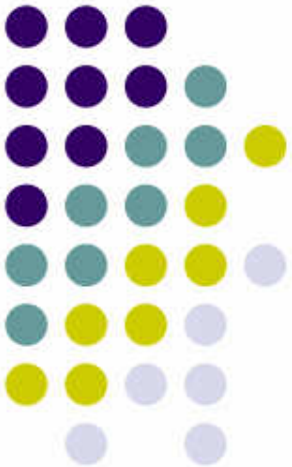
# 10g Tuning Highlights



# Presenter

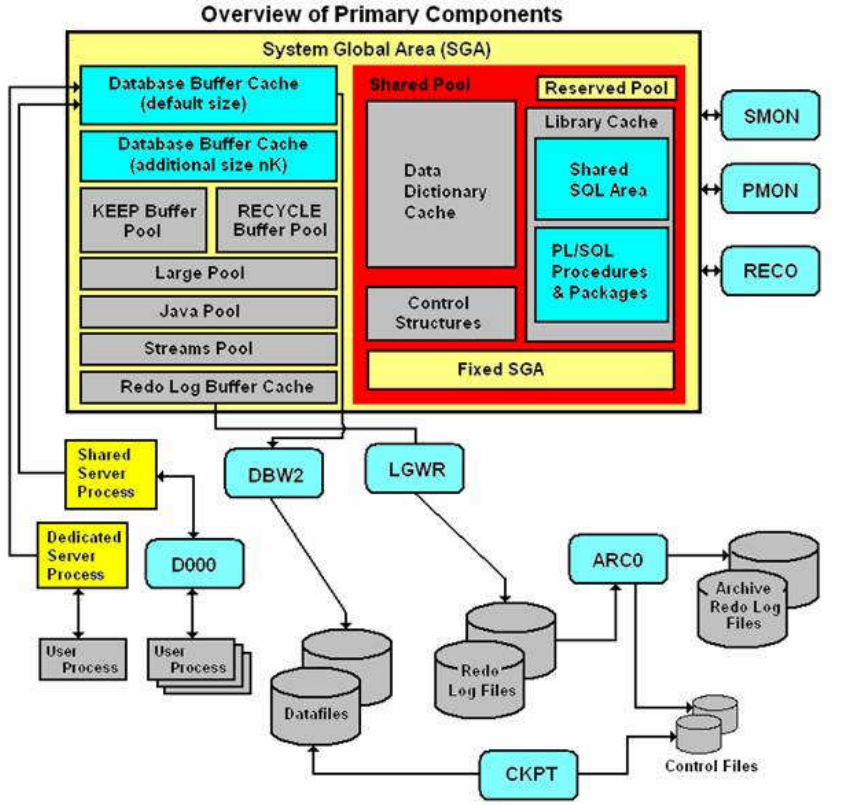
**JEREMY SCHNEIDER**

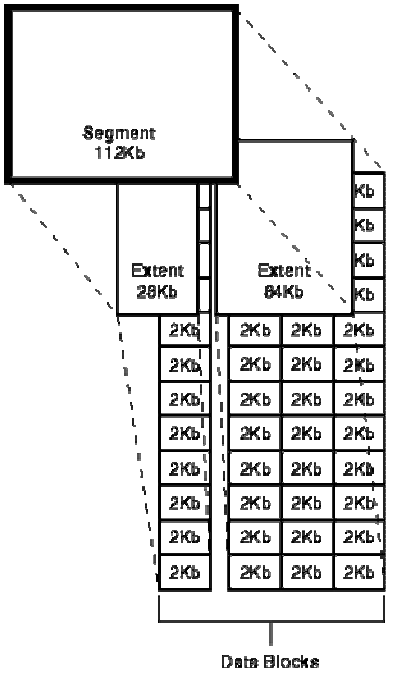
jeremy.schneider@ardentperf.com

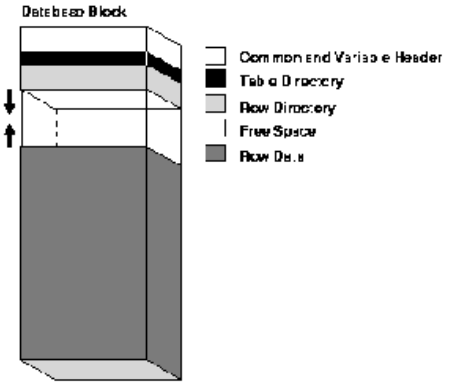
Senior Consultant,
ITC Technology Services
OCP, RAC since 2002, Systems Admin
and Developer in previous lives
http://www.ardentperf.com

# Fundamentals

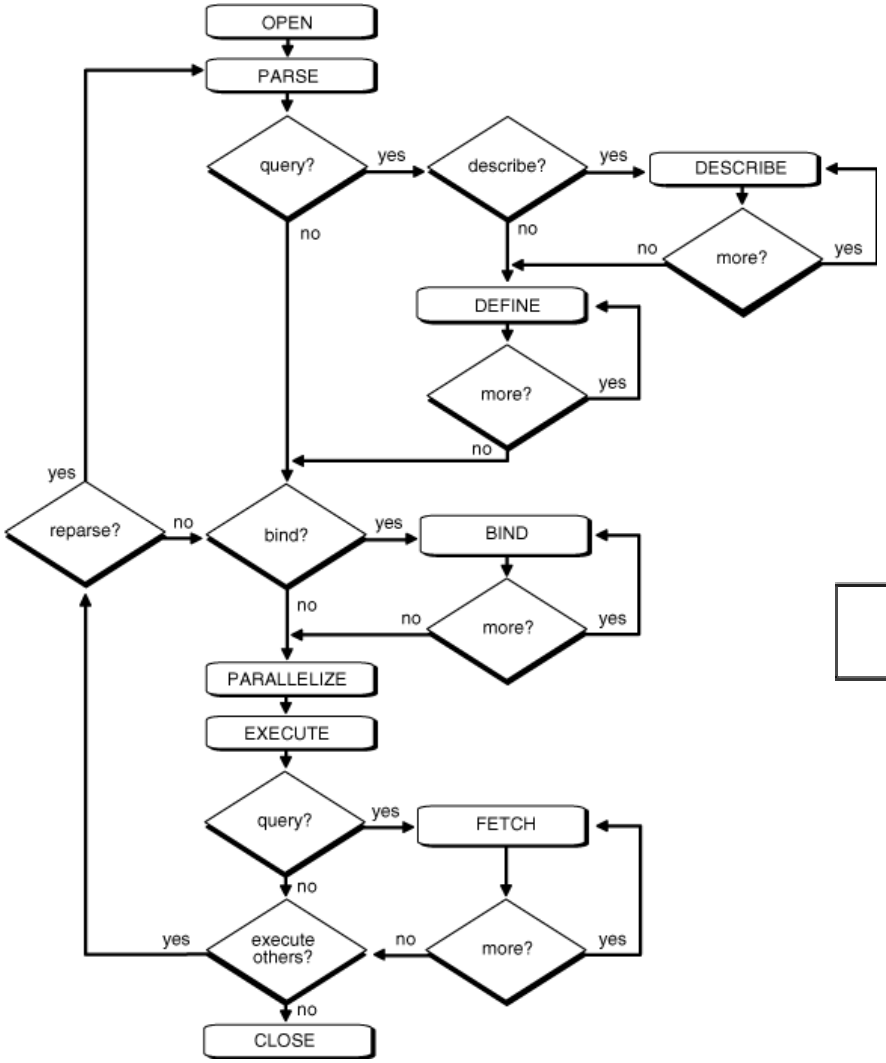
- Memory structures
- Storage structures

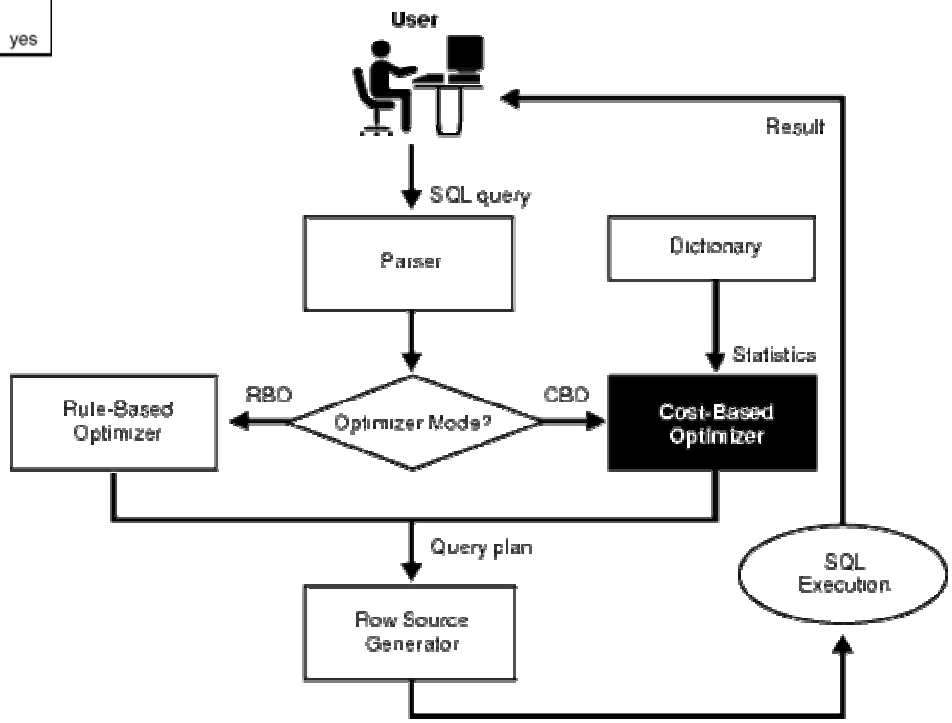


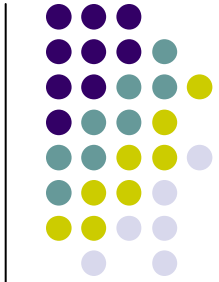



# RBO vs CBO


OPEN
PARSE
query?
yes
describe?
yes
DESCRIBE
no
no
more?
yes
no
DEFINE
more?
yes
no
yes
reparse?
no
bind?
yes
BIND
no
more?
yes
no
PARALLELIZE
EXECUTE
query?
yes
FETCH
no
yes
execute others?
no
more?
yes
no
CLOSE


User
Result
SQL query
Parser
Dictionary
Statistics
Rule-Based Optimizer
RBO
Optimizer Mode?
CBO
Cost-Based Optimizer
Query plan
Row Source Generator
SQL Execution

# RBO vs CBO



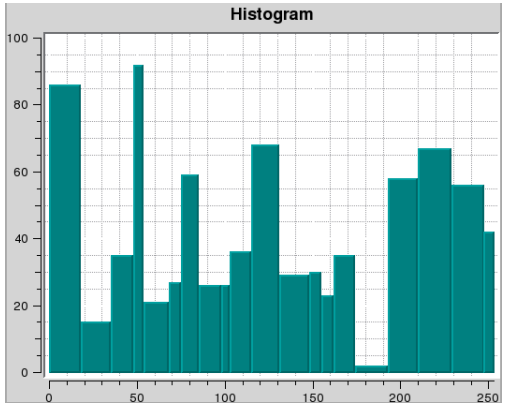
1. Single Row by Rowid
2. Single Row by Cluster Join
3. Single Row by Hash Cluster Key with Unique or Primary Key
4. Single Row by Unique or Primary Key
5. Clustered Join
6. Hash Cluster Key
7. Indexed Cluster Key
8. Composite Index
9. Single-Column Indexes
10. Bounded Range Search on Indexed Columns
11. Unbounded Range Search on Indexed Columns
12. Sort Merge Join
13. MAX or MIN of Indexed Column
14. ORDER BY on Indexed Column
15. Full Table Scan

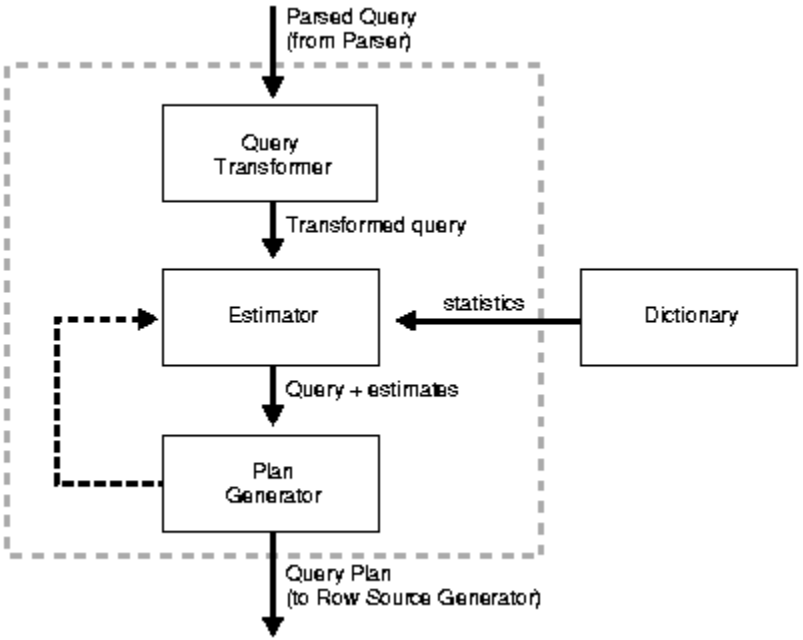
- Only single column indexes are ever merged.
- If all columns in an index are specified in the WHERE clause, that index will be used in preference to other indexes for which *some* columns are referenced.
- If multiple indexes can be applied to a WHERE clause, and they all have an equal number of columns specified, only the index created last will be used.
- If multiple columns of an index are being accessed with an = operator, that will override other operators such as LIKE or BETWEEN. Two ='s will override two ='s and a LIKE.
- A higher percentage of columns accessed will override a lower percentage of columns accessed. So generally, the optimizer will choose to use the index from which you specify the highest percentage of columns. However, as stated previously, all columns specified in a unique or primary key index will override the use of all other indexes.
- The RBO uses rules to select the driving table.
- If a WHERE clause has a column that is the leading column on any index, the rule-based optimizer will use that index. The exception is if a function is placed on the leading index column in the WHERE clause.

*Check out http://www.oreilly.com/catalog/orsqltunpr/chapter/excerpt.html

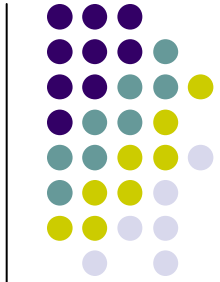
# RBO vs CBO

- Based on advanced mathematics and statistics
- Default 80,000 permutations (optimizer_max_permutations)



# RBO vs CBO



```
SQL> desc all_tables
 Name                          Null?    Type
 ----------------------------- -------- ------------
...
 NUM_ROWS                               NUMBER
 BLOCKS                                 NUMBER
 EMPTY_BLOCKS                           NUMBER
 AVG_SPACE                              NUMBER
 CHAIN_CNT                              NUMBER
 AVG_ROW_LEN                            NUMBER
...
 SAMPLE_SIZE                            NUMBER
 LAST_ANALYZED                          DATE
...
 GLOBAL_STATS                           VARCHAR2(3)
 USER_STATS                             VARCHAR2(3)
...
```

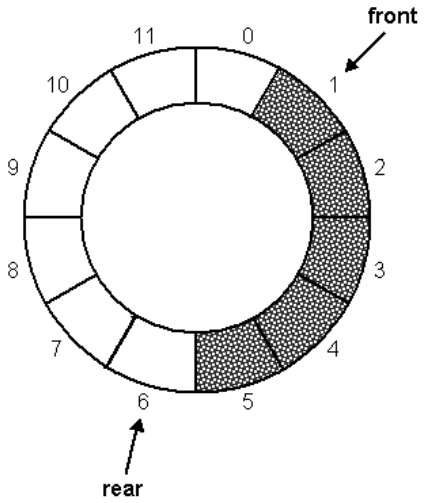
```
SQL> desc all_tab_columns
 Name                          Null?    Type
 ----------------------------- -------- ------------
...
 NUM_DISTINCT                           NUMBER
 LOW_VALUE                              RAW(32)
 HIGH_VALUE                             RAW(32)
 DENSITY                                NUMBER
 NUM_NULLS                              NUMBER
 NUM_BUCKETS                            NUMBER
 LAST_ANALYZED                          DATE
 SAMPLE_SIZE                            NUMBER
...
 GLOBAL_STATS                           VARCHAR2(3)
 USER_STATS                             VARCHAR2(3)
 AVG_COL_LEN                            NUMBER
...
 HISTOGRAM                              VARCHAR2(15)
```

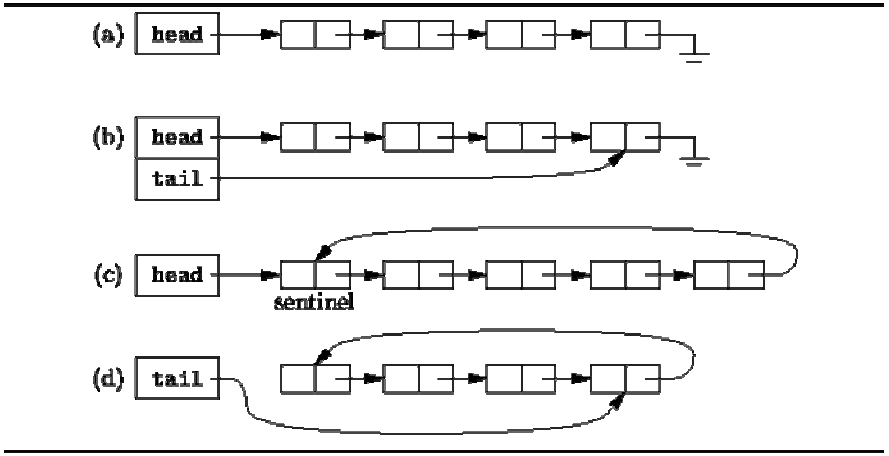
# Internal Structures & Algorithms

- Latches
- Enqueues
- Locks
- Pins

- Bitmaps
- Linked Lists
- Hash Maps

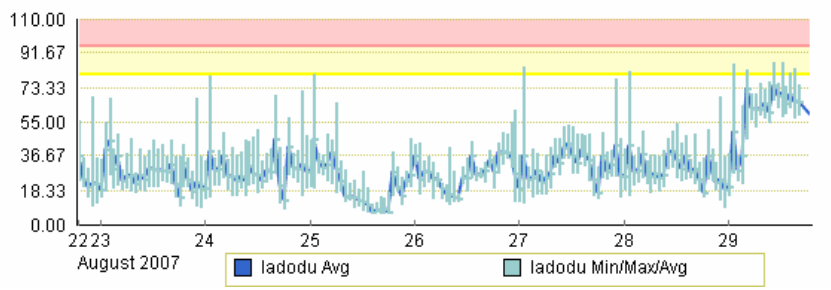
- Caching
- Sorting (opt,single,multi)

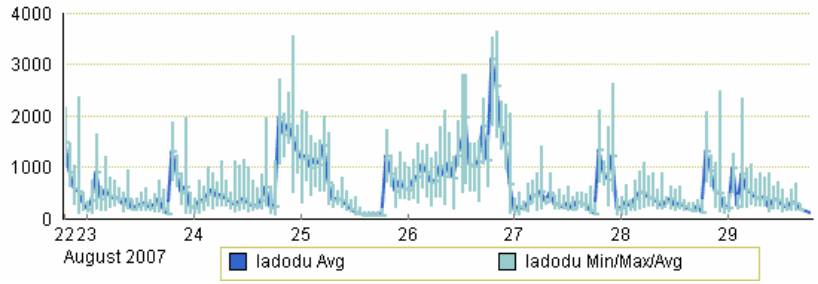


# System Health

- Statspack Reports or AWR Reports (10g)
- DBControl



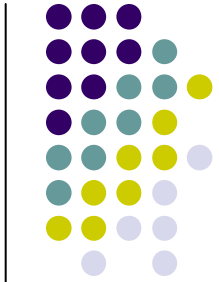

**Instance Efficiency Percentages (Target 100%)**

| | | | |
|---|---|---|---|
| Buffer Nowait %: | 99.85 | Redo NoWait %: | 100.00 |
| Buffer Hit %: | 91.42 | In-memory Sort %: | 100.00 |
| Library Hit %: | 92.60 | Soft Parse %: | 93.24 |
| Execute to Parse %: | 40.65 | Latch Hit %: | 99.21 |
| Parse CPU to Parse Elapsd %: | 19.74 | % Non-Parse CPU: | 92.83 |

**Top 5 Timed Events**

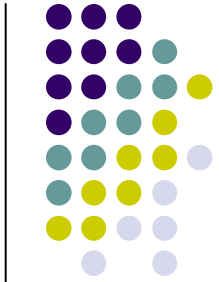
| Event | Waits | Time(s) | Avg Wait(ms) | % Total Call Time | Wait Class |
|---|---|---|---|---|---|
| CPU time | | 1,509 | | 30.1 | |
| db file scattered read | 154,959 | 841 | 5 | 16.8 | User I/O |
| latch: library cache | 1,707 | 416 | 244 | 8.3 | Concurrency |
| latch: shared pool | 1,965 | 389 | 198 | 7.8 | Concurrency |
| db file sequential read | 112,975 | 368 | 3 | 7.3 | User I/O |

# SQL Statement Tuning Methodology



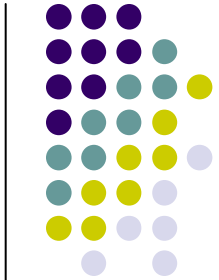
- Don’t guess!  Set goals and have a plan.
- Investigate scientifically.  Oracle is very highly instrumented.

# SQL Statement Top 10 Mistakes



1. Bad Connection Management
2. Bad Use of Cursors in the Shared Pool
3. Bad SQL
4. Nonstandard Initialization Parameters
5. Datafiles layed out poorly for I/O
6. Too few or too small redo logs (checkpoint errors)
7. Serialization – shortage of freelists, transactions slots or rollback segs
8. Inefficient full table scans
9. Large amount of recursive (SYS) SQL
10. Deployment/migration errors (e.g. missing indexes)

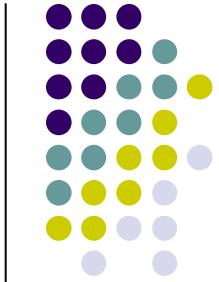
# SQL Statement Tuning



What can cause SQL execution to change?

1. Change in text of SQL statement (including hints)
2. Change in schema (indexes, parallel declarations, etc)
3. Change in object statistics
4. Change in system statistics
5. Change in system (init/spfile) or session parameters
6. Manipulation of Stored Outline or SQL Profile
7. Change in CBO code (i.e. upgrade or patch)

# 10g Tools for SQL Statement Tuning

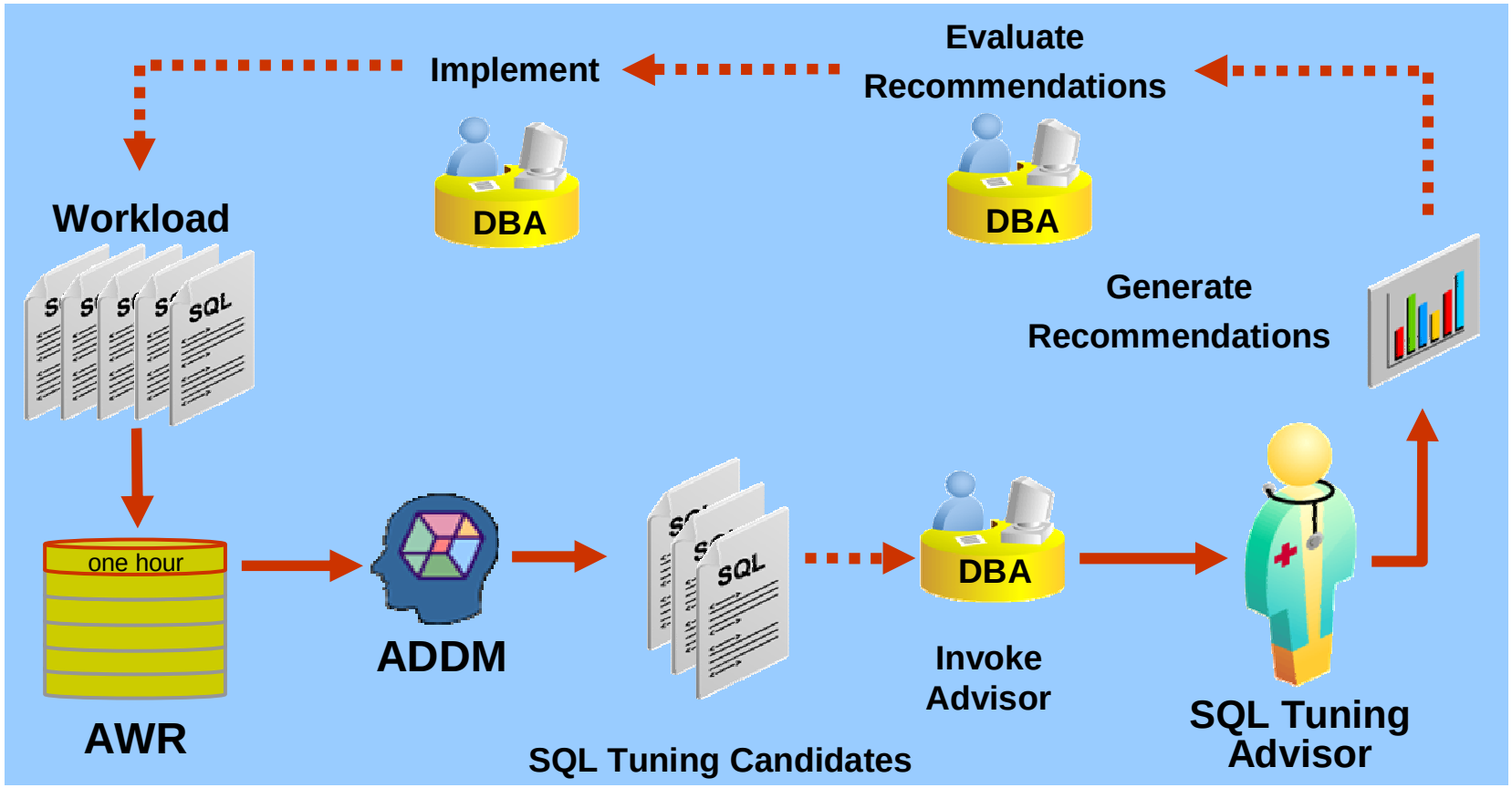


- Explain plan (DBMS_XPLAN)
- Runtime Statistics (V$SESSTAT, V$SYSSTAT, V$PGASTAT)
- SQL Advisor
- SQLTrace (10046) and tkprof
- CBO Trace (10059)
- Stored Outlines

Can use SQLPlus or DBControl

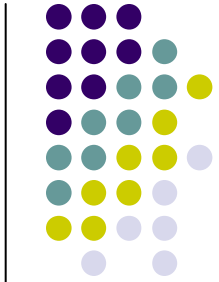
# SQL Tuning Advisor

- SQL Tuning Advisor in Oracle 10g

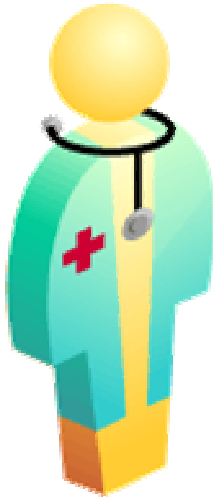



Animation from Oracle

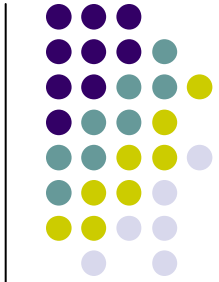
# SQL Tuning Advisor



1. Create Index
2. Rewrite SQL (UNION ALL, NOT EXISTS, etc)
3. Gather Stats (stale or missing)
4. Create SQL Profile (correct cardinality or selectivity estimates, optimizer mode – only in comprehensive mode, not in limited mode)



# Where to Learn More



- Documentation – tahiti.oracle.com
- Metalink and OTN
- Local User Group
- Online Community – Oracle Forums, oracle-l, c.d.o.s (usenet)
- Websites – asktom, oracle-base, orafaq, dizwell, jlcomp
- Blogs – Kyte, Lewis, Closson, Rittman
- Try things out on your own!


# Q&A

**Questions, comments, suggestions?**