# Xtreme SQL Tuning: The Tuning Limbo

Iggy Fernandez
Database Specialists
Session #553

# Speaker Qualifications

- Oracle DBA at Database Specialists
- Editor of the Journal of the Northern California Oracle Users Group (NoCOUG)
- Author of *Beginning Oracle Database 11*g *Administration* (Apress, 2009)

# Definition of SQL Efficiency

- Amount of computing resources used in producing the output
- Elapsed time is not a good proxy
- Logical reads is a good proxy

# Identifying Inefficient SQL Statements

- Enterprise Manager, SQL Developer, Toad
- Tracing sessions
  - dbms_monitor.session_trace_enable
  - dbms_monitor.session_trace_disable
- Statspack reports
- Diagnostic Pack
  - AWR
  - ADDM

# Causes of Inefficient SQL

- Optimizer limitations
- Many ways to write a query
- Failure to use advanced features
  - Analytic Functions
- Ad-hoc queries
- Poor logical and physical database design
- Inadequate database maintenance

# Other Performance Inhibitors

- Hardware limitations
- Mixed workloads
- Contention

# Ways to Improve SQL—Physical Database Design

- Indexes
  - B-tree indexes
  - Reverse key indexes
  - Function-based indexes
  - Indexes on virtual columns
  - Bitmap indexes
- Clusters
- IOTs
- Partitioning

# Ways To Improve SQL—Hints

- LEADING
- ORDERED
- INDEX
- FULL
- NO_MERGE
- USE_NL, USE_HASH, USE_MERGE

# Ways To Improve SQL—Statistics

- ENABLE_JOB, DISABLE_JOB, START_JOB
- GATHER_*_STATS
- DELETE_*_STATS
- EXPORT_*_STATS
- IMPORT_*_STATS
- RESTORE_*_STATS
- LOCK_*_STATS
- SET_*_PREFS

# Tuning By Example

```
CREATE TABLE my_tables AS
SELECT dba_tables.*
  FROM dba_tables;

CREATE TABLE my_indexes AS
SELECT dba_indexes.*
  FROM dba_tables, dba_indexes
 WHERE dba_tables.owner = dba_indexes.table_owner
   AND dba_tables.table_name = dba_indexes.table_name;
```

# Tables Which Have a Bitmap Index

```
EXEC :index_type := 'BITMAP';

SELECT DISTINCT my_tables.owner,
                my_tables.table_name,
                my_tables.tablespace_name
           FROM my_tables, my_indexes
          WHERE my_tables.owner =
my_indexes.table_owner
            AND my_tables.table_name =
my_indexes.table_name
            AND my_indexes.index_type = :index_type;
```

# Autotrace

```
ALTER SYSTEM FLUSH SHARED_POOL;
ALTER SYSTEM FLUSH BUFFER_CACHE;
Statistics
----------------------------------------------------
       1653  recursive calls
          0  db block gets
        498  consistent gets
        137  physical reads
          0  redo size
        645  bytes sent via SQL*Net to client
        381  bytes received via SQL*Net from clie
          2  SQL*Net roundtrips to/from client
         37  sorts (memory)
          0  sorts (disk)
          5  rows processed
```

# Autotrace

```
ALTER SYSTEM FLUSH BUFFER_CACHE;

Statistics
----------------------------------------------------
          0  recursive calls
          0  db block gets
        108  consistent gets
        104  physical reads
          0  redo size
        645  bytes sent via SQL*Net to client
        381  bytes received via SQL*Net from clie
          2  SQL*Net roundtrips to/from client
          0  sorts (memory)
          0  sorts (disk)
          5  rows processed
```

# Baseline

```
Statistics
----------------------------------------------------
          0  recursive calls
          0  db block gets
        108  consistent gets
          0  physical reads
          0  redo size
        645  bytes sent via SQL*Net to client
        381  bytes received via SQL*Net from clie
          2  SQL*Net roundtrips to/from client
          0  sorts (memory)
          0  sorts (disk)
          5  rows processed
```

# Execution Plan I

```
-----------------------------------------------
| Operation                | Name        | Buffers |
-----------------------------------------------
|  HASH UNIQUE             |             |     108 |
|   HASH JOIN              |             |     108 |
|    TABLE ACCESS FULL| MY_INDEXES |      58 |
|    TABLE ACCESS FULL| MY_TABLES  |      50 |
-----------------------------------------------
```

# Constraints

```
ALTER TABLE my_tables
ADD (CONSTRAINT my_tables_pk PRIMARY KEY (owner,
table_name));

ALTER TABLE my_indexes
ADD (CONSTRAINT my_indexes_pk PRIMARY KEY (owner,
index_name));

ALTER TABLE my_indexes
ADD (CONSTRAINT my_indexes_fk1 FOREIGN KEY
(table_owner, table_name) REFERENCES my_tables);
```

# Indexes

```
CREATE INDEX my_indexes_i1 ON my_indexes
(index_type);

CREATE INDEX my_indexes_fk1 ON my_indexes
(table_owner, table_name);
```

# Statistics

```
EXEC
DBMS_STATS.gather_table_stats(ownname=>'IFERNANDE
Z',tabname=>'MY_TABLES');
EXEC
DBMS_STATS.gather_table_stats(ownname=>'IFERNANDE
Z',tabname=>'MY_INDEXES');
EXEC
DBMS_STATS.gather_index_stats(ownname=>'IFERNANDE
Z',indname=>'MY_TABLES_PK');
EXEC
DBMS_STATS.gather_index_stats(ownname=>'IFERNANDE
Z',indname=>'MY_INDEXES_I1');
EXEC
DBMS_STATS.gather_index_stats(ownname=>'IFERNANDE
Z',indname=>'MY_INDEXES_FK1');
```

# Execution Plan II

| Operation | Name | Buffers |
|---|---|---|
| HASH UNIQUE | | 55 |
| HASH JOIN | | 55 |
| TABLE ACCESS BY INDEX ROWID | MY_INDEXES | 5 |
| INDEX RANGE SCAN | MY_INDEXES_I1 | 2 |
| TABLE ACCESS FULL | MY_TABLES | 50 |

# SQL Access Advisor

```
VARIABLE tuning_task VARCHAR2(32);
EXEC :tuning_task :=
dbms_sqltune.create_tuning_task (sql_id =>
'&sqlID');

EXEC dbms_sqltune.execute_tuning_task(task_name
=> :tuning_task);

SELECT DBMS_SQLTUNE.report_tuning_task
(:tuning_task) AS recommendations
  FROM DUAL;
```

# Recommendations

```
The execution plan of this statement can be
improved by creating one or more indices.

 Recommendation (estimated benefit: 100%)
 ----------------------------------------
 - Consider running the Access Advisor to improv
   the physical schema design or creating the
   recommended index.
   create index IFERNANDEZ.IDX$$_00470001 on
   IFERNANDEZ.MY_TABLES('OWNER',
   'TABLE_NAME','TABLESPACE_NAME');
```

# Hints

```
EXEC :index_type := 'BITMAP';

SELECT              /*+ INDEX(MY_INDEXES (INDEX_TYPE))
                        INDEX(MY_TABLES (OWNER TABLE_NAME))
                        LEADING(MY_INDEXES MY_TABLES)
                        USE_NL(MY_TABLES)
                    */
          DISTINCT my_tables.owner,
                   my_tables.table_name,
                   my_tables.tablespace_name
              FROM my_tables, my_indexes
             WHERE my_tables.owner = my_indexes.table_owner
               AND my_tables.table_name = my_indexes.table_name
               AND my_indexes.index_type = :index_type;
```

# Execution Plan III

| Operation | Name | Buffers |
|---|---|---|
| HASH UNIQUE | | 37 |
| NESTED LOOPS | | 37 |
| TABLE ACCESS BY INDEX ROWID | MY_INDEXES | 5 |
| INDEX RANGE SCAN | MY_INDEXES_I1 | 2 |
| TABLE ACCESS BY INDEX ROWID | MY_TABLES | 32 |
| INDEX UNIQUE SCAN | MY_TABLES_PK | 17 |

# Cluster

```
CREATE CLUSTER my_cluster (index_type
VARCHAR2(27))
SIZE 8192 HASHKEYS 5;
```

# Materialized View

```
CREATE MATERIALIZED VIEW LOG ON my_tables WITH
ROWID;
CREATE MATERIALIZED VIEW LOG ON my_indexes WITH
ROWID;
CREATE MATERIALIZED VIEW my_mv
CLUSTER my_cluster (index_type)
REFRESH FAST ON COMMIT ENABLE QUERY REWRITE AS
SELECT t.ROWID AS table_rowid,
       t.owner AS table_owner,
       t.table_name,
       t.tablespace_name,
       i.ROWID AS index_rowid,
       i.index_type
  FROM my_tables t, my_indexes i
 WHERE t.owner = i.table_owner
   AND t.table_name = i.table_name;
```

# Execution Plan IV

```
-----------------------------------------
| Operation           | Name   | Buffers |
-----------------------------------------
|  HASH UNIQUE        |        |       1 |
|   TABLE ACCESS HASH| MY_MV  |       1 |
-----------------------------------------
```

# Result Cache

```
SELECT          /*+ RESULT_CACHE */
       DISTINCT my_tables.owner,
                my_tables.table_name,
                my_tables.tablespace_name
           FROM my_tables, my_indexes
          WHERE my_tables.owner = my_indexes.table_owner
            AND my_tables.table_name = my_indexes.table_name
            AND my_indexes.index_type = :index_type;
```

# Execution Plan V

| Operation | Name |
|---|---|
| SELECT STATEMENT | |
| RESULT CACHE | afscr8p240b168b5az0dkd4k65 |
| HASH UNIQUE | |
| TABLE ACCESS HASH | MY_MV |

# Contact Information

Iggy Fernandez
Database Specialists, Inc.
388 Market Street, Suite 400
San Francisco, CA 94111

Tel: 415-344-0500 Ext. 43
Email: ifernandez@dbspecialists.com
Web: www.dbspecialists.com

# There's No Substitute For Experience

- Proven track record with emerging to Fortune 500 clients since 1995.
- Services and support plans tailored to your business needs and budget.
- Team of recognized industry experts and thought leaders.

**Database Specialists helps you increase uptime, improve performance, minimize risk, and reduce costs**

DatabaseSpecialists

# About Database Specialists

- Database Specialists, Inc. provides Oracle database consulting in Solaris, Linux, HP-UX, AIX, and Windows environments.
- Our DBA Pro offering and Database Rx™ tools provide remote database support and 24/7 coverage at an attractive price point.
- We specialize in short term projects including upgrades, performance tuning and health checks.
- Our Oracle DBAs each have a minimum of 10 years of Oracle experience with a focus on Oracle technology, mission-critical production support and RAC environments.
- Database Specialists is US-based.

**Database Specialists helps you increase uptime, improve performance, minimize risk, and reduce costs**

DatabaseSpecialists

# Please complete evaluation form

Iggy Fernandez

Xtreme SQL Tuning:

The Tuning Limbo

Session# 553

**iggy_fernandez@hotmail.com**