The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle.
Agenda

• SQL Tuning Challenges
• Oracle Database 11g Solutions
  • Automatic SQL Tuning
  • Real-time SQL Monitoring
  • Partition Advisor
• Q & A
SQL Tuning Challenges

• Oracle Database 10g introduced SQL advisors to simplify application and SQL tuning

• Remaining challenges
  • SQL Tuning still reactive
  • Painful to find and investigate long-running SQL
  • Partitioning excluded from schema optimization advice

• Oracle Database 11g solutions
  • Automatic SQL Tuning
  • Real-time SQL Monitoring
  • Partition Advisor component of SQL Access Advisor
Automatic SQL Tuning

The Self-Managing Database
Challenges of Manual SQL Tuning

- Requires expertise in several domains
  - SQL optimization: adjust the execution plan
  - Access design: provide fast data access
  - SQL design: use appropriate SQL constructs

- Time consuming
  - Plans are complicated
  - Each SQL statement is unique and each execution can be different
  - Potentially large number of statements to tune
  - Testing proposed changes is labor-intensive
  - Many possible ways to a solution

- Never ending task
  - SQL workload always evolving
  - Plan regressions
Simplifying SQL Tuning

SQL Tuning Advisor, since Oracle Database 10g

Automatic Tuning Optimizer

- Statistics Analysis
- SQL Profiling
- Access Path Analysis
- SQL Structure Analysis

SQL Tuning Advisor

- SQL Tuning Recommendations
  - Gather Missing or Stale Statistics
  - Create a SQL Profile
  - Add Missing Indexes
  - Modify SQL Constructs

DBA
Improvements in Oracle Database 11g

Better SQL Profiling

Automatic Tuning Optimizer

- Statistics Analysis
- SQL Profiling
  - Fix potential regression after upgrade
  - Verify benefit through test-execution
- Access Path Analysis
- SQL Structure Analysis

SQL Tuning Advisor

- Gather Missing or Stale Statistics

SQL Tuning Recommendations

- Create a SQL Profile – show verified benefit
- Add Missing Indexes
- Modify SQL Constructs
Testing SQL Profiles (1)
Measuring actual benefit with test-execution

Naïve: Execute in Order

P1 → P2

Finish, P2 wins!

But what if P1 never completes?

P1

Timeout!

It would be great to run them concurrently….

P1 → P2

P2 wins, kill P1!

But then I take 2 CPUs, and N in the general case…
Testing SQL Profiles (2)

Measuring actual benefit with test-execution

Solution: Tournament Execution

Round 1:

15 sec

P1

15 sec

P2

Round 2:

30 sec

P1

16 sec

P2

Your winner, with a knockout in the second round, P2!
SQL Tuning in Oracle Database 10g

End-to-end Workflow

- Workload
- ADDM
  - one hour
  - AWR
- SQL Tuning Candidates
- SQL Tuning Advisor
- Evaluate Recommendations
  - DBA
  - Generate Recommendations
  - Invoke Advisor
  - DBA
- Implement
  - DBA
- A good end-to-end solution, but manual intervention is required.
Automatic SQL Tuning in Oracle 11g

The Self-Managing Database

Workload

SQL

one week

AWR

Implement SQL Profiles

Test SQL Profiles

Generate Recommendations

Choose Candidate SQL

SQL Tuning Candidates

It’s Automatic!

View Reports / Control Process

DBA
Picking Candidate SQL (1)

I could just pick from the top down…
But I will miss SQLs with important hotspots!

Let’s try a more balanced approach:

OK, but where do I start?
Eventually we need one list to tune from: merge the buckets.

All buckets are not created equal: focus on the week, but don’t forget about the others.

Focus on the SQLs we have not seen recently: Don’t re-tune SQLs if nothing has changed!
Tuning Flow

Tuning activities per SQL

Candidate SQLs
- Store findings, exec stats
- Fetch next SQL

Accept Profile
- Require 3X benefit in CPU and IO time
- Still recommend if < 3X

Test Profile
- Tournament competition

Tune SQL
- Fix potential regressions
- Look for indexes, statistics, as with standard tuning
Focus on SQL Profiles
First step in automating SQL tuning

Auto-testing/implementing is limited to profiles because:

• No lengthy, expensive set-up process (building an index takes time)
• Private to the current compilation
• No change to user SQL (does not change semantics)
• SQL-level recommendation, can be effectively tested
• Easily reversed by the DBA

Testing is done for regular SQL Tuning Advisor tasks as well!
Automatic SQL Tuning Defaults
Sensible defaults with flexible configurations

- Out-of-the-box defaults:
  - Runs in each maintenance window (MAINTENANCE_WINDOW_GROUP)
  - SQL profiles are tested but not implemented

- DBA can configure using EM:
  - Whether / When / How long it runs
  - Resources it uses
  - Whether it implements profiles
  - How many profiles it implements
Automatic SQL Tuning Task

Automated Maintenance Tasks

Status: Enabled

TIP: If the status is Disabled, there are no future windows.

<table>
<thead>
<tr>
<th>Task Name</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimizer Statistics Gathering</td>
<td></td>
</tr>
<tr>
<td>Segment Advisor</td>
<td></td>
</tr>
<tr>
<td>Automatic SQL Tuning</td>
<td></td>
</tr>
</tbody>
</table>

Logged in As SYSTEM

Begin Date: Nov 9, 2007
Completed: Nov 9, 2007

Logged in As SYSTEM

Automatic SQL Tuning

Nov 8, 2007
Automatic SQL Tuning Configuration

Automated Maintenance Tasks Configuration

Global Status
- Enabled
- Disabled

Task Settings
- Optimizer Statistics Gathering
  - Enabled
  - Disabled
- Segment Advisor
  - Enabled
  - Disabled
- Automatic SQL Tuning
  - Enabled
  - Disabled

Maintenance Window Group Assignment

<table>
<thead>
<tr>
<th>Window</th>
<th>Optimizer Statistics Gathering</th>
<th>Segment Advisor</th>
<th>Automatic SQL Tuning</th>
</tr>
</thead>
<tbody>
<tr>
<td>FRIDAY_WINDOW</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SATURDAY_WINDOW</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SUNDAY_WINDOW</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>MONDAY_WINDOW</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TUESDAY_WINDOW</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>WEDNESDAY_WINDOW</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>THURSDAY_WINDOW</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Automatic SQL Tuning Settings

- Maximum Time Spent Per SQL During Tuning (sec): 200
- Automatic Implementation of SQL Profiles: Yes
- Maximum SQL Profiles Implemented Per Execution: 20
- Maximum SQL Profiles Implemented (Overall): 10000
Automatic SQL Tuning Result Summary

Task Activity Summary

The activity summary graph shows the benefit of the analysis tools on the system's high-level SQL queries that significantly improve SQL performance were implemented.

Begin Date: Nov 8, 2007 10:00:01 PM GMT-08:00  End Date: Nov 9, 2007 12:28:52 PM GMT-08:00

Overall Task Summary

Breakdown by Finding Type

Profile Effect Statistics

- DB Time ($) vs Implemented Recommended
- Before vs After

Profile Effect Report

Profile Effect Details

Profile Effect Summary

Profile Effect Summary Report

Profile Effect Summary Details

Profile Effect Summary Details Report

Profile Effect Summary Details Summary

Profile Effect Summary Details Summary Report

Profile Effect Summary Details Summary Details

Profile Effect Summary Details Summary Details Report

Profile Effect Summary Details Summary Details Summary

Profile Effect Summary Details Summary Details Summary Report

Profile Effect Summary Details Summary Details Summary Summary

Profile Effect Summary Details Summary Details Summary Summary Report

Profile Effect Summary Details Summary Details Summary Summary Summary

Profile Effect Summary Details Summary Details Summary Summary Summary Report

Profile Effect Summary Details Summary Details Summary Summary Summary Summary

Profile Effect Summary Details Summary Details Summary Summary Summary Summary Report

Profile Effect Summary Details Summary Details Summary Summary Summary Summary Summary

Profile Effect Summary Details Summary Details Summary Summary Summary Summary Summary Report

Profile Effect Summary Details Summary Details Summary Summary Summary Summary Summary Summary

Profile Effect Summary Details Summary Details Summary Summary Summary Summary Summary Summary Report

Profile Effect Summary Details Summary Details Summary Summary Summary Summary Summary Summary Summary

Profile Effect Summary Details Summary Details Summary Summary Summary Summary Summary Summary Summary Report

Profile Effect Summary Details Summary Details Summary Summary Summary Summary Summary Summary Summary Summary

Profile Effect Summary Details Summary Details Summary Summary Summary Summary Summary Summary Summary Summary Report

Profile Effect Summary Details Summary Details Summary Summary Summary Summary Summary Summary Summary Summary Summary

Profile Effect Summary Details Summary Details Summary Summary Summary Summary Summary Summary Summary Summary Summary Report

Profile Effect Summary Details Summary Details Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary

Profile Effect Summary Details Summary Details Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary Report

Profile Effect Summary Details Summary Details Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary

Profile Effect Summary Details Summary Details Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary Report

Profile Effect Summary Details Summary Details Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary

Profile Effect Summary Details Summary Details Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary Report

Profile Effect Summary Details Summary Details Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary

Profile Effect Summary Details Summary Details Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary Summary Report
<table>
<thead>
<tr>
<th>SQL Text</th>
<th>Parsing Schema</th>
<th>SQL Profile</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>SELECT * FROM t1, t2, t3 FROM t1, t2 WHERE t1.id = t2.id AND t2.id = t3.id</td>
<td>ParseTest1</td>
<td>45.65%</td>
<td>11/8/07</td>
</tr>
<tr>
<td>SELECT COUNT(*) FROM t1</td>
<td>ParseTest2</td>
<td>35.78%</td>
<td>11/8/07</td>
</tr>
<tr>
<td>SELECT SUM(salary) FROM employees</td>
<td>ParseTest3</td>
<td>54.32%</td>
<td>11/8/07</td>
</tr>
<tr>
<td>SELECT COUNT(*) FROM orders</td>
<td>ParseTest4</td>
<td>65.47%</td>
<td>11/8/07</td>
</tr>
<tr>
<td>SELECT AVG(price) FROM products</td>
<td>ParseTest5</td>
<td>78.90%</td>
<td>11/8/07</td>
</tr>
<tr>
<td>SELECT COUNT(DISTINCT product_id) FROM order_details</td>
<td>ParseTest6</td>
<td>34.65%</td>
<td>11/8/07</td>
</tr>
<tr>
<td>SELECT SUM(price) FROM order_details</td>
<td>ParseTest7</td>
<td>67.89%</td>
<td>11/8/07</td>
</tr>
<tr>
<td>SELECT COUNT(*) FROM order_details</td>
<td>ParseTest8</td>
<td>89.76%</td>
<td>11/8/07</td>
</tr>
<tr>
<td>SELECT COUNT(DISTINCT order_id) FROM orders</td>
<td>ParseTest9</td>
<td>78.90%</td>
<td>11/8/07</td>
</tr>
<tr>
<td>SELECT SUM(quantity) FROM order_details</td>
<td>ParseTest10</td>
<td>65.47%</td>
<td>11/8/07</td>
</tr>
</tbody>
</table>

**Automatic SQL Tuning Result Recommendations**
Automatically Tuned SQL Details Drilldown

<table>
<thead>
<tr>
<th>Advisor Task Name</th>
<th>Advisor Task Owner</th>
<th>Task Completion</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYS_AUTO_SQL_TUNING_TASK</td>
<td>SYS</td>
<td>Nov 8, 2007 10:35:12 PM</td>
</tr>
</tbody>
</table>

### SQL Details

```sql
SELECT /*+ INDEX (ch featurevalue_08_id) ch featurevalue_09_id */ /*+ INDEX (ttl_pg_featurevalue_13_id pg_featurevalue_13_id) */ /*+ INDEX (ttl_pg_featurevalue_15_id pg_featurevalue_15_id) */ /*+ INDEX (ttl_pg_featurevalue_01_elig_id pg_featurevalue_01_id) */...
```

### Statistics

<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Top SQL by DB Time</td>
<td>2 of 19</td>
</tr>
<tr>
<td>Top SQL by SQL Time</td>
<td>2 of 15</td>
</tr>
</tbody>
</table>

### Summary

- **Time Range:** 12 AM to 4 PM
- **Task Details:**
  - **Task Name:** SYS_AUTO_SQL_TUNING_TASK
  - **Task Owner:** SYS
  - **Completion Time:** Nov 8, 2007 10:35:12 PM
Conclusions

- *Manual SQL tuning* is painful even for the experts

- *Oracle 10g SQL Tuning Advisor* quickly gives DBA good choices

- *Oracle 11g Automatic SQL Tuning* automates the process by making the easy decisions

- *DBA can control* as much of the process as he wants
Just when you thought it was safe to run your SQLs…

There’s a lot more to SQL performance than bad plans!
• Potential run-time issues
• Finding high response-time SQL is no piece of cake
• Keeping tabs on Parallel SQL is even harder
Real-Time SQL Monitoring

Shining new light on SQL Performance
Problem: Managing High Response-Time SQLs

- **Monitoring**: tracking high response-time SQL
  - What is that expensive SQL (ETL, DDL, batch, report, …) I started up to?
  - Do I have any high response-time SQL running on my OLTP system?
  - Any SQL executing parallel?

- **Investigating**: why is this execution so expensive?
  - Plan has hundreds of operations -- where is the time being spent?
  - Why is a particular operation so expensive?
  - SQL runs parallel, is DOP appropriate? is there a skew?

➤ What is going on inside a SQL execution???
Solution: Real-time SQL Monitoring
Looking inside the SQL

- Enabled **out-of-the-box** with no performance impact
- **Automatically** monitors SQL executions that:
  - consume more than 5 seconds of CPU or I/O time
  - are running parallel: PQ, PDML, PDDL
- Monitors each execution **independently**
- Exposes monitoring statistics at **multiple levels**
  - Global execution level
  - Plan operation level (Plan Tuning)
  - Parallel Execution level (PX Tuning)
- Guides your tuning efforts
How does it work?

- Exposes monitoring statistics in:
  - V$SQL_MONITOR
    - Cumulative DB time breakdown (CPU, IO, Application, etc)
    - PL/SQL, Java Exec Times
  - V$SQL_PLAN_MONITOR
    - #rows, #executions, memory, temp space per plan operation
    - Plan operation begin and end times
  - V$ACTIVE_SESSION_HISTORY (ASH)
    - Each execution of each SQL identifiable in ASH
      execution key: (SQL_ID, SQL_EXEC_START, SQL_EXEC_ID)
    - Parallel Execution Servers share an execution key with QC, but use a
      separate Session ID
- Separate entries for each Parallel Execution Server
- Refreshes statistics every second, during query execution
- Statistics available for at least 5 minutes, even with cursor age-outs
How do I use it?

• 11g Enterprise Manager Grid Control
• Additional reporting (available today):
  DBMS_SQLTUNE.REPORT_SQL_MONITOR
Enterprise Manager Flow (1)

SQL Details

Top Activity

Monitoring Details

Session Details
Enterprise Manager Flow (2)

Monitoring List

Monitoring Details
### SQL Monitoring List

<table>
<thead>
<tr>
<th>Status</th>
<th>Duration</th>
<th>SQL ID</th>
<th>Session</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>ft2mfz40z1ud6</td>
<td>85</td>
</tr>
<tr>
<td></td>
<td></td>
<td>203z7cxfh0d26</td>
<td>85</td>
</tr>
</tbody>
</table>

- **Duration**: 6.3mn
- **SQL ID**: 203z7cxfh0d26
- **Session**: 85
- **Disk Reads**: 1,557,158 (92%)
SQL Monitoring Details

Session: 85
Plan Hash Value: 3554797743
First Refresh Time: Fri Nov 9 2007 12:36:02 PM
Last Refresh Time: Fri Nov 9 2007 12:45:26 PM

Buffer Gets: 910K
IO Count: 1257K
Database Time: 9.4mn
Wait Activity %: 100

User I/O: direct path read temp - 377 samples (73%)
# SQL Monitoring Details (Parallelism)

## SQL Monitoring Detail

### Global Execution Detail

- **SQL ID**: 203970edf08ed89d
- **Execution Started**: Fri Nov 9 2007 12:45:26 PM
- **Execution ID**: 16777246
- **Instance ID**: 1
- **Session**: 85
- **Plan Hash Value**: 2755622702

### Buffer Gets

<table>
<thead>
<tr>
<th>Parallel Server</th>
<th>Buffer Gets</th>
<th>IO Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Parallel Servers</td>
<td>225,586</td>
<td>225K</td>
</tr>
<tr>
<td>Instance 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parallel Coordinator</td>
<td>226K</td>
<td>225K</td>
</tr>
<tr>
<td>Instance 2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parallel Set 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Parallel Server 1 (p000)</td>
<td>226K</td>
<td>225K</td>
</tr>
<tr>
<td>Parallel Server 2 (p001)</td>
<td>226K</td>
<td>225K</td>
</tr>
<tr>
<td>Parallel Server 3 (p002)</td>
<td>226K</td>
<td>225K</td>
</tr>
<tr>
<td>Parallel Server 4 (p003)</td>
<td>226K</td>
<td>225K</td>
</tr>
<tr>
<td>Parallel Server 1 (p004)</td>
<td></td>
<td>87K</td>
</tr>
<tr>
<td>Parallel Server 2 (p005)</td>
<td></td>
<td>44K</td>
</tr>
<tr>
<td>Parallel Server 3 (p006)</td>
<td></td>
<td>87K</td>
</tr>
<tr>
<td>Parallel Server 4 (p007)</td>
<td></td>
<td>87K</td>
</tr>
</tbody>
</table>

### Wait Activity Soo
Conclusion

• Real-Time SQL Monitoring is
  • Monitoring and tuning for high response-time SQLs
  • New, fine-grained SQL statistics
    • tracked automatically
    • updated while the SQL runs
    • highly visible and accessible
    • at no cost to your production system
  • The only way to know what’s happening inside single SQL execution
  • The quickest way to the root cause of a performance problem:
    If you can find the problem, you can fix it!
Partition Advisor
Problem

- SQLs on large tables run too long or timeout
- High I/O counts
- Too much pressure on buffer pool
- Disgruntled users
- Low transaction rates
- Too many complex SQLs to figure out on my own
- Put out a fire here, another starts over there
Solution

- Get new 11g partition advice along with other advice from the new 11g SQL Access Advisor
  - Recommendations targeted at partition elimination in query processing
  - Recommendations to aid certain join processing
CREATE TABLE emp
  (empno NUMBER(6),
   first_name VARCHAR(20),
   last_name VARCHAR(20),
   deptno NUMBER(6))
PARTITION BY RANGE (deptno) INTERVAL 100
PARTITION p1 VALUES LESS THAN 100
Interval Partitioning

CREATE TABLE emp
  (empno NUMBER(6),
   first_name VARCHAR(20),
   last_name VARCHAR(20),
   deptno NUMBER(6))
PARTITION BY RANGE (deptno) INTERVAL 100
PARTITION p1 VALUES LESS THAN 100

Interval partition is a new, automated form of range partitioning.
Partition Elimination

CREATE TABLE emp
  (empno NUMBER(6),
   first_name VARCHAR(20),
   last_name VARCHAR(20),
   deptno NUMBER(6))
PARTITION BY RANGE (deptno) INTERVAL 100
PARTITION p1 VALUES LESS THAN 100

SELECT empno, last_name, first_name
FROM emp
WHERE deptno = 123
Partition Elimination

CREATE TABLE emp
    (empno NUMBER(6),
     first_name VARCHAR(20),
     last_name VARCHAR(20),
     deptno NUMBER(6))
PARTITION BY RANGE (deptno) INTERVAL 100
PARTITION p1 VALUES LESS THAN 100

SELECT empno, last_name, first_name
FROM emp
WHERE deptno = 123
When joining two tables that are partitioned on the join-key, Oracle may choose to join on a per-partition basis.
How does SAA work?

Standard STS Workload

SQL Access Advisor w/new Partition Advice

Analyzes access patterns, column usage, etc.

Determines best partitioning strategy for the entire workload in concert with best index and materialized view solutions

Recommendations:
- partition
- index
- mv

SQL cache, user defined, etc.
How does SAA work?

Expensive Qs on BIG tables

Table+Index Partition Analysis

Workload

Rank Qs

Partition annotated workload

Index Analysis

MV Analysis

MV Partition Analysis

feedback output

Recommendations
How does SAA work?

Expensive Qs on BIG tables

Table+Index Partition Analysis

Workload

Rank Qs

Partition annotated workload

Index Analysis

MV Analysis

MV Partition Analysis

feedback output

Recommendations
Partition Advisor Problem Space

• Fact: If I partition table T1, all Qs referencing T1 are affected (+ or -)
• Fact: If I also partition table T2, the same applies
• Fact: Lots of Qs reference multiple tables forming a network of inter-relationships
• Therefore: A potential partitioning scheme on each different table affects each potential partitioning scheme on other tables in that network
Partition Advisor

- Enumeration
- Pruning
- Heuristics
- Evaluation
Partition Advisor

Enumeration
Pruning
Heuristics
Evaluation
Partition Advisor

- Enumeration
- Pruning
- Heuristics
- Evaluation
Partition Advisor

Enumeration
Pruning
Heuristics
Evaluation
How does SAA work?

- Expensive Qs on BIG tables
- Table+Index Partition Analysis
- Workload
- Rank Qs
- Partition annotated workload
- Index Analysis
- MV Analysis
- MV Partition Analysis
- feedback output
- Recommendations
How does SAA work?

- Expensive Qs on BIG tables
- Table+Index Partition Analysis
- Workload
- Rank Qs
- Partition annotated workload
- Index Analysis
- MV Analysis
- MV Partition Analysis

Output

Recommendations

Feedback
MV & Index Advisor

MV Analysis: joins group bys dimensions

Index Analysis: predicates group bys joins index-only access bitmap access

MV candidates

Index candidates

Evaluate

MV Partition Advisor

Optimizer / Query Rewrite

Associated groups of access candidates

Global access optimization

Recommendations
What does SAA do?
What does SAA do?

Recommends:

Partitioning
What does SAA do?

Recommends:

Partitioning  Tables
What does SAA do?

Recommends:

- Partitioning
- Tables
- Materialized Views
What does SAA do?

Recommends:

Partitioning

- Tables
- Materialized Views
- Indexes
What does SAA do?

Recommends:

- Partitioning
- Tables
- Materialized Views
- Indexes

Supported Partitioning Types: Interval Hash
What does SAA do?

Recommends:

Partitioning

Supported Partitioning Types: Interval Hash

Materialized Views

Supported Partition Key Types: Date Number

Indexes
What does SAA do?

Recommends:

Partitioning

- Tables
- Materialized Views
- Indexes

Creating
What does SAA do?

Recommends:

- Partitioning
  - Tables
  - Materialized Views
  - Indexes

- Creating
  - Materialized Views
What does SAA do?

Recommends:

- Partitioning
  - Tables
  - Materialized Views
  - Indexes

- Creating
  - Materialized Views
  - Indexes
What does SAA do?

Recommends:

Partitioning
- Tables
- Materialized Views
- Indexes

Creating
- Materialized Views
- Indexes
- MV Logs
What does SAA do?

Recommends:

Partitioning
- Tables
- Materialized Views
- Indexes

Creating
- Materialized Views
- Indexes
- MV Logs

Holistic Advice
Choosing Partition Advice

SQL Access Advisor: Recommendation Options

Access Structures to Recommend
- Indexes
- Materialized Views
- Partitioning

Scope
The advisor can run in one of two modes, Limited or Comprehensive. Limited Mode is meant to return quickly after processing the statements with the highest cost, potentially ignoring statements with a cost below a certain threshold. Comprehensive Mode will perform an exhaustive analysis.

Advanced Options
Recommendation summary
Partition recommendations

<table>
<thead>
<tr>
<th>Implementation Status</th>
<th>Recommendation IDs</th>
<th>Action</th>
<th>Object Name</th>
<th>Object Attributes</th>
<th>Indexed Columns</th>
<th>Base Table</th>
<th>Schema</th>
<th>Tablespace</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1,4,5,6,7,8,9,11,12</td>
<td>PARTITION_TABLE</td>
<td>PRODUCTS</td>
<td></td>
<td></td>
<td></td>
<td>SH</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2,3</td>
<td>PARTITION_TABLE</td>
<td>CUSTOMERS</td>
<td></td>
<td></td>
<td></td>
<td>SH</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>CREATE_MATERIALIZED_VIEW_LOG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>_SH.SALES _SH</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>CREATE_MATERIALIZED_VIEW_LOG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>_SH.CUSTOMERS _SH</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>CREATE_MATERIALIZED_VIEW</td>
<td>MV$_00130000</td>
<td>General Match</td>
<td></td>
<td></td>
<td>SH</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>GATHER_TABLE_STATISTICS</td>
<td>MV$_00130000</td>
<td></td>
<td></td>
<td></td>
<td>SH</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>RETAIN_INDEX</td>
<td>SALES_CUSTOM_IDX</td>
<td>BITMAP</td>
<td>CUST_ID</td>
<td>_SH.SALES</td>
<td>SH</td>
<td></td>
</tr>
<tr>
<td></td>
<td>12,13</td>
<td>RETAIN_INDEX</td>
<td>PRODUCTS_PK</td>
<td>Btree</td>
<td>PROD_ID</td>
<td>_SH.PRODUCTS</td>
<td>SH</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>RETAIN_INDEX</td>
<td>CUSTOMERS_GENDER_IDX</td>
<td></td>
<td></td>
<td></td>
<td>_SH.CUSTOMERS _SH.CUSTOMERS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7,8,11</td>
<td>RETAIN_INDEX</td>
<td>SALES_PROD_IDX</td>
<td>BITMAP</td>
<td></td>
<td></td>
<td>_SH.SALES</td>
<td></td>
</tr>
</tbody>
</table>
### Recommendation 1

SQL Access Advisor generates default object names and uses the default schemas and table spaces specified during task creation, but you can change them. If you edit any name, dependent names, which are shown as red text, will be updated accordingly. If the Table space field is left blank the default table space of the schema will be used. When you click OK, the SQL script is modified, but it is not actually executed until you select 'Schedule Implementation' on the Recommendations or SQL Statements pages.

### Actions

<table>
<thead>
<tr>
<th>Implementation Status</th>
<th>Action</th>
<th>Object Name</th>
<th>Object Attributes</th>
<th>Indexed Columns</th>
<th>Base Table</th>
<th>Schema</th>
<th>Tablespace</th>
<th>Partition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PARTITION_TABLE</td>
<td>PRODUCTS</td>
<td></td>
<td></td>
<td></td>
<td>SH</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CREATE_INDEX</td>
<td>PRODUCTS_IDX$00130003</td>
<td>BTREE, LOCAL</td>
<td>SUPPLIER_ID, PROD_LIST_PRICE, PROD_NAME</td>
<td>SH, PRODUCTS</td>
<td>SH</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### SQL Affected by Recommendation: 1

<table>
<thead>
<tr>
<th>Statement ID</th>
<th>Statement</th>
<th>Original Cost</th>
<th>New Cost</th>
<th>Cost Improvement</th>
<th>Cost Improvement (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>116</td>
<td>SELECT prod_name FROM products p WHERE prod_list_price=123 AND supplier_id=267</td>
<td>10</td>
<td>2</td>
<td>8</td>
<td>80.00</td>
</tr>
<tr>
<td>119</td>
<td>SELECT prod_name FROM products p WHERE prod_list_price=375 AND supplier_id=282</td>
<td>10</td>
<td>2</td>
<td>8</td>
<td>80.00</td>
</tr>
<tr>
<td>122</td>
<td>SELECT prod_name FROM products p WHERE prod_list_price=461 AND supplier_id=8</td>
<td>10</td>
<td>2</td>
<td>8</td>
<td>80.00</td>
</tr>
<tr>
<td>123</td>
<td>SELECT prod_name FROM products p WHERE prod_list_price=431 AND supplier_id=240</td>
<td>10</td>
<td>2</td>
<td>8</td>
<td>80.00</td>
</tr>
<tr>
<td>126</td>
<td>SELECT prod_name FROM products p WHERE prod_list_price=355 AND supplier_id=5</td>
<td>10</td>
<td>2</td>
<td>8</td>
<td>80.00</td>
</tr>
<tr>
<td>128</td>
<td>SELECT prod_name FROM products p WHERE prod_list_price=139 AND supplier_id=98</td>
<td>10</td>
<td>2</td>
<td>8</td>
<td>80.00</td>
</tr>
<tr>
<td>131</td>
<td>SELECT prod_name FROM products p WHERE prod_list_price=441 AND supplier_id=103</td>
<td>10</td>
<td>2</td>
<td>8</td>
<td>80.00</td>
</tr>
<tr>
<td>133</td>
<td>SELECT prod_name FROM products p WHERE prod_list_price=451 AND supplier_id=12</td>
<td>10</td>
<td>2</td>
<td>8</td>
<td>80.00</td>
</tr>
<tr>
<td>136</td>
<td>SELECT prod_name FROM products p WHERE prod_list_price=594 AND supplier_id=257</td>
<td>10</td>
<td>2</td>
<td>8</td>
<td>80.00</td>
</tr>
<tr>
<td>137</td>
<td>SELECT prod_name FROM products p WHERE prod_list_price=445 AND supplier_id=243</td>
<td>10</td>
<td>2</td>
<td>8</td>
<td>80.00</td>
</tr>
</tbody>
</table>
Conclusions

• SAA now covers your data access problems with all possible access solutions
• New for 11g:
  • Partition advice, including hash and new interval on date and number
  • Incremental advice
• Partition recommendations are holistically generated, simultaneously considering all possible access solutions across an entire SQL workload
• SAA is easy to use as ever – partition advice is yours for click of a checkbox!
Navigating to SQL Access Advisor

**EM Home Page**

**Advisor Central Page**

**SQL Advisor Page**
Using SQL Access Advisor

Choose initial options

Select a workload
Running advisor job

Select job options

Review & submit

Review results