SQL Tuning in Oracle 10g: The Do’s and Don’ts

Mughees A. Minhas
Group Product Manager
Database Manageability
Oracle Corp.
Introduction

- Automatic SQL Tuning solution consists of
  - SQL Tuning Advisor
  - SQL Access Advisor
- Provides comprehensive, automatic, and cost-effective solution for application tuning
- Reduces SQL tuning time by up to 80%
- Reduces management cost
Manual Tuning Challenges

• 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
  – Each SQL statement is unique
  – Potentially large number of statements to tune

• Never ending task
  – SQL workload always evolving
  – Plan regressions
Manual Tuning Example
Packaged App Tuning Scenario

**Manual Tuning**

1. Get explain plan
2. Examine query objects and their sizes
3. Review and compare explain plan statistics with execution statistics (stored in V$SQL view)
4. Identify the problem, e.g., “first rows” issue because only recent data is ever displayed despite large history being queried
5. Contact application vendor
6. Produce test case for vendor
7. Get a patch with appropriate code modifications from the vendor
8. Install the patch in next maintenance cycle
SQL Tuning & Access Advisor Overview

How do I tune my SQL workload?

Solution

No expert required

Component of CBO

Provides implementation script

SQL Tuning & Access Advisors

ORACLE
SQL Tuning Advisor
Oracle 10g Automates the SQL Tuning Process

Oracle 10g Automates the SQL Tuning Process

I can do it for you!

SQL Workload

ADDM

High-Load SQL

SQL Tuning Advisor

DBA

Oracle 10g

Automates the SQL Tuning Process

I can do it for you!
Automatic SQL Tuning Overview

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
Automatic Tuning Optimizer (ATO)

- It is the query optimizer running in tuning mode
  - Uses same plan generation process but performs additional steps that require significantly more time
- It performs verification steps
  - To validate statistics and its own estimates
    - Uses dynamic sampling and partial executions to validate
- It performs exploratory steps
  - To investigate the use of new indexes that could provide significant speed-up
  - To analyze SQL constructs that led to expensive plan operators
Statistics Analysis

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

ORACLE
Statistics Analysis

• Motivation
  – Statistics are key input to the query optimizer
    • Their availability and accuracy is very important
  – In Oracle10g, the Automatic Statistics Collection maintains statistics up to date…
  – But it may not be enabled or properly configured!
• The ATO verifies statistics that it needs/uses
  – Generates auxiliary information to compensate for missing or stale statistics
  – Generates recommendations to gather statistics where appropriate
SQL Profiling

Automatic Tuning Optimizer

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

SQL Profiling Advisor

SQL Tuning Recommendations

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

DBA
SQL Profiling

• Motivation
  – Empower query optimizer to find better plan by gathering additional information on query behavior

• The query optimizer has time constraints
  – Makes compromises while finding right plan

• The ATO is allowed a lot more time
  – Uses the time to gather customized information about the SQL statement, known as SQL Profile
  – Builds a SQL Profile and recommends it
  – Once implemented, SQL Profile is used by the query optimizer to generate a well-tuned plan
SQL Profile

- Contains auxiliary information collected by the ATO for a SQL statement
  - Customized optimizer settings
    - Based on past execution history (e.g., first_rows vs. all_rows)
  - Compensation for missing or stale statistics
  - Compensation for errors in optimizer estimates
    - Estimation errors occur due to data skews and correlations, complex filters and joins
- Doesn’t require any change to the SQL text
  - Ideal for Packaged Apps
- Persistence: Works across shutdowns & upgrades
- Transportable across databases (10.2)
SQL Profiling Flow

**SQL Profiling**

1. **SQL Tuning Advisor**
   - **submit**
   - **create** SQL Profile

After …

2. **Database Users**
   - **submit**
   - **use**
   - **output** Well-Tuned Plan

3. **Optimizer (Normal Mode)**
   - **submit**
   - **output** Well-Tuned Plan
Access Path Analysis

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
Access Path Analysis

• Motivation
  – Adding an index may significantly improve the performance of a SQL statement

• Problem: A critical access path is missing
  – Index not created or mistakenly dropped

• ATO explores the use of new indexes
  – Recommends index if major performance boost provided
  – Also recommends running SQL Access Advisor to get comprehensive index analysis for entire workload
    • SQL Access Advisor also uses this analysis mode
SQL Structure Analysis

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
SQL Structure Analysis

• Motivation
  – Help application developers identify poorly written SQL statements
  – Suggest restructuring of SQL for efficiency

• Problem categories
  – Semantic changes of SQL operators (e.g., use UNION ALL instead of UNION)
    • Subject to user acceptance of new result
  – Syntactic changes to predicates on indexed columns (e.g., remove type mismatch in column = :bind)
  – SQL design issues (e.g., add missing join predicate to eliminate a large Cartesian join)
SQL Tuning Usage Scenarios

Automatic Selection

AWR → ADDM → High-load SQL

Manual Selection

SQL Sources

AWR → Cursor Cache → User-defined

Filter / Rank → SQL Tuning Set (STS) → SQL Tuning Advisor
SQL Tuning Set (STS)

- **Motivation**
  - Enable user to tune custom set of SQL statements
- **New object in Oracle10g for capturing and managing SQL workload**
- **Stores SQL statements along with:**
  - Execution context: parsing user, bind values, etc.
  - Execution statistics: buffer gets, CPU time, elapse time, number of executions, etc.
- **Transportable across databases (10.2)**
- **Created from any SQL source**
  - AWR, cursor cache, user-defined workload, STS
SQL Tuning Set Benefits

- Allows selective, on-demand, custom SQL workload tuning
- Simplifies tuning of large number of SQL statements
- Is persistent
- Facilitates workload capture/management
- Provides a common infrastructure for dealing with SQL workloads
  - Can be used as a source for different tuning tasks
SQL Tuning Advisor User Interface

- GUI: Enterprise Manager
  - Launch SQL Tuning Advisor from a SQL Source page
    - ADDM Finding page, or
    - Top SQL page, or
    - SQL Tuning Set (STS) page
- View/Implement SQL Tuning Recommendations
- Command line: DBMS_SQLTUNE package
**SQL Source: ADDM Finding**

### Recommendations

**Select All** | **Select None** | **Show All Details** | **Hide All Details**
--- | --- | --- | ---

#### Select Details

<table>
<thead>
<tr>
<th>Category</th>
<th>Benefit (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL Tuning</td>
<td>26.1</td>
</tr>
</tbody>
</table>

### Action

1. **Investigate the SQL statement with SQL_ID “g0fyfq5fatsbh” for possible performance improvements.**
   - **SQL Text:** UPDATE PARTS_CAT SET FNUM=1 WHERE PCAT='SPORTS'
   - **SQL ID:** g0fyfq5fatsbh

2. **Rationale**
   - The SQL statement with SQL_ID “g0fyfq5fatsbh” was executed 84 times and had an average elapsed time of 7.3 seconds.
   - Waiting for event “enq: TX - row lock contention” in wait class “Application” accounted for 93% of the database time spent in processing the SQL statement with SQL_ID “g0fyfq5fatsbh”.
   - Waiting for event “buffer busy waits” in wait class “Concurrency” accounted for 4% of the database time spent in processing the SQL statement with SQL_ID “g0fyfq5fatsbh”.

### Additional Actions

- **Run SQL Tuning Advisor on the SQL statement with SQL_ID “gqtf76mt3amcy”**
  - **SQL Text:** SELECT /* DSS_Q6O */ "E" || t12.ch_featurevalue_09_id t12.ch_featurevalue_09_id, ...
  - **SQL ID:** gqtf76mt3amcy

- **View Tuning History**

---

**Oracle**
### SQL Source: SQL Tuning Set

**A SQL Tuning Set is a collection of SQL Statements that can be used for tuning purposes.**

#### Create SQL Tuning Set From

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Created</th>
<th>Last Modified</th>
</tr>
</thead>
</table>
Enterprise Manager Interface

- Launch SQL Tuning Advisor from a SQL Source page
  - ADDM Finding page, or
  - Top SQL page, or
  - SQL Tuning Set (STS) page

- View/Implement SQL Tuning Recommendations
SQL Tuning Recommendations — Overview

Database Instance: r2e > Advisor Central > SQL Tuning Results:SQL_TUNING_1129151335852

SQL Tuning Results:SQL_TUNING_1129151335852

Status: COMPLETED
SQL ID: 7jwb3bmbc4gcd
Time Limit (seconds): 1800

Started: Oct 12, 2005 2:09:00 PM
Completed: Oct 12, 2005 2:09:03 PM
Running Time (seconds): 3

Recommendations

View Recommendations
DBMS_SQLTUNE PL/SQL Package

- Contains API for SQL Tuning

**Tuning Task Management**
- Create Tuning Task
- Execute Tuning Task
- Display Advisor Recommendations
- Drop Tuning Task

**STS Management**
- Create STS
- Populate STS
- Query STS Contents
- Drop STS

**SQL Profile Management**
- Accept SQL Profile
- Drop SQL Profile
- Alter SQL Profile Attribute
Automatic vs. Manual SQL Tuning

**Manual Tuning**
1. Get explain plan
2. Examine query objects and their sizes
3. Review and compare explain plan statistics with execution statistics (stored in V$SQL view)
4. Identify the problem, e.g., “first rows” issue because only recent data is ever displayed despite large history being queried
5. Contact application vendor
6. Produce test case for vendor
7. Get a patch with appropriate code modifications from the vendor
8. Install the patch in next maintenance cycle

**Automatic Tuning**
1. Run SQL Tuning Advisor
2. Implement recommendations.
SQL Access Advisor
SQL Access Advisor

- Workload
- SQL Access Advisor
- Indexes
- Materialized Views
- Materialized View Logs
SQL Access Advisor Features

- De-mystifies access structure design for optimal application performance
- Recommends indexes, materialized views, and materialized view logs to create and/or drop for faster performance
- Analyzes entire workload and not just independent SQL statements
- Takes into account impact of new access structures on DML operations
- Considers storage, creation and maintenance costs
SQL Access Advisor Features

- Simultaneously considers
  - index solutions
  - materialized view solutions
  - combinations of both
- Optimizes materialized views for
  - maximum query rewrite usage
  - fast refresh
- Recommends materialized view logs for fast refresh
- Combines similar indexes into single index
- Impact of new access structures on DML considered
- Considers storage, creation & maintenance costs
Usage Scenarios

Filter Options

Cursor Cache
User Defined
Hypothetical
STS

Workload

SQL Access Advisor
Filter Options

- Don’t have to use the entire workload
- Filter by
  - Application or module name
  - Number of SQL statements
  - Queries during a specified time window
  - Username
  - Tables
    - must be in this list
    - not in this list
SQL Access Advisor User Interface

- GUI: Enterprise Manager
  - Launch SQL Access Advisor from Advisor Central
  - Select workload source
  - Set options
    - Workload
    - Recommendation
    - Advanced
  - Schedule job
  - Review job and submit
  - Monitor job
  - View Recommendations
  - Implement Recommendations

- Command line: DBMS_ADVISOR package
Launch SQL Access Advisor
Providing the SQL Access Advisor with an accurate workload is crucial to the effectiveness of the recommendations the advisor will generate. The performance of SQL not found in the workload may be adversely affected by recommendations to improve the SQL in the workload. The best workload is one that fully represents all the statements used to access the underlying tables.

**Workload Source**

- Current and recent SQL activity
  SQL will be selected from the cache.
- User-Defined Workload, Import SQL from a table
  Table must contain at least the SQL_TEXT and USERNAME columns.
- Create a hypothetical workload from the following schemas
  The advisor can create a hypothetical workload if the schema contains dimension or primary/foreign key constraints.
- Import Workload from SQL Repository
  Enter the name of the SQL Tuning Set
  You may choose any SQL Tuning set from the SQL Repository.
Set Workload Options

**SQL Access Advisor: Workload Advanced Options**

### Workload Type

- [ ] Read Only (Data Warehouse)
- [ ] Allow Advisor to determine workload type based on workload

### Drop Unused Indexes

- [ ] No, there are statements missing from the workload that might be adversely affected by an index removal.
- [ ] Yes, generate recommendations to drop unused access structures.

### Filter Options

You can apply filters to reduce the scope of the statements found in the workload. This will allow you to direct the advisor to make recommendations based on a specific subset of statements from the workload. Application and Action are strings that can be associated with SQL statements via the DBMS_APPLICATION_INFO package to allow for better SQL statement identification and tuning.

- [ ] Evaluate entire workload.
- [ ] Filter workload based on these options.

- [ ] Only the top resource consuming SQL statements

  Number of Statements: 25

  Order By [ ] Optimizer Cost

- [ ] Only SQL statements executed by the following users
Set Recommendation Options

SQL Access Advisor: Recommendation Options

Database  DemoDB

The SQL Access Advisor recommends index and materialized views to improve the performance of the SQL statements in the workload. These indexes and materialized views reduce the time it takes to read data. This benefit is balanced against the time required to maintain the index or materialized view. Through the analysis of the SQL in the workload and the volatility of the underlying tables the SQL Access Advisor makes the appropriate recommendations. You may choose to limit the advisor to recommendations based on a single access method.

Recommendation Types

- [x] Indexes
- [ ] Materialized Views

Advisor Mode

- [x] Limited Mode: Perform analysis that includes highest cost statements
- [ ] Comprehensive Mode: Perform an exhaustive analysis

Advanced Options
Advanced Options

Space Restrictions
Indexes and materialized views increase performance at the cost of space. When the SQL Access Advisor is invoked with no space limitations it will make the best possible performance recommendations. In the review step you can pick and choose which recommendations to implement. If you have a hard space limit you can pass that to the advisor such that any generated recommendation will fit within the space requirement. Do you wish the sum of the recommendations sizes to fit within a certain space limit?
- No, show me all recommendations
- Yes, space is limited. Space Limit: 10 GB

Tuning Options
- Prioritize tuning of SQL statements by Optimizer Cost
  SQL statements will be analyzed in descending order of the value of the prioritized statistic. Optimizer Cost will be used as the secondary sort criteria
- Limit the number of indexes on a table. Index Limit: 10

Default Storage Locations
By default indexes will be placed in the schema and tablespace of the table they reference. Materialized views will be placed in the schema and tablespace of the first table referenced in the query. You may override these defaults.
- Index Tablespace
- Schema
- Materialized View Tablespace
- Schema
Advanced Tuning Options

- SQL statements will be tuned according to the resources they use.
Schedule Job

SQL Access Advisor: Schedule

Database demoDB

The Advisor Task will be scheduled to run immediately by default. You may choose to schedule the task to run at a later time.

Advisor Task Name
Task Name: Lilian_hyper_run
Task Description: SQL Access Advisor - Hypothetical

Scheduling Options
Time Zone: GMT-4:00

Start
• Immediately
• Later
  Date: Jul-23-2003
  Time: 05:00

Repeat
• One Time Only
• Interval
  Frequency: 1
  Days

Repeat Until
• Indefinite
• Custom
  Date: Jul-23-2003
  Time: 05:00
Review & Submit

SQL Access Advisor: Review

Database: demoDB

Task Name: Lillian_hyp_run
Task Description: SQL Access Advisor - Hypothetical Run
Scheduled Start Time: 23 Jul 2003 07:05:00

Advisor Options

This is a summary of the SQL Access Advisor options you have requested. Please review this information. To change any of these options, you must use the Back button to revisit the page where the option was initially set.

<table>
<thead>
<tr>
<th>OPTION</th>
<th>VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advisor Mode</td>
<td>Limited Mode</td>
</tr>
<tr>
<td>Recommendation Type</td>
<td>Indexes Only</td>
</tr>
<tr>
<td>Storage Limitation</td>
<td>10(GB)</td>
</tr>
<tr>
<td>Workload Scope</td>
<td>Partial Workload</td>
</tr>
<tr>
<td>Workload Source</td>
<td>Hypothetical</td>
</tr>
<tr>
<td>Workload Type</td>
<td>Allow Advisor to determine workload type based on workload content</td>
</tr>
</tbody>
</table>
Monitor the Job

ORACLE Enterprise Manager

Host: gollison-sun.us.oracle.com > Database: database > Advisor Central

Advisors
- ADDM
- SQL Tuning Advisor
- SQL Access Advisor
- Memory Advisor
- MTTR Advisor
- Segment Advisor
- Undo Management

Advisor Tasks

Search
Select an advisory type and optionally enter a task name to filter the data that is displayed in your results set.

Advisory Type Task Name Advisor Runs
- SQL Access Advisor Last 24 Hours

Results

<table>
<thead>
<tr>
<th>Select</th>
<th>Advisory Type</th>
<th>Name</th>
<th>Description</th>
<th>User Status</th>
<th>Start Time</th>
<th>End Time</th>
<th>Expires In (days)</th>
</tr>
</thead>
</table>
View Recommendations

SQL Access Advisor: View Recommendations by SQL
The SQL Access Advisor can improve the following sql statements. By selecting a SQL statement you will choose to implement, edit or generate a report on all the recommendations that affect that statement.

SQL Statements Improved by Recommendations
The following chart and table initially list SQL statements ordered by their percentage improvement by the recommendations. The top SQL statement will be the SQL statement that is improved the most by the recommendations. Selecting a column header in the table will re-sort the table by that column and also change the chart to show graphically the column values.

<table>
<thead>
<tr>
<th>Statement</th>
<th>Recommendation Id</th>
<th>Workload Improvement</th>
<th>Original Cost</th>
<th>New Cost</th>
<th>Execution Count</th>
<th>Access Structures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Select sum(e.sal), d.deptno, m.mgr from emp e, dept d, emp f...</td>
<td>1</td>
<td>23</td>
<td>400</td>
<td>365</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>Select sum(d.amount) from debits d where ...</td>
<td>2</td>
<td>15</td>
<td>400</td>
<td>370</td>
<td>125</td>
<td>5</td>
</tr>
<tr>
<td>Select deptno,count(*) from scott.emp group by deptno</td>
<td>1</td>
<td>10</td>
<td>20</td>
<td>18</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>Select empno from scott.emp where sal &lt; 10000</td>
<td>3</td>
<td>0</td>
<td>10</td>
<td>10</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>
## Implement Recommendations

### SQL Access Advisor: Review Recommendations by SQL

The SQL Access Advisor can improve the following SQL statements. By selecting a SQL statement you will choose to implement, edit or generate a report on all the recommendations that affect that statement.

#### SQL Statements Improved by Recommendations

The following chart and table initially list SQL statements ordered by their percentage improvement by the recommendations. The top SQL statement will be the SQL statement that is improved the most by the recommendations. Selecting a column header in the table will re-sort the table by that column and also change the chart to show graphically the column values.

#### Workload Improvement %

![Graph showing bar chart for workload improvement](image)

#### Select SQL statements to be improved

<table>
<thead>
<tr>
<th>Select</th>
<th>Statement</th>
<th>Recommendation Id</th>
<th>Workload Improvement %</th>
<th>Original Cost</th>
<th>New Cost</th>
<th>Execution Count</th>
<th>Access Structures</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><code>SELECT SUM(e.sal), d.deptno, m.mgr FROM emp e, dept d, emp f ...</code></td>
<td>1</td>
<td>23</td>
<td>400</td>
<td>365</td>
<td>100</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td><code>SELECT SUM(d.amount) FROM debits d WHERE ...</code></td>
<td>2</td>
<td>15</td>
<td>400</td>
<td>370</td>
<td>125</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td><code>SELECT deptno, COUNT(*) FROM scott.emp GROUP BY deptno</code></td>
<td>1</td>
<td>10</td>
<td>20</td>
<td>18</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td><code>SELECT empno FROM scott.emp WHERE sal &lt; 10000</code></td>
<td>3</td>
<td>0</td>
<td>10</td>
<td>10</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>
Tips, Do’s, Don’t’s
Capturing SQL Workload

- Use cursor cache capture capability of STS
  - Run workload
  - Capture workload in STS simultaneously

```sql
DBMS_SQLTUNE.CAPTURE_CURSOR_CACHE_SQLSET(
  sqlset_name => 'MY_STS',
  time_limit => 3600,
  repeat_interval => 60,
  sqlset_owner => own);
```
Cursor sharing can be enforced at database or SQL level
- Database level: CURSOR_SHARING=FORCE
- SQL level: Set SQL Profile attribute
  FORCE_MATCH=TRUE

```sql
DBMS_SQLTUNE.ACCEPT_SQL_PROFILE (task_name => 'my_sql_tuning_task,',
                                  name => 'my_sql_profile',)
force_match => TRUE);
```
Tuning Considerations

• Resource consumption
  – Running SQL advisors consumes CPU, I/O, memory and can affect system performance

• Potential negative impact of recommendation
  – Implementing recommendations *may* impact system negatively, e.g., optimizer stats refresh

• Before initiating tuning, answer the following questions
  • How much resources will be consumed by tuning activity?
  • Can the system spare resources needed for tuning?
  • How can the production system be shielded from possible negative impact of tuning actions?
Resource Consumption of SQL Advisors

- SQL Tuning Advisor
  - **Limited mode**: Resource consumption minimal
    - Stats, index and SQL restructure analysis is cheap
    - Average is less than 1 second per SQL statement
  - **Comprehensive mode**: Resource consumption may be significant
    - SQL Profiling can potentially consume non-trivial resources
    - Roughly comparable to amount of resources/time consumed when executing SQL statement(s)

- SQL Access Advisor
  - Resource consumption depends on size of SQL workload
  - For small number of SQL, resource consumption not very high
Tuning Options

• Options
  – Direct tuning of live system
  – Remote tuning

• Live system tuning
  – Run SQL Tuning Advisor in Comprehensive mode
  – Run SQL Tuning Advisor in Limited mode only if …
    • System does not have spare resources to tune SQL
  – Run SQL Access Advisor for few SQL at a time

• Perform remote tuning, if …
  – Cumulative resources/time consumed by all SQL statements being tuned significant
  – System cannot spare resources
Live System Tuning Tips

1. Ensure tables referenced in SQL have representative optimizer stats

2. Run SQL Access Advisor
   - For individual SQL, set “Recommendation Type” to indexes
   - MV not suitable for tuning individual statements

3. Run SQL Tuning Advisor
   - Test profile before making it PUBLIC

```sql
DBMS_SQLTUNE.ACCEPT_SQL_PROFILE (task_name => '<tuning task name>',
category => 'MY_CATEGORY');

ALTER SESSION SET SQLTUNE_CATEGORY=‘MY_CATEGORY’ ;
```

- Once satisfied with results set category to DEFAULT

```sql
DBMS_SQLTUNE.ALTER_SQL_PROFILE();
```
Remote Tuning Tips

- Performed to shield production system from performance impact of running SQL advisors

1. Move SQL to Test
2. Tune SQL
3. Apply recomm. on production
Remote Tuning Tips

• Move SQL to test
  – Use Transportable STS
    • DBMS_SQLTUNE.XXX_STGTAB_SQLSET

• Apply recommendation on production
  – Use Transportable SQL Profiles
    • DBMS_SQLTUNE.XXX_STGTAB_SQLPROF
## Automatic SQL Tuning Summary

SQL Tuning Advisor and SQL Access Advisor together completely automate SQL tuning.

<table>
<thead>
<tr>
<th>Analysis Types</th>
<th>Performed By</th>
</tr>
</thead>
<tbody>
<tr>
<td>Statistics</td>
<td>SQL Tuning Advisor</td>
</tr>
<tr>
<td>SQL Profile</td>
<td>SQL Tuning Advisor</td>
</tr>
<tr>
<td>SQL Structure</td>
<td>SQL Tuning Advisor</td>
</tr>
<tr>
<td>Access Path: Indexes</td>
<td>SQL Tuning/Access Advisor</td>
</tr>
<tr>
<td>Access Path: Materialized Views</td>
<td>SQL Access Advisor</td>
</tr>
<tr>
<td>Access Path: Materialized View Logs</td>
<td>SQL Access Advisor</td>
</tr>
</tbody>
</table>
Conclusion

• SQL Advisors help address critical SQL tuning challenges
• Provides targeted and automated tuning
• Makes possible comprehensive tuning of packaged applications
•Eliminates need for highly skilled performance experts
Thank You!