

Oracle Database 11gR2 Syntax for Developers

or

What you should know before the release of 12c

Daniel A. Morgan



Oracle ACE Director



Consultant to Harvard University



University of Washington Oracle Instructor, ret.



The Morgan of Morgan's Library on the web



Board Member: Western Washington OUG

■ Upcoming Presentations

- May 30-31: OUG Harmony Finland
- Jun 1: OUG Harmony Latvia
- Jun 20: Vancouver Canada
- Jun 21: Victoria Canada
- Sep: OpenWorld 2012: San Francisco

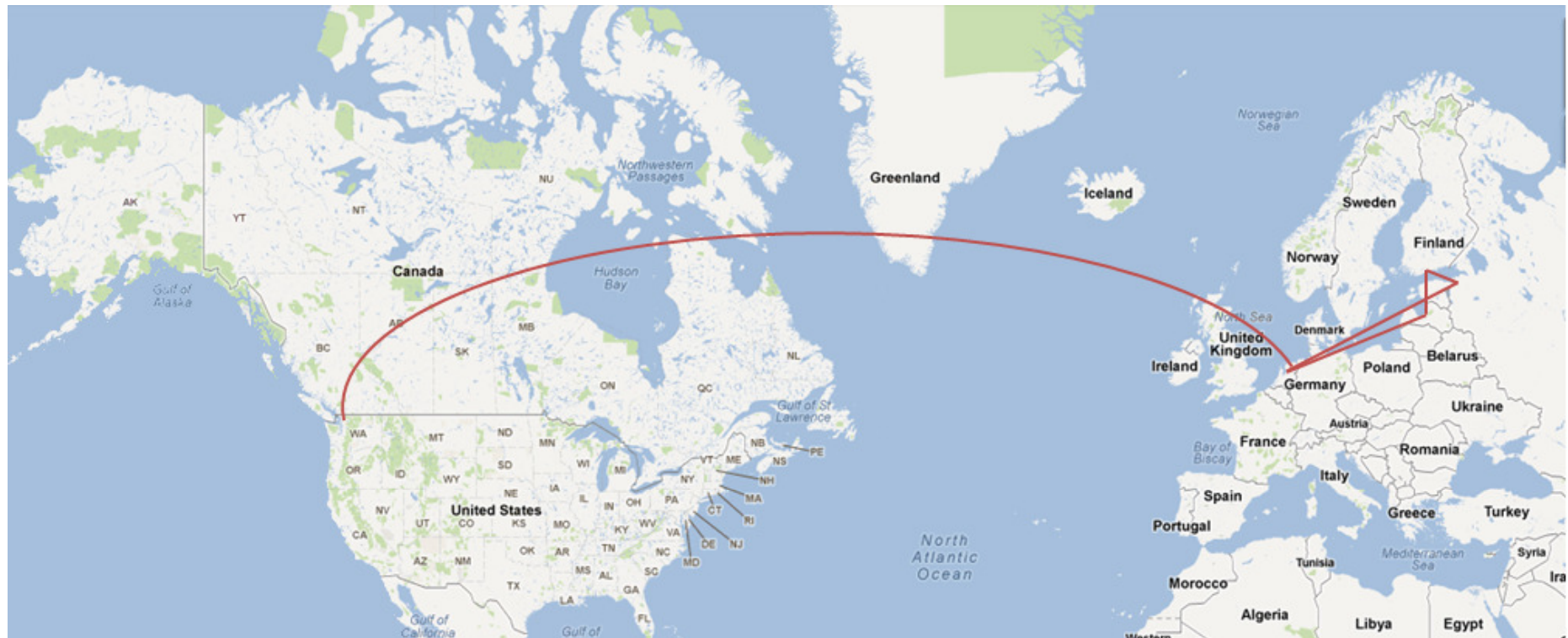


Official Beta Site
ORACLE
DATABASE **11g**

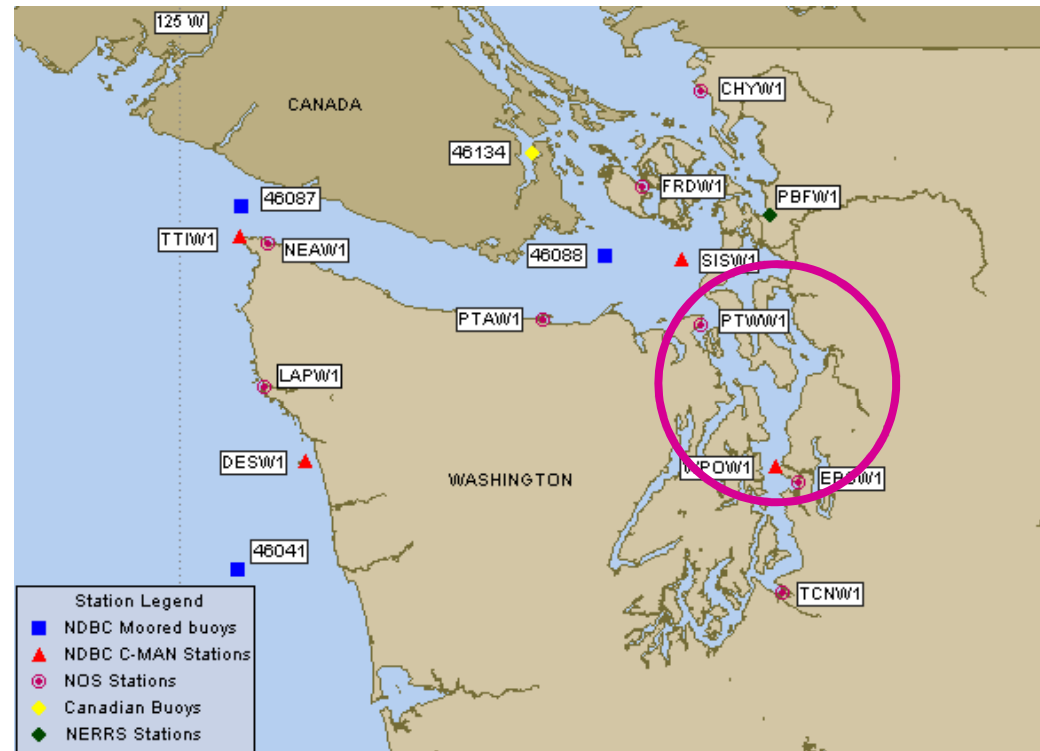
ORACLE
RAC SIG

International
zSeries
Oracle SIG

cd \$MORGAN_HOME



cd \$MORGAN_HOME



```
cd $MORGAN_HOME
```



Morgan's Library: www.morganslibrary.org



Morgan's Library

[www](#) [library](#)

Morgan's 2010 - 2011 Calendar

May	Jun	Jul	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----

EMEA Harmony Conference

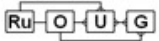
Tallinn, Estonia

May 20-21, 2010



A joint conference of the Estonian, Finnish, Latvian and Russian user groups
EMEA Harmony will focus on Technology, Middleware and BI
Featured speakers include Tom Kyte, Mogen Norgaard, Tanel Poder, and Dan Morgan





Community

- [Events](#)
- [Training](#)
- [Evening Workshops](#)

Resources

- [Library](#)
- [How Can I?](#)
- [Code Samples](#)
- [Presentations](#)
- [Links](#)
- [Book Reviews](#)
- [Downloads](#)
- [User Groups](#)

General

- [Contact](#)
- [About](#)
- [Services](#)
- [Legal Notice & Terms of Use](#)
- [Privacy Statement](#)

Presentations Map



The Mad Dog ACE



Morgan



aboard USA-71

Training Events

-  [EMEA Harmony](#) - May 20 - 21, Tallinn, Estonia
-  [NoCOUG](#) - August 2010,
-  [AIOUG](#) Sep 3 - 4, Hyderabad, India
-  [OOW](#) - Sep 19 - 23, San Francisco CA
-    [LAD Tour](#) - October
-  [DOAG](#) - Nov 16 - 18, Nurnberg, Germany
-  [UKOUG](#) - Nov 29 - Dec 1, Birmingham UK

Library News

- [Morgan's Notepad vi \(Blog\)](#) UPDATED
- [Join the Western Washington OUG](#)
- [Morgan's Oracle Podcast](#)
- [DBA Best Practice Guidelines](#)
- [Bryn Llewellyn's PL/SQL White Paper](#)
- [Bryn Llewellyn's Editioning White Paper](#)
- [Troubleshooting Performance](#)

Oracle Events



[EMEA Harmony - Tallinn Estonia - May 20-21](#)

ACE News

 Would you like to become an Oracle ACE? 

Learn more about becoming an ACE



- [ACE Directory](#)
- [ACE Google Map](#)
- [ACE Nomination Form](#)
- [Stanley's Blog](#)

ODA's are not childproof



Syllabus

- What can I tell you about Database Version 12c?
- What should everyone know about Oracle 8.0i?
- What should everyone know about Oracle 8.i?
- What should everyone know about Oracle 9.0?
- What should everyone know about Oracle 9.2?
- What should everyone know about Oracle 10gR1?
- What should everyone know about Oracle 10gR2?
- What should everyone know about Oracle 11gR1?
- What should everyone know about Oracle 11gR2?

One Take-Away: Get your money's worth!

- Some people complain about the cost of Oracle Database software
 - While not using a fraction of what they paid for
 - While reinventing the wheel
- Would anyone purchase a house and not use the kitchen?

Basic SQL

- DDL statements
 - Tables and Columns
 - Constraints
 - Indexes
 - Database Links

Tables and Columns

- Heap
- Partitioned and Composite Partitioned
- Compressed
- External
- Global Temporary
- Index Organized
- XML
- Read Only
- CLOB, BLOB, XMLTYPE, Virtual

Production Code

```
CREATE TABLE orders OF XMLType
XMLTYPE STORE AS BINARY XML
VIRTUAL COLUMNS (
  SITE_ID AS (XMLCast(XMLQuery('/Order/@SiteId' PASSING OBJECT_VALUE RETURNING CONTENT) AS NUMBER)))
PARTITION BY RANGE (site_id) (
  PARTITION p1 VALUES LESS THAN (10),
  PARTITION p2 VALUES LESS THAN (20),
  PARTITION pm VALUES LESS THAN (MAXVALUE));

DECLARE
  x XMLTYPE;
BEGIN
  x := XMLTYPE('<?xml version="1.0" encoding="utf-8"?>
    <Order orderId="1" orderRevision="1" orderTimeStamp="01-JAN-2012">
      <OrderHeader>
        <AlternateIds>
          <AlternateId altIdType="SiteId">12</AlternateId>
          <AlternateId altIdType="MerchantOrderNumber">Merch</AlternateId>
          <AlternateId altIdType="MarketplaceOrderNumber">Place</AlternateId>
          <AlternateId altIdType="CustomerReferenceId">Ref</AlternateId>
          <AlternateId altIdType="CartId">Cart</AlternateId>
          <AlternateId altIdType="SessionId">1</AlternateId>
        </AlternateIds>
      </OrderHeader>
    </Order>');
  INSERT INTO orders VALUES (x);
END;
/

...
```


READ ONLY Optimization

- When consistent with their usage:
- Set tablespaces to READ ONLY
- Set tables to READ ONLY
- Set partitions to READ ONLY

READ ONLY Optimization

```
CREATE TABLE t1 AS SELECT * FROM sh.sales;

ALTER SESSION SET tracefile_identifier = 'test_plan1';
ALTER SESSION SET EVENTS '10046 trace name context forever, level 12';

SELECT COUNT(*) FROM (SELECT * FROM t1);

ALTER SESSION SET EVENTS '10046 trace name context OFF';
```

call	count	cpu	elapsed	disk	query	current	rows
Parse	1	0.00	0.00	0	1	0	0
Execute	1	0.00	0.00	0	0	0	0
Fetch	2	0.67	1.32	4433	4438	0	1
total	4	0.67	1.32	4433	4439	0	1

Read-Write

```
CREATE TABLE t2 AS SELECT * FROM sh.sales;
ALTER TABLE t2 READ ONLY;

ALTER SESSION SET tracefile_identifier = 'test_plan2';
ALTER SESSION SET EVENTS '10046 trace name context forever, level 12';

SELECT COUNT(*) FROM (SELECT * FROM t2);

ALTER SESSION SET EVENTS '10046 trace name context OFF';
```

call	count	cpu	elapsed	disk	query	current	rows
Parse	1	0.00	0.00	0	1	0	0
Execute	1	0.00	0.00	0	0	0	0
Fetch	2	0.28	1.14	4433	4438	0	1
total	4	0.28	1.15	4433	4439	0	1

Read-Only

Constraints

- Why do primary keys and unique constraints have unique indexes?
- When is it appropriate to create deferrable constraints
- Unique and Check constraints on Virtual Columns
- Why does everyone create NOT NULL constraints on Primary Key columns?

Indexes

- B*Tree
- Bitmap
- Bitmap Join (Star)
- Compressed
- Descending
- Function Based
- Invisible
- No Segment
- Reverse
- Indexes on Virtual Columns

Database Links

```
CREATE DATABASE LINK fixed_user  
CONNECT TO hr IDENTIFIED BY hr;
```

```
col owner format a20  
col db_link format a15  
col username format a20  
col host format a12
```

```
SELECT * FROM dba_db_links;
```

```
ALTER DATABASE LINK fixed_user  
CONNECT TO hr IDENTIFIED BY hrnew;
```

Basic SQL

- DML statements
 - SELECT with SAMPLE clause
 - SELECT into a ROW
 - INSERT WHEN
 - INSERT ALL
 - INSERT FIRST
 - INSERT into a SELECT statement
 - INSERT with the RETURNING clause
 - INSERT statement with records
 - UPDATE with RETURNING clause
 - UPDATE with SET ROW
 - DELETE from a SELECT statement
 - DELETE with RETURNING clause
 - MERGE can also include a DELETE

SELECT SAMPLE CLAUSE

```
SELECT *  
FROM t  
SAMPLE (1) ;
```

```
SELECT /*+ FULL */ COUNT (*)  
FROM t  
SAMPLE BLOCK (1) ;
```

```
SELECT *  
FROM t  
SAMPLE (1) SEED (5) ;
```

SELECT (PL/SQL)

```
DECLARE
```

```
    trec t%ROWTYPE;
```

```
BEGIN
```

```
    SELECT *
```

```
    INTO trec
```

```
    FROM t
```

```
    WHERE rownum = 1;
```

```
    dbms_output.put_line(trec.table_name);
```

```
END;
```

```
/
```

I am not not afraid to show mine in public



Optimizer Plans (customer's SQL before "tuning")

```
SELECT DISTINCT E1_2.OBJECT_ID
FROM PMCM.ELEMENT_DETAIL E1_1, PMCM.ELEMENT_DETAIL E1_2, PMCM.MARK_NETW_HIERARCHY H1,
     PMCM.ELEMENT_DETAIL E2_1, PMCM.ELEMENT_DETAIL E2_2, PMCM.MARK_NETW_HIERARCHY H2
WHERE E1_1.OBJECT_ID = H1.PARENT_ID
      AND E1_2.OBJECT_ID = H1.OBJECT_ID
      AND E2_1.OBJECT_ID = H2.PARENT_ID
      AND E2_2.OBJECT_ID = H2.OBJECT_ID
      AND E1_1.CURRENT_IND = 'Y' AND E2_1.CURRENT_IND = 'Y'
      AND E2_1.CURRENT_IND = 'Y' AND E2_2.CURRENT_IND = 'Y'
      AND H1.CURRENT_IND = 'Y' AND H2.CURRENT_IND = 'Y'
      AND H1.HIERARCHY_TYPE = 'NETWORK' AND H2.HIERARCHY_TYPE = 'NETWORK'
      AND H1.PARENT_TYPE IN ('BSC','RNC') AND H2.PARENT_TYPE IN ('BSC','RNC')
      AND E2_2.ELEMENT_TYPE = 'CELL' AND E1_2.ELEMENT_TYPE = 'CELL'
      AND H1.PARENT_TYPE IN ('BSC','RNC')
      AND E1_1.ELEMENT_NAME = E2_1.ELEMENT_NAME
      AND E1_1.ELEMENT_ID = E2_1.ELEMENT_ID
      AND E1_2.ELEMENT_NAME = E2_2.ELEMENT_NAME
      AND E1_2.ELEMENT_ID = E2_2.ELEMENT_ID
      AND E1_2.USEID LIKE '*' AND E2_2.USEID NOT LIKE '*%';
```

Id	Operation	Name	Rows	Bytes	TempSpc	Cost (%CPU)	Time	Pstart	Pstop
0	SELECT STATEMENT		1	78		74M (40)	50:54:42		
1	TEMP TABLE TRANSFORMATION								
2	LOAD AS SELECT								
3	PARTITION RANGE ALL		22M	1111M		38153 (11)	00:01:34	1	29
* 4	TABLE ACCESS FULL	ELEMENT_DETAIL	22M	1111M		38153 (11)	00:01:34		
5	LOAD AS SELECT								
6	PARTITION HASH ALL		337K	9231K		3514 (15)	00:00:09	1	16
* 7	TABLE ACCESS FULL	MARK_NETW_HIERARCHY	337K	9231K		3514 (15)	00:00:09		
8	SORT AGGREGATE		1	78					
* 9	HASH JOIN		927G	65T	534M	74M (40)	50:53:00		
10	VIEW		22M	277M		16808 (12)	00:00:42		
11	TABLE ACCESS FULL	SYS_TEMP_OFDA7485F_6A66C42E	22M	1111M		16808 (12)	00:00:00		
* 12	HASH JOIN		21G	1272G	534M	1616K (43)	01:06:04		
13	VIEW		22M	277M		16808 (12)	00:00:42		
14	TABLE ACCESS FULL	SYS_TEMP_OFDA7485F_6A66C42E	22M	1111M		16808 (12)	00:00:00		
* 15	HASH JOIN		476M	23G	524M	97327 (22)	00:03:59		
* 16	HASH JOIN		10M	401M	8704K	34520 (10)	00:01:25		
* 17	HASH JOIN		234K	5948K	8256K	783 (10)	00:00:02		
18	VIEW		337K	4286K		142 (14)	00:00:01		
19	TABLE ACCESS FULL	SYS_TEMP_OFDA74860_6A66C42E	337K	3956K		142 (14)	00:00:01		
20	VIEW		337K	4286K		142 (14)	00:00:01		
21	TABLE ACCESS FULL	SYS_TEMP_OFDA74860_6A66C42E	337K	3956K		142 (14)	00:00:01		
22	VIEW		22M	277M		16808 (12)	00:00:42		
23	TABLE ACCESS FULL	SYS_TEMP_OFDA7485F_6A66C42E	22M	1111M		16808 (12)	00:00:42		
24	VIEW		22M	277M		16808 (12)	00:00:42		
25	TABLE ACCESS FULL	SYS_TEMP_OFDA7485F_6A66C42E	22M	1111M		16808 (12)	00:00:00		

Optimizer Plans (tuning gone terribly wrong)

Id	Operation	Name	Rows	Bytes	TempSpc	Cost (%CPU)	Time	Pstart	Pstop
0	SELECT STATEMENT		1	78		14T(100)	999:59:59		
1	TEMP TABLE TRANSFORMATION								
2	LOAD AS SELECT								
3	PARTITION RANGE ALL		22M	1111M		38153 (11)	00:01:34	1	29
* 4	TABLE ACCESS FULL	ELEMENT_DETAIL	22M	1111M		38153 (11)	00:01:34		
5	LOAD AS SELECT								
6	PARTITION HASH ALL		337K	9231K		3514 (15)	00:00:09	1	16
* 7	TABLE ACCESS FULL	MARK_NETW_HIERARCHY	337K	9231K		3514 (15)	00:00:09		
8	SORT AGGREGATE		1	78					
9	MERGE JOIN		471P	15E		14T(100)	999:59:59		
10	MERGE JOIN		10P	616P		694G (81)	999:59:59		
11	MERGE JOIN		231T	10P		377G (64)	999:59:59		
12	SORT JOIN		334T	11P	28P	377G (64)	999:59:59		
13	MERGE JOIN CARTESIAN		334T	11P		140G (14)	999:59:59		
* 14	HASH JOIN		989M	23G	534M	96010 (38)	00:03:56		
15	VIEW		22M	277M		16808 (12)	00:00:42		
16	TABLE ACCESS FULL	SYS_TEMP_OFDA7485B_6A66C42E	22M	1111M		16808 (12)	00:00:42		
17	VIEW		22M	277M		16808 (12)	00:00:42		
18	TABLE ACCESS FULL	SYS_TEMP_OFDA7485B_6A66C42E	22M	1111M		16808 (12)	00:00:42		
19	BUFFER SORT		337K	4286K		140G (14)	999:59:59		
20	VIEW		337K	4286K		142 (14)	00:00:01		
21	TABLE ACCESS FULL	SYS_TEMP_OFDA7485C_6A66C42E	337K	3956K		142 (14)	00:00:01		
* 22	SORT JOIN		337K	4286K	12M	844 (14)	00:00:03		
23	VIEW		337K	4286K		142 (14)	00:00:01		
24	TABLE ACCESS FULL	SYS_TEMP_OFDA7485C_6A66C42E	337K	3956K		142 (14)	00:00:01		
* 25	SORT JOIN		22M	277M	855M	65084 (16)	00:02:40		
26	VIEW		22M	277M		16808 (12)	00:00:42		
27	TABLE ACCESS FULL	SYS_TEMP_OFDA7485B_6A66C42E	22M	1111M		16808 (12)	0		
* 28	SORT JOIN		22M	277M	855M	65084 (16)	00:02:40		
29	VIEW		22M	277M		16808 (12)	00:00:42		
30	TABLE ACCESS FULL	SYS_TEMP_OFDA7485B_6A66C42E	22M	1111M		16808 (12)	0		

Optimizer Plans (making it better)

```

WITH ed AS (SELECT object_id, element_id, element_name, element_type, useid
             FROM pmcm.element_detail
             WHERE element_type = 'CELL'
             AND current_ind = 'Y'),
     mn AS (SELECT parent_id, object_id
             FROM pmcm.mark_netw_hierarchy
             WHERE current_ind = 'Y'
             AND hierarchy_type = 'NETWORK'
             AND parent_type IN ('BSC', 'RNC'))
SELECT COUNT(*)
FROM ed e1_1, ed e1_2, ed e2_1, ed e2_2, mn h1, mn h2
WHERE e1_1.object_id = h1.parent_id AND e1_2.object_id = h1.object_id
AND e2_1.object_id = h2.parent_id AND e2_2.object_id = h2.object_id
AND e1_1.element_name = e2_1.element_name
AND e1_1.element_id = e2_1.element_id
AND e1_2.element_name = e2_2.element_name
AND e1_2.element_id = e2_2.element_id
AND e1_2.useid LIKE '%%'
AND e2_2.useid NOT LIKE '%%';

```

Id	Operation	Name	Rows	Bytes	TempSpc	Cost (%CPU)	Time
0	SELECT STATEMENT		1	214		100K (6)	00:04:08
1	HASH UNIQUE		1	214		100K (6)	00:04:08
* 2	HASH JOIN		1	214	12M	100K (6)	00:04:08
3	PARTITION HASH ALL		337K	9231K		3514 (15)	00:00:09
* 4	TABLE ACCESS FULL	MARK_NETW_HIERARCHY	337K	9231K		3514 (15)	00:00:00
* 5	HASH JOIN		207K	36M	22M	95860 (6)	00:03:56
6	PARTITION RANGE ALL		586K	15M		16233 (2)	00:00:40
7	TABLE ACCESS BY LOCAL INDEX ROWID	ELEMENT_DETAIL	586K	15M		16233	??:??:??
* 8	INDEX SKIP SCAN	ED_ET_TECH_CI	586K			12791 (1)	00:00:3?
* 9	HASH JOIN		207K	31M	22M	77982 (7)	00:03:12
10	PARTITION RANGE ALL		586K	15M		16233 (2)	00:00:40
11	TABLE ACCESS BY LOCAL INDEX ROWID	ELEMENT_DETAIL	586K	15M		16233	??:??:??
* 12	INDEX SKIP SCAN	ED_ET_TECH_CI	586K			12791 (1)	00:00:??
* 13	HASH JOIN		179K	22M	12M	60372 (8)	00:02:29
14	PARTITION HASH ALL		337K	9231K		3514 (15)	00:00:09
* 15	TABLE ACCESS FULL	MARK_NETW_HIERARCHY	337K	9231K		3514 (15)	00:00:??
* 16	HASH JOIN		184K	17M	10M	55886 (8)	00:02:18
17	PARTITION RANGE ALL		184K	9008K		37137 (8)	00:01:32
* 18	TABLE ACCESS FULL	ELEMENT_DETAIL	184K	9008K		37137 (8)	00:01:32
19	PARTITION RANGE ALL		576K	28M		17383 (8)	00:00:43
* 20	TABLE ACCESS BY LOCAL INDEX ROWID	ELEMENT_DETAIL	576K	28M		17383 (8)	??:??:??
* 21	INDEX SKIP SCAN	ED_ET_TECH_CI	583K			13939 (9)	00:00:35

INSERT WHEN

INSERT

```
WHEN (deptno=10) THEN
    INTO emp_10 (empno,ename,job,mgr,sal,deptno)
    VALUES (empno,ename,job,mgr,sal,deptno)
WHEN (deptno=20) THEN
    INTO emp_20 (empno,ename,job,mgr,sal,deptno)
    VALUES (empno,ename,job,mgr,sal,deptno)
WHEN (deptno=30) THEN
    INTO emp_30 (empno,ename,job,mgr,sal,deptno)
    VALUES (empno,ename,job,mgr,sal,deptno)
ELSE
    INTO leftover (empno,ename,job,mgr,sal,deptno)
    VALUES (empno,ename,job,mgr,sal,deptno)
SELECT * FROM emp;
```

INSERT ALL

```
INSERT ALL  
  INTO ap_cust  
    VALUES (customer_id, program_id, delivered_date)  
  INTO ap_orders  
    VALUES (order_date, program_id)  
SELECT program_id, delivered_date, customer_id, order_date  
FROM airplanes;
```


INSERT RETURNING

```
DECLARE
  x emp.empno%TYPE;
  r rowid;
BEGIN
  INSERT INTO emp
    (empno, ename)
  VALUES
    (seq_emp.NEXTVAL, 'Harmony')
  RETURNING rowid, empno
  INTO r, x;

  dbms_output.put_line(r);
  dbms_output.put_line(x);
END;
/
```

INSERT (PL/SQL)

```
DECLARE
```

```
    trec  t%ROWTYPE;
```

```
BEGIN
```

```
    trec.table_name := 'NEW';
```

```
    trec.tablespace_name := 'NEW_TBSP';
```

```
    INSERT INTO t
```

```
    VALUES trec;
```

```
    COMMIT;
```

```
END;
```

```
/
```

UPDATE: SET ROW

```
DECLARE
    trec  t%ROWTYPE;
BEGIN
    trec.table_name := 'DUAL';
    trec.tablespace_name := 'NEW_TBSP';

    UPDATE t
    SET ROW = trec
    WHERE table_name = 'DUAL';

    COMMIT;
END;
/
```

UPDATE RETURNING

```
var bnd1 NUMBER  
var bnd2 VARCHAR2(30)
```

```
UPDATE employees  
SET job_id = 'SA_MAN',  
    salary = salary + 1000,  
    department_id = 140  
WHERE last_name = 'Jones'  
RETURNING salary*0.25, last_name  
INTO :bnd1, :bnd2
```

```
print bnd1  
print bnd2
```

DELETE FROM SELECT

```
DELETE FROM (  
    SELECT * FROM t WHERE table_name LIKE '%MAP');
```

DELETE RETURNING

```
DECLARE
  r urowid;
BEGIN
  DELETE FROM t
  WHERE rownum = 1
  RETURNING rowid INTO r;

  dbms_output.put_line(r);
END;
/
```

Basic PL/SQL

- Explicitly declare CONSTANTS
- CONTINUE clause
- Native compilation
- NOCOPY compiler hint
- PL/SQL Warnings
- PRAGMA INLINE
- PRAGMA SERIALLY REUSABLE
- Result Cache
- Triggers
- Array Processing
 - BULK COLLECT & FORALL
 - SAVE EXCEPTIONS

Basic PL/SQL

- Table Triggers
 - COMPOUND
 - FOLLOWS clause
 - OF clause
 - WHEN clause

CONSTANT Optimization

- Identify constants as constants ... don't define them as variables if the value will not vary
- Using the constant keyword can, under some circumstances, tell the PL/SQL compiler that particular optimizations are safe where, without this information, they would have to be assumed to be unsafe.
- Reduces opportunities for SQL Injection

Edition Based Redefinition

- Editions
- Editioning Views
- Cross-Edition Triggers

GLOGIN.SQL

```
COL column_name FORMAT a30
COL data_type FORMAT a20
COL file_name FORMAT a60
COL name FORMAT a30
COL object_name FORMAT a30
COL owner FORMAT a25
COL segment_name FORMAT a30
COL service_name FORMAT a30
COL triggering_event FORMAT a35
COL value FORMAT a30

DEFINE _EDITOR=VI

SET ARRAYSIZE 500
SET DEFINE OFF
SET LINESIZE 121
SET LONG 1000000
SET PAGESIZE 45
SET SERVEROUTPUT ON
SET TRIM ON
SET TRIMSPOOL ON

SET sqlprompt '_CONNECT_IDENTIFIER> '

ALTER SESSION SET NLS_DATE_FORMAT = 'DD-MON-YYYY HH24:MI:SS';
ALTER SESSION SET PLSQL_WARNINGS='ENABLE:ALL';
```

Craft Your Own DBA Role

```
-- the default Oracle DBA role will be granted to no users: ever!  
prompt Creating X2 DBA Role
```

```
CREATE ROLE x2dba;  
GRANT advisor TO x2dba;  
GRANT alter any index TO x2dba;  
GRANT alter any table TO x2dba;  
GRANT alter database TO x2dba;  
GRANT alter profile TO x2dba;  
GRANT alter system TO x2dba;  
GRANT alter session TO x2dba;  
GRANT alter tablespace TO x2dba;  
GRANT analyze any TO x2dba;  
GRANT analyze any dictionary TO x2dba;  
GRANT create any edition TO x2dba;  
GRANT create any index TO x2dba;  
GRANT create any job TO x2dba;  
GRANT create any procedure TO x2dba;  
GRANT create any synonym TO x2dba;  
GRANT create any table TO x2dba;  
GRANT create any view TO x2dba;  
GRANT create external job TO x2dba;  
GRANT create session TO x2dba;  
GRANT create tablespace TO x2dba;  
GRANT drop any edition TO x2dba;  
GRANT drop any index TO x2dba;  
GRANT drop any procedure TO x2dba;  
GRANT drop any synonym TO x2dba;  
GRANT drop any table TO x2dba;  
GRANT drop user TO x2dba;  
GRANT flashback any table TO x2dba;  
GRANT grant any object privilege TO x2dba;  
GRANT manage scheduler TO x2dba;  
...
```

Create Customized Profiles

```
CREATE PROFILE readonly LIMIT
COMPOSITE_LIMIT          UNLIMITED
CONNECT_TIME             600
CPU_PER_CALL             UNLIMITED
CPU_PER_SESSION          UNLIMITED
FAILED_LOGIN_ATTEMPTS    3
IDLE_TIME                60
LOGICAL_READS_PER_CALL   UNLIMITED
LOGICAL_READS_PER_SESSION UNLIMITED
PASSWORD_GRACE_TIME      7
PASSWORD_LIFE_TIME       60
PASSWORD_LOCK_TIME       2
PASSWORD_REUSE_MAX       0
PASSWORD_REUSE_TIME      0
PASSWORD_VERIFY_FUNCTION verify_function_11g
PRIVATE_SGA              UNLIMITED
SESSIONS_PER_USER        2;
```

```
CREATE PROFILE mechid LIMIT
COMPOSITE_LIMIT          UNLIMITED
CONNECT_TIME             UNLIMITED
CPU_PER_CALL             UNLIMITED
CPU_PER_SESSION          UNLIMITED
FAILED_LOGIN_ATTEMPTS    4
IDLE_TIME                1440
LOGICAL_READS_PER_CALL   UNLIMITED
LOGICAL_READS_PER_SESSION UNLIMITED
PASSWORD_GRACE_TIME      30
PASSWORD_LIFE_TIME       60
PASSWORD_LOCK_TIME       1
PASSWORD_REUSE_MAX       0
PASSWORD_REUSE_TIME      0
PASSWORD_VERIFY_FUNCTION NULL
PRIVATE_SGA              UNLIMITED
SESSIONS_PER_USER        UNLIMITED;
```

...

Lock Accounts ... use Proxy Users

```
-- presentment (PCI)
CREATE USER
IDENTIFIED BY "N0way!n"
DEFAULT TABLESPACE pat_tbs
TEMPORARY TABLESPACE temp
QUOTA 0 ON SYSTEM
QUOTA 0 ON SYSAUX
QUOTA UNLIMITED ON pat_tbs
PROFILE x2opera
ACCOUNT LOCK;

GRANT x2readonly TO pat;

CREATE USER events
IDENTIFIED BY "N0way!n"
DEFAULT TABLESPACE events_tbs
TEMPORARY TABLESPACE temp
QUOTA UNLIMITED ON events_tbs
PROFILE x2opera
ACCOUNT LOCK;

GRANT x2readonly TO events;

CREATE USER c_dmorgan
IDENTIFIED BY oracle1
DEFAULT TABLESPACE opera_tbs
TEMPORARY tablespace temp
QUOTA UNLIMITED ON opera_tbs
PROFILE dbateam;

GRANT create session, alter user TO c_dmorgan;
ALTER USER c_dmorgan GRANT CONNECT THROUGH dbadmin;
AUDIT CONNECT BY dbadmin ON BEHALF OF c_dmorgan;

...
```

Discussion

Developer Practices

Unnecessary SQL

```
PROCEDURE get_records(pCustID IN NUMBER) AUTHID DEFINER IS
  vCount PLS_INTEGER;
BEGIN
  SELECT COUNT(*)
  INTO vCount
  FROM customers
  WHERE cust_id = pCustID;

  IF vCount > 0 THEN
    process_customer_recs(pCustID);
  END IF;
END get_records;
/
```

```
PROCEDURE process_customer_recs(pCustID IN NUMBER) AUTHID DEFINER IS
  CURSOR cust_cur IS
  SELECT transno, trans_type, transdate
  FROM customers
  WHERE cust_id = pCustID;
BEGIN
  ... do stuff ...
END process_customer_recs;
/
```


Unnecessary Dynamic SQL

```
PROCEDURE gather_schema_stats(pschema IN VARCHAR2, pGlobalOption IN BOOLEAN := TRUE) IS
BEGIN
    IF pschema IS NOT NULL THEN
        DBMS_STATS.UNLOCK_SCHEMA_STATS (pschema);

        IF pGlobalOption = TRUE THEN
            EXECUTE IMMEDIATE 'BEGIN DBMS_STATS.GATHER_SCHEMA_STATS(OWNNAME => ''' ||
                pschema || ''', ' ||
                ' ESTIMATE_PERCENT => 25, METHOD_OPT => ''FOR ALL INDEXED COLUMNS SIZE 254'', DEGREE => 4, ' ||
                ' CASCADE => TRUE, granularity => ''GLOBAL'', OPTIONS => ''GATHER STALE'', no_invalidate => TRUE); END;';
        ELSE
            EXECUTE IMMEDIATE 'BEGIN DBMS_STATS.GATHER_SCHEMA_STATS(OWNNAME => ''' ||
                pschema || ''', ' ||
                ' ESTIMATE_PERCENT => 25, METHOD_OPT => ''FOR ALL INDEXED COLUMNS SIZE 254'', DEGREE => 4, ' ||
                ' CASCADE => TRUE, OPTIONS => ''GATHER STALE'', no_invalidate => TRUE); END;';
        END IF;

        add_to_log_table(clogInformation, 827533, 'Schema stats gathered for schema ' || upper(pschema));
        -- Lock schema stats
        DBMS_STATS.LOCK_SCHEMA_STATS (pschema);
    END IF;
EXCEPTION
    WHEN OTHERS THEN
        --log error in COMMON_LOGS table
        add_to_log_table(clogmajor, 827516, 'Error in Gather_Schema_stats for schema ' ||
            upper(pschema) || ' - message:' || SQLERRM);
END;
```

```
dbms_stats.gather_schema_stats(pschema, ESTIMATE_PERCENT => 25,
METHOD_OPT => 'FOR ALL INDEXED COLUMNS SIZE 254', DEGREE => 4,
CASCADE => TRUE, granularity => 'GLOBAL', OPTIONS => 'GATHER STALE', no_invalidate => TRUE);

dbms_stats.gather_schema_stats(pschema, ESTIMATE_PERCENT => 25,
METHOD_OPT => 'FOR ALL INDEXED COLUMNS SIZE 254', DEGREE => gcDegree,
CASCADE => TRUE, granularity => 'GLOBAL', OPTIONS => 'GATHER STALE', no_invalidate => TRUE);
```

Unnecessary Concatenation

```
FUNCTION delete_partitions_in_range(pschema    IN VARCHAR2,
                                   ptable     IN VARCHAR2,
                                   pretention IN INTEGER,
                                   parttype   IN INTEGER) RETURN PLS_INTEGER

IS
    ... variable declarations ...

BEGIN
    --add starting processing
    IF ptable <> 'COMMON_LOGS' THEN
        add_to_log_table(cloginformation,
                        827600,
                        upper(pschema) || '.' || upper(ptable) ||
                        ' - Processing (Delete partitions in range)');
    END IF;
    ... code here ...
    LOOP
        BEGIN
            ... code here ...
        EXCEPTION
            WHEN OTHERS THEN
                add_to_log_table(clogmajor,
                                827604,
                                upper(pschema) || '.' || upper(ptable) ||
                                ' - Unable to Delete Partition ' ||
                                partTable(i).Partition_Name || ':' ||
                                SQLERRM);

                RAISE;
            END;
        END LOOP;

        --Add log entry to summarise the partitions added
        IF ((vSummaryMinDate < vmindate) AND (ptable <> 'COMMON_LOGS')) THEN
            add_to_log_table(cloginformation,
                            827605,
                            upper(pschema) || '.' || upper(ptable) ||
                            ' - Deleted partitions from ' ||
                            to_char(vSummaryMinDate, 'YYYY-MM-DD HH24:MI:SS') ||
                            ' to ' || to_char(vmindate, 'YYYY-MM-DD HH24:MI:SS'));
        END IF;
        RETURN partTable.count;
    END;
```

Replaced by a Variable Assignment

```
FUNCTION delete_partitions_in_range(pschema    IN VARCHAR2,
                                   ptable     IN VARCHAR2,
                                   pretention IN INTEGER,
                                   parttype   IN INTEGER) RETURN PLS_INTEGER

IS
    ... variable declarations ...
    cObject VARCHAR2(61) := UPPER(pschema || '.' || ptable);
BEGIN
    --add starting processing
    IF ptable <> 'COMMON_LOGS' THEN
        add_to_log_table(cloginformation,
                        827600,
                        cObject ||
                        ' - Processing (Delete partitions in range)');
    END IF;
    ... code here ...
    LOOP
        BEGIN
            ... code here ...
        EXCEPTION
            WHEN OTHERS THEN
                add_to_log_table(clogmajor,
                                827604,
                                cObject ||
                                ' - Unable to Delete Partition ' ||
                                partTable(i).Partition_Name || ':' ||
                                SQLERRM);

                RAISE;
            END;
        END LOOP;

        --Add log entry to summarise the partitions added
        IF ((vSummaryMinDate < vmindate) AND (ptable <> 'COMMON_LOGS')) THEN
            add_to_log_table(cloginformation,
                            827605,
                            cObject ||
                            ' - Deleted partitions from ' ||
                            to_char(vSummaryMinDate, 'YYYY-MM-DD HH24:MI:SS') ||
                            ' to ' || to_char(vmindate, 'YYYY-MM-DD HH24:MI:SS'));
        END IF;
        RETURN partTable.count;
    END;
```

Deterministic Functions

- Indicates that the function returns the same result value whenever it is called with the same values for its parameters

```
SQL> SELECT owner, COUNT(*)
2  FROM dba_arguments
3  WHERE sequence = 0
4  AND owner NOT LIKE '%SYS%'
5  GROUP BY owner
6  ORDER BY 1;
```

OWNER	COUNT (*)
PMCM	16
OSSBACKEND	3
OSS_DICTIONARY	8
SRE	6
WEBWIZ	2
ZZYZX	17

```
SQL> SELECT owner, COUNT(*)
2  FROM dba_source
3  WHERE owner NOT LIKE '%SYS%'
4  AND UPPER(text) LIKE '%DETERMINISTIC%'
5  GROUP BY owner
6  ORDER BY 1;
```

OWNER	COUNT (*)
ZZYZX	1
SRE	79

```
CREATE OR REPLACE FUNCTION get_date_determ RETURN DATE DETERMINISTIC IS
BEGIN
    RETURN TRUNC(SYSDATE);
END get_date_determ;
/
```



GV\$ Dynamic Performance Views

- V\$ is the local instance
- GV\$ view hit all instances
 - Creating delays and increased cache fusion interconnect traffic
- Do not query a gv\$ unless you really need to collect information from more than one node

```
SELECT name FROM v$database;
```

```
SELECT name FROM gv$database;
```

Result Cache

SQL ordered by Executions

- Total Executions: 29,717,627
- Captured SQL account for 77.4% of Total

Executions	Rows Processed	Rows per Exec	CPU per Exec (s)	Elap per Exec (s)	SQL Id	SQL Module	SQL Text
10,128,178	2,506,529	0.25	0.00	0.00	932srzgt1krc33	ASN_07B_DIP(004110016)	SELECT NE_TIMEZONE FROM CMPM.E...
7,576,759	7,579,197	1.00	0.00	0.00	1h698sb62un99	asci_56_RANAPProtocolStats(016110006)	SELECT DISTINCT NE_TIMEZONE FR...
3,914,621	3,848,268	0.98	0.00	0.00	5tbzddgguu8cc	asci_56_RANAPProtocolStats(016110006)	SELECT SYS_VERSION FROM CMPM.T...
311,645	311,604	1.00	0.00	0.00	7gtztzv329wg0		select c.name, u.name from co...
301,428	301,325	1.00	0.00	0.00	36s446f9cnwhw		SELECT C.NAME FROM COL\$ C WHER...
200,692	200,669	1.00	0.00	0.00	4vs91dcv7u1p6	OMS	insert into sys.aud\$(sessioni...
65,044	65,035	1.00	0.00	0.00	fz9xwpt2cvt0k		SELECT par_type, param_clob, ...
64,949	3,945,482	60.75	0.00	0.00	f5ra7dru5fk5n	XML_P7R_RNC_RCS(003110008)	SELECT NAME, PATH, READ, WR...
64,801	64,807	1.00	0.00	0.00	fhzi09a7fmrnb	XML_V7I_IN_LP_DC(00811000V)	SELECT DBTIMEZONE, LENGTH(DBT...
64,632	64,542	1.00	0.00	0.00	15inrrb6016nd	XML_V7I_IN_LP_DC(00811000V)	SELECT SESSIONTIMEZONE, LENGT...

```
CREATE OR REPLACE FUNCTION rcache(p_srvr_id IN servers.srvr_id%TYPE) RETURN BOOLEAN
RESULT_CACHE RELIES_ON(servers) IS
```

```
    srvrow servers%ROWTYPE;
BEGIN
    SELECT *
    INTO srvrow
    FROM servers
    WHERE srvr_id = p_srvr_id;
    RETURN TRUE;
EXCEPTION
    WHEN OTHERS THEN
        RETURN FALSE;
END rcache;
/
```

```
SELECT /*+ RESULT_CACHE */ srvr_id
FROM (
    SELECT srvr_id, SUM(cnt) SUMCNT
    FROM (
        SELECT DISTINCT srvr_id, 1 AS CNT
        FROM servers
        UNION ALL
        SELECT DISTINCT srvr_id, 1
        FROM serv_inst)
    GROUP BY srvr_id)
WHERE sumcnt = 2;
```

Sequence Optimization

- High volume insert intensive applications using sequence generated keys add extra overhead
- NOCACHE is the worst for performance
- CACHE NOORDER is the best for performance
- Services may help improve performance even for CACHE NOORDER sequences if the insert application uses a service with only one Preferred Instance
- This eliminates the Interconnect traffic and synchronization overhead required to coordinate the next value from amongst all the instances when using CACHE ORDER

Sequence Caching

```
SQL> SELECT sequence_owner, cache_size, COUNT(*)
  2  FROM dba_sequences
  3  WHERE sequence_owner IN ('ADMIN', 'PMCM', 'OPS', 'OSSBACKEND', 'OSS_DICTIONARY', 'SRE', 'WEBWIZ', 'ZZYZX')
  4  GROUP BY sequence_owner, cache_size
  5  ORDER BY 1,2;
```

SEQUENCE_OWNER	CACHE_SIZE	COUNT (*)
ADMIN	20	1
PMCM	1000	2
OPS	0	1
OSSBACKEND	0	11
OSS_DICTIONARY	0	23
SRE	0	33
SRE	20	7
WEBWIZ	20	24
ZZYZX	0	21
ZZYZX	20	44

“You would be amazed what setting a sequence cache via alter sequence to 100,000 or more can do during a large load -- amazed.”

~ Tom Kyte

Sequence Ordering

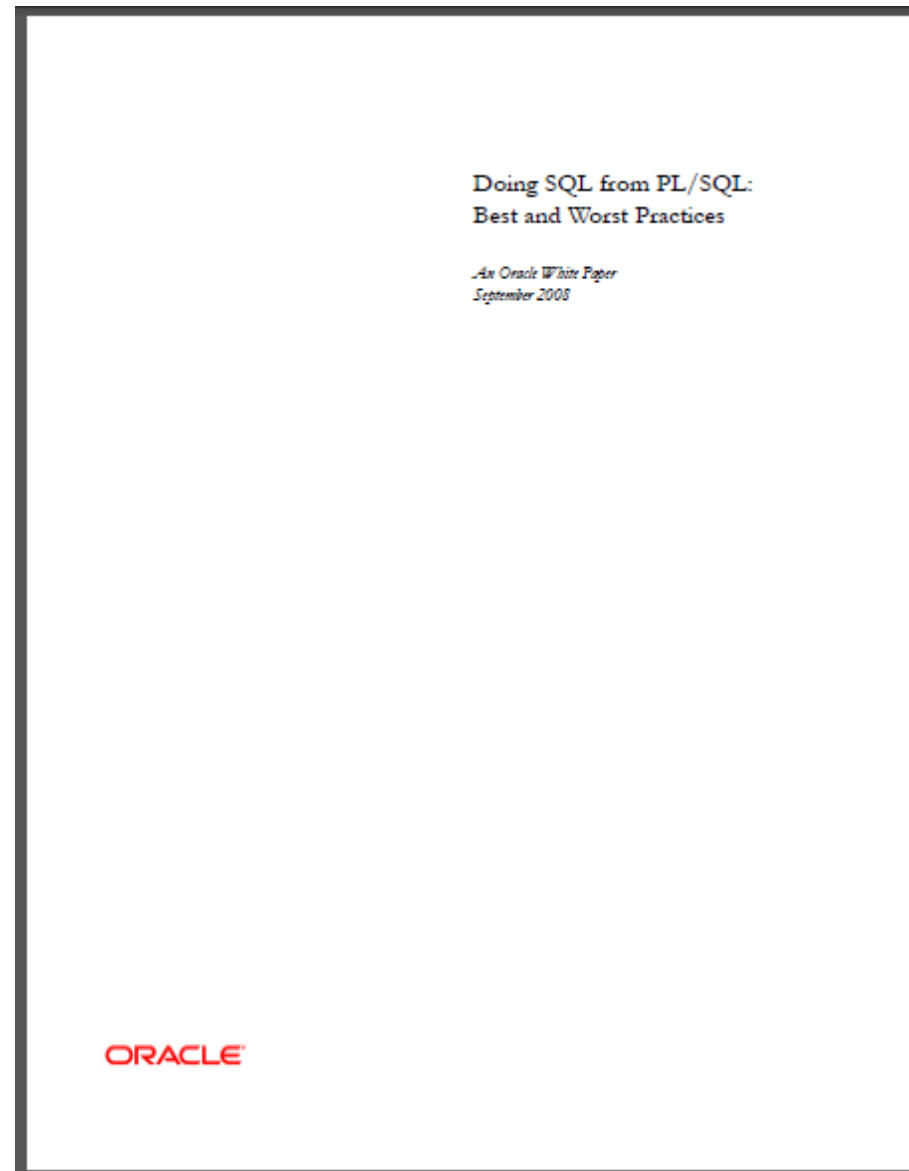
```
SQL> SELECT sequence_owner, order_flag, COUNT(*)
  2  FROM dba_sequences
  3  WHERE sequence_owner IN ('ADMIN', 'PMCM', 'OPS', 'OSSBACKEND', 'OSS_DICTIONARY', 'SRE', 'WEBWIZ', 'ZZYZX')
  4  GROUP BY sequence_owner, order_flag
  5  ORDER BY 1,2;
```

SEQUENCE_OWNER	O	COUNT (*)
ADMIN	N	1
PMCM	N	2
OPS	N	1
OSSBACKEND	N	5
OSSBACKEND	Y	6
OSS_DICTIONARY	N	7
OSS_DICTIONARY	Y	16
SRE	N	39
SRE	Y	1
WEBWIZ	Y	24
ZZYZX	N	59
ZZYZX	Y	6

Built-in PL/SQL Packages

- DBMS_APPLICATION_INFO
- DBMS_ASSERT
- DBMS_DB_VERSION
- DBMS_METADATA
- DBMS_RESULT_CACHE
- OWA_UTIL

Brynn's White Paper



Two More Presentations

- Thursday 31, May: 13:00 - 14:00

Oracle Database Coding Practices for Security and Data Security

- Thursday 31, May: 14:10 - 15:10

Oracle Database Appliances

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



All demo code at morganslibrary.org

Thank you