I DO choose the SAL and/or COMM columns.

• You could also use TDE (Transparent Data Encryption) to encrypt the data for a given column. This is part of Oracle's Database Advanced Security Options and has certain restrictions.
The Result Cache

• Function Results of queries and query fragments can be cached in memory for future executions.
  – Choose calculations that frequently run
  – Choose data that does NOT frequently change
• RESULT_CACHE & RELIES_ON clauses
• Takes its memory from the Shared Pool
  – Set with RESULT_CACHE_MAX_SIZE
  – RESULT_CACHE_MODE=force (auto/manual)
• DBMS_RESULT_CACHE_FLUSH to clear
• Is NOT passed between RAC/Grid nodes
• Check the docs for other Restrictions & Rules!!
select *
from (select *
    from (select t.country_name, t.city_name,
            sum(t.salary) a_sum, max(t.salary) a_max
        from emps t
        group by t.country_name, t.city_name)
    order by a_max desc)
where rownum < 2;
Result Cache

Example Performance

**Step 1 - In Session 1**
Executed query without hint and it returned an elapsed time of 3.80 seconds *(not cached)*.

**Step 2 - In Session 2**
Executed query without hint and it returned an elapsed time of 3.20 seconds *(not cached)*.
Step 3 - In Session 2

Executed query with the RESULT_CACHE hint and it returned an elapsed time of 3.18 seconds (cache it).

Step 4 - In Session 1

Executed query without the RESULT_CACHE hint, but with RESULT_CACHE_MODE=force and it returned an elapsed time of 0.86 seconds (cached!!).
The **RELIES_ON Clause** specifies tables or views that the Function Results are dependent on.

-- Package specification

```sql
CREATE OR REPLACE PACKAGE HR IS
...

type DeptInfoRec IS RECORD (avgSal NUMBER, numberEmployees NUMBER);
```

-- Function declaration

```sql
FUNCTION GetDeptInfo (dept_id NUMBER) RETURN DeptInfoRec RESULT_CACHE;
```

```sql
...
END HR;
```
PACKAGE BODY HR IS

-- Function definition
FUNCTION GetDeptInfo (dept_id NUMBER) RETURN DeptInfoRec
  RESULT_CACHE RELIES_ON (EMP);
IS
  result DeptInfoRec;
BEGIN
  SELECT AVG(sal), count(*) INTO result
  FROM EMP
  WHERE deptno = dept_id;
  RETURN result;
END;

END HR;
The Result Cache – V$ Views

- **V$RESULT_CACHE_STATISTICS** – Displays the amount of memory to help you determine memory currently allocated to the result cache.

**Other V$ views:**
- **V$RESULT_CACHE_MEMORY**
- **V$RESULT_CACHE_OBJECTS**
- **V$RESULT_CACHE_DEPENDENCY**
## The Result Cache – FYI Only

### Digging Deeper

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>client_result_cache_lag</td>
<td>big integer</td>
<td>3000</td>
</tr>
<tr>
<td>client_result_cache_size</td>
<td>big integer</td>
<td>0</td>
</tr>
<tr>
<td>result_cache_max_result</td>
<td>integer</td>
<td>5</td>
</tr>
<tr>
<td>result_cache_max_size</td>
<td>big integer</td>
<td>1536K</td>
</tr>
<tr>
<td>result_cache_mode</td>
<td>string</td>
<td>MANUAL</td>
</tr>
<tr>
<td>result_cache_remote_expiration</td>
<td>integer</td>
<td>0</td>
</tr>
</tbody>
</table>

- **_result_cache_max_result** 100 maximum result size as percent of cache size
- **_result_cache_remote_expiration** 0 maximum life time (min) for any result using a remote object
- **_result_cache_timeout** 60 maximum time (sec) a session waits for a result
The Invisible Index
The Invisible Index

- Set an index to VISIBLE or INVISIBLE
  - ALTER INDEX idx1 INVISIBLE;
  - ALTER INDEX idx1 VISIBLE;
  - CREATE INDEX... INVISIBLE;

- Great to turn off indexes for a while when you think they’re not being used, but BEFORE you drop them.
- Can NOT use INDEX (to override invisibility) anymore, but CAN use NO_INDEX (to turn off visible indexes).
- The index IS MAINTAINED during DML
- Great for testing!
The Invisible Index

create index deptno_invisible_idx on dept_rich(deptno) invisible;
Index created.

select count(*) from dept_rich where deptno = 30; (doesn’t see the index)

COUNT(*)
---------
      512

Execution Plan
---------------------------------------------
Plan hash value: 3024595593

<table>
<thead>
<tr>
<th>Id</th>
<th>Operation</th>
<th>Name</th>
<th>Rows</th>
<th>Bytes</th>
<th>Cost (%CPU)</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SELECT STATEMENT</td>
<td></td>
<td>1</td>
<td>2</td>
<td>4 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>1</td>
<td>SORT AGGREGATE</td>
<td></td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>* 2</td>
<td>TABLE ACCESS FULL</td>
<td>DEPT_RICH</td>
<td>512</td>
<td>1024</td>
<td>4 (0)</td>
<td>0:00:01</td>
</tr>
</tbody>
</table>
The Invisible Index (set visible)

```sql
alter index dept_rich_inv_idx visible;
Index altered.
```

```sql
select count(*) from dept_rich where deptno = 30;
(it does see the index)
```

```
COUNT(*)
---------
  512
```

Execution Plan

```
<table>
<thead>
<tr>
<th>Id</th>
<th>Operation</th>
<th>Name</th>
<th>Rows</th>
<th>Bytes</th>
<th>Cost (%CPU)</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SELECT STATEMENT</td>
<td></td>
<td>1</td>
<td>2</td>
<td>1   (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>1</td>
<td>SORT AGGREGATE</td>
<td></td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>* 2</td>
<td>INDEX RANGE SCAN</td>
<td>DEPT_RICH_INV_IDX</td>
<td>512</td>
<td>1024</td>
<td>1 (0)</td>
<td>00:00:01</td>
</tr>
</tbody>
</table>
```

Plan hash value: 3699452051
The Invisible Index (set visible)

```
select /*+ no_index(dept_rich dept_rich_inv_idx) */ count(*)
from dept_rich
where deptno = 30; (forces not using the index with hint)
```

```
COUNT(*)
---------
  512
```

Execution Plan
```
Plan hash value: 3024595593
```
```
<table>
<thead>
<tr>
<th>Id</th>
<th>Operation</th>
<th>Name</th>
<th>Rows</th>
<th>Bytes</th>
<th>Cost (%CPU)</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SELECT STATEMENT</td>
<td></td>
<td>1</td>
<td>2</td>
<td>4 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>1</td>
<td>SORT AGGREGATE</td>
<td></td>
<td>1</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>* 2</td>
<td>TABLE ACCESS FULL</td>
<td>DEPT_RICH</td>
<td>512</td>
<td>1024</td>
<td>4 (0)</td>
<td>00:00:01</td>
</tr>
</tbody>
</table>
```
The Invisible Index (check it)

alter index dept_rich_inv_idx invisible;
Index altered.

select index_name, visibility
from dba_indexes  (or go to USER_INDEXES)
where index_name = 'DEPT_RICH_INV_IDX';

<table>
<thead>
<tr>
<th>INDEX_NAME</th>
<th>VISIBILITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEPT_RICH_INV_IDX</td>
<td>INVISIBLE</td>
</tr>
</tbody>
</table>
Multiple *Types* of Indexes on the *Same Column* *(Using the Invisible Index even more)*
Multiple Types of Indexes on the Same Column(s)

- Create MORE than one index on a column
- Set only ONE index to VISIBLE
- Ok to have ONE + any Function Based Index (exception)
- Great to use different types of indexes for *batch, query, or data warehousing at different times.*
- Some restrictions apply...for a give column(s)
  - You can not create a B-tree AND B-tree cluster index
  - You can not create a B-tree and an index-organized table (IOT)
- All indexes ARE MAINTAINED during DML
  - DML could be slow if TOO MANY indexes are created
- Great for *variable* workloads!
Multiple Types of Indexes on the Same Column(s)

Basic SELECT to DEPT Table:

SELECT * FROM DEPT;

<table>
<thead>
<tr>
<th>DEPTNO</th>
<th>DNAME</th>
<th>LOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>ACCOUNTING</td>
<td>NEW YORK</td>
</tr>
<tr>
<td>20</td>
<td>RESEARCH</td>
<td>DALLAS</td>
</tr>
<tr>
<td>30</td>
<td>SALES</td>
<td>CHICAGO</td>
</tr>
<tr>
<td>40</td>
<td>OPERATIONS</td>
<td>BOSTON</td>
</tr>
</tbody>
</table>
create unique index dept_unique1 on dept(deptno);
Index created.

insert into dept(deptno) values (10);
insert into dept(deptno) values (10)
*
ERROR at line 1:
ORA-00001: unique constraint (SYS.DEPT_UNIQUE1) violated
Multiple Types of Indexes on the Same Column(s)

Make Index Invisible… *Still* can NOT insert duplicate!

```
alter index dept_unique1 invisible;
Index altered.

SQL> insert into dept(deptno) values(10);
insert into dept(deptno) values(10)
* 
ERROR at line 1:
ORA-00001: unique constraint (SYS.DEPT_UNIQUE1) violated
```
Multiple Types of Indexes on the Same Column(s)

Check the Indexes Views - Index is Invisible

```sql
select a.table_name, a.index_name, b.column_name, a.uniqueness, a.visibility
from user_indexes a, user_ind_columns b
where a.index_name = b.index_name
and a.table_name = 'DEPT';
```

<table>
<thead>
<tr>
<th>TABLE_NAME</th>
<th>INDEX_NAME</th>
<th>COLUMN_NAME</th>
<th>UNIQUENESS</th>
<th>VISIBILITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEPT</td>
<td>DEPT_UNIQUE1</td>
<td>DEPTNO</td>
<td>UNIQUE</td>
<td>INVISIBLE</td>
</tr>
</tbody>
</table>
Multiple Types of Indexes on the Same Column(s)

Make Index Visible again:

```
alter index dept_unique1 visible;
Index altered.
```

```
select a.table_name, a.index_name, 
       b.column_name, a.uniqueness, a.visibility 
from   user_indexes a, user_ind_columns b 
where  a.index_name = b.index_name 
and    a.table_name = 'DEPT';
```

```
TABLE_NAME  INDEX_NAME     COLUMN_NAME  UNIQUENESS  VISIBILITY
----------  --------------  -----------  -----------  ----------
DEPT        DEPT_UNIQUE1   DEPTNO       UNIQUE      VISIBLE
```
Multiple Types of Indexes on the Same Column(s)

```
create index dept_normal on dept(deptno);
create index dept_normal on dept(deptno) *
ERROR at line 1:
ORA-01408: such column list already indexed

Make FIRST Index Invisible & can now create SECOND index:
alter index dept_unique1 invisible;
Index altered.

create index dept_normal on dept(deptno);
Index created.
```
Multiple Types of Indexes on the Same Column(s)

Check the Indexes Views – TWO Indexes on the same column:

```sql
select a.table_name, a.index_name, 
    b.column_name, a.uniqueness, a.visibility 
from   user_indexes a, user_ind_columns b 
where  a.index_name = b.index_name 
and    a.table_name = 'DEPT';
```

<table>
<thead>
<tr>
<th>TABLE_NAME</th>
<th>INDEX_NAME</th>
<th>COLUMN_NAME</th>
<th>UNIQUENESS</th>
<th>VISIBILITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEPT</td>
<td>DEPT_UNIQUE1</td>
<td>DEPTNO</td>
<td>UNIQUE</td>
<td>INVISIBLE</td>
</tr>
<tr>
<td>DEPT</td>
<td>DEPT_NORMAL</td>
<td>DEPTNO</td>
<td>NONUNIQUE</td>
<td>VISIBLE</td>
</tr>
</tbody>
</table>
Multiple Types of Indexes on the Same Column(s)

Try to make both Indexes Visible… *ERROR!*

```
alter index dept_unique1 visible;
```

* ERROR at line 1:
ORA-14147: There is an existing VISIBLE index defined on the same set of columns.

*Only ONE index may be visible at a time (except function-based indexes)*
Multiple Types of Indexes on the Same Column(s)

Despite a unique index that’s invisible, uses visible index only:

```sql
select deptno
from   dept
where  deptno=10;
```

```
DEPTNO
------
10
```

<table>
<thead>
<tr>
<th>Id</th>
<th>Operation</th>
<th>Name</th>
<th>Rows</th>
<th>Bytes</th>
<th>Cost (%CPU)</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SELECT STATEMENT</td>
<td></td>
<td>1</td>
<td>13</td>
<td>1 (0)</td>
<td>00:00:01</td>
</tr>
<tr>
<td>* 1</td>
<td>INDEX RANGE SCAN</td>
<td>DEPT_NORMAL</td>
<td>1</td>
<td>13</td>
<td>1 (0)</td>
<td>00:00:01</td>
</tr>
</tbody>
</table>

```sql
alter index dept_normal invisible;
```

Index altered.
Multiple Types of Indexes on the Same Column(s)

Make FIRST TWO Indexes Invisible & create THIRD index:

create index dept_reverse on dept(deptno) reverse;
Index created.

select a.table_name, a.index_name, b.column_name, a.uniqueness, a.visibility
from user_indexes a, user_ind_columns b
where a.index_name = b.index_name
and a.table_name = 'DEPT';

<table>
<thead>
<tr>
<th>TABLE_NAME</th>
<th>INDEX_NAME</th>
<th>COLUMN_NAME</th>
<th>UNIQUENESS</th>
<th>VISIBILITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEPT</td>
<td>DEPT_UNIQUE1</td>
<td>DEPTNO</td>
<td>UNIQUE</td>
<td>INVISIBLE</td>
</tr>
<tr>
<td>DEPT</td>
<td>DEPT_REVERSE</td>
<td>DEPTNO</td>
<td>NONUNIQUE</td>
<td>VISIBLE</td>
</tr>
<tr>
<td>DEPT</td>
<td>DEPT_NORMAL</td>
<td>DEPTNO</td>
<td>NONUNIQUE</td>
<td>INVISIBLE</td>
</tr>
</tbody>
</table>
Multiple Types of Indexes on the Same Column(s)

Now I create a Bitmap Index & Function-Based Index:

```
alter index dept_reverse invisible;
Index altered.
```

```
create bitmap index dept_bitmap on dept(deptno);
Index created.
```

```
create index dept_fb on dept(substr(deptno,1,1));
Index created.
```

**OK to Create TWO VISIBLE indexes if one is a Function-Based Index!**
Multiple Types of Indexes on the Same Column(s)

Check the Indexes Views – FIVE Indexes on the same column:

```sql
select a.table_name, a.index_name,
       b.column_name, a.uniqueness, a.visibility
from user_indexes a, user_ind_columns b
where a.index_name = b.index_name
and a.table_name = 'DEPT';
```

<table>
<thead>
<tr>
<th>TABLE_NAME</th>
<th>INDEX_NAME</th>
<th>COLUMN_NAME</th>
<th>UNIQUENESS</th>
<th>VISIBILITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEPT</td>
<td>DEPT_UNIQUE1</td>
<td>DEPTNO</td>
<td>UNIQUE</td>
<td>INVISIBLE</td>
</tr>
<tr>
<td>DEPT</td>
<td>DEPT_REVERSE</td>
<td>DEPTNO</td>
<td>NONUNIQUE</td>
<td>INVISIBLE</td>
</tr>
<tr>
<td>DEPT</td>
<td>DEPT_NORMAL</td>
<td>DEPTNO</td>
<td>NONUNIQUE</td>
<td>INVISIBLE</td>
</tr>
<tr>
<td>DEPT</td>
<td>DEPT_BITMAP</td>
<td>DEPTNO</td>
<td>NONUNIQUE</td>
<td>VISIBLE</td>
</tr>
<tr>
<td>DEPT</td>
<td>DEPT_FB</td>
<td>SYS_NC00004$</td>
<td>NONUNIQUE</td>
<td>VISIBLE</td>
</tr>
</tbody>
</table>

(Index types: NORMAL, NORMAL/REV, UNIQUE, BITMAP, FUNCTION-BASED NORMAL)
Multiple Types of Indexes on the Same Column(s)

Interesting note on Index Suppression:

```
alter index dept_bitmap invisible;
Index altered.

select /*+ index(dept dept_fb) */ deptno
from dept
where substr(deptno,1,1)=1;

DEPTNO
------
10

Predicate Information (identified by operation id):
-----------------------------------------------
1 - filter(TO_NUMBER(SUBSTR(TO_CHAR("DEPTNO"),1,1))=1)
```

---

```
select /*+ index(dept dept_fb) */ deptno
from dept
where substr(deptno,1,1)=1;

DEPTNO
------
10

Predicate Information (identified by operation id):
-----------------------------------------------
1 - filter(TO_NUMBER(SUBSTR(TO_CHAR("DEPTNO"),1,1))=1)
```
Multiple Types of Indexes on the Same Column(s)

Interesting note on Index Suppression:

```sql
select /*+ index(dept dept_fb) */ deptno
from dept
where substr(deptno,1,1)='1';
```

<table>
<thead>
<tr>
<th>DEPTNO</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
</tr>
</tbody>
</table>

```
<table>
<thead>
<tr>
<th>Id</th>
<th>Operation</th>
<th>Name</th>
<th>Rows</th>
<th>Bytes</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>SELECT STATEMENT</td>
<td></td>
<td>1</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>TABLE ACCESS BY INDEX ROWID BATCHED</td>
<td>DEPT</td>
<td>1</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>* 2</td>
<td>INDEX RANGE SCAN</td>
<td>DEPT_FB</td>
<td>1</td>
<td></td>
<td>1</td>
</tr>
</tbody>
</table>
```

Predicate Information (identified by operation id):

```
2 - access(SUBSTR(TO_CHAR("DEPTNO"),1,1)='1')
```
Optimizer Statistics & Other Optimizer Advances

Special Thanks: Maria Colgan, Penny Avril & Debbie Migliore
Corporate data often has correlations between different columns of a table. For example:

- A job title is correlated to the salary.
- The season affects the sold amounts of items such as swim suits sell more in the summer and snow shoes sell more in the winter.
- The make of a car and color are often used together but are not really correlated well so the filter doesn’t reduce the result set.

Optimizer has to estimate the correct cardinality

- *Will the additional column condition reduce the result set or not? Should it be used.*

Oracle calculates correlated statistics so the optimizer will make great decisions. Single column statistics and histograms are not enough!
Example

SELECT make, price, color
FROM   cars_dot_com
WHERE make = 'CORVETTE';

<table>
<thead>
<tr>
<th>Make</th>
<th>Price</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>CORVETTE</td>
<td>40,000</td>
<td>RED</td>
</tr>
<tr>
<td>CORVETTE</td>
<td>60,000</td>
<td>BLACK</td>
</tr>
<tr>
<td>CORVETTE</td>
<td>50,000</td>
<td>SILVER</td>
</tr>
</tbody>
</table>

- Three records selected.
- Single column statistics are accurate
Example, cont.

SELECT make, price, color
FROM   cars_dot_com
WHERE  make = 'CORVETTE'
AND    COLOR = 'RED';

• One record selected.
  • No correlated columns
  • Additional predicate reduces result set
  • Single column statistics are STILL sufficient
Example, cont.

Three records selected.

- Correlated columns
- Additional predicate has no effect
- Single column statistics are NOT sufficient
- Must use ‘=‘ and not < or >

```sql
SELECT make, price, color
FROM   cars_dot_com
WHERE  make = 'CORVETTE'
AND    PRICE = 50000;
```

<table>
<thead>
<tr>
<th>Make</th>
<th>Price</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>CORVETTE</td>
<td>50,000</td>
<td>RED</td>
</tr>
<tr>
<td>CORVETTE</td>
<td>50,000</td>
<td>BLACK</td>
</tr>
<tr>
<td>CORVETTE</td>
<td>50,000</td>
<td>SLIVER</td>
</tr>
<tr>
<td>CADILLAC</td>
<td>90,000</td>
<td>RED</td>
</tr>
<tr>
<td>JEEP</td>
<td>35,000</td>
<td>BLACK</td>
</tr>
<tr>
<td>JEEP</td>
<td>45,000</td>
<td>SLIVER</td>
</tr>
</tbody>
</table>
Manage New Statistics – FYI Only

EXTENDED Statistic Group

- Provides a way to collect stats on a group of columns
- Full integration into existing statistics framework
  - Automatically maintained with column statistics
  - Instantaneous and transparent benefit for any application
- Accurate cardinalities for inter-related columns
  - Multiple predicates on the same table are estimated correctly
Manage New Statistics – FYI Only After normal Statistics Creation

```sql
select column_name, num_distinct, histogram
from user_tab_col_statistics where table_name = 'CUSTOMERS';
```

<table>
<thead>
<tr>
<th>COLUMN_NAME</th>
<th>NUM_DISTINCT</th>
<th>HISTOGRAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUST_VALID</td>
<td>2</td>
<td>NONE</td>
</tr>
<tr>
<td>COUNTRY_ID</td>
<td>19</td>
<td>FREQUENCY</td>
</tr>
<tr>
<td>CUST_STATE_PROVINCE</td>
<td>145</td>
<td>NONE</td>
</tr>
<tr>
<td>CUST_CITY_ID</td>
<td>620</td>
<td>HEIGHT BALANCED</td>
</tr>
<tr>
<td>CUST_CITY</td>
<td>620</td>
<td>NONE</td>
</tr>
<tr>
<td>CUST_LAST_NAME</td>
<td>908</td>
<td>NONE</td>
</tr>
<tr>
<td>CUST_FIRST_NAME</td>
<td>1300</td>
<td>NONE</td>
</tr>
<tr>
<td>CUST_ID</td>
<td>55500</td>
<td>NONE</td>
</tr>
</tbody>
</table>

... 23 rows selected.
Now let's create the extended statistics group & re-gather statistics on the CUSTOMER table (query user_tab_col_statistics to see new column):

```sql
select dbms_stats.create_extended_stats('SH','CUSTOMERS', '(country_id, cust_state_province)') from dual;
```

DBMS_STATS.CREATE_EXTENDED_STATS('SH','CUSTOMERS','(CO

SYS_STUJGVLRVH5USVDU$XNV4_IR#4

exec dbms_stats.gather_table_stats('SH','CUSTOMERS', method_opt =>
  'for all columns size skewonly');

PL/SQL procedure successfully completed.
```
select column_name, num_distinct, histogram
from user_tab_col_statistics where table_name = 'CUSTOMERS';
```

<table>
<thead>
<tr>
<th>COLUMN_NAME</th>
<th>NUM_DISTINCT</th>
<th>HISTOGRAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYS_STUJGVRVH5USVDU$XNV4_IR#4</td>
<td>145</td>
<td>FREQUENCY</td>
</tr>
<tr>
<td>CUST_VALID</td>
<td>2</td>
<td>FREQUENCY</td>
</tr>
<tr>
<td>COUNTRY_ID</td>
<td>19</td>
<td>FREQUENCY</td>
</tr>
<tr>
<td>CUST_STATE_PROVINCE</td>
<td>145</td>
<td>FREQUENCY</td>
</tr>
<tr>
<td>CUST_CITY_ID</td>
<td>620</td>
<td>HEIGHT BALANCED</td>
</tr>
<tr>
<td>CUST_CITY</td>
<td>620</td>
<td>HEIGHT BALANCED</td>
</tr>
<tr>
<td>CUST_LAST_NAME</td>
<td>908</td>
<td>HEIGHT BALANCED</td>
</tr>
<tr>
<td>CUST_FIRST_NAME</td>
<td>1300</td>
<td>HEIGHT BALANCED</td>
</tr>
<tr>
<td>CUST_ID</td>
<td>55500</td>
<td>HEIGHT BALANCED</td>
</tr>
</tbody>
</table>

... 

24 rows selected.
exec dbms_stats.drop_extended_stats('SH', 'CUSTOMERS', '(country_id, cust_state_province)');
PL/SQL procedure successfully completed.

select column_name, num_distinct, histogram
from user_tab_col_statistics where table_name = 'CUSTOMERS';

<table>
<thead>
<tr>
<th>COLUMN_NAME</th>
<th>NUM_DISTINCT</th>
<th>HISTOGRAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUST_VALID</td>
<td>2</td>
<td>NONE</td>
</tr>
<tr>
<td>COUNTRY_ID</td>
<td>19</td>
<td>FREQUENCY</td>
</tr>
<tr>
<td>CUST_STATE_PROVINCE</td>
<td>145</td>
<td>NONE</td>
</tr>
<tr>
<td>CUST_CITY_ID</td>
<td>620</td>
<td>HEIGHT BALANCED</td>
</tr>
<tr>
<td>CUST_CITY</td>
<td>620</td>
<td>NONE</td>
</tr>
<tr>
<td>CUST_LAST_NAME</td>
<td>908</td>
<td>NONE</td>
</tr>
<tr>
<td>CUST_FIRST_NAME</td>
<td>1300</td>
<td>NONE</td>
</tr>
<tr>
<td>CUST_ID</td>
<td>55500</td>
<td>NONE</td>
</tr>
</tbody>
</table>

... 23 rows selected.
Adaptive Cursor Sharing

- The optimizer **peeks at user-defined bind values** during plan selection on the hard parse.
- Initial value of the binds determines the plan for all future binds (hopefully the first peek covers most queries)
- Same execution plan shared regardless of future bind values
- One plan is not always appropriate for all bind values for a given SQL statement
  - Where `job = 'PRESIDENT'` (use an index – only one row)
  - Where `job = 'OPERATOR'` (don’t use an index – 90% of the table)
- If Oracle “peeks” and sees the President, it will use the index. Future queries also use the index without peeking after that (bad for the OPERATOR query).
Bind Peeking – Pre-11g

• If you need to tune a query that you suspect has issues related to bind peeking, use v$sql_plan or tkprof output using different values for bind variables and compare execution plans in both cases.

• If you wish to deactivate bind peeking you can set:

  alter system set "_OPTIM_PEEK_USER_BINDS"=FALSE;

**Note:** When running tkprof "explain=username/password" argument should NOT be used. That will cause tkprof to issue an explain plan whose output could differ from the execution plan info inside the raw 10046/sql_trace file.
Consider a Telephone Company:

SELECT Ename, Empno, Job
FROM   Emp
WHERE Job = :B1
Value of B1 = ‘OPERATOR’;

• If ‘OPERATOR’ is the bind value at hard parse, most records will be selected. Execution plan will be a full table scan

• If ‘PRESIDENT’ is the bind value at hard parse, few records will be selected. Execution plan will be an index search
Adaptive Cursor Sharing

Solution:

- In 11g, Oracle uses **bind-aware cursor matching**.
- Share the plan when binds values are “equivalent”
  - Plans are marked with selectivity range
  - If current bind values fall within range they use the same plan
- Create a new plan if binds are not equivalent
  - Generating a new plan with a different selectivity range
select sql_id, peeked, executions, rows_processed, cpu_time from v$sql_cs_statistics; (using the peeked value on the 2nd+ execution)

<table>
<thead>
<tr>
<th>SQL_ID</th>
<th>P EXECUTIONS</th>
<th>ROWS_PROCESSED</th>
<th>CPU_TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>5wfj3qs71nd7m</td>
<td>Y</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>2rad83pp613m1</td>
<td>Y</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>dr78c03uv97bp</td>
<td>N</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>dr78c03uv97bp</td>
<td>N</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>dr78c03uv97bp</td>
<td>Y</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>9qv6tq9ag5b80</td>
<td>Y</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>a2k4gkh681fzx</td>
<td>Y</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>413zr99jf9h72</td>
<td>N</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>413zr99jf9h72</td>
<td>N</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>413zr99jf9h72</td>
<td>Y</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>fd69nfzww1mhm</td>
<td>Y</td>
<td>6</td>
<td>0</td>
</tr>
</tbody>
</table>
### Bind Peeking – V$SQL

```sql
select sql_id, executions, is_bind_sensitive, is_bind_aware
from v$sql;
```

<table>
<thead>
<tr>
<th>SQL_ID</th>
<th>EXECUTIONS</th>
<th>is_bind_sensitive</th>
<th>is_bind_aware</th>
</tr>
</thead>
<tbody>
<tr>
<td>9ugwm6xmvw06u</td>
<td>11</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>bdfrydpbzw07g</td>
<td>11</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>57pfs5p8xc07w</td>
<td>20</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>

- **is_bind_sensitive** – If ‘Y’, then Oracle is using multiple plans depending on bind variable.
- **is_bind_aware** – Oracle knows that the different data patterns may result depending on bind value. Oracle switches to a bind-aware cursor and may hard parse the statement.
Adaptive Optimization
Adaptive Query Optimization

- Adaptive query optimization allows optimizer to adjust execution plan at run time when additional/better information is available.
  - **Adaptive Plans**: Different Join Methods (change NL to HASH) or Parallel Distribution
  - **Adaptive Statistics**: Dynamic stats, Auto Reoptimization, and SQL Plan Directives

- Adaptive Plans does not pick the final plan until execution time based on statistics collection. Information learned at execution time is used in future executions. You’ll see the plan table output in the note section:

  ```
  Note
  --------------------------
  - this is an adaptive plan
  ```

- The 12c Adaptive Optimizer adapts plans based on not just the original tables stats, but also additional adaptive statistics

- There are three types of Adaptive Statistics:
  - **Dynamic Statistics** (previously dynamic sampling in 10g/11g) or runtime statistics
  - **Automatic Reoptimization** or statistics generated after the initial execution
  - **SQL Plan Directives** direct optimizer to dynamic statistics & gets accurate cardinality
Adaptive Query Optimization

• The Adaptive Optimizer adapts the execution plan based on stats collected at run time (a sample of stats – dynamic statistics)… it makes runtime optimizations (perhaps changing join method)
• Adaptive Query Optimization is set ON by default. To turn it OFF set:
  OPTIMIZER_ADAPTIVE_REPORTING_ONLY = TRUE
  OPTIMIZER_FEATURES_ENABLE=12.1.0.1 (or higher)
• You can also set it for a given session:
  SQL> alter session set optimizer_adaptive_reporting_only=false;
• Set it to TRUE for reporting only. You can then check to get notes about runtime optimizations, such as dynamic plans switching from NL to HASH joins.
  (Use DBMS_XPLAN.DISPLAY_CURSOR … use ADAPTIVE for format…)

SQL> sho parameter OPTIMIZER_ADAPTIVE_REPORTING_ONLY

<table>
<thead>
<tr>
<th>NAME</th>
<th>TYPE</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>optimizer_adaptive_reporting_only</td>
<td>boolean</td>
<td>FALSE</td>
</tr>
</tbody>
</table>
Adaptive Query Optimization

- Previously called **dynamic sampling** in 10g/11g, **Dynamic Statistics** was used only in absence of stats on one of the tables in a multi-table join; This is helpful when existing statistics are not sufficient.

- SQL Plan Management (SPM) builds SQL plan baselines to use a verified plan

- In 12c, **Adaptive SPM can be used by using the SPM Evolve Advisor (checks for better plans)**

- Adaptive query optimization uses runtime stats to get an adaptive plan that may be better than the default plan.

- **Automatic Reoptimization** – When actual stats (after query executes) vary greatly compared to the original plan statistics (when the original plan was created), the optimizer records the new statistics (actual vs. estimated) & applies them **next time** *(see below Note from Plan Table)*.

**Note**

- **statistics feedback used for this statement**
Adaptive Query Optimization

• Reoptimization is called performance feedback. This is improving SQL statements that are repeated & optimized over time.

• This allows the optimizer to choose PARALLEL and to set a degree. It is set to MANUAL by default. To turn it on set:

  PARALLEL_DEGREE_POLICY = ADAPTIVE

• Even if you DON’T set the parameter above, reoptimization based on statistics may influence degree of parallelism that the optimizer uses.

• Reoptimization creates SQL Plan Directives (information/instructions for optimizer to use to generate a more optimal plan next time).
  – SQL Plan Directives stored in SYSAUX tablespace & initially created in Shared Pool
  – SQL Plan Directives may be managed using the DBMS_SPD package
  – DBA_SQL_PLAN_DIRECTIVES will show directives/reasons (e.g. MISESTIMATE)

Note
----------------------------------------------------
  - dynamic sampling used for this statement (level=2)
  - 1 Sql Plan Directive used for this statement
Runaway Query Management
Runaway Query Management

- **Resource Manager now pro-actively manages problems** queries and takes action based on settings for a given consumer group when:
  - CPU is exceeded
  - Physical I/O is exceeded (disk)
  - Logical I/O is exceeded (memory)
  - Elapsed Time is exceeded

- This can be automated!

- **New views** allow the DBA to see problem queries that are over the limit for each Consumer Group (can be set to automatically be terminated or can be switched to a new group with lower resources)

- Views are persisted in the AWR

- Must have the appropriate resources to manage this

- Can be set based on start of session or start of SQL or PL/SQL:
  - `SWITCH_FOR_CALL` resource plan directive
Runaway Query Management
(Oracle 12c DBA Guide example…)

Create a Resource plan Directive that kills any session that exceeds 60 seconds of CPU time

BEGIN
DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE ( PLAN => 'DAYTIME',
GROUP_OR_SUBPLAN => 'OLTP',
COMMENT => 'OLTP group',
MGMT_P1 => 75,
SWITCH_GROUP => 'KILL_SESSION',
SWITCH_TIME => 60);
END;
/

BEGIN
DBMS_RESOURCE_MANAGER.CREATE_PLAN_DIRECTIVE ( PLAN => 'DAYTIME',
GROUP_OR_SUBPLAN => 'OLTP',
COMMENT => 'OLTP group',
MGMT_P1 => 75,
SWITCH_GROUP => 'LOW_GROUP',
SWITCH_IOREQS => 10000,
SWITCH_IOMEGABYTES => 2500,
SWITCHFOR_CALL => TRUE);
END;
/

Create a Resource plan Directive that switches sessions to the low_group if they exceed 10000 physical IO’s or 2500M of data transferred. Session returns back to original group after bad query ends
Runaway Query Management

(Oracle 12c DBA Guide example...)

Check the statistics for sessions and consumer groups:

```sql
SELECT se.sid sess_id, co.name consumer_group, se.state,
       se.consumed_cpu_time cpu_time,
       se.cpu_wait_time,
       se.queued_time
FROM v$rsrc_session_info se,
     v$rsrc_consumer_group co
WHERE se.current_consumer_group_id = co.id;
```

<table>
<thead>
<tr>
<th>SESS_ID</th>
<th>CONSUMER_GROUP</th>
<th>STATE</th>
<th>CPU_TIME</th>
<th>CPU_WAIT_TIME</th>
<th>QUEUED_TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>113</td>
<td>OLTP_ORDER_ENTRY</td>
<td>WAITING</td>
<td>137947</td>
<td>28846</td>
<td>0</td>
</tr>
<tr>
<td>135</td>
<td>OTHER_GROUPS</td>
<td>IDLE</td>
<td>785669</td>
<td>11126</td>
<td>0</td>
</tr>
<tr>
<td>124</td>
<td>OTHER_GROUPS</td>
<td>WAITING</td>
<td>50401</td>
<td>14326</td>
<td>0</td>
</tr>
<tr>
<td>114</td>
<td>SYS_GROUP</td>
<td>RUNNING</td>
<td>495</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>102</td>
<td>SYS_GROUP</td>
<td>IDLE</td>
<td>88054</td>
<td>80</td>
<td>0</td>
</tr>
<tr>
<td>147</td>
<td>DSS_QUERIES</td>
<td>WAITING</td>
<td>460910</td>
<td>512154</td>
<td>0</td>
</tr>
</tbody>
</table>
Nice DBA Tools

Change Table Compression at Import Time
&
Data Pump Export View as a Table

(Also: No redo logging option of table load/Index creation)
Change Table Compression at Import Time

- Use impdp command line option (or use DBMS_DATAPUMP)
- Use the TABLE_COMPRESSION_CLAUSE:
  
  TABLE_COMPRESSION_CLAUSE=NONE
  TABLE_COMPRESSION_CLAUSE=NOCOMPRESS
  TABLE_COMPRESSION_CLAUSE=COMPRESS BASIC
  TABLE_COMPRESSION_CLAUSE=COMPRESS ROW STORE COMPRESS ADVANCED (used for OLTP)

  Warehouse compression (low is faster load):
  TABLE_COMPRESSION_CLAUSE=COMPRESS COLUMN STORE COMPRESS FOR QUERY LOW
  TABLE_COMPRESSION_CLAUSE=COMPRESS COLUMN STORE COMPRESS FOR QUERY HIGH

  Archive compression (low is faster load):
  TABLE_COMPRESSION_CLAUSE=COMPRESS COLUMN STORE COMPRESS FOR ARCHIVE LOW
  TABLE_COMPRESSION_CLAUSE=COMPRESS COLUMN STORE COMPRESS FOR ARCHIVE HIGH

  impdp hr TABLES=hr.employees DIRECTORY=dpump_dir1 DUMPFILE=hr.dmp
  TRANSFORM=TABLE_COMPRESSION_CLAUSE=NOCOMPRESS

- This is especially helpful for Exadata migrations where more compression options (HCC) are available.
A Basic Example changing a table to COMPRESS:

$ impdp scott2/tiger TABLES=dept2
   TRANSFORM=TABLE_COMPRESSION_CLAUSE:compress:table

Master table "SCOTT2"."SYS_IMPORT_TABLE_01" successfully loaded/unloaded
Starting "SCOTT2"."SYS_IMPORT_TABLE_01": scott2/******** TABLES=dept2
   TRANSFORM=TABLE_COMPRESSION_CLAUSE:compress:table
Processing object type TABLE_EXPORT/TABLE/TABLE
Processing object type TABLE_EXPORT/TABLE/TABLE_DATA
   imported "SCOTT2"."DEPT2" 5.937 KB 4 rows
Processing object type TABLE_EXPORT/TABLE/STATISTICS/TABLE_STATISTICS
Job "SCOTT2"."SYS_IMPORT_TABLE_01" successfully completed at Sat Mar 2 03:59:51 2013 elapsed 0 00:00:12

A Basic Example creating views as tables:

$ impdp scott2/tiger VIEWS_AS_TABLES…
   VIEWS_AS_TABLES=schema.view_name:table, …
Create Views as Tables Example

Export a view as a table and then import it:

create view emp_dept as
(select a.empno, a.ename, b.deptno, b.dname, b.loc
from emp a, dept b
where a.deptno=b.deptno);
View created.

$ expdp scott2/tiger VIEWS_AS_TABLES=emp_dept

Processing object type TABLE_EXPORT/VIEWS_AS_TABLES/TABLE
.. exported "SCOTT2"."EMP_DEPT"
  7.140 KB  14 rows
Create Views as Tables Example

SQL> rename emp_dept to emp_dept_view;
$ impdp scott2/tiger VIEWS_AS_TABLES=emp_dept

Processing object type
  TABLE_EXPORT/VIEWS_AS_TABLES/TABLE_DATA
  . . imported "SCOTT2"."EMP_DEPT"
  7.140 KB    14 rows

select segment_name, segment_type
from   dba_segments
where  segment_name = 'EMP_DEPT';

SEGMENT_NAME    SEGMENT_TYPE
---------------    --------------
EMP_DEPT         TABLE
Compression History – Timeline (FYI Only)

- Index Compression since 8i
- Table Compression since 9i
  - No Additional License Requirement
  - Only for direct inserts
  - Compression Not Maintained with updates and normal inserts
  - Had to re-org table to re-compress over time.
- 11g Advanced Compression
  - Additional License Requirement
  - Compression Maintained with all DML activity
  - No re-orgs required after initial compression
- 11gR2 – Hybrid Columnar Compression (with Exadata)
- 12c – Change Table Compression at Import Time
Partitioning: (FYI Only)

- Online Move Partition – 12c
- Partial Indexes for Partitioned Table – 12C
- WHAT ELSE IS NEW IN ORACLE 12c
  - Partition Maintenance Operation on Multiple Partitions (12c fyi)
  - Interval Reference Partitioning (12c fyi) (use in parent/passes to child)
CREATE TABLE DEPT
(DEPTNO NUMBER(2),
DEPT_NAME VARCHAR2(30))

PARTITION BY RANGE(DEPTNO)
(PARTITION D1 VALUES LESS THAN (10) TABLESPACE DEPT1,
PARTITION D2 VALUES LESS THAN (20) TABLESPACE DEPT2,
PARTITION D3 VALUES LESS THAN (MAXVALUE) TABLESPACE DEPT3);

INSERT INTO DEPT VALUES (1, ‘DEPT 1’);
INSERT INTO DEPT VALUES (7, ‘DEPT 7’);
INSERT INTO DEPT VALUES (10, ‘DEPT 10’);
INSERT INTO DEPT VALUES (15, ‘DEPT 15’);
INSERT INTO DEPT VALUES (22, ‘DEPT 22’);
create table cust_sales ( 
  acct_no number(5),
  cust_name char(30),
  sale_day integer not null,
  sale_mth integer not null,
  sale_yr integer not null)
partition by range (sale_yr, sale_mth, sale_day)
  (partition cust_sales_q1 values less than (1998, 04, 01) tablespace users1,
   partition cust_sales_q2 values less than (1998, 07, 01) tablespace users2,
   partition cust_sales_q3 values less than (1998, 10, 01) tablespace users3,
   partition cust_sales_q4 values less than (1999, 01, 01) tablespace users4,
   partition cust_sales_qx values less than (maxvalue, maxvalue, maxvalue) tablespace users4);
create table cust_sales_hash (
acct_no number(5),
cust_name  char(30),
sale_day integer not null,
sale_mth  integer not null,
sale_yr  integer not null)
partition by hash (acct_no)
partitions 4
store in (users1, users2, users3, users4);
CREATE TABLE test5 (data_item INTEGER, length_of_item INTEGER, storage_type VARCHAR(30), owning_dept NUMBER, storage_date DATE) PARTITION BY RANGE (storage_date) SUBPARTITION BY HASH(data_item) SUBPARTITIONS 4 STORE IN (data_tbs1, data_tbs2, data_tbs3, data_tbs4) (PARTITION q1_1999 VALUES LESS THAN (TO_DATE('01-apr-1999', 'dd-mon-yyyy')), PARTITION q2_1999 VALUES LESS THAN (TO_DATE('01-jul-1999', 'dd-mon-yyyy')), PARTITION q3_1999 VALUES LESS THAN (TO_DATE('01-oct-1999', 'dd-mon-yyyy')) (SUBPARTITION q3_1999_s1 TABLESPACE data_tbs1, SUBPARTITION q3_1999_s2 TABLESPACE data_tbs2), PARTITION q4_1999 VALUES LESS THAN (TO_DATE('01-jan-2000', 'dd-mon-yyyy')) SUBPARTITIONS 8 STORE IN (q4_tbs1, q4_tbs2, q4_tbs3, q4_tbs4, q4_tbs5, q4_tbs6, q4_tbs7, q4_tbs8), PARTITION q1_2000 VALUES LESS THAN (TO_DATE('01-apr-2000', 'dd-mon-yyyy')));
create table dept_part
(deptno number(2),
dname varchar2(14),
loc varchar2(13))
partition by list (dname)
(partition d1_east values ('BOSTON', 'NEW YORK'),
partition d2_west values ('SAN FRANCISCO', 'LOS ANGELES'),
partition d3_south values ('ATLANTA', 'DALLAS', 'HOUSTON'),
partition d4_north values ('CHICAGO', 'DETROIT', 'ATLANTA'));

Table created.
This is a helpful addition to range partitioning where Oracle automatically creates a partition when the inserted value exceeds all other partition ranges. **11g also has Ref & Virtual Column Partitioning & Oracle 12c has Interval Ref Partitioning. (not covered here).**

There are the following restrictions:

- You can only specify one partitioning key column, and it **must be of NUMBER or DATE type.**
- Interval partitioning is **NOT supported for index-organized tables.**
- Interval Partitioning supports **composite partitioning:**
  - Interval-range *** Interval-hash *** Interval-list
- You can **NOT create a domain index on an interval-partitioned table.**
CREATE TABLE DEPT_new 
(DEPTNO NUMBER(2), 
DEPT_NAME VARCHAR2(30))  
PARTITION BY RANGE(DEPTNO)  
(PARTITION D1 VALUES LESS THAN (10),  
PARTITION D2 VALUES LESS THAN (20),  
PARTITION D3 VALUES LESS THAN (30));

Table created.

SQL> insert into dept_new values(40, 'test2');
insert into dept_new values(40, 'test2')
  *
ERROR at line 1:
ORA-14400: inserted partition key does not map to any partition
select segment_name, partition_name
from dba_segments
where segment_name = 'DEPT_NEW';

<table>
<thead>
<tr>
<th>SEGMENT_NAME</th>
<th>PARTITION_NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEPT_NEW</td>
<td>D1</td>
</tr>
<tr>
<td>DEPT_NEW</td>
<td>D2</td>
</tr>
<tr>
<td>DEPT_NEW</td>
<td>D3</td>
</tr>
</tbody>
</table>
CREATE TABLE DEPT_NEW2
(DEPTNO NUMBER(2),
 DEPT_NAME VARCHAR2(30))
PARTITION BY RANGE(DEPTNO)
INTERVAL(10)
  (PARTITION D1 VALUES LESS THAN (10),
   PARTITION D2 VALUES LESS THAN (20),
   PARTITION D3 VALUES LESS THAN (30))

Table created.
insert into dept_new2 values(40,null);
insert into dept_new2 values(50,null);
insert into dept_new2 values(99,null);

select  segment_name, partition_name
from   dba_segments
where  segment_name = 'DEPT_NEW2'

<table>
<thead>
<tr>
<th>SEGMENT_NAME</th>
<th>PARTITION_NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEPT_NEW2</td>
<td>D1</td>
</tr>
<tr>
<td>DEPT_NEW2</td>
<td>D2</td>
</tr>
<tr>
<td>DEPT_NEW2</td>
<td>D3</td>
</tr>
<tr>
<td>DEPT_NEW2</td>
<td>SYS_P41</td>
</tr>
<tr>
<td>DEPT_NEW2</td>
<td>SYS_P42</td>
</tr>
<tr>
<td>DEPT_NEW2</td>
<td>SYS_P43</td>
</tr>
</tbody>
</table>
CREATE TABLE DEPT_new3
(DEPTNO NUMBER(2),
DEPT_NAME VARCHAR2(30))
COMPRESS FOR OLTP
PARTITION BY RANGE(DEPTNO)
interval(10)
(PARTITION D1 VALUES LESS THAN (10),
PARTITION D2 VALUES LESS THAN (20) NOCOMPRESS,
PARTITION D3 VALUES LESS THAN (30));

Table created.

- **NOCOMPRESS** - The table or partition is not compressed. This is the default action
- **COMPRESS** - Suitable for data warehouse. Compression enabled during direct-path inserts only.
- **COMPRESS FOR DIRECT_LOAD OPERATIONS** - Same affect as the simple COMPRESS.
- **COMPRESS FOR ALL OPERATIONS** - Suitable for OLTP systems. Compression for all operations, including regular DML statements. Requires COMPATIBLE to be set to 11.1.0 or higher.
- **COMPRESS FOR OLTP** - Suitable for OLTP systems. Enables compression for OLTP operations, including regular DML statements. Requires COMPATIBLE to be set to 11.1.0 or higher and in 11.2 replaces the COMPRESS FOR ALL OPERATIONS Syntax, but COMPRESS FOR ALL OPERATIONS syntax still exists and is still valid.
Partition Compression

```
select table_name, partition_name, compression
from   dba_tab_partitions
where table_name = 'DEPT_NEW3';
```

<table>
<thead>
<tr>
<th>TABLE_NAME</th>
<th>PARTITION_NAME</th>
<th>COMPRESS</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEPT_NEW3</td>
<td>D1</td>
<td>ENABLED</td>
</tr>
<tr>
<td>DEPT_NEW3</td>
<td>D2</td>
<td>DISABLED</td>
</tr>
<tr>
<td>DEPT_NEW3</td>
<td>D3</td>
<td>ENABLED</td>
</tr>
<tr>
<td>DEPT_NEW3</td>
<td>SYS_P64</td>
<td>ENABLED</td>
</tr>
<tr>
<td>DEPT_NEW3</td>
<td>SYS_P65</td>
<td>ENABLED</td>
</tr>
<tr>
<td>DEPT_NEW3</td>
<td>SYS_P66</td>
<td>ENABLED</td>
</tr>
</tbody>
</table>

6 rows selected.
Partial Indexes for Partitioned Table – **NO NO NO NO...**

Create an index on a subset of the partitions of a table:

```sql
CREATE INDEX dept_index ON dept3 (deptno) LOCAL
    (PARTITION d1 TABLESPACE users,
     PARTITION d2 TABLESPACE users);
```

```
CREATE INDEX dept_index ON dept3 (deptno) LOCAL;
```

*ERROR at line 1:
ORA-14024: number of partitions of LOCAL index must equal that of the underlying table

```sql
CREATE PARTIAL INDEX dept_index ON dept3 (deptno) LOCAL
    (PARTITION d1 TABLESPACE users,
     PARTITION d2 TABLESPACE users);
```

```
CREATE PARTIAL INDEX dept_index ON dept3 (deptno) LOCAL;
```

*ERROR at line 1:
ORA-00901: invalid CREATE command
CREATE TABLE DEPT3
(DEPTNO NUMBER(2), DEPT_NAME VARCHAR2(30))
INDEXING OFF
PARTITION BY RANGE (DEPTNO)
(PARTITION D1 VALUES LESS THAN (10) indexing on,
PARTITION D2 VALUES LESS THAN (20) indexing on,
PARTITION D3 VALUES LESS THAN (MAXVALUE));

Table created.

SQL> create index dept3_partial on dept3 (dept_name)
    2 local indexing partial;

Index created.

(Local Index Partitions D1 & D2 will be usable – can create global index instead)
Online Move Partition

- You can now move partitions *real time*:
  - `ALTER TABLE MOVE PARTITION`...
- Now a non-blocking DDL!
- DML on the partition continue to run before/during/after the move!
- Global indexes are maintained as well.

```
alter table dept3 move partition d1 tablespace users;
alter table dept3 *
ERROR at line 1:
ORA-00054: resource busy and acquire with NOWAIT specified or timeout expired
```

```
alter table dept3 move partition d1 tablespace sysaux online;
Table altered.
```
Large-Scale Data Warehouses*

Feature Usage

* Oracle Survey
Pluggable Databases
(Plug into the Power of the Database!)

Thanks: Penny Avril & Byrn Liewellyn

ORA-65052: statement involves operations with different container scope
ORA-65040: operation not allowed from within a pluggable database
ORA-65017: seed pluggable database may not be dropped or altered
Start with a Pristine Oracle System and Brand New Oracle Database

- Non-CDB
- Pristine DB
- Separate PDB
- Install New DB
- Add User Data
- More Data

Keep Pristine DB Separated
Pluggable Databases are Coming!

Cloning Databases for Test, Development
Fast, flexible copy and snapshot of pluggable databases
Pluggable Databases

- **CDB** = Container Database (has Root DB & also has a seed PDB)
- **PDB** = Pluggable Database (plugged into a CDB)
- **Non-CDB** = Original type of Database (neither a CDB or PDB)
- **Why?:** Can’t consolidate 100’s of database on one machine … too many resources required when you add the SGAs up! Enter PDBs.
  - **Share:** Big Data Sources, Acquisitions, Partners, Shared Research, Governments
- Quickly create a new database (PDB) or copy existing one (PDB)
- Move existing PDBs to new platform or location or clone it (snapshot)
- Patch/Upgrade PDB by plugging it into a CDB at a later version
- Physical machine runs more PDBs old way: Easier to manage/tune
- Backup entire CDB + any number of PDBs
- New syntax for commands: PLUGGABLE DATABASE
Pluggable Databases…
Is the database a CDB or non-CDB?

```sql
SQL> SELECT NAME, CREATED, CDB, CON_ID FROM V$DATABASE;

<table>
<thead>
<tr>
<th>NAME</th>
<th>CREATED</th>
<th>CDB</th>
<th>CON_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>CDB1</td>
<td>19-FEB-12</td>
<td>YES</td>
<td>0</td>
</tr>
</tbody>
</table>
```
Pluggable Databases

- In a CDB: Only one CDB$ROOT (Root), only one PDB$SEED (Seed), plus any PDBs (up to 252 more – 253 including the seed) that you create or plug in.

- CDB Root has schemas, schema objects, data dictionary information about PDBs

- Seed database – Can’t add any objects – only to create new PDBs (clone it to create others)

- PDB – appears to users/applications as if it was a non-CDB. Accessing a PDB is like accessing a non-CDB

- PDBs are how you split applications physically
Containers 0 - 254

- Entire CDB => Container ID = 0
- Root (CDB$ROOT) => Container ID = 1
- Seed (PDB$SEED) => Container ID = 2
- PDBs => Container ID = 3 to 254

(While in PDB1):

```
SQL> SHO CON_ID CON_NAME

CON_ID   ----------------------------
3

CON_NAME --------------------------
PDB1
```

(Connect to ROOT):

```
SQL> connect / as sysdba

SQL> SHO CON_ID CON_NAME

CON_ID -----------------------------
1

CON_NAME ---------------------------
CDB$ROOT
```

(integer overflöw!)

CDB or PDB created...

- **Background Processes / SGA** *(shared by root & all PDBs)*
- **Character Set** *(shared by root & all PDB’s)*
- **Redo** *(shared by root and all PDB’s)*
- **Undo** *(shared by root and all PDB’s)*
- **Temporary Tablespace** – *can create for each PDB*
- **Time Zones** – *can be set for each PDB*
- **Initialization parameters** – *some can be set by PDB*
- **Separate SYSTEM & SYSAUX** for root & each PDB
- **Data files** *separate for root & each PDB (same block size)*
Pluggable Databases...
Query the PDBs

select name, open_mode, open_time
from v$pdb$;

<table>
<thead>
<tr>
<th>NAME</th>
<th>OPEN_MODE</th>
<th>OPEN_TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDB$SEED</td>
<td>READ ONLY</td>
<td>23-FEB-13 05.29.19.861 AM</td>
</tr>
<tr>
<td>PDB1</td>
<td>READ WRITE</td>
<td>23-FEB-13 05.29.25.846 AM</td>
</tr>
<tr>
<td>PDB_SS</td>
<td>READ WRITE</td>
<td>23-FEB-13 05.29.37.587 AM</td>
</tr>
</tbody>
</table>
Pluggable Databases

- PDB is backward compatible with pre-12c database.
- **Common users** like SYS, SYSTEM connect to the CDB and also across all PDBS in which it has privileges (you can create your own **common users** as well). Common users create/plug/unplug PDBs.
- Privileged common users can even switch CDBs
- **Local users** are only in a SINGLE PDB (dwadm, erpadm ...etc.)
- Listener, Service Name, ..etc. needed
- One CDB has same software version, Active Data Guard, RMAN Backups, Initialization parameters related to database level (character set for instance)
- **Plug or unplug a PDB into a CDB.** Plug it in to associate it with the CDB, consisting of the XML file describing the PDB and its files (database files and wallet file)
CREATE DATABASE newcdb

USER SYS IDENTIFIED BY sys_password USER SYSTEM IDENTIFIED BY system_password

LOGFILE GROUP 1 ('/u01/logs/my/redo01a.log', '/u02/logs/my/redo01b.log') SIZE 100M BLOCKSIZE 512,

GROUP 2 ('/u01/logs/my/redo02a.log', '/u02/logs/my/redo02b.log') SIZE 100M BLOCKSIZE 512,

GROUP 3 ('/u01/logs/my/redo03a.log', '/u02/logs/my/redo03b.log') SIZE 100M BLOCKSIZE 512 MAXLOGHISTORY 1

MAXLOGFILES 16 MAXLOGMEMBERS 3 MAXDATAFILES 1024 CHARACTER SET AL32UTF8 NATIONAL
CHARACTER SET AL16UTF16

EXTENT MANAGEMENT LOCAL

DATAFILE '/u01/app/oracle/oradata/newcdb/system01.dbf' SIZE 700M REUSE AUTOEXTEND ON NEXT 10240K MAXSIZE UNLIMITED

SYSAUX DATAFILE '/u01/app/oracle/oradata/newcdb/sysaux01.dbf' SIZE 550M REUSE AUTOEXTEND ON NEXT 10240K MAXSIZE UNLIMITED

DEFAULT TABLESPACE deftbs DATAFILE '/u01/app/oracle/oradata/newcdb/deftbs01.dbf' SIZE 500M REUSE AUTOEXTEND ON MAXSIZE UNLIMITED

DEFAULT TEMPORARY TABLESPACE temp01 TEMPFILE '/u01/app/oracle/oradata/newcdb/temp01.dbf' SIZE 20M REUSE AUTOEXTEND ON NEXT 640K MAXSIZE UNLIMITED

UNDO TABLESPACE undotbs1 DATAFILE '/u01/app/oracle/oradata/newcdb/undotbs01.dbf' SIZE 200M REUSE AUTOEXTEND ON NEXT 5120K MAXSIZE UNLIMITED

ENABLE PLUGGABLE DATABASE

SEED FILE_NAME_CONVERT = ('/u01/app/oracle/oradata/newcdb/', '/u01/app/oracle/oradata/pdbseed/')

SYSTEM DATAFILES SIZE 125M AUTOEXTEND ON NEXT 10M MAXSIZE UNLIMITED

SYSAUX DATAFILES SIZE 100M

USER_DATA TABLESPACE usertbs DATAFILE '/u01/app/oracle/oradata/pdbseed/usertbs01.dbf' SIZE 200M REUSE AUTOEXTEND ON MAXSIZE UNLIMITED;
Creating a PDB
Many ways to do it...

• Create a PDB by copying the seed PDB
• Create a PDB by cloning another PDB
• Create a PDB by using the XML metadata files and other files and plugging them into a CDB
• Create a PDB using a non-CDB (multiple ways)
  – Use DBMS_PDB to create an unplugged PDB
  – Create an empty PDB and use data pump to move data
  – Using GoldenGate replication to create
Create a PDB – fyi only…
(Parameters to possibly set)

- **PATH_PREFIX** = '/disk1/oracle/dbs/dwpdb/'
- This **PATH_PREFIX** clause restricts the location of files and directory object paths associated with a PDB to the /disk1/oracle/dbs/dwpdb directory.
- **FILE_NAME_CONVERT** = ('/oracle/dbs/', '/oracle/dwpdb/')
  This **FILE_NAME_CONVERT** clause generates file names for the new PDB in the /oracle/dwpdb directory using file names in the /oracle/dbs directory. This is when you want to move file location upon creation.
- **SOURCE_FILE_NAME_CONVERT** = ('/disk1/oracle/pdb1/', '/disk2/oracle/pdb1/'). This **SOURCE_FILE_NAME_CONVERT** clause uses the files in the /disk2/oracle/pdb1 directory instead of the /disk1/oracle/pdb1 directory. In this case, the XML file describing a PDB specifies the /disk1/oracle/pdb1 directory, but the PDB should use the files in the /disk2/oracle/pdb1 directory. **NONE** if location is correct.
CREATE PLUGGABLE DATABASE dwpdb ADMIN USER dwadm IDENTIFIED BY password;

CREATE PLUGGABLE DATABASE dwpdb ADMIN USER dwadm IDENTIFIED BY password ROLES=(SELECT_CATALOG_ROLE, GATHER_SYSTEM_STATISTICS);

(PDB_DBA role is also granted in addition to the above specifically granted roles.)

CREATE PLUGGABLE DATABASE dwpdb ADMIN USER dwadm IDENTIFIED BY password STORAGE (MAXSIZE 10G MAX_SHARED_TEMP_SIZE 100M) DEFAULT TABLESPACE dw DATAFILE '/disk1/oracle/dbs/dwpdb/dw1.dbf' SIZE 2G AUTOEXTEND ON PATH_PREFIX = '/disk1/oracle/dbs/dwpdb/' FILE_NAME_CONVERT = ('/disk1/oracle/dbs/pdbseed/', '/disk1/oracle/dbs/dwpdb/');
Cloning a PDB

CREATE PLUGGABLE DATABASE pdb2 FROM pdb1;
CREATE PLUGGABLE DATABASE pdb2 FROM pdb1
PATH_PREFIX = '/disk2/oracle/pdb2'
FILE_NAME_CONVERT = ('/disk1/oracle/pdb1/',
'/disk2/oracle/pdb2/');
CREATE PLUGGABLE DATABASE pdb2 FROM pdb1
FILE_NAME.Convert = ('/disk1/oracle/pdb1/',
'/disk2/oracle/pdb2/') STORAGE (MAXSIZE 2G
MAX_SHARED_TEMP_SIZE 100M);
CREATE PLUGGABLE DATABASE pdb2 FROM pdb1
pdb1@pdb1_link;
Create PDB from non-CDB (3 ways) (fyi only...)

- Use Oracle Data Pump with or without transportable tablespaces (11.2.0.3 – full transportable export). Create an empty PDB and then import into it.
- Use Oracle GoldenGate replication ... replicate from non-CDB to PDB & fail over when replication catches up with non-CDB.
- Execute DBMS_PDB.DESCRIBE on a non-CDB in Oracle Database 12c Release 1 (12.1) ... creates the .XML Metadata file. You can then use this with the database files to create a PDB (see next slide).
Use DBMS_PDB to create PDB from non-CDB (fyi only)

- Ensure non-CDB is in a transactionally-consistent state and place it in read-only mode.
- Generate an XML file (ncdb.xml) in /disk1/oracle directory:

  ```sql
  BEGIN
    DBMS_PDB.DESCRIBE(
      pdb_descr_file => '/disk1/oracle/ncdb.xml');
  END;
  /
  ```

- Shutdown the non-CDB.
- Plug in the non-CDB, Access the PDB.
- Run the noncdb_to_pdb.sql script:
  ```
  @$ORACLE_HOME/rdbms/admin/noncdb_to_pdb.sql
  ```
- Open the new PDB in read/write mode & Back up the PDB.
Unplug/Plug-in a 12.1 PDB ...

CDB with 2 PDBs

Unplug PDB

CDB with 1 PDB

Plug into a Different CDB (12.2):
Plug in an Unplugged PDB

CREATE PLUGGABLE DATABASE dwpdb USING '/disk1/usr/dwpdb.xml' NOCOPY TEMPFILEFILE REUSE;

CREATE PLUGGABLE DATABASE dwpdb USING '/disk1/usr/dwpdb.xml'
SOURCE_FILE_NAME_CONVERT = ('/disk1/oracle/dw/', '/disk2/oracle/dw/') NOCOPY
STORAGE (MAXSIZE 4G MAX_SHARED_TEMP_SIZE 100M) TEMPFILEFILE REUSE;
Unplugging & Dropping PDBs

ALTER PLUGGABLE DATABASE dwpdb
UNPLUG INTO '/oracle/data/dwpdb.xml';

DROP PLUGGABLE DATABASE dwpdb KEEP DATAFILES;

DROP PLUGGABLE DATABASE dwpdb INCLUDING DATAFILES;
Go to the **CDB target** & manage storage & objects.

Under Provisioning & Patching – Provision *(create or clone)* or Unplug Pluggable databases

You can create multiple PDBs at once

View Job Details under Procedure Activity

Go to **CDB target** *(as a common user)* and then look at PDB level … you can switch containers and refresh to look at specific PDB information.
Connecting to CDB/PDB
Using sqlplus…

- To connect to the root … must be a common user & must have create session privilege in the root.
- To connect to a PBD, must either be common user with local create session or local PDB user with create session.
- Use SQLPLUS /nolog … and then CONNECT
- Connect / as sysdba (to root, just as a non-CDB)
Moving between CDB/PDBs
Switch Containers...

SQL> ALTER SESSION SET CONTAINER=PDB1;
Session altered.

SQL> alter session set container=CDB1;
ERROR:
ORA-65011: Pluggable database does not exist

ALTER SESSION SET CONTAINER=CDB$ROOT;
Session altered.

ALTER SESSION SET CONTAINER=PDB$SEED;
Session altered.

ALTER SESSION SET CONTAINER=pdb_ss; (not case sensitive)
Session altered.
### DBA_CONTAINER_DATA

SQL> desc dba_container_data

<table>
<thead>
<tr>
<th>Name</th>
<th>Null?</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>USERNAME</td>
<td></td>
<td>VARCHAR2(128)</td>
</tr>
<tr>
<td>DEFAULT_ATTR</td>
<td></td>
<td>CHAR(1)</td>
</tr>
<tr>
<td>OWNER</td>
<td></td>
<td>VARCHAR2(128)</td>
</tr>
<tr>
<td>OBJECT_NAME</td>
<td></td>
<td>VARCHAR2(128)</td>
</tr>
<tr>
<td>ALL_CONTAINERS</td>
<td></td>
<td>VARCHAR2(1)</td>
</tr>
<tr>
<td>CONTAINER_NAME</td>
<td></td>
<td>VARCHAR2(128)</td>
</tr>
</tbody>
</table>

SELECT * FROM DBA_CONTAINER_DATA;

<table>
<thead>
<tr>
<th>USERNAME</th>
<th>D</th>
<th>OWNER</th>
<th>OBJECT_NAME</th>
<th>A</th>
<th>CONTAINER_NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM</td>
<td>Y</td>
<td></td>
<td></td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>DBSNMP</td>
<td>Y</td>
<td></td>
<td></td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>SYS</td>
<td>Y</td>
<td></td>
<td></td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>SYSBACKUP</td>
<td>Y</td>
<td></td>
<td></td>
<td>Y</td>
<td></td>
</tr>
</tbody>
</table>
Open/Close PDBs

SQL> ALTER PLUGGABLE DATABASE CLOSE IMMEDIATE;
Pluggable database altered.

SQL> ALTER PLUGGABLE DATABASE OPEN READ WRITE;
Pluggable database altered.

SQL> ALTER PLUGGABLE DATABASE CLOSE; (shutdown)
Pluggable database altered.

Alter pluggable database open upgrade; (to migrate)
Pluggable database altered.
ALTER PLUGGABLE DATABASE PDB_SS, PDB1 CLOSE; (not in CDB)
ALTER PLUGGABLE DATABASE PDB_SS, PDB1 CLOSE

* 
ERROR at line 1:
ORA-65040: operation not allowed from within a pluggable database

alter session set container=CDB$ROOT;
Session altered.

alter pluggable database ALL open read only; (from CDB)
Pluggable database altered.

ALTER PLUGGABLE DATABASE PDB_SS, PDB1 CLOSE;
Pluggable database altered.
select name, open_mode, open_time
from  v$pdb$;

<table>
<thead>
<tr>
<th>NAME</th>
<th>OPEN_MODE</th>
<th>OPEN_TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDB$SEED</td>
<td>READ ONLY</td>
<td>11-MAR-13 09.29.18.284 PM</td>
</tr>
<tr>
<td>PDB1</td>
<td>MOUNTED</td>
<td>27-MAR-13 01.19.02.666 AM</td>
</tr>
<tr>
<td>PDB_SS</td>
<td>MOUNTED</td>
<td>27-MAR-13 01.19.02.985 AM</td>
</tr>
</tbody>
</table>
ALTER PLUGGABLE DATABASE PDB_SS, PDB1 open;
Pluggable database altered.

```
select name, open_mode, open_time
from v$pdbs;
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>OPEN_MODE</th>
<th>OPEN_TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDB$SEED</td>
<td>READ ONLY</td>
<td>11-MAR-13 09.29.18.284 PM</td>
</tr>
<tr>
<td>PDB1</td>
<td>READ WRITE</td>
<td>27-MAR-13 01.26.32.905 AM</td>
</tr>
<tr>
<td>PDB_SS</td>
<td>READ WRITE</td>
<td>27-MAR-13 01.26.36.559 AM</td>
</tr>
</tbody>
</table>
Open/Close PDBs

```
alter pluggable database all except pdb1 close immediate;
Pluggable database altered.

select name, open_mode, open_time
from v$pdb$s;

NAME       OPEN_MODE  OPEN_TIME
----------  -----------  -------------------------
PDB$SEED    READ ONLY  11-MAR-13 09.29.18.284 PM
PDB1        READ WRITE  27-MAR-13 01.26.32.905 AM
PDB_SS      MOUNTED     27-MAR-13 01.29.47.225 AM

alter pluggable database pdb$seed close immediate;
alter pluggable database pdb$seed close immediate
```

ERROR at line 1:
ORA-65017: seed pluggable database may not be dropped or altered
Startup pluggable database `pdb1 open`; (read/write)
Pluggable Database opened.
(or while in pdb1 just run `STARTUP`)

Startup pluggable database `pdb1 open read only`;
Pluggable Database opened.

Startup pluggable database `pdb1 force`; (closes/opens)
Pluggable Database opened.
(or while in pdb1 just run `STARTUP FORCE`)
Careful – New commands!

SQL> SHUTDOWN PLUGGABLE DATABASE PDB1;

SP2-0717: illegal SHUTDOWN option

SQL> STARTUP

Pluggable Database opened.

SQL> SHUTDOWN (also SHUTDOWN ABORT works)

ORACLE instance shut down.

select name, open_mode, open_time
from v$pdbs;

<table>
<thead>
<tr>
<th>NAME</th>
<th>OPEN_MODE</th>
<th>OPEN_TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDB1</td>
<td>MOUNTED</td>
<td>27-Mar-13 01.50.25.345 AM</td>
</tr>
</tbody>
</table>
Query CDB before PDB1 startup…

```sql
SQL> connect / as sysdba
Connected.

select name, open_mode, open_time
from   v$pdb$s;
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>OPEN MODE</th>
<th>OPEN TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDB$SEED</td>
<td>READ ONLY</td>
<td>11-MAR-13 09.29.18.284 PM</td>
</tr>
<tr>
<td>PDB1</td>
<td>MOUNTED</td>
<td>27-MAR-13 02.00.06.536 AM</td>
</tr>
<tr>
<td>PDB_SS</td>
<td>READ WRITE</td>
<td>27-MAR-13 01.41.58.049 AM</td>
</tr>
</tbody>
</table>
When you **startup the CDB...**

```
SQL> startup
ORACLE instance started.

Total System Global Area   626327552 bytes
Fixed Size                  2276008 bytes
Variable Size              524289368 bytes
Database Buffers           92274688 bytes
Redo Buffers                7487488 bytes
Database mounted.
Database opened.

select name, open_mode, open_time
from v$pdb$s;
```

<table>
<thead>
<tr>
<th>NAME</th>
<th>OPEN_MODE</th>
<th>OPEN_TIME</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDB$SEED</td>
<td>READ ONLY</td>
<td>27-MAR-13 02.04.46.883 AM</td>
</tr>
<tr>
<td>PDB1</td>
<td>MOUNTED</td>
<td></td>
</tr>
<tr>
<td>PDB_SS</td>
<td>MOUNTED</td>
<td></td>
</tr>
</tbody>
</table>
alter pluggable database all open;
(great command!)

RMAN> alter pluggable database pdb1 close;
RMAN> restore pluggable database pdb1;
RMAN> recover pluggable database pdb1 until SCN 777070;
RMAN> alter pluggable database pdb1 open resetlogs;

srvctl add service … –pdb <pdb_name>
V$ Views…

• New views start with CDB_ are CDB only
• Dictionary/Performance views (V$) – show only PDB when queried from that PDB (isolation)
• Query performance views from root & will show all PDB’s
• PDB’s have container identifier – when you look at root… see all of the id’s
• V$SESSION & V$INSTANCE have a CON_ID column for containers (& new V$IO_OUTLIER)
SELECT d.con_ID, p.PDB_NAME, d.FILE_ID, d.TABLESPACE_NAME, d.FILE_NAME
FROM CDB_PDBS p, CDB_DATA_FILES d
WHERE p.PDB_ID(+) = d.CON_ID
order by d.con_id;

<table>
<thead>
<tr>
<th>CON_ID</th>
<th>PDB</th>
<th>FILE_ID</th>
<th>TABLESPACE_NAME</th>
<th>FILE_NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>6</td>
<td>USERS</td>
<td>/u01/app/oracle/oradata/cdb1/users01.dbf</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>4</td>
<td>UNDOTBS1</td>
<td>/u01/app/oracle/oradata/cdb1/undotbs01.dbf</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>3</td>
<td>SYSAUX</td>
<td>/u01/app/oracle/oradata/cdb1/sysaux01.dbf</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>1</td>
<td>SYSTEM</td>
<td>/u01/app/oracle/oradata/cdb1/system01.dbf</td>
</tr>
<tr>
<td>2</td>
<td>PDB$SEED</td>
<td>2</td>
<td>SYSTEM</td>
<td>/u01/app/oracle/oradata/cdb1/pdbseed/system01.dbf</td>
</tr>
<tr>
<td>2</td>
<td>PDB$SEED</td>
<td>5</td>
<td>SYSAUX</td>
<td>/u01/app/oracle/oradata/cdb1/pdbseed/sysaux01.dbf</td>
</tr>
<tr>
<td>3</td>
<td>PDB1</td>
<td>7</td>
<td>SYSTEM</td>
<td>/u01/app/oracle/oradata/cdb1/pdb1/system01.dbf</td>
</tr>
<tr>
<td>3</td>
<td>PDB1</td>
<td>8</td>
<td>SYSAUX</td>
<td>/u01/app/oracle/oradata/cdb1/pdb1/sysaux01.dbf</td>
</tr>
<tr>
<td>4</td>
<td>PDB_SS</td>
<td>9</td>
<td>SYSTEM</td>
<td>/u01/app/oracle/oradata/cdb1/pdb_ss/system01.dbf</td>
</tr>
<tr>
<td>4</td>
<td>PDB_SS</td>
<td>10</td>
<td>SYSAUX</td>
<td>/u01/app/oracle/oradata/cdb1/pdb_ss/sysaux01.dbf</td>
</tr>
<tr>
<td>4</td>
<td>PDB_SS</td>
<td>11</td>
<td>EXAMPLE</td>
<td>/u01/app/oracle/oradata/cdb1/pdb_ss/example.dbf</td>
</tr>
</tbody>
</table>

11 rows selected.
Map tables to PDBs...

```
SELECT p.PDB_ID, p.PDB_NAME, t.OWNER, t.TABLE_NAME
FROM CDB_PDBS p, CDB_TABLES t
where p.PDB_ID = t.CON_ID
AND T.OWNER = 'ORDDATA'
ORDER BY t.TABLE_NAME;
```

<table>
<thead>
<tr>
<th>PDB_ID</th>
<th>PDB_NAME</th>
<th>OWNER</th>
<th>TABLE_NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>PDB$SEED</td>
<td>ORDDATA</td>
<td>ORDDCM_ANON_ACTION_TYPES</td>
</tr>
<tr>
<td>3</td>
<td>PDB1</td>
<td>ORDDATA</td>
<td>ORDDCM_ANON_ACTION_TYPES</td>
</tr>
<tr>
<td>2</td>
<td>PDB$SEED</td>
<td>ORDDATA</td>
<td>ORDDCM_ANON_ATTRS</td>
</tr>
<tr>
<td>3</td>
<td>PDB1</td>
<td>ORDDATA</td>
<td>ORDDCM_ANON_ATTRS</td>
</tr>
<tr>
<td>3</td>
<td>PDB1</td>
<td>ORDDATA</td>
<td>ORDDCM_ANON_ATTRS_TMP</td>
</tr>
<tr>
<td>2</td>
<td>PDB$SEED</td>
<td>ORDDATA</td>
<td>ORDDCM_ANON_ATTRS_TMP</td>
</tr>
<tr>
<td>3</td>
<td>PDB1</td>
<td>ORDDATA</td>
<td>ORDDCM_ANON_ATTRS_WRK</td>
</tr>
<tr>
<td>2</td>
<td>PDB$SEED</td>
<td>ORDDATA</td>
<td>ORDDCM_ANON_ATTRS_WRK</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Sanity check - what do I have ...

```sql
select name, service_id, con_name, con_id
from v$active_services
order by 1;

<table>
<thead>
<tr>
<th>NAME</th>
<th>SERVICE_ID</th>
<th>CON_NAME</th>
<th>CON_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYS$BACKGROUND</td>
<td>1</td>
<td>CDB$ROOT</td>
<td>1</td>
</tr>
<tr>
<td>SYS$USERS</td>
<td>2</td>
<td>CDB$ROOT</td>
<td>1</td>
</tr>
<tr>
<td>cdb1</td>
<td>6</td>
<td>CDB$ROOT</td>
<td>1</td>
</tr>
<tr>
<td>cdb1XDB</td>
<td>5</td>
<td>CDB$ROOT</td>
<td>1</td>
</tr>
<tr>
<td>pdb1</td>
<td>3</td>
<td>PDB1</td>
<td>3</td>
</tr>
<tr>
<td>pdb_ss</td>
<td>3</td>
<td>PDB_SS</td>
<td>4</td>
</tr>
</tbody>
</table>

6 rows selected.
```
ALTER SYSTEM **while in PDB**

**Effect of flushing shared pool or buffer cache at different levels**
- ALTER SYSTEM FLUSH SHARED_POOL
- ALTER SYSTEM FLUSH BUFFER_CACHE
- ALTER SYSTEM SET USE_STORED_OUTLINES
- ALTER SYSTEM SUSPEND/RESUME
- ALTER SYSTEM CHECKPOINT
- ALTER SYSTEM KILL SESSION
- ALTER SYSTEM DISCONNECT SESSION
- ALTER SYSTEM SET initialization_parameter

*(Great commands to run at the PDB level)*
Able to modify initialization parameter for a given PDB...

```sql
SELECT NAME FROM V$PARAMETER
WHERE ISPDB_MODIFIABLE = 'TRUE'
AND NAME LIKE 'optim%';
```

(without condition – can set 147 parameters out of 357)
(There were 341 parameters in 11gR2)

<table>
<thead>
<tr>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>optimizer_adaptive_reporting_only</td>
</tr>
<tr>
<td>optimizer_capture_sql_plan_baselines</td>
</tr>
<tr>
<td>optimizer_dynamic_sampling</td>
</tr>
<tr>
<td>optimizer_features_enable</td>
</tr>
<tr>
<td>optimizer_index_caching</td>
</tr>
<tr>
<td>optimizer_index_cost_adj</td>
</tr>
<tr>
<td>optimizer_mode</td>
</tr>
<tr>
<td>optimizer_use_invisible_indexes</td>
</tr>
<tr>
<td>optimizer_use_pending_statistics</td>
</tr>
<tr>
<td>optimizer_use_sql_plan_baselines</td>
</tr>
</tbody>
</table>

10 rows selected.

**Key ones modifiable:** cursor_sharing, open_cursors, result_cache_mode, sort_area_size

**Key ones NOT modifiable:** shared_pool_size, db_cache_size, memory_target, pga...
Set PDB Resource Plans …

- Keep runaway PDBs from affecting other PDBs
- Allocate appropriate resource plans (between/within PDBs)
- Set min/max CPU / I/O / Parallelism / (Future: Memory / Network / I/O on non-Exadata)

```sql
alter system set RESOURCE_LIMIT = TRUE_CONTAINER = ALL
(dynamically enable resource limits for all containers)
alter system set RESOURCE_LIMIT = TRUE_CONTAINER = CURRENT
(dynamically enable resource limits for the root)
```
Set **PDB Resource Plans** …

- If 4 PDBs have 3 shares each, there are 12 shares total and each has 3/12 or 1/4\(^\text{th}\) of the CPU resources.
- If 2 PDBs have 3 shares & 2 PDBs have 1 share, then the ones with 3 shares have 3/8ths of the CPU resources and are 3x more likely to queue parallel queries than the ones that have 1 share.
- CPU utilization_limit and parallel_server_limit percents also be set.

```sql
BEGIN DBMSRESOURCEMANAGER.CREATE_CDB_PLAN_DIRECTIVE(
    plan => 'newcdb_plan',
    pluggable_database => 'pdb1',
    shares => 3,
    utilization_limit => 70,
    parallel_server_limit => 70);
END;
/
```
### Resource Plan Queries...

```sql
SELECT PLAN, STATUS, COMMENTS
FROM DBA_CDB_RSRC_PLANS
ORDER BY PLAN;
```

<table>
<thead>
<tr>
<th>PLAN</th>
<th>STATUS</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULT_CDB_PLAN</td>
<td>ACTIVE</td>
<td>Default CDB plan</td>
</tr>
<tr>
<td>ORA$INTERNAL_CDB_PLAN</td>
<td>ACTIVE</td>
<td>Internal CDB plan</td>
</tr>
</tbody>
</table>

```sql
SELECT PLAN, PLUGGABLE_DATABASE, SHARES, UTILIZATION_LIMIT, PARALLEL_SERVER_LIMIT
FROM DBA_CDB_RSRC_PLAN_DIRECTIVES
ORDER BY PLAN;
```

<table>
<thead>
<tr>
<th>Plan</th>
<th>Pluggable Database</th>
<th>Utilization Shares</th>
<th>Parallel Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEFAULT_CDB_PLAN</td>
<td>ORA$DEFAULT_PDB_DIRECTIVE</td>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>DEFAULT_CDB_PLAN</td>
<td>ORA$AUTOTASK</td>
<td>90</td>
<td>100</td>
</tr>
</tbody>
</table>
SELECT DB_NAME, CON_ID, PDB_NAME, OPERATION, OP_TIMESTAMP, CLONED_FROM_PDB_NAME
FROM CDB_PDB_HISTORY
WHERE CON_ID > 2
ORDER BY CON_ID;

Sample output:

<table>
<thead>
<tr>
<th>DB_NAME</th>
<th>CON_ID</th>
<th>PDB_NAME</th>
<th>OPERATION</th>
<th>OP_TIMESTAMP</th>
<th>CLONED_FROM_PDB_NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEWCDB</td>
<td>3</td>
<td>PDB1</td>
<td>CREATE</td>
<td>01-APR-13</td>
<td>PDB$SEED</td>
</tr>
<tr>
<td>NEWCDB</td>
<td>4</td>
<td>PDB_SS</td>
<td>CREATE</td>
<td>01-APR-13</td>
<td>PDB$SEED</td>
</tr>
<tr>
<td>NEWCDB</td>
<td>5</td>
<td>PDB2</td>
<td>CLONE</td>
<td>02-APR-13</td>
<td>PDB1</td>
</tr>
</tbody>
</table>
Get Ready for **Pluggable Databases**!

This guy and his team working hard to make your life easier!
What is your System of the Future?
12c Cloud Control
Manages 12c Database & RAC
Wait Class – Top Dimensions
By SQL ID (Click on LOAD MAP)
Replay OOW Keynote if you missed it...

Heat Map... A lot of cold data
Replay OOW Keynote if you missed it…
(compress the cold data)

New in 12c Database:
- **Heat Map** tracks modifications of rows (block level), table, partition levels
- Automate *policy-driven* data movement and compression using **Heat Map**
Nice Developer
Tools/Improvements

DDL_LOCK_TIMEOUT – 11g
Enhanced DDL Capabilities – 12c
The DDL Lock Timeout

- DDL Statements (Create/Alter/Drop) require exclusive locks and thus sometimes fail due to bad timing.
- The parameter DDL_LOCK_TIMEOUT specifies the amount of time (in seconds) the DDL statement will wait for the lock before timing out and failing.
- The default value is 0, the max value is 100000 (27.77 hours).
- Example:
  
  alter session set DDL_LOCK_TIMEOUT = 30
  Session altered.

You can specify a lock timeout in seconds for FINISH_REDEF_TABLE
Enhanced DDL Online

Many schema level DDL maintenance commands no longer have blocking locks. Less of an issue for online use while there are users using the objects. This DDL non-blocking operations include:

- **DROP INDEX ONLINE**
- **DROP CONSTRAINT ONLINE**
- **SET UNUSED COLUMN ONLINE**
- **ALTER INDEX VISIBLE**
- **ALTER INDEX INVISIBLE**
- **SET UNUSED COLUMN ONLINE**

*Can also now move a Data File while Online and is open and being accessed!*
Real Application Testing!
Database workload capture and replay
Database workload capture and replay

- Used to **capture database workload** on one system and replay later on a different system. Useful to **compare two different systems**.
- Could rival LoadRunner in the future (may be more precise!)

**Brief Steps:**
- **Capture** workload on a database even from 10gR2
- **Restore** the database on a **test system** to the SCN when capture begins
- Perform **upgrade** and make changes to the test system as needed
- **Preprocess** the captured workload if it is not preprocessed
- **Configure** the test system for replay (I don’t do this here)
- **Replay workload** on the restored database (I don’t have this in this presentation, but will show some of the screens to do it)
- Great to test upgrade to 11g (Capture 10gR2 then test against 11g)
Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client

Pre-Change (could be 9.2.0.8 or 10g Capture) Production System

Capture Workload

Processed Captured Workload

Backup

Can use Snapshot Standby as test system

Post-Change Test System

Client

Client

Client

App Server

App Server

App Server

Process

Process

Process

Process

Process

Process

Process

Client
Replay Options...

• **Synchronized Replay**
  – Exact Concurrency, commits & data divergence minimal

• **Unsynchronized Replay**
  – Not the same concurrency or commits
  – Data divergence can be large depending on load test performed

• **Creates Report – Better Reporting in**
  – Data Divergence, Error Divergence, Performance Divergence

• **NEW in 12c: Consolidated Database Replays**
  – Take multiple workloads on different databases and consolidate into a single replay (either manually with non-CDBs or with PDBs).
Automatic Diagnostic Repository (ADR)
Oracle includes a Fault Diagnosability Infrastructure to prevent, detect, diagnose, resolve issues related to bugs, corruption, etc.

When a critical error occurs it is assigned an incident number and all diagnostic data tagged with this in ADR.

ADR is a file based repository outside of the database.

ADR helps detect problems proactively.

ADR helps limit the damage of interruptions.

ADR helps reduce problem diagnostic time.

ADR simplifies Oracle Support / Customer interaction.

The ADR also contains Health Reports, Trace Files, Dump Files, SQL Test Cases and Data Repair Records.
ADR Directory Structure for a Database Instance

ADR Base
- diag
- rdbms
- `<database name>`
- `<SID>`
  - alert
  - cdump
  - incident
  - trace
  - (others)

Alert Log: /u01/app/oracle/diag/rdbms/cdb1/cdb1(trace
ORACLE_HOME: /u01/app/oracle/product/12.1.0/dbhome_1
**ADR – V$ Diagnostic Info**

**12c – No changes (that I saw)**

```sql
1* select * from V$diag_info
SYS@sillgr2> /

<table>
<thead>
<tr>
<th>INST_ID</th>
<th>NAME</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Diag Enabled</td>
<td>TRUE</td>
</tr>
<tr>
<td>1</td>
<td>ADR Base</td>
<td>/u01/app/oracle</td>
</tr>
<tr>
<td>1</td>
<td>ADR Home</td>
<td>/u01/app/oracle/diag/rdbms/cdb1/cdb1</td>
</tr>
<tr>
<td>1</td>
<td>Diag Trace</td>
<td>/u01/app/oracle/diag/rdbms/cdb1/cdb1/trace</td>
</tr>
<tr>
<td>1</td>
<td>Diag Alert</td>
<td>/u01/app/oracle/diag/rdbms/cdb1/cdb1/alert</td>
</tr>
<tr>
<td>1</td>
<td>Diag Incident</td>
<td>/u01/app/oracle/diag/rdbms/cdb1/cdb1/incident</td>
</tr>
<tr>
<td>1</td>
<td>Diag Cdump</td>
<td>/u01/app/oracle/diag/rdbms/cdb1/cdb1/cdump</td>
</tr>
<tr>
<td>1</td>
<td>Health Monitor</td>
<td>/u01/app/oracle/diag/rdbms/cdb1/cdb1/hm</td>
</tr>
<tr>
<td>1</td>
<td>Default Trace File</td>
<td>/u01/app/oracle/diag/rdbms/cdb1/cdb1/trac...</td>
</tr>
<tr>
<td>1</td>
<td>Active Problem Count</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>Active Incident Count</td>
<td>17</td>
</tr>
</tbody>
</table>

11 rows selected.
```

SQL> This is 12c Query Output above!
Security Enhancements

Enhanced security of Audit Data with new **AUDIT_ADMIN** role

- Also SYSBACKUP privilege (don’t need SYSDBA for RMAN)
- Update strong user authentication using **kerberos**
- Simplified **Vault** administration
Oracle Database Security
Built over MANY years...

Oracle Audit Vault
Oracle Database Vault
DB Security Evaluation #19
Transparent Data Encryption
EM Configuration Scanning
Fine Grained Auditing (9i)
Secure application roles
Client Identifier / Identity propagation
Proxy authentication
Enterprise User Security
Global roles
Virtual Private Database (8i)
Database Encryption API
Strong authentication (PKI, Kerberos, RADIUS)
Native Network Encryption (Oracle7)

1977 Government customer

2007
Other 12c Features …

- Database Instance Smart Flash Cache Support for Multiple Devices (can access/combine) without the overhead of the local volume manager.
- Supports In-Memory Jobs & In-Memory Temporary Tablespaces
- Active Data Guard Security has in-memory table of failed login attempts
- Heat Map that tracks modifications of rows (block level), table, partition levels
- Automate policy-driven data movement and compression using Heat Map
- Move partitions while ONLINE with DML happening
- Improved query performance against OLAP cubes (especially Exadata)
- Automatic extended stats for groups of columns accessed together
- DBMS_STATS.GATHER_TABLE_STATS run on a partitioned table when CONCURRENT is set to TRUE will gather stats using multiple jobs concurrently
- Online statistics gathered during a bulk load (similar to rebuild index command)
- Flashback Data Archive (FDA) can be fully used on HCC tables on Exadata
- Enterprise Manager Database Express 12c ships with every database (NICE!)
- “Spot ADDM” triggered by high CPU or I/O into AWR Reports
- Mask Data At Source for testing & Oracle Masking templates for E-Business
- Oracle Data Redaction (prevents things like SSN from being displayed)
Other 12c Features …

- Full **Transportable** support & Point-in-time recovery for PDBs
- **TRUNCATE TABLE …CASCADE** (truncate child tables too)
- Data Pump No Logging Option for import
- No-echo of Encryption Passwords on expdp/impdp commands
- Sql*Loader Express Mode – no control file!
- **In-Database MapReduce (Big Data)**
- Update strong user authentication using kerberos & Simplified Vault administration
- Many **Windows enhancements** (if you must use Windoze)
- Fast Application Notification (FAN) gets improved with Application Continuity which helps recover incomplete requests without executing more than once.
- **Real-Time Apply** (redo) is now default for Data Guard vs. applying archive logs
- SQL Apply Support for Objects, Collections, XML Type, & SecureFiles LOBs
- **Oracle Spacial is now Oracle Spacial & Graph** – Enhancements include routing engine enhancements, caching of index metadata, vector performance, Asian address support (geocoding), raster algebra & analytics, enhance image processing
- Many ACFS, Oracle Multimedia, Oracle Text & Oracle XML enhancements
- **VARCHAR2(32767)** – not default/4K stored inline/ >4K out of line (like a LOB)
12c Deprecated Features …fyi
(could be desupported in future releases)

- **IGNORCASE** argument of ORAPWD
- Single character options with SVRCTL (accepts full-word options now)
- *_SCHEDULER_CREDENTIALS*

(This list will certainly change in the future…)

---

180
The Future: 8 Exabytes
Look what fits in one 12c Database!

2K – A typewritten page
5M – The complete works of Shakespeare
10M – One minute of high fidelity sound
2T – Information generated on YouTube in one day
10T – 530,000,000 miles of bookshelves at the Library of Congress
20P – All hard-disk drives in 1995 (or your database in 2010)
700P – Data of 700,000 companies with Revenues less than $200M
1E – Combined Fortune 1000 company databases (average 1P each)
1E – Next 9000 world company databases (average 100T each)
8E – Capacity of ONE Oracle12c Database (CURRENT)
12E to 16E – Info generated before 1999 (memory resident in 64-bit)
16E – Addressable memory with 64-bit (CURRENT)
161E – New information in 2006 (mostly images not stored in DB)
1Z – 1000E (Zettabyte - Grains of sand on beaches -125 Oracle DBs)
100TY - 100T-Yottabytes – Addressable memory 128-bit (FUTURE)
8 Exabytes:
Look what fits in one 12c Database!

- All databases of the largest 1,000,000 companies in the world (3E).

or
- All Information generated in the world in 1999 (2E)

or
- All Information generated in the world in 2003 (5E)

or
- All Email generated in the world in 2006 (6E)

or
- 1 Mount Everest filled with Documents (approx.)
Bigger Data – Get Ready for it…

- Worldwide, data is growing rapidly*:
  - 2000: 800 Terabytes ($10^{12}$)
  - 2006: 160 Exabytes ($10^{18}$)
  - 2009: 500 Exabytes (just Internet)
  - 2012: 2.7 Zettabytes ($10^{21}$)
  - 2020: 35 Zettabytes …?

- Data generated in ONE day*….?
  - Twitter: 7 TB
  - Facebook: > 10 TB

Big data: The next frontier for innovation, competition, and productivity McKinsey Global Institute 2011

We are drowning in data, but thirsting for Information

* Data collated from various online sources
## V$ Views over the years

<table>
<thead>
<tr>
<th>Version</th>
<th>V$ Views</th>
<th>X$ Tables</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>23</td>
<td>? (35)</td>
</tr>
<tr>
<td>7</td>
<td>72</td>
<td>126</td>
</tr>
<tr>
<td>8.0</td>
<td>132</td>
<td>200</td>
</tr>
<tr>
<td>8.1</td>
<td>185</td>
<td>271</td>
</tr>
<tr>
<td>9.0</td>
<td>227</td>
<td>352</td>
</tr>
<tr>
<td>9.2</td>
<td>259</td>
<td>394</td>
</tr>
<tr>
<td>10gR1</td>
<td>340 (+31%)</td>
<td>543 (+38%)</td>
</tr>
<tr>
<td>10gR2</td>
<td>396</td>
<td>613</td>
</tr>
<tr>
<td>11gR1</td>
<td>484 (+42%)</td>
<td>798 (+46%)</td>
</tr>
<tr>
<td>11gR2</td>
<td>525 (+33%)</td>
<td>945 (+54%)</td>
</tr>
<tr>
<td>12cR1</td>
<td>606 (+25%)</td>
<td>1062 (+33%)</td>
</tr>
</tbody>
</table>
Exadata = Paradigm Shift!

Exadata 101 Presentation Wednesday 8:15-9:15: Mile High Ballroom 1C
More SPEED Coming… Get Ready…
This guy does not ever slow down!!
Exadata X-3: In-Memory Database
4 T DRAM / 22 T Flash Cache

Exadata X3 Database In-Memory Machine

- X3 mass memory hierarchy delivers extreme performance
  - Automatically moves all active data from disk to memory

- DRAM memory expanded to 2 or 4 TB for hottest data
  - Up to 40 TB of compressed user data

- Flash memory expanded 4X to 22 TB per rack
  - Up to 220 TB of compressed user data – ALL active data
  - 1.5 Million SQL random read I/Os per second for OLTP
    - Comparable to 15,000 disk drives in 150 array frames
  - 100 GB/sec SQL data scan rate for reporting and warehouses
    - Comparable to 1,000 disk drives in 10 array frames
SQL Performance Analyzer allows you to test and to analyze the effects of changes on the execution performance of SQL contained in a SQL Tuning Set.

**SQL Performance Analyzer Workflows**

Create and execute SQL Performance Analyzer Task experiments of different types using the following links:

- **Upgrade from 9i or 10.1**
  - Test and analyze the effects of database upgrade from 9i or 10.1 on SQL Tuning Set performance.
- **Upgrade from 10.2 or 11g**
  - Test and analyze the effects of database upgrade from 10.2 or 11g on SQL Tuning Set performance.
- **Parameter Change**
  - Test and compare an initialization parameter change on SQL Tuning Set performance.
- **Optimizer Statistics**
  - Test and analyze the effects of optimizer statistics changes on SQL Tuning Set performance.
- **Exadata Simulation**
  - Simulate the effects of a Exadata Storage Server installation on SQL Tuning Set performance.
- **Guided Workflow**
  - Create a SQL Performance Analyzer Task and execute custom experiments using manually created SQL trials.

<table>
<thead>
<tr>
<th>Select</th>
<th>Name</th>
<th>Owner</th>
<th>Last Modified</th>
<th>Current Step Name</th>
<th>Type</th>
<th>Last Run Status</th>
<th>SQLs Processed</th>
<th>Steps Completed</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔</td>
<td>TEST2_W_TEST</td>
<td>SYSTEM</td>
<td>May 30, 2012 7:35:03 PM</td>
<td>EXEC_52291</td>
<td>Compare</td>
<td>Completed</td>
<td>4 of 4</td>
<td></td>
</tr>
<tr>
<td>✔</td>
<td>TEST_W_TEST</td>
<td>SYSTEM</td>
<td>May 30, 2012 7:19:17 PM</td>
<td>EXEC_52276</td>
<td>Compare</td>
<td>Completed</td>
<td>4 of 4</td>
<td></td>
</tr>
<tr>
<td>✔</td>
<td>TASK_2_ABC</td>
<td>SYSTEM</td>
<td>Mar 13, 2012 11:41:49 AM</td>
<td>EXEC_51834</td>
<td>Compare</td>
<td>Completed</td>
<td>4 of 4</td>
<td></td>
</tr>
</tbody>
</table>

**TIP** For an explanation of the icons and symbols used in the following table, see the Icon Key.
Exadata Simulation

Oracle Database > SQL Performance Analyzer > SQL Performance Analyzer Task: SYSTEM.TOP3 > SQL Performance Analyzer Task Report: SYSTEM.TOP3

SQL Performance Analyzer Task Report: SYSTEM.TOP3

SQL Tuning Set Name: TOP_ACTIVITY_1317427853462
STS Owner: SYS
Total SQL Statements: 1
SQL Statements With Errors: 0

Global Statistics

Projected Workload I/O Interconnect Bytes

<table>
<thead>
<tr>
<th>I/O Interconnect Bytes</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.05</td>
</tr>
<tr>
<td>0.10</td>
</tr>
<tr>
<td>0.15</td>
</tr>
<tr>
<td>0.20</td>
</tr>
<tr>
<td>0.25</td>
</tr>
<tr>
<td>0.30</td>
</tr>
<tr>
<td>0.35</td>
</tr>
<tr>
<td>0.40</td>
</tr>
<tr>
<td>0.45</td>
</tr>
<tr>
<td>0.50</td>
</tr>
<tr>
<td>0.55</td>
</tr>
<tr>
<td>0.60</td>
</tr>
<tr>
<td>0.65</td>
</tr>
<tr>
<td>0.70</td>
</tr>
<tr>
<td>0.75</td>
</tr>
<tr>
<td>0.80</td>
</tr>
<tr>
<td>0.85</td>
</tr>
<tr>
<td>0.90</td>
</tr>
<tr>
<td>0.95</td>
</tr>
<tr>
<td>1.00</td>
</tr>
<tr>
<td>1.05</td>
</tr>
<tr>
<td>1.10</td>
</tr>
<tr>
<td>1.15</td>
</tr>
<tr>
<td>1.20</td>
</tr>
<tr>
<td>1.25</td>
</tr>
<tr>
<td>1.30</td>
</tr>
<tr>
<td>1.35</td>
</tr>
<tr>
<td>1.40</td>
</tr>
</tbody>
</table>

SQL Statement Count

<table>
<thead>
<tr>
<th>SQL Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.4</td>
</tr>
<tr>
<td>1.05</td>
</tr>
<tr>
<td>0.7</td>
</tr>
<tr>
<td>0.35</td>
</tr>
<tr>
<td>0.05</td>
</tr>
</tbody>
</table>

Change in I/O Interconnect Bytes

<table>
<thead>
<tr>
<th>Improved</th>
<th>Regressed</th>
<th>Unchanged</th>
</tr>
</thead>
<tbody>
<tr>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
</tbody>
</table>

Top 10 SQL Statements Based on Impact on Workload

<table>
<thead>
<tr>
<th>SQL ID</th>
<th>Net Impact on Workload (%)</th>
<th>I/O Interconnect Bytes</th>
<th>Net Impact on SQL (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>6mnb3c0j4h7n</td>
<td>0.000</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

TIP A '-' means that the value is not applicable.
Cloud Control 12c – Monitor Exadata

Administration
- Initialization Parameters
- Security
- Storage
- Oracle Scheduler
- Streams and Replication
- Exadata
- Migrate to ASM
- Resource Manager
- Database Feature Usage

DB Exadata System Home
DB Machine Home
Put it all together – Oracle’s picture of the X3-2

8 Compute Servers
- 8 x 2 sockets x 8 cores = 128 cores
- 2T DRAM

InfiniBand Network
- 40 Gb/sec each direction
- Fault Tolerant

14 Storage Servers
- 14x12=168 Disks
- 100T SAS or 504T SAS
- 22.41TB flash storage!
NEW X3-2 - One more time…
How they got these NUMBERS?

- 8 compute servers
  - 8 servers x 2 CPU sockets x 8 cores = 128 cores (Xeon E5-2690)
- 8 servers x 256G DRAM = 2T DRAM
- 14 Storage Servers total 336G DRAM = 2.3T+ Total DRAM
- 3 InfiniBand Switches x 36 ports = 108 ports
- 14 Storage Servers (100-504T) with Flash Cache (22.4T)
  - 400G x 4 banks = 1.6T flash cache per storage server
  - 14 storage servers x 1.6T = 22.4T Flash Cache
  - 12 disks per storage server x 14 servers = 168 disks
  - 168 disks x 600G SAS = 101T High Performance SAS
  - 168 disks x 3T SAS = 504T High Capacity SAS
  - Additional total storage of 9.6T on Database Servers (300G drives)
- 14 storage servers x 2 six core L5640 = 168 additional cores
The **X3-2** is much more than **X2-2** …

- 4x more Flash Memory
- 20x increase in Write Performance (Smart Flash Cache Write-Back – could age out in months/years)
- 33% more Data Throughput
- 10–30% more Power Savings (3KW per rack)
- 33% faster CPUs & 75% more Memory
- Same price except 6-core to 8-core software increase
- Can expand V2 or X-2 (½ or ¼ rack & add X3-2)

*** If on 11.2.3.2+ … Does not require Database/ASM / Cluster upgrade
Benefits Multiply*: Access 1/2000\textsuperscript{th} the data; It’s like getting 8P memory resident in 4T of an X3-8

10 TB of user data Requires 10 TB of IO

1 TB with compression

100 GB with partition pruning

20 GB with Storage Indexes

5 GB with Smart Scans

Sub second On Database Machine

Data is 10x Smaller, Scans are 2000x faster

*Oracle Slide – Thanks!
Oracle is never caught from behind
Oracle’s 36th Anniversary in 2013

- Great Sales/Marketing
- Great Database
- Applications Leader
- BI Leader /
- Already in the lead
- Game Over!
- Hardware/Software Engineering!
“We make a Living by what we get; We make a Life by what we give.”

Dedicated to the memory of Mark Townsend (and Rod Serling).
Summary – 12c Database

- Know the Oracle!
- Start Me Up – Using Memory Target, The Buffer Cache & The Result Cache
- Invisible Columns (12c) & virtual columns (11g)
- Multiple indexes on the same Column (12c) & Invisible Indexes (11g)
- Adaptive Execution Plans (12c) & Adaptive Cursor Sharing & Bind Peeking (11g)
- Runaway query Management (12c)
- Change Table Compression at import Time (12c) & (Partition Compression – 11g)
- Create Views as Tables (12c)
- Online Move Partition (12c) & Interval Partitioning (11g)
- Partial Indexes for Partitioned Table (12c)
- Pluggable Databases (12c)
- Enhanced DDL Online (12c)
- Exadata and Big Data (In-Database MapReduce in 12c)
- Consolidated Database Replays & Better Reporting (12c)
- Automatic Diagnostics Repository (12c)
- Security Enhancements (12c)
- Other 12c New Features
For More Information

Oracle 11g Release 2 Performance Tuning Tips & Techniques; Richard J. Niemiec; Oracle Press (Available now)

“If you are going through hell, keep going” - Churchill
#1 Selling Oracle Database Book on Amazon since it came out in February!

## Best Sellers in Oracle Databases

1. **Oracle Database 11g Release 2 Performance Tuning Tips & Techniques**  
   (Oracle Press)  
   by Richard Niemiec  
   Rating: 5 stars  
   List Price: $60.00  
   Price: $37.80  
   You Save: $22.20 (37%)  
   57 used & new from $32.35

2. **Oracle Database 11g PL/SQL Programming**  
   (Oracle Press)  
   by Michael McLaughlin  
   Rating: 4 stars  
   List Price: $59.99  
   Price: $31.91  
   You Save: $28.08 (47%)  
   78 used & new from $17.98
Amazon Most Wished For

Our products most often added to Wish Lists and Registries. Updated daily.

Most Wished For in Oracle Databases

1. Oracle Database 11g Release 2 Performance Tuning Tips & Techniques (Oracle Press)
   by Richard Niemiec
   ★★★★★ (12)
   Paperback
   List Price: $60.00
   Price: $37.80
   You Save: $22.20 (37%)
   57 used & new from $32.35

2. Oracle PL/SQL Best Practices
   by Steven Feuerstein
   ★★★★★ (10)
   Paperback
   List Price: $69.95
   Price: $57.99
   You Save: $21.96 (31%)
Database References

- Oracle 12c Beta Documentation & Beta Database
- *Oracle11g Performance Tuning Tips & Techniques; Richard J. Niemiec; Oracle Press*
- [www.tusc.com](http://www.tusc.com), [www.rolta.com](http://www.rolta.com)
- Database Secure Configuration Initiative: Enhancements with Oracle Database 11g, [www.oracle.com](http://www.oracle.com)
- All Oracle11g Documentation from Oracle Beta Site
- Introduction to Oracle Database 11g, Ken Jacobs
- Oracle Database 11g New Features, Linda Smith
- New Optimizer Features in 11g, Maria Colgan
- [www.ioug.org](http://www.ioug.org), [www.oracle.com](http://www.oracle.com), en.wikipedia.org & technet.oracle.com
- Thanks Dan M., Bob T., Brad, Joe, Heidi, Mike K., Debbie, Maria, Linda
- All companies and product names are trademarks or registered trademarks of the respective owners
Exadata References

- Exadata V2 – Sun Oracle DB Machine, Oracle
- Oracle Exadata Implementation Workshop, Oracle Corporation, McLean, Virginia - Multiple Exadata sessions
- Oracle Learning Library – multiple sessions/topics
- Oracle 11g R1/R2 Best Features, Rich Niemiec
- Oracle Enterprise Manager Deployment and High Availability Best Practices, Jim Viscusi (Oracle Corporation), Jim Bulloch (Oracle Corporation), Steve Colebrook-Taylor (Barclays Global Investors)
- Advanced Compression with Oracle Database 11g Release 2, Oracle Corporation, Steven Lu
- Tech Crunch
- Twilight Zone Series
- Rod Serling; Submitted for Your Approval, American Masters
- YouTube/oracle Oracle OpenWorld On Demand
• www.tusc.com

• Oracle9i Performance Tuning Tips & Techniques; Richard J. Niemiec; Oracle Press (May 2003)

• Oracle 10g Tuning (June 11, 2007)

“成功只访问那些没空追求它的人。”

- Henry David Thoreau
V$ View Poster – Booth 1355
Rolta—Your Partner…. Accomplished in Oracle!

2012 Oracle Excellence Award
(9 Partner of the Year / Titans / Excellence Awards)


*Won 2 Awards
How to Make a Difference in the World!
Rolta’s Oracle Services

Oracle
- E-Business Suite implementation, R12 upgrades, migration & support
- Fusion Middleware and Open Systems development
- Business Intelligence (OBIEE) development
- Hyperion Financial Performance Management
- DBA and Database tactical services
- Strategic Global Sourcing

• IT Infrastructure
  - IT Roadmap - Security & Compliance - Infrastructure Management
  - Enterprise Integration / SOA - High Availability and Disaster Planning

• Profitability & Cost Management
  - Financial Consolidation - Budgeting & Forecasting
  - Profitability & Risk Analysis - Enterprise Performance Management
  - Operational, Financial & Management Reporting

• Rolta Software Solutions
  - iPerspective™ - rapid data & systems integration
  - Geospatial Fusion™ - spatial integration & visualization
  - OneView™ - business & operational intelligence
Neither Rolta nor the author guarantee this document to be error-free. Please provide comments/questions to rich.niemiec@roltasolutions.com. I am always looking to improve!

Rich Niemiec/ Rolta © 2013. This document cannot be reproduced without expressed written consent from Rich Niemiec or an officer of Rolta TUSC, but may be reproduced or copied for presentation/conference use.

Contact Information

Rich Niemiec: rich.niemiec@roltasolutions.com
www.rolta.com
Rich’s Overview…

- Advisor to Rolta International Board
- Former President of TUSC
  - Inc. 500 Company (Fastest Growing 500 Private Companies)
  - 10 Offices in the United States (U.S.); Based in Chicago
  - Oracle Advantage Partner in Tech & Applications
- Former President Rolta TUSC & President Rolta EICT International
- Author (3 Oracle Best Sellers – #1 Oracle Tuning Book for a Decade):
  - Oracle Performing Tips & Techniques (Covers Oracle7 & 8i)
  - Oracle9i Performance Tips & Techniques
  - Oracle Database 10g Performance Tips & Techniques
- Former President of the International Oracle Users Group
- Current President of the Midwest Oracle Users Group
- Chicago Entrepreneur Hall of Fame - 1998
- E&Y Entrepreneur of the Year & National Hall of Fame - 2001
- MOUG Top Speaker Twelve Times
- National Trio Achiever award - 2006
- Oracle Certified Master & Oracle Ace Director
- Purdue Outstanding Electrical & Computer and Engineer - 2007