DOUG

DALLAS ORACLE USERS GROUP

What's New In Database 18c ... You Won't Hear About From Oracle

Daniel A. Morgan
dmorgan@siriuscom.com
+1 206-669-2949

28 June, 2018

Unsafe Harbor

- This room is an unsafe harbor
- You can rely on the information in this presentation to help you protect your data, your databases, your organization, and your career
- No one from Oracle has previewed this presentation
- No one from Oracle knows what I'm going to say
- No one from Oracle has supplied any of my materials
- Everything I will present is existing, proven, functionality

Introduction

META7™ A Division of Sirius

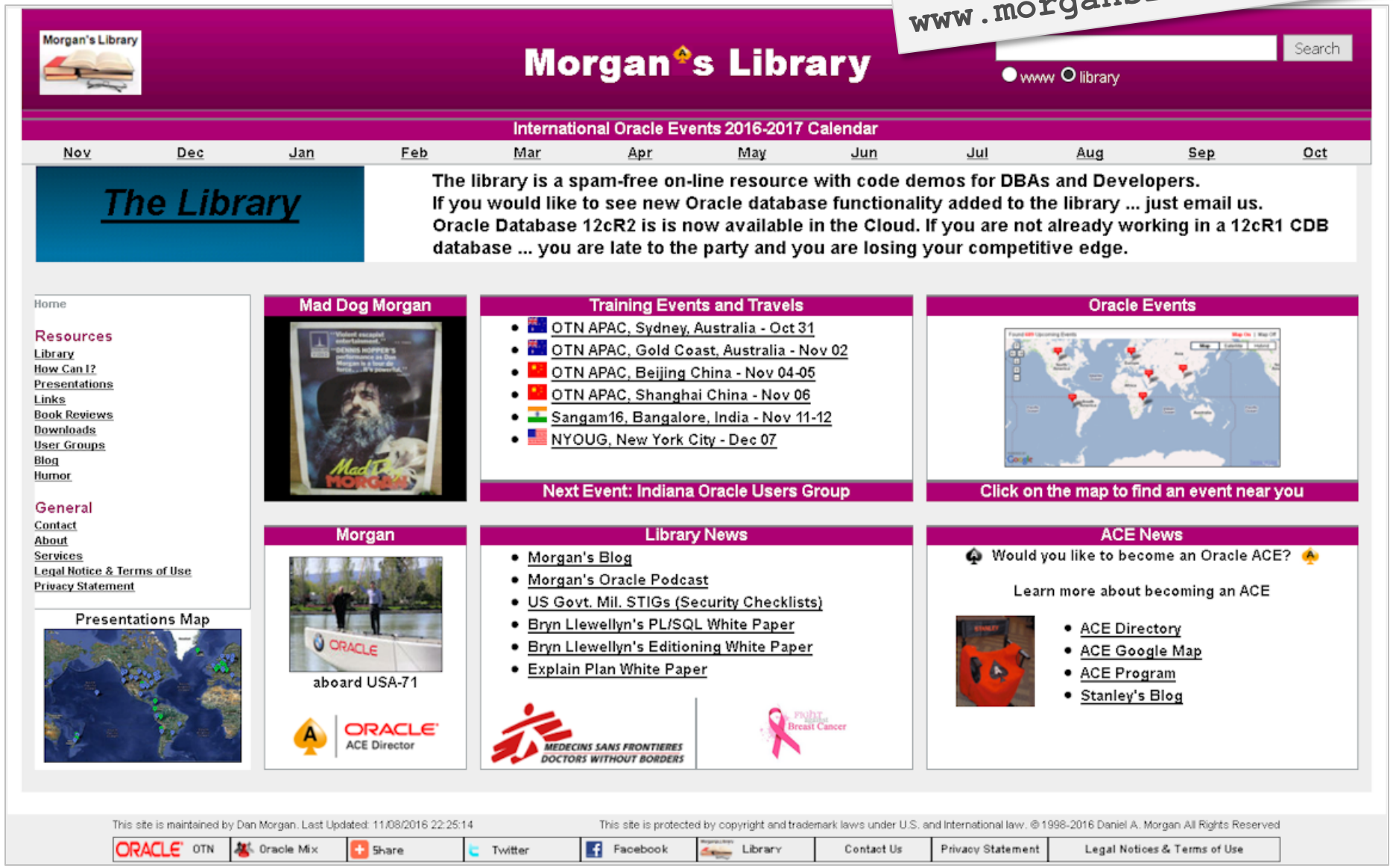
Daniel Morgan

- Oracle ACE Director Alumni
- Oracle Educator
  - Curriculum author and primary program instructor at University of Washington
  - Consultant: Harvard University
  - University Guest Lecturers
    - APAC: University of Canterbury (NZ)
    - EMEA: University of Oslo (Norway)
    - Latin America: Universidad Cenfotec, Universidad Latina de Panama, Technologico de Costa Rica
- IT Professional
  - First computer: IBM 360/40 in 1969: Fortran IV
  - Oracle Database since 1988-9 and Oracle Beta tester
  - The Morgan behind www.morganslibrary.org
  - Member Oracle Data Integration Solutions Partner Advisory Council
  - Vice President Twin Cities Oracle Users Group (Minneapolis-St. Paul)
  - Co-Founder International GoldenGate Oracle Users Group
- Principal Adviser: Sirius **Meta7**

System/370-145 system console

My Websites: Morgan's Library

www.morganslibrary.org



www.morganslibrary.org

Meta7 In Forbes Magazine

ForbesBrandVoice® What is this?

JAN 15, 2018 @ 05:00 AM 20,020

3 Essential DBA Career Priorities For 2018

OracleVoice

Simplify IT, Drive Innovation FULL BIO

Jeff Erickson, Oracle

Many database administrators (DBAs) will go into 2018 wondering if "self-driving" databases will weaken their career prospects. More likely, 2018 will be a year that database technology leaps forward and these valuable data experts take on other, more important responsibilities.

"History is repeating itself," says longtime DBA Dan Morgan, founder of Morgan's Library and principal adviser at tech firm Meta7. Morgan has seen the DBA role evolve amid a long series of technical advances in storage, management, and performance. And each advance asked DBAs to adjust the way they work.

Who We Are

Maggie Mello

- Sirius Computer Solutions
  - National integrator of technology-based business solutions that span the enterprise
  - Built on products and services from the world's top technology companies
  - Second largest security integrator in North America
  - More than 2000 employees and more than $4B in annual sales
  - Providing business and Data Center Solutions from the Brightest Minds In The Business

- Meta7
  - Sirius' Oracle and Data focused division
  - Home of subject matter expertise from silicon up tech the stack to Data Integration
  - Industry expertise in aerospace, banking, finance, health care, insurance, manufacturing, retail, and telecom
  - Focused on solving business problems for and with our customers

Meta7 is a Division of Sirius

ORACLE Platinum Partner

Forbes AMERICA'S BEST MANAGEMENT CONSULTING FIRMS 2016

Forbes | 2017 AMERICA'S BEST MANAGEMENT CONSULTING FIRMS Powered by Statista

7th straight year CRN Top 50 Providers

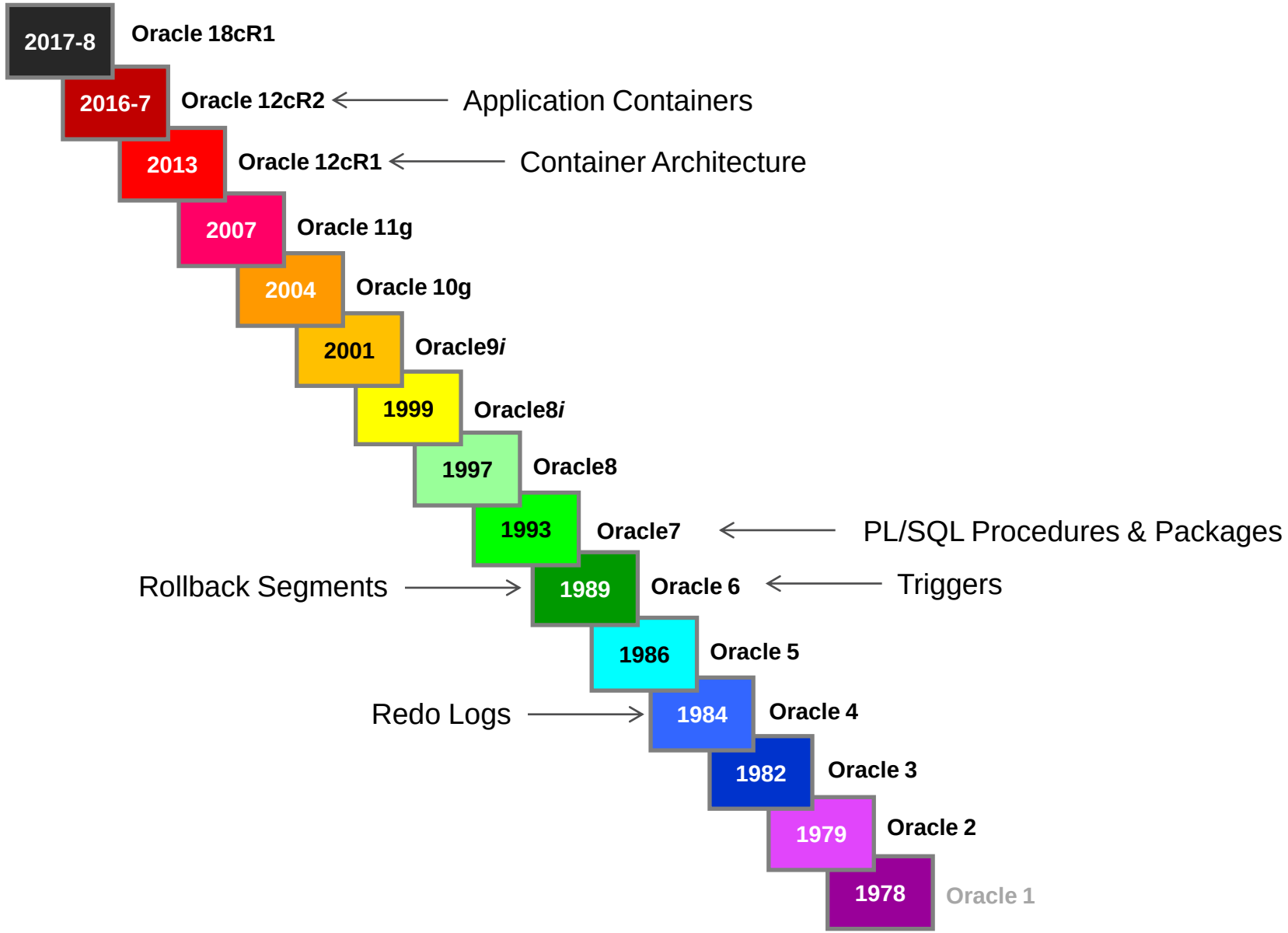
THE CHANNEL CO. CRN SOLUTION PROVIDER 500 2017

THE CHANNEL CO. CRN TECH ELITE 250 2017

THE CHANNEL CO. CRN MSP 500 2017

- Sirius acquired Forsythe Nov 1, 2017
- Combined we are a $3.5B consultancy and VAR
- World's largest IBM integrator
- Second largest security integrator in North America
- Our focus areas
  - Silicon up through Data Integration
  - Stability
  - Security
  - Scalability

A Brief History of the Oracle Database



Installation (1:2)

```
[opc@oem13c2-demo-db18c ~]$ sudo su - oracle
[oracle@oem13c2-demo-db18c ~]$ pwd
/home/oracle
[oracle@oem13c2-demo-db18c ~]$ ls -al
total 3372948
drwx------  8 oracle oinstall       4096 May 15 01:00 .
drwxr-xr-x. 5 root   root           4096 May  1 16:24 ..
-rw-r--r--  1 oracle oinstall        181 May 14 23:10 afiedt.buf
-rw-------  1 oracle oinstall      13202 May 16 02:49 .bash_history
-rw-r--r--  1 oracle oinstall         18 Mar 22  2017 .bash_logout
-rw-r--r--  1 oracle oinstall        175 May  9 18:02 .bash_profile
-rw-r--r--  1 oracle oinstall       1383 May  9 19:42 .bashrc
-rw-r--r--  1 root   root           135 Mar 13 18:23 .bashrc2018-03-13_18:23:24
-rw-r--r--  1 root   root           207 Mar 13 18:24 .bashrc2018-03-13_18:24:49
-rw-r--r--  1 root   root           788 Mar 13 18:42 .bashrc2018-03-13_18:42:36
drwxr-xr-x  3 root   root          4096 Mar 13 18:33 bkup
drwxr-xr-x  7 root   root          4096 Jan 26  2017 database
-rw-r-----  1 oracle oinstall      26662 May 12 18:37 dbca_122_container.rsp
-rw-r-----  1 oracle oinstall      26577 May  8 16:50 dbca_noncontainer.rsp
-rw-r--r--  1 root   root          5500 Mar 13 18:44 dbsetup.out.2872
-rwxr-xr-x  1 oracle oinstall      14204 Jan 24 00:43 dbsetup.sh
-rw-r--r--  1 oracle oinstall       4657 May  6 00:29 initparams.txt
-rwxr-xr-x  1 root   root          2892 Jan 24 00:43 dinject-sshkeys.sh
-rw-r--r--  1 oracle oinstall        171 Nov 15 18:39 .kshrc
-rw-rw-r--  1 oracle oinstall 3453696911 May  9 16:15 linuxx64_12201_database.zip
drwxr-xr-x  4 oracle oinstall       4096 Jan  9 22:32 .mozilla
drwxr-xr-x  2 oracle oinstall       4096 May  9 18:15 .oracle_jre_usage
drwx------  2 oracle oinstall       4096 Mar 13 18:19 .ssh
drwxr-xr-x  2 oracle oinstall       4096 Mar 13 18:19 tmp
-rw-------  1 oracle oinstall      10376 May 15 01:00 .viminfo
-rw-------  1 oracle oinstall         64 May 12 18:32 .Xauthority
```

Installation (2:2)

```
[oracle@oem13c2-demo-db18c ~]$ cd database
[oracle@oem13c2-demo-db18c database]$ ls -al
total 44
drwxr-xr-x  7 root   root     4096 Jan 26  2017 .
drwx------  8 oracle oinstall 4096 May 15 01:00 ..
drwxr-xr-x  4 root   root     4096 Jan 26  2017 install
drwxrwxr-x  2 root   root     4096 Jan 26  2017 response
drwxr-xr-x  2 root   root     4096 Jan 26  2017 rpm
-rwxr-xr-x  1 root   root     8771 Jan 26  2017 runInstaller
drwxrwxr-x  2 root   root     4096 Jan 26  2017 sshsetup
drwxr-xr-x 14 root   root     4096 Jan 26  2017 stage
-rwxr-xr-x  1 root   root      500 Feb  6  2013 welcome.html
[oracle@oem13c2-demo-db18c database]$
```

Read Only Oracle Home (1:4)

- This is one of the most important new Oracle 18c features and is a game changer with respect to how database software is installed
- It is something that was needed for decades for security and now has appeared to satisfy the requirements of the Oracle Cloud and Docker
- Docker containers are read-only ... so how can you deploy an Oracle Database in a Docker container if every ALTER SYSTEM that alers the spfile is non-persistent?
  - Of course you can create symbolic links to the spfile, to sqlnet.ora, listener.ora, tnsnames.ora, the password file, etc.
  - But it is incredibly clumsy
- The Cloud also benefits from a read only home when looked at from the standpoint of Oracle wanting to make claims for security and high availability in the Oracle Cloud

Read Only Oracle Home (2:4)

Read Only Oracle Home (3:4)

- $ORACLE_HOME/bin
  - executable: roohctl

```
[oracle@oem13c2-demo-db18c bin]$ pwd
/u01/app/oracle/product/18.0.0/dbhome_1/bin
[oracle@oem13c2-demo-db18c bin]$ ls -al *ctl
-rwxr-x--x 1 oracle oinstall 234586 Mar 13 18:23 agtctl
-rwxr-x--x 1 oracle oinstall   1578 Feb  8 08:45 aqxmlctl
-rwxr-x--x 1 oracle oinstall 254444 Mar 13 18:24 wdrdactl
-rwxr-x--x 1 oracle oinstall 178299 Mar 13 18:24 lsnrctl
-rwxr-x--- 1 oracle oinstall  35759 Feb  7 18:55 mtactl
-rwxr-xr-x 1 oracle oinstall  33548 Nov 27 08:12 wolfsctl
-rwxr-xr-x 1 oracle oinstall  14558 Feb  8 08:45 onsctl
-rwxr-xr-x 1 oracle oinstall   5440 Nov 23 06:18 wrhpctl
-rwxr-x--- 1 oracle oinstall   4631 Feb  8 08:45 roohctl
-rwxr-xr-x 1 oracle oinstall  11460 Feb  8 08:45 srvctl
[oracle@oem13c2-demo-db18c bin]$
```


Read Only Oracle Home (4:4)

```
[oracle@oem13c2-demo-db18c bin]$ roohctl -help
Usage:  roohctl [<flag>] [<command> <option>]
Following are the possible flags:
        -help

Following are the possible commands:
        -enable Enable Read-only Oracle Home
                [-nodeList List of nodes in a cluster environment]
```

- Now we have 4 different environment variables to improve our deployments
- ORACLE_BASE
  - Used to externalize the mutable files outside of the ORACLE_HOME directory structure
- ORACLE_HOME
  - The name provided when installing using OUI and DBCA ... findable in the Inventory
- ORACLEBASE_HOME
  - Mutable SQL*NET config, log and trace files and the /assistant DBCA templates
- ORACLEBASE_CONFIG
  - Mutable configuration files (.ora and .dat usually found in /dbs)

Users

New: 12cR2

APEX_050100
APEX_INSTANCE_ADMIN_USER
APEX_LISTENER
APEX_REST_PUBLIC_USER
DBJSON
DBSFWUSER
GGSYS
HRREST
OBE
ORDS_METADATA
ORDS_PUBLIC_USER
REMOTE_SCHEDULER_AGENT
RESTFUL
SYS$UMF
SYSRAC
XDBEXT
XDBPM
XFILES

Dropped:

SPATIAL_WFS_USR

New: 18cR1

None

Dropped:

SPATIAL_CSW_ADMIN_USR

Roles

New: 18cR1

None

Dropped:

JAVA_DEPLOY
SPATIAL_CSW_ADMIN
XFILES_ADMINISTRATOR
XFILES_USER

System Privileges

New: 18cR1

READ ANY ANALYTIC VIEW CACHE
TEXT DATASTORE ACCESS
WRITE ANY ANALYTIC VIEW CACHE

Dropped:

EXEMPT DDL REDACTION POLICY
EXEMPT DML REDACTION POLICY

Initialization Parameters (1:3)

New: 18cR1

ADG_ACCOUNT_INFO_TRACKING
AWR_PDB_MAX_PARALLEL_SLAVES
DBFIPS_140
FORWARD_LISTENER
INMEMORY_AUTOMATIC_LEVEL
INMEMORY_OPTIMIZED_ARITHMETIC
INMEMORY_PREFER_XMEM_MEMCOMPRESS
INMEMORY_PREFER_XMEM_PRIORITY
INMEMORY_XMEM_SIZE
MEMOPTIMIZE_POOL_SIZE
MULTISHARD_QUERY_DATA_CONSISTENCY
MULTISHARD_QUERY_PARTIAL_RESULTS
OPTIMIZER_IGNORE_HINTS
OPTIMIZER_IGNORE_PARALLEL_HINTS
PARALLEL_MIN_DEGREE
PDB_TEMPLATE
PRIVATE_TEMP_TABLE_PREFIX
RESOURCE_MANAGER_CPU_ALLOCATION
STANDBY_PDB_SOURCE_FILE_DBLINK
STANDBY_PDB_SOURCE_FILE_DIRECTORY
TDE_CONFIGURATION
UNIFIED_AUDIT_SYSTEMLOG
WALLET_ROOT

Changed Values: 18cR1

DB_BLOCK_CHECKING

Desupported / Changed Values: 18cR1

DBA_REGISTERED_MVIEW_GROUPS

GLOBAL_CONTEXT_POOL_SIZE
MAX_ENABLED_ROLES
OPTIMIZER_ADAPTIVE_FEATURES
PARALLEL_AUTOMATIC_TUNING
PARALLEL_IO_CAP_ENABLED
PARALLEL_SERVER
PARALLEL_SERVER_INSTANCES
STANDBY_ARCHIVE_DEST
USE_INDIRECT_DATA_BUFFERS
UTL_FILE_DIR

Initialization Parameters (2:3)

- ADG_ACCOUNT_INFO_TRACKING
  - Controls login attempts of users on Active Data Guard Standby databases by extending the control of user account security information
- AWR_PDB_MAX_PARALLEL_SLAVES
  - Enables a DBA to allocate the correct amount of resources to enable quick and timely Automatic Workload Repository (AWR) flushes
- for multitenant container databases (CDBs).
- DBFIPS_140
  - Enables Transparent Data Encryption (TDE) and DBMS_CRYPTO PL/SQL package program units to run in a mode compliant to the Federal Information Processing Standard (known as "FIPS mode")
- FORWARD_LISTENER
  - Specifies the name of a listener to which a connection must be forwarded by an existing set of remote listeners
- MEMOPTIMIZE_POOL_SIZE
  - Sets the size of the Memoptimized Rowstore in the SGA

Initialization Parameters (3:3)

- OPTIMIZER_IGNORE_HINTS
  - Causes the optimizer to ignore all hints ... this should be the default setting in most databases
- OPTIMIZER_IGNORE_PARALLEL_HINTS
  - Causes the optimizer to ignore all parallel hints ... this should be the default setting in all databases
- PRIVATE_TEMP_TABLE_PREFIX
  - Specifies the prefix that the database uses for private temporary tables
- TDE_CONFIGURATION
  - Used for per-PDB configuration for Transparent Data Encryption (TDE)
  - Before 18c, each PDB stored their separate encryption keys in the CDB's keystore (united mode)
  - Starting with Oracle Database 18c, a PDB can optionally store its encryption keys in a separate keystore (isolation mode) allowing protection by a separate keystore password
  - The WALLET_ROOT initialization parameter must be set for TDE_CONFIGURATION to take effect

Dropped Built-In PL/SQL Packages

- All of Streams Change Data Capture (CDC)
  - DBMS_CDC_EXPDP
  - DBMS_CDC_EXPVDP
  - DBMS_CDC_IMPDP
  - DBMS_CDC_IMPDPV
  - DBMS_CDC_IPUBLISH
  - DBMS_CDC_ISUBSCRIBE
  - DBMS_CDC_PUBLISH
  - DBMS_CDC_SUBSCRIBE
  - DBMS_CDC_SYS_IPUBLISH
  - DBMS_CDC_DPUTIL
  - DBMS_CDC_UTILITY
- DBMS_XMLQUERY
- DBMS_XMLSAVE
- Oracle Multimedia and DICOM

Temporary Tables (1:3)

- Global Temporary Tables are persistent tables defined in the data dictionary but created in the temporary tablespace

```
CREATE GLOBAL TEMPORARY TABLE gtt_zip (
zip_code      VARCHAR2(5),
by_user       VARCHAR2(30),
entry_date    DATE)
ON COMMIT DELETE ROWS;
```

```
CREATE GLOBAL TEMPORARY TABLE gtt_zip3 (
zip_code      VARCHAR2(5),
by_user       VARCHAR2(30),
entry_date    DATE)
ON COMMIT PRESERVE ROWS;
```

- Private Temporary Tables have similar characteristics but are created in memory

```
CREATE PRIVATE TEMPORARY TABLE ora$ptt_ocdr(
rid   NUMBER(10),
rname VARCHAR2(20))
ON COMMIT PRESERVE DEFINITION
ON COMMIT DELETE ROWS AS
SELECT * FROM servers;
```

```
CREATE PRIVATE TEMPORARY TABLE uwclass.ora$ptt_ocpr(
ON COMMIT DROP DEFINITION
ON COMMIT PRESERVE ROWS AS
SELECT * FROM uwclass.servers;
```

Temporary Tables (2:3)

- The Oracle docs are incomplete about PTTs so keep the following in mind when use them
  - A PTT's name must be prefixed with the parameter string value for "private_temp_table_prefix". If you don't like the Oracle Corp default, and I don't (too many bytes) change it

```
SQL> show parameter private

NAME                           TYPE    VALUE
------------------------------ ------- ---------
private_temp_table_prefix      string  ORA$PTT_
```

  - You cannot create a PTT as SYS and possibly with other privileged accounts. If you try to do so the error message you get will be misleading: Ignore it and move to a non-privileged schema.

```
SQL> sho user
USER is "SYS"

SQL> CREATE PRIVATE TEMPORARY TABLE ora$ptt_msg_fail
  2  ON COMMIT PRESERVE DEFINITION
  3  ON COMMIT DELETE ROWS AS
  4* SELECT * FROM user_objects;
ON COMMIT DELETE ROWS AS
*
ERROR at line 3:
ORA-00922: missing or invalid option
```

Temporary Tables (3:3)

- All DDL contains 2 implicit commits
- If you create a Temporary Table with ON COMMIT DELETE ROWS ... any DDL will empty the temporary table(s)

Built-In Functions (1:3)

- APPROX_COUNT
  - Returns the approximate count of an expression. With MAX_ERROR the function returns the maximum error between the actual and approximate count.

```
APPROX_COUNT(<expression> [, 'MAX_ERROR']) RETURN NUMBER;

SELECT department_id, job_id, APPROX_COUNT(*)
FROM employees
GROUP BY department_id, job_id
HAVING APPROX_RANK (PARTITION BY department_id
ORDER BY APPROX_COUNT(*) DESC) <= 10;
```

- APPROX_RANK
  - Returns the approximate rank from an optional PARTITION BY clause followed by a mandatory ORDER BY ... DESC clause. The PARTITION BY key must be a subset of the GROUP BY key. The ORDER BY clause must include either APPROX_COUNT or APPROX_SUM.

```
APPROX_MEDIAN(<expression> [PARTITION BY <partition_by_clause> [ORDER BY <order_by_clause> DESC])

SELECT department_id, job_id, APPROX_COUNT(*)
FROM employees
GROUP BY department_id, job_id
HAVING APPROX_RANK (PARTITION BY department_id ORDER BY APPROX_COUNT(*) DESC) <= 10;
```

Built-In Functions (2:3)

- APPROX_SUM
  - Returns the approximate sum of an expression. If you supply MAX_ERROR as the second argument, then the function returns the maximum error between the actual and approximate sum. You must use this function with a corresponding APPROX_RANK function in the HAVING clause. If a query uses APPROX_COUNT, APPROX_SUM, or APPROX_RANK, then the query must not use any other aggregation functions.

```
APPROX_COUNT(<expression> [, 'MAX_ERROR']) RETURN NUMBER;

SELECT department_id, job_id, APPROX_SUM(salary)
FROM employees
GROUP BY department_id, job_id
HAVING APPROX_RANK (PARTITION BY department_id
ORDER BY APPROX_SUM(salary) DESC) <= 10;
```

Built-In Functions (3:3)

- ROUND_TIED_TO_EVEN
  - Returns n rounded to integer places according to the following rules:
    - 1. If integer is positive, n is rounded to integer places to the right of the decimal point
    - 2. If integer is not specified, then n is rounded to 0 places
    - 3. If integer is negative, then n is rounded to integer places to the left of the decimal point

```
ROUND_TIES_TO_EVEN(n [, INTEGER DESC])

SQL> SELECT round_ties_to_even(0.05, 1) "ROUND_EVEN"
  2  FROM dual;

ROUND_EVEN
----------
         0

SQL> SELECT round_ties_to_even(41.572,-1) "ROUND_EVEN"
  2  FROM dual;

ROUND_EVEN
----------
        40
```

```
SQL> SELECT round_ties_to_even(41.572,1) "ROUND_EVEN"
  2  FROM dual;

ROUND_EVEN
----------
      41.6

SQL> SELECT round_ties_to_even(41.572,2) "ROUND_EVEN"
  2  FROM dual;

ROUND_EVEN
----------
     41.57
```

Polymorphic Table Functions (1:4)

- PTFs are a new type of table function, a function that returns a collection of rows, whose return type is determined by the arguments passed into the PTF
- The new PTFs provides an efficient and scalable framework to extend the analytical capabilities of the Oracle Database
- A query writer is able to call these functions without knowing the details of the implementation and the PTF doesn't need to know about the details or how the function is being executed or whether the input rows are partitioned or ordered
- PTFs are useful when SQL developers and database administrators want to provide generic extensions which work for arbitrary input tables or queries
- Making possible queries like this producing JSON as output

```
SELECT * FROM to_doc(scott.dept)

{"DEPTNO":10, "DNAME":"ACCOUNTING", "LOC":"NEW YORK"}
{"DEPTNO":20, "DNAME":"RESEARCH", "LOC":"DALLAS"}
{"DEPTNO":30, "DNAME":"SALES", "LOC":"CHICAGO"}
{"DEPTNO":40, "DNAME":"OPERATIONS", "LOC":"BOSTON"}
```

Polymorphic Table Functions (2:4)

- The DBMS_TF package was initially released in version 12.2 and is now extended with new capabilities in 18.1
  - Contains types, constants, and subprograms that can be used by Polymorphic Table Functions (PTFs)
  - Provides server and client services to get rows from the database and send back new rows

12.2

```
GET_COL
GET_ENV
GET_XID
GET_ROW_SET
PUT_COL
PUT_ROW_SET
SUPPORTED_TYPE
TRACE
```

18.1

```
COLUMN_TYPE_NAME
COL_TO_CHAR
CSTORE_EXISTS
CSTORE_GET
ROW_REPLICATION
ROW_TO_CHAR
XSTORE_CLEAR
XSTORE_EXISTS
XSTORE_GET
XSTORE_REMOVE
XSTORE_SET
```

Polymorphic Table Functions (3:4)

- Examples from the web of dbms_tf usage

```
CREATE PACKAGE to_doc_p AS
 FUNCTION desc(tab IN OUT dbms_tf.table_t,cols IN dbms_tf.columns_t DEFAULT NULL) RETURN dbms_tf.describe_t;
END to_doc_p;
```

```
CREATE PACKAGE BODY to_doc_p AS
 FUNCTION desc(tab IN OUT dbms_tf.table_t,cols IN dbms_tf.columns_t DEFAULT NULL) RETURN dbms_tf.describe_t AS
 BEGIN
   FOR i IN 1 .. tab.column.COUNT LOOP
     CONTINUE WHEN NOT DBMS_TF.supported_type(tab.column(i).description.TYPE);
     IF cols IS NULL THEN
       tab.column(i).for_read := TRUE;
       tab.column(i).pass_through := FALSE;
       CONTINUE;
     END IF;
     FOR j IN 1 .. cols.COUNT LOOP
       IF (tab.column(i).description.name = cols(j)) THEN
         tab.column(i).for_read     := TRUE;
         tab.column(i).pass_through := FALSE;
       END IF;
     END LOOP;
   END LOOP;
   RETURN dbms_tf.describe_t(new_columns => dbms_tf.columns_new_t(1 =>
          dbms_tf.column_metadata_t(name =>'DOCUMENT')));
 END;
END;
```

Polymorphic Table Functions (4:4)

- Examples from the web of dbms_tf usage

```
dbms_tf.get_col(
ColumnId   IN            NUMBER,
Collection IN OUT NOCOPY "<V2_TABLE_1>");
pragma interface(c, Get_Col);
```

```
CREATE OR REPLACE PROCEDURE fetch_rows AUTHID CURRENT_USER IS
 col1 dbms_tf.tab_clob_t;
 col2 dbms_tf.tab_colb_t;
 out1 dbms_tf.tab_clob_t;
 out2 dbms_tf.tab_clob_t;
BEGIN
  dbms_tf.get_col(1, col1);
  dbms_tf.get_col(2, col2);

  FOR i IN 1 .. col1.COUNT LOOP
    out1(i) := 'ECHO-' || col1(i);
  END LOOP;

  FOR i IN 1 .. col2.COUNT LOOP
    out2(i) := 'ECHO-' || col2(i);
  END LOOP;

  dbms_tf.put_col(1, out1);
  dbms_tf.put_col(2, out2);
END fetch_rows;
/
```

SYS_CONTEXT (1:2)

- SYS_CONTEXT is a function that returns information about the environment in which an operation is running
- Here are some examples from earlier versions of SYS_CONTEXT

```
SELECT sys_context('USERENV', 'AUTHENTICATION_METHOD') FROM dual;

 SYS_CONTEXT('USERENV','AUTHENTICATION_METHOD')
 ----------------------------------------------
 PASSWORD


 SELECT sys_context('USERENV', 'IS_DG_ROLLING_UPGRADE')
 FROM dual;

 SYS_CONTEXT('USERENV','IS_DG_ROLLING_UPGRADE')
 ----------------------------------------------
 FALSE

 SELECT sys_context('USERENV', 'ORACLE_HOME')
 FROM dual;

 SYS_CONTEXT('USERENV','ORACLE_HOME')
 ----------------------------------------
 /u01/app/oracle/product/18.1.0\dbhome_1
```

SYS_CONTEXT (2:2)

- In 18c LDAP_SERVER_TYPE returns the configured LDAP server type, one of OID, AD(Active Directory), OID_G, or OPENLDAP

```
SQL> SELECT sys_context('USERENV', 'LDAP_SERVER_TYPE') FROM dual;

SYS_CONTEXT('USERENV','IS_DG_ROLLING_UPGRADE')
 ----------------------------------------------
OID
```

18c Feature Usage Procs

- DBMS_FEATURE_VPD
  - Collects metadata about the use of Virtual Private Database (DBMS_RLS)

```
dbms_feature_vpd(
feature_boolean OUT NUMBER,
aux_count       OUT NUMBER,
feature_info    OUT CLOB);
```

```
set serveroutput on

DECLARE
 i NUMBER;
 j NUMBER;
 k CLOB;
BEGIN
  dbms_feature_vpd(i, j, k);
  dbms_output.put_line('1: ' || i);
  dbms_output.put_line('2: ' || j);
  dbms_output.put_line('3: ' || k);
END;
/
1: 0
2: 0
3: Number of policies=0, Number of enabled policies=0, Number of objects that have VPD policies=0, Number f
policies on SELECT statement=0, Number of policies on INSERT statement=0, Number of policies on UPDATE
statement=0, Number of policies on DELETE statement=0, Number of policies on INDEX statement=0, Number of
DYNAMIC policies=0, Number of STATIC policies=0, Number of SHARED_STATIC policies=0, Number of
CONTEXT_SENSITIVE policies=0, Number of SHARED_CONTEXT_SENSITIVE policies=0, Number of attribute associated
CONTEXT_SENSITIVE policies=0, Number of policies with long predicate=0, Number of COLUMN LEVEL policies=0,
Number of COMMON policies=0, Number of INHERITED policies=0
```

New Built-In Packages

- DBMS_AWR_PROTECTED
- DBMS_ISCHEDFW
- DBMS_ISCHED_AGENT
- DBMS_ISCHED_UTL
- DBMS_MEMOPTIMIZE
- DBMS_PDB_APP_CON
- DBMS_SODA
- DBMS_SQLSET
- DBMS_STATS_INTERNAL_AGG
- DBMS_WORKLOAD_CAPTURE_I
- DBMS_WORKLOAD_REPLAY_I
- DBMS_WRR_REPORT
- DBMS_XDS_INT

DBMS_MEMOPTIMIZE

- Provides an interface for managing data in the memoptimize pool which is an SGA cache that stores table data and hash index related to the Memoptimized Rowstore
- The package provides the following functionality
  - DROP_OBJECT
    - Removes a table's in-memory hash index

```
dbms_memoptimize.drop_object(
schema_name    IN VARCHAR2,
table_name     IN VARCHAR2,
partition_name IN VARCHAR2 DEFAULT NULL);
```

```
SQL> exec dbms_memooptimize.drop_object('UWCLASS', 'SERVERS');
```

  - POPULATE
    - Populates a table's in-memory hash index

```
dbms_memoptimize.populate(
schema_name    IN VARCHAR2,
table_name     IN VARCHAR2,
partition_name IN VARCHAR2 DEFAULT NULL);
```

```
SQL> exec dbms_memooptimize.populate('UWCLASS', 'SERVERS');
```

DBMS_SODA

- A PL/SQL package implementing Simple Oracle Document Access (SODA)
- SODA allows use of the Oracle Database as a NoSQL document store
- The core abstraction provided by SODA is that of document collections
- The DBMS_SODA package allows developers to create, list, and delete document collections with PL/SQL, and to perform CRUD (create, replace, update, delete) operations on documents
- All DDL functions are encapsulated within this package
- The package contains the following objects
  - CREATE_COLLECTION
  - DROP_COLLECTION
  - LIST_COLLECTION_NAMES
  - OPEN_COLLECTION

DBMS_SQLSET (1:16)

- The DBMS_SQLSET package provides a new an interface for managing SQL tuning sets
- It provides the same subprograms, although in some cases with slightly different names, as the SQL tuning set subprograms in DBMS_SQLTUNE
- The important difference is that use of DBMS_SQLSET **does not require the Oracle Tuning Pack license**
- Execute is granted to PUBLIC
  - Which I recommend you **revoke** unless you can come up with valid a justification for why a user with no priv other than CREATE SESSION has the skills to tune SQL Sets

```
SQL> REVOKE EXECUTE ON DBMS_SQLSET FROM PUBLIC;
```

Oracle ... this is an example of security irresponsibility like granting SELECT on ALL_SOURCE to PUBLIC ... please stop compromising the integrity of your products

DBMS_SQLSET (2:16)

- ADD_REFERENCE
  - Adds a new reference to an existing SQL tuning set to indicate its use by a client

```
dbms_sqlset.add_reference(
sqlset_name  IN VARCHAR2,
description  IN VARCHAR2 := NULL,
sqlset_owner IN VARCHAR2 := NULL)
RETURN NUMBER;
```

```
DECLARE
 retVal NUMBER;
BEGIN
  retVal := dbms_sqlset.add_reference('UW_SQLSET', 'Test Add Ref');
  dbms_output.put_line(retVal);
END;
/
```

DBMS_SQLSET (3:16)

- CAPTURE_CURSOR_CACHE
  - Polls the cache multiple times over a time period, and updates the workload data stored there. It can execute over as long a period as required to capture an entire system workload.

```
dbms_sqlset.capture_cursor_cache(
sqlset_name     IN VARCHAR2,
time_limit      IN POSITIVE := 1800,
repeat_interval IN POSITIVE := 300,
capture_option  IN VARCHAR2 := 'MERGE',
capture_mode    IN NUMBER   := MODE_REPLACE_OLD_STATS,
basic_filter    IN VARCHAR2 := NULL,
sqlset_owner    IN VARCHAR2 := NULL,
recursive_sql   IN VARCHAR2 := HAS_RECURSIVE_SQL);
```

```
DECLARE
 retVal NUMBER;
BEGIN
  retVal := dbms_sqlset.add_reference('UW_SQLSET', 'Test Add Ref');
  dbms_output.put_line(retVal);
END;
/
```

DBMS_SQLSET (4:16)

- CREATE_SQLSET
  - Creates a SQL tuning set object in the database
  - Overload 1

```
dbms_sqlset.create_sqlset(
sqlset_name  IN VARCHAR2,
description  IN VARCHAR2 := NULL,
sqlset_owner IN VARCHAR2 := NULL);
```

```
SQL> exec dbms_sqlset.create_sqlset('UW_SQLSET', 'MLib workload');
```

  - Overload 2

```
dbms_sqlset.create_sqlset(
sqlset_name  IN VARCHAR2 := NULL,
description  IN VARCHAR2 := NULL,
sqlset_owner IN VARCHAR2 := NULL)
RETURN VARCHAR2;
```

```
DECLARE
 retVal VARCHAR2(60);
BEGIN
 retVal := dbms_sqlset.create_sqlset('UW_SQLSET', 'MLib workload', 'UWCLASS');
 dbms_output.put_line(retVal);
END;
/
```

DBMS_SQLSET (5:16)

- CREATE_STGTAB
  - Creates a staging table through which SQL tuning sets are imported and exported

```
dbms_sqlset.create_stgtab(
table_name      IN VARCHAR2,
schema_name     IN VARCHAR2 := NULL,
tablespace_name IN VARCHAR2 := NULL,
db_version      IN NUMBER   := NULL);
```

```
SQL> exec dbms_sqlset.create_stgtab('SQLSET_TAB', 'UWCLASS', 'UWDATA',
dbms_sqlset.sts_stgtab_18_1_version);
```

- DELETE_SQLSET
  - Deletes a set of SQL statements from a SQL tuning set

```
dbms_sqlset.delete_sqlset(
sqlset_name  IN VARCHAR2,
basic_filter IN VARCHAR2 := NULL,
sqlset_owner IN VARCHAR2 := NULL);
```

```
SQL> exec dbms_sqlset.delete_sqlset('UW_SQLSET', 'elapsed_time < 250000', 'UWCLASS');
```

DBMS_SQLSET (6:16)

- DROP_SQLSET
  - Drops a SQL tuning set if it is not active

```
dbms_sqlset.drop_sqlset(
sqlset_name  IN VARCHAR2,
sqlset_owner IN VARCHAR2 := NULL);
```

```
exec dbms_sqlset.drop_sqlset('UW_SQLSET', 'UWCLASS');
```

DBMS_SQLSET (7:16)

- LOAD_SQLSET
  - Populates the sqlset with a set of selected SQL

```
dbms_sqlset.load_sqlset(
sqlset_name       IN VARCHAR2,
populate_cursor   IN sqlset_cursor,
load_option       IN VARCHAR2 := 'INSERT',
update_option     IN VARCHAR2 := 'REPLACE',
update_condition  IN VARCHAR2 := NULL,
update_attributes IN VARCHAR2 := NULL,
ignore_null       IN BOOLEAN  := TRUE,
commit_rows       IN POSITIVE := NULL,
sqlset_owner      IN VARCHAR2 := NULL);
```

```
DECLARE
 l_cursor dbms_sqltune.sqlset_cursor;
BEGIN
  OPEN l_cursor FOR
  SELECT VALUE(p)
  FROM TABLE(dbms_sqltune.select_workload_repository (
  765,    -- begin_snap
  766,    -- end_snap
  NULL,   -- basic_filter
  NULL,   -- object_filter
  NULL,   -- ranking_measure1
  NULL,   -- ranking_measure2
  NULL,   -- ranking_measure3
  NULL,   -- result_percentage
  10)) p; -- result_limit

  dbms_sqltune.load_sqlset('UW_SQLSET', l_cursor);
  dbms_sqlset.load_sqlset('UW_SQLSET', l_cursor);
END;
/
```

DBMS_SQLSET (8:16)

- PACK_STGTAB
  - Moves one or more STS from their location in the SYS schema to a staging table created by the create_stgtab function

```
dbms_sqlset.pack_stgtab(
sqlset_name          IN VARCHAR2,
sqlset_owner         IN VARCHAR2 := NULL,
staging_table_name   IN VARCHAR2,
staging_schema_owner IN VARCHAR2 := NULL,
db_version           IN NUMBER   := NULL);
```

```
SQL> exec dbms_sqlset.pack_stgtab('UW_SQLSET', 'UW_SQSET_TAB');
```

DBMS_SQLSET (9:16)

- REMAP_STGTAB
  - Changes the sqlset names and owners in the staging table so that they can be unpacked with different values than they had on the host system

```
dbms_sqlset.remap_stgtab(
old_sqlset_name       IN VARCHAR2,
old_sqlset_owner      IN VARCHAR2 := NULL,
new_sqlset_name       IN VARCHAR2 := NULL,
new_sqlset_owner      IN VARCHAR2 := NULL,
staging_table_name    IN VARCHAR2,
staging_schema_owner IN VARCHAR2 := NULL,
old_con_dbid          IN NUMBER   := NULL,
new_con_dbid          IN NUMBER   := NULL);
```

```
SQL> exec dbms_sqlset.remap_stgtab('UW_SQLSET', 'UW_SQSET_TAB');
```

DBMS_SQLSET (10:16)

- REMOVE_REFERENCE
  - Deactivates a sqlset to indicate it is no longer used by the client

```
dbms_sqlset.remove_reference(
sqlset_name  IN VARCHAR2,
reference_id IN NUMBER,
sqlset_owner IN VARCHAR2 := NULL,
force_remove IN NUMBER   := 0);
```

- SELECT_CURSOR_CACHE
  - Provided to be able to collect SQL statements from the Cursor Cache

```
dbms_sqlset.select_cursor_cache(
basic_filter      IN VARCHAR2 := NULL,
object_filter     IN VARCHAR2 := NULL,
ranking_measure1  IN VARCHAR2 := NULL,
ranking_measure2  IN VARCHAR2 := NULL,
ranking_measure3  IN VARCHAR2 := NULL,
result_percentage IN NUMBER   := 1,
result_limit      IN NUMBER   := NULL,
attribute_list    IN VARCHAR2 := 'TYPICAL',
recursive_sql     IN VARCHAR2 := HAS_RECURSIVE_SQL)
RETURN sys.sqlset PIPELINED;
```

DBMS_SQLSET (11:16)

- SELECT_SQL_TRACE
  - Reads the content of one or more trace files and returns the SQL statements it finds in the format of sqlset_row

```
dbms_sqlset.select_sql_trace(
directory           IN VARCHAR2,
file_name           IN VARCHAR2       := NULL,
mapping_table_name  IN VARCHAR2       := NULL,
mapping_table_owner IN VARCHAR2       := NULL,
select_mode         IN POSITIVE       := SINGLE_EXECUTION,
options             IN BINARY_INTEGER := LIMITED_COMMAND_TYPE,
pattern_start       IN VARCHAR2       := NULL,
pattern_end         IN VARCHAR2       := NULL,
result_limit        IN POSITIVE       := NULL)
RETURN sys.sqlset PIPELINED;
```

DBMS_SQLSET (12:16)

- SELECT_SQLPA_TASK
  - Collects SQL statements from a Performance Analyzer task for creating a SQL Tuning Set containing the subset of SQL statements that regressed during a SQL Performance Analyzer (SPA) run

```
dbms_sqlset.select_sqlpa_task(
task_name      IN VARCHAR2,
task_owner     IN VARCHAR2 := NULL,
execution_name IN VARCHAR2 := NULL,
level_filter   IN VARCHAR2 := 'REGRESSED',
basic_filter   IN VARCHAR2 := NULL,
object_filter  IN VARCHAR2 := NULL,
attribute_list IN VARCHAR2 := 'TYPICAL')
RETURN sys.sqlset PIPELINED;
```

DBMS_SQLSET (13:16)

- SELECT_SQLSET
  - Reads SQL tuning set contents

```
dbms_sqlset.select_sqlset(
sqlset_name       IN VARCHAR2,
basic_filter      IN VARCHAR2 := NULL,
object_filter     IN VARCHAR2 := NULL,
ranking_measure1  IN VARCHAR2 := NULL,
ranking_measure2  IN VARCHAR2 := NULL,
ranking_measure3  IN VARCHAR2 := NULL,
result_percentage IN NUMBER   := 1,
result_limit      IN NUMBER   := NULL,
attribute_list    IN VARCHAR2 := 'TYPICAL',
plan_filter       IN VARCHAR2 := NULL,
sqlset_owner      IN VARCHAR2 := NULL,
recursive_sql     IN VARCHAR2 := HAS_RECURSIVE_SQL)
RETURN sys.sqlset PIPELINED;
```

```
SQL> exec dbms_sqlset.select_sqlset('UW_SQLSET');
```

DBMS_SQLSET (14:16)

- SELECT_WORKLOAD_REPOSITORY
  - Overload 1: Collects SQL statements from the workload repository to collect SQL statements from all snapshots between begin_snap and and end_snap

```
dbms_sqlset.select_workload_repository(
begin_snap         IN NUMBER,
end_snap           IN NUMBER,
basic_filter       IN VARCHAR2 := NULL,
object_filter      IN VARCHAR2 := NULL,
ranking_measure1   IN VARCHAR2 := NULL,
ranking_measure2   IN VARCHAR2 := NULL,
ranking_measure3   IN VARCHAR2 := NULL,
result_percentage  IN NUMBER   := 1,
result_limit       IN NUMBER   := NULL,
attribute_list     IN VARCHAR2 := 'TYPICAL',
recursive_sql      IN VARCHAR2 := HAS_RECURSIVE_SQL,
dbid               IN NUMBER   := NULL)
RETURN sys.sqlset PIPELINED;
```

  - Overload 2: Collects SQL statements from the workload repository to collect SQL statements from a specified baseline

```
dbms_sqlset.select_workload_repository(
baseline_name      IN VARCHAR2,
basic_filter       IN VARCHAR2 := NULL,
object_filter      IN VARCHAR2 := NULL,
ranking_measure1   IN VARCHAR2 := NULL,
ranking_measure2   IN VARCHAR2 := NULL,
ranking_measure3   IN VARCHAR2 := NULL,
result_percentage  IN NUMBER   := 1,
result_limit       IN NUMBER   := NULL,
attribute_list     IN VARCHAR2 := 'TYPICAL',
recursive_sql      IN VARCHAR2 := HAS_RECURSIVE_SQL,
dbid               IN NUMBER   := NULL)
RETURN sys.sqlset PIPELINED;
```

DBMS_SQLSET (15:16)

- UNPACK_STGTAB
  - Moves one or more STS from the staging table, as populated by a call to pack_stgtab and moved by the user, into the STS schema, making them proper STS
  - Users can drop the staging table after this procedure completes successfully

```
dbms_sqlset.unpack_stgtab(
sqlset_name          IN VARCHAR2 := '%',
sqlset_owner         IN VARCHAR2 := NULL,
replace              IN BOOLEAN,
staging_table_name   IN VARCHAR2,
staging_schema_owner IN VARCHAR2 := NULL);
```

```
SQL> exec dbms_sqlset.unpack_stgtab(replace => TRUE, staging_table_name => 'UW_SQSET_TAB');
```

DBMS_SQLSET (16:16)

- UPDATE_SQLSET
  - updates selected string fields for a SQL statement in a sqlset (2 overloads)

```
dbms_sqlset.update_sqlset(
sqlset_name     IN VARCHAR2,
sql_id          IN VARCHAR2,
plan_hash_value IN NUMBER   := NULL,
attribute_name  IN VARCHAR2,
attribute_value IN VARCHAR2 := NULL,
sqlset_owner    IN VARCHAR2 := NULL);
```

  - Overload 2

```
dbms_sqlset.update_sqlset(
sqlset_name IN VARCHAR2,
sql_id          IN VARCHAR2,
plan_hash_value IN NUMBER   := NULL,
attribute_name  IN VARCHAR2,
attribute_value IN NUMBER   := NULL,
sqlset_owner    IN VARCHAR2 := NULL);
```

Modified Built-In Packages

- DBMS_DISTRUPT
- DBMS_DRS
- DBMS_DST
- DBMS_DB_VERSION
- DBMS_SESSION
- DBMS_WORKLOAD_CAPTURE
- DBMS_WORKLOAD_REPLAY
- DBMS_XPLAN

DBMS_DISRUPT

- The DBMS_DISRUPT built-in package is officially undocumented in the Types and Packages reference but worthy of your attention as it provides a PL/SQL interface for disrupting sessions and services and contains subprograms related to the following
  - DISRUPT SERVICES
  - DISRUPT SESSION
- Originally released in version 12.2.0.1 the DISRUPT_SESSIONS procedure added a new parameter in 18.1.0.0

```
dbms_disrupt.disrupt_sessions(
job_name       IN VARCHAR2,
user_names     IN VARCHAR2 := '?',
service_names  IN VARCHAR2 := '*',
instance_names IN VARCHAR2 := '?',
module_names   IN VARCHAR2 := '*',
percentage     IN NUMBER,
sleep_interval IN NUMBER,
duration       IN NUMBER   := 0,
output_file    IN VARCHAR2 := NULL);
```

- The package is owned by SYS with no privileges granted and you might want to keep an eye on the fact that it stays that way

DBMS_DRS (1:5)

- The DBMS_DRS built-in package supporting Data Guard has 27 new objects
- DBMS_DRS is not documented and supported for DBAs and Developers but these new functions provide an excellent window into changes Oracle is making to Data Guard Physical Standbys and are an excellent way of learning how it changing and improving
  - ADD_DATABASE
    - Add a standby database to a broker configuration. database_ci is the connection identifier
  - ADD_FAR_SYNC
    - Add a far sync instance to a broker configuration. far_sync_ci is the connection identifier
  - CHECK_CONNECT
    - Check network connectivity to the specified member

```
dbms_drs.check_connect(
member_name   IN VARCHAR2,
instance_name IN VARCHAR2);
```

```
SQL> exec dbms_drs.check_connect('ORABASE_DR', 'ORABASE1');
```

DBMS_DRS (2:5)

- CREATE_CONFIGURATION
  - Creates a broker configuration. The primary database will be automatically added to the configuration by this procedure. Must be called on a primary database.
- DISABLE_FS_FAILOVER
  - Disables Fast Start Failover
- DO_OBSERVE
  - Observer operation API - observer's operation to control FSFO since 12.2. (replaces Ping, ReadyToFailover, and StateChangeRecorded)
- DUMP_BROKER
  - Dumps critical internal data of the broker process to a file
- DUMP_OBSERVER
  - Dumps critical internal data of client-side observer process to a file
- ENABLE_CONFIGURATION
  - Enables broker management of a Data Guard configuration. It must be called on the primary database. Return 0 means enable was successful, otherwise returns an error number.

DBMS_DRS (3:5)

- ENABLE_DATABASE
  - Used to enable broker management of a database within the broker configuration. It must be called on the primary database.
- ENABLE_FAR_SYNC
  - Enable broker management of a far sync instance within the broker configuration. It must be called on the primary database.
- ENABLE_FS_FAILOVER
  - Used to enable fast-start failover
- REMOVE_CONFIGURATION
  - Removes a broker configuration. It must be called on the primary database.
- REMOVE_DATABASE
  - Used to remove a database from the broker configuration. It must be called on the primary database.
- REMOVE_FAR_SYNC
  - Removes a far sync instance from the broker configuration. It must be called on the primary database.

DBMS_DRS (4:5)

- REPLACE_MEMBER_NAME_IN_PROPS
  - Replaces a member name with another member name in all broker properties
- RESET_CONFIGURATION_PROPERTY
  - Resets configuration-level property, not database or far sync instance property, to its default value.
- RESET_DATABASE_PROPERTY
  - Resets a database configurable property to its default value
- RESET_FAR_SYNC_PROPERTY
  - Resets a far sync instance configurable property to its default value
- SET_CONFIGURATION_PROPERTY
  - Used to set configuration-level property (not a database or far sync property). Can be used to set both integer and character string properties.
- SET_DATABASE_PROPERTY
  - Used to set a database configurable property. Can be used to set both integer and character string properties.

DBMS_DRS (5:5)

- SET_FAR_SYNC_PROPERTY
  - Used to set a far sync instance's configurable property. Can be used to set both integer and character string properties.
- SET_PROTECTION_MODE
  - Changes the protection mode to the mode specified. To prevent including database restart logic this procedure does not support the promotion of the protection mode from maximum performance to maximum protection.
- STOP_OBSERVER
  - Stops the fast-start failover observers in a data guard broker configuration
- WAIT
  - Waits up to the number of seconds specified by the max_wait_time argument for the event specified by the event_type parameter to prevail

DBMS_DST

- Subprograms in this package allow users to apply Daylight Saving Time (DST) patches to the TIMESTAMP WITH TIME ZONE (TSTZ) data type
- FIND_AFFECTED_TABLES has a new PARALLEL parameter
  - During a "prepare window" finds all the tables which have affected TSTZ data due to the new time zone version

```
dbms_dst.find_affected_tables(
 affected_tables  IN VARCHAR2 := 'sys.dst$affected_tables',
 log_errors       IN BOOLEAN  := FALSE,
 log_errors_table IN VARCHAR2 := 'sys.dst$error_table',
 parallel         IN BOOLEAN  := FALSE);
```

```
BEGIN
  dbms_dst.begin_prepare(31);
  dbms_dst.find_affected_tables;
  dbms_dst.end_prepare;
END;
/

SELECT * FROM sys.dst$affected_tables;
```

DBMS_DB_VERSION

- This package can be used to specify the Oracle version numbers and other information useful for simple conditional compilation selections based on the Oracle or TimesTen Database version
- New Constant: VERSION_LE_18

```
BEGIN
  $IF dbms_db_version.ver_le_10 $THEN
    dbms_output.put_line('version 10 and earlier');
  $ELSIF dbms_db_version.ver_le_11 $THEN
    dbms_output.put_line('version 11');
  $ELSIF dbms_db_version.ver_le_12 $THEN
    dbms_output.put_line('version 12c');
  $ELSIF dbms_db_version.ver_le_18 $THEN
    dbms_output.put_line('version 18c');
  $ELSE
    dbms_output.put_line('Unknown version');
  $END  -- note that there is no semi-colon
END;
/
```

DBMS_HPROF (1:2)

- This package is used for hierarchical profiling of PL/SQL objects
  - ANALYZE
    - 2 new overloads for analyzing the raw profiler output and produces hierarchical profiler information in database tables

Overload 4

```
dbms_hprof.analyze(
trace_id      IN NUMBER,
summary_mode  IN BOOLEAN      DEFAULT FALSE,
trace         IN VARCHAR2     DEFAULT NULL,
skip          IN PLS_INTEGER  DEFAULT 0,
collect       IN PLS_INTEGER  DEFAULT NULL,
run_comment   IN VARCHAR2     DEFAULT NULL,
profile_uga   IN BOOLEAN      DEFAULT NULL,
profile_pga   IN BOOLEAN      DEFAULT NULL)
RETURN NUMBER;
```

Overload 5

```
dbms_hprof.analyze(
trace_id     IN  NUMBER,
report_clob OUT CLOB,
trace        IN  VARCHAR2     DEFAULT NULL,
skip         IN  PLS_INTEGER  DEFAULT 0,
collect      IN  PLS_INTEGER  DEFAULT NULL,
profile_uga  IN  BOOLEAN      DEFAULT NULL,
profile_pga  IN  BOOLEAN      DEFAULT NULL);
```

  - CREATE TABLES
    - Creates the table dbmshp_trace_data and sequence dbmshp_tracenumber sequence

```
dbms_hprof.start_profiling(force_it IN BOOLEAN DEFAULT FALSE);
```

```
exec dbms_hprof.create_tables(TRUE);
```

DBMS_HPROF (2:2)

- START_PROFILING
  - 1 new overload for starting PL/SQL profiling

```
dbms_hprof.start_profiling(
max_depth   IN PLS_INTEGER DEFAULT NULL,
profile_uga IN BOOLEAN     DEFAULT NULL,
profile_pga IN BOOLEAN     DEFAULT NULL,
sqlmonitor  IN BOOLEAN     DEFAULT TRUE,
run_comment IN VARCHAR2    DEFAULT NULL)
RETURN NUMBER;
```

DBMS_PDB (1:3)

- Removed Subprogram
  - DBMS_PDB.REMOVE_LINK
- New Subprograms
  - CLEAR_PLUGIN_VIOLATIONS
    - Undocumented but you need to know what it is and how it works
    - Determines whether a pluggable database described by file pdb_descr_file is compatibile with the current CDB

```
SQL> desc pdb_plug_in_violations
 Name                                                      Null?    Type
 --------------------------------------------------------- -------- --------------
 TIME                                                      NOT NULL TIMESTAMP(6)
 NAME                                                      NOT NULL VARCHAR2(30)
 CAUSE                                                              VARCHAR2(64)
 TYPE                                                               VARCHAR2(9)
 ERROR_NUMBER                                                       NUMBER
 LINE                                                      NOT NULL NUMBER
 MESSAGE                                                   NOT NULL VARCHAR2(4000)
 STATUS                                                             VARCHAR2(9)
 ACTION                                                             VARCHAR2(4000)
```

DBMS_PDB (2:3)

```
SQL> SELECT time, name, cause, type, message, status, action
  2  FROM pdb_plug_in_violations
  3* WHERE rownum = 1;

TIME                          NAME    TYPE     MESSAGE
----------------------------- ------- -------- ---------------------------------------------------------------
04-DEC-14 10.20.20.929000 PM  PDBDEV  WARNING  CDB parameter shared_pool_size mismatch: Previous 800M Current 0

STATUS   ACTION
-------- ----------------------------------------------
RESOLVED Please check the parameter in the current CDB
```

- IS_VALID_PATH
  - Undocumented but a potentially valuable tool

```
BEGIN
  IF dbms_pdb.is_valid_path('/u04/app/oracle/oradata/uwapp/') THEN
    dbms_output.put_line('T');
  ELSE
    dbms_output.put_line('F');
  END IF;
END;
/
```

DBMS_PDB (3:3)

- SET_SHARING_NONE
  - Undocumented but you need to know what it is and how it works
  - Uused to set SHARTING=NONE status on an object in an App Root. It is intended to be used in migration cases where an application was already installed in a PDB or a non-CDB, where there was no support for application containers.

```
dbms_pdb.set_sharing_none(
schema_name  IN VARCHAR2,
object_name  IN VARCHAR2,
namespace    IN NUMBER,
edition_name IN VARCHAR2 DEFAULT NULL);
```

```
exec dbms_pdb.set_sharing_none('UWCLASS', 'SSNONE', 1, 'ORA$BASE');
```

DBMS_SESSION (1:2)

- ADD_SQL_CONNECTION_TEST
  - Creates connection test an application servers can use to check the health of a database connection before using it

```
dbms_session.add_sql_connection_test(
connection_test IN VARCHAR2,
service_name    IN VARCHAR2 DEFAULT NULL);
```

```
SQL> exec dbms_session.add_sql_connection_test('OurTest', 'SYS$USERS');
```

- DELETE_SQL_CONNECTION_TEST
  - Remove a connection test

```
SQL> exec dbms_session.add_sql_connection_test('OurTest', 'SYS$USERS');
```

DBMS_SESSION (2:2)

- DISABLE_CONNECTION_TEST
  - Disables an application connection test

```
dbms_session.disable_connection_test(
 connection_test_type IN NUMBER,
 connection_test      IN VARCHAR2 DEFAULT NULL,
 service_name         IN VARCHAR2 DEFAULT NULL);;
```

Connection Test Types
- ENDREQUEST_TEST
- PING_TEST
- SQL_TEST

```
SQL> exec dbms_session.disable_connection_test(dbms_session.ping_test, 'OurTest', 'SYS$USERS');
```

- ENABLE_CONNECTION_TEST
  - Enables an application connection test

```
SQL> exec dbms_session.disable_connection_test(dbms_session.sql_test, 'OurTest', 'SYS$USERS');
```

- SLEEP
  - Because having SLEEP in dbms_backup_restore, dbms_drs, dbms_lock, and user_lock was not sufficient

```
SQL> exec dbms_session.sleep(10);
```

DBMS_WORKLOAD_CAPTURE (1:3)

- We have had a major security hole in the Oracle Database that has now been plugged with version 18 ... we can now encrypt capture for Real Application Testing
- DECRYPT_CAPTURE
  - Decrypts a previously encrypted workload capture (outside of replay)

```
dbms_workload_capture.decrypt_capture(
src_dir IN VARCHAR2,
dst_dir IN VARCHAR2);
```

```
SQL> exec dbms_workload_capture.decrypt_capture('SRCDIR', 'TGTDIR');
```

- ENCRYPT_CAPTURE
  - Encrypts an unencrypted workload capture

```
dbms_workload_capture.encrypt_capture(
 src_dir    IN VARCHAR2,
 dst_dir    IN VARCHAR2,
 encryption IN VARCHAR2 DEFAULT 'AES256'); -- options: 'AES128', 'AES192', 'AES256'
```

```
SQL> exec dbms_workload_capture.encrypt_capture('SRCDIR', 'TGTDIR', 'AES256');
```

DBMS_WORKLOAD_CAPTURE (2:3)

- New Public Capabilities
  - START_CAPTURE (new parameters)
    - PLSQL_MODE
      - TOP_LEVEL: only top-level PL/SQL calls are captured
      - EXTENDED: both top-level PL/SQL calls and SQL called from PL/SQL are captured
    - ENCRYPTION
      - NULL: no encryption
      - AES128
      - AES 192
      - AES256

```
dbms_workload_capture.start_capture(
name              IN VARCHAR2,
dir               IN VARCHAR2,
duration          IN NUMBER   DEFAULT NULL,
default_action    IN VARCHAR2 DEFAULT 'INCLUDE',
auto_unrestrict   IN BOOLEAN  DEFAULT TRUE,
capture_sts       IN BOOLEAN  DEFAULT FALSE,
sts_cap_interval  IN NUMBER   DEFAULT 300,
plsql_mode        IN VARCHAR2 DEFAULT 'TOP_LEVEL',
encryption        IN VARCHAR2 DEFAULT NULL);
```

DBMS_WORKLOAD_CAPTURE (3:3)

- GET_STATE (unsupported)
  - Returns 1 if a session is being capture: Else 0

```
SQL> SELECT dbms_workload_capture.get_state
   2  FROM dual;

  GET_STATE
 ----------
          0
```

- Not supported but worth knowing are coming in the future
  - START_BATCH_CAPTURE
    - Starts a workload capture and stores data in different buckets
    - For instance, workload in 9AM - 10AM will be stored in bucket 1 while workload in 10AM - 12PM will be stored in bucket 2
  - SWITCH_BUCKET
    - Signals all connected sessions to store workload captures into a new bucket
    - By default, SWITCH_BUCKET will create an AWR snapshot for the workload captured in the current bucket

DBMS_WORKLOAD_REPLAY (1:2)

- ASSIGN_GROUP_TO_INSTANCE
  - Assigns a group of capture files to be processed by a particular node in a RAC cluster

```
dbms_workload_replay.assign_group_to_instance(
 group_id        IN NUMBER,
 instance_number IN NUMBER);
```

```
SQL> exec dbms_workload_replay.assign_group_to_instance(6, 2);
```

- LOAD_LONG_SQLTEXT
  - Loads captured SQL statements whose length is greater than 1000 characters

```
dbms_workload_replay.load_long_sqltext(capture_id IN NUMBER);
```

```
SQL> exec dbms_workload_replay.load_long_sqltext(1107);
```

DBMS_WORKLOAD_REPLAY (2:2)

- SET_SQL_MAPPING
  - Specifies SQL statements to be skipped or replaced in replay

```
dbms_workload_replay.set_sql_mapping(
 schedule_cap_id      IN NUMBER,
 sql_id               IN VARCHAR2,
 operation            IN VARCHAR2,
 replacement_sql_text IN VARCHAR2);
```

  - Overload 2

```
dbms_workload_replay.set_sql_mapping(
 sql_id               IN VARCHAR2,
 operation            IN VARCHAR2,
 replacement_sql_text IN VARCHAR2);
```

DBMS_XPLAN

- DISPLAY_CURSOR has 2 new overloads
  - The original pipelined table function is now Overload 3

```
dbms_xplan.display_cursor(
 sql_id          IN VARCHAR2 DEFAULT NULL,
 cursor_child_no IN INTEGER  DEFAULT 0,
 format          IN VARCHAR2 DEFAULT 'TYPICAL',
 shard_id        IN NUMBER)
 RETURN dbms_xplan_type_table PIPELINED;
```

```
dbms_xplan.display_cursor(
 sql_id          IN VARCHAR2 DEFAULT NULL,
 cursor_child_no IN INTEGER  DEFAULT 0,
 format          IN VARCHAR2 DEFAULT 'TYPICAL',
 shard_ids       IN num_tab_type)
 RETURN dbms_xplan_type_table PIPELINED;
```

```
*
ERROR at line 1:
ORA-00028: your session has been killed
```

Thank You

META7™

Questions? dmorgan@siriuscom.com