

Managing Performance Through Versioning of Statistics

Claudia Fernandez
Technical Services Manager
claudia@leccotech.com

LECCOTECH
www.leccotech.com

NoCOUG, August 2003

"Managing Performance Through Versioning of Statistics"

Learn how to use versioning of statistics in Oracle 9i to manage and enhance application performance. This presentation will exemplify how statistics between different environments such as development and production can be imported/exported to simulate production data volumes and manage performance changes.

Cost-based Optimizer

- Evaluates a SQL statement and determines the execution plan
- CBO can be influenced by factors such as:
 - Configuration parameters
 - Hints
 - Statistics
 - Others..

LECCOTECH – delivering ultimate database optimization solutions

This presentation focuses on the importance of gathering stats for the CBO with the DBMS_STATS package. Also, we will review different techniques to maintain performance through management of statistics.

Execution Plans

The screenshot displays the Oracle SQL Developer interface with the following components:

- SQL Statement:**

```

SELECT EMP_NAME,
       DPT_NAME,
       GRD_DESC
INTO EMPLOYEE_NAME,
       DEPARTMENT_NAME,
       GRADE_DESC
FROM EMPLOYEE,
     DEPARTMENT DEPARTMENT1,
     GRADE
WHERE EMP_GRADE = GRD_ID
AND EMP_DEPT = DPT_ID
AND EXISTS (SELECT 'X'
            FROM DEPARTMENT DEPA
            WHERE DPT_AVG_SALARY
              < DPT_ID = EMPLOY

```
- Execution Plan Table:**

Execution Step	Object Name	Object Owner	Est. Cost	Est. Row Count
11	Select Statement [CHOOSE]		1,693	14,999
10	Filter			2
5	Hash Join	DEPARTMENT	1,693	14,999
4	Hash Join	DEPARTMENT	1,689	14,999
2	Table Access (Full) [ANALYZED]	GRADE	2	1,100
3	Table Access (Full) [ANALYZED]	EMPLOYEE	1,688	14,989
9	Table Access (By Index Rows) [ANALYZED]	DEPARTMENT	2	1
6	Index (Unique Scan) [ANALYZED]	DEPARTMENT_PK	1	1
8	Sort (Aggregate)			1
7	Table Access (Full) [ANALYZED]	DEPARTMENT	2	306
- Information:**

Problematic SQL Statement
5 Table operations found in execution plan exceeding the upper limit of the Complex SQL Table Range.
Full table scan with table size larger than the Problematic SQL Full Table Scan Threshold (8 Kbytes) (SQLEXP.DEPARTMENT 60, SQLEXP.GRADE 60, SQLEXP.EMPLOYEE 64804, SQLEXP.DEPARTMENT 60)
- Connection Information:**

User: SQLEXP | Host: Sting LECCO

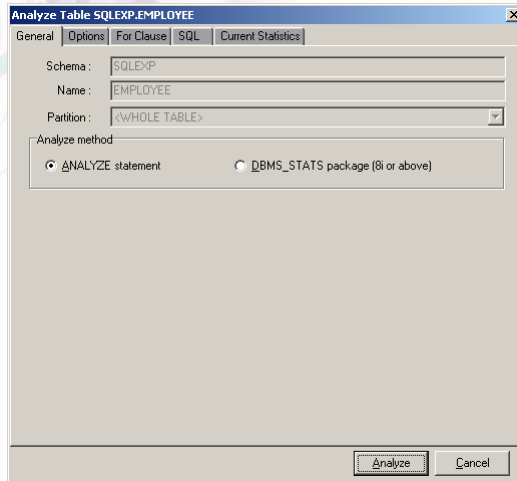
LECCOTECH – delivering ultimate database optimization solutions

This slide exemplifies a cost-based execution plan.

CBO Statistics

- CBO needs stats to make intelligent decisions when generating execution plans.
- Good choices on execution plans depend on having good statistics

Statistics



LECCOTECH – delivering ultimate database optimization solutions

Statistics can be collected using the ANALYZE command or with the DBMS_STATS package.

The DBMS_STATS package was introduced in Oracle 8i.

Analyze

- Compiles data distribution stats for tables and indexes
- ANALYZE always runs serially
- Calculates global statistics for partitioned tables and indexes which can lead to inaccuracies
- ANALYZE can gather stats that are not used by the CBO such as chained rows
- Not recommended for Oracle 8i+

LECCOTECH – delivering ultimate database optimization solutions

- Command that can be executed to gather stats for different objects.
- Oracle recommends that for Oracle 8i and above DBMS_STATS is used instead of ANALYZE. The Analyze command has several disadvantages such as that it only runs serially and it does not collect accurate info for tables and indexes with partitions and subpartitions.
- Oracle 9i Performance and Tuning Guide mentions that the future of ANALYZE will be to collect stats that are not used for the CBO.
- ANALYZE currently collects stats that the CBO uses plus other stats not used by the CBO such as chained rows.

DBMS_STATS package (Oracle 8i+)

- DBMS_STATS found in the SYS schema
- Used to gather, modify, view, export, import, and delete statistics
- Allows for generation and management of statistics for the CBO
 - Indexes
 - Tables
 - Columns
 - Partitions
 - Schema
 - System
- Gathers statistics only for cost-based optimization

LECCOTECH – delivering ultimate database optimization solutions

This package is used to gather stats. Available from Oracle 8i.

Oracle recommends the use of DBMS_STATS for Oracle8i and above.

DBMS_STATS package (Oracle 8i+)

- **Table Statistics**
 - Number of blocks
 - Number of rows
 - Avg. row length
- **Column Statistics**
 - Number of distinct values in columns
 - Number of nulls in columns
 - Data distribution (histogram)
- **Index Statistics**
 - Number of leaf blocks
 - Levels
 - Clustering factor
- **System Statistics**
 - I/O performance and utilization
 - CPU performance and utilization

Oracle9i +

LECCOTECH – delivering ultimate database optimization solutions

System Statistics are available from Oracle9i.

SYS.DBMS_STATS package (Oracle 8i+)

GATHER_INDEX_STATS	Collects index statistics.
GATHER_TABLE_STATS	Collects table, column, and index statistics.
GATHER_SCHEMA_STATS	Collects statistics for all objects in a schema.
GATHER_DATABASE_STATS	Collects statistics for all objects in a database.
GATHER_SYSTEM_STATS	Collects CPU and I/O statistics for the system. <small>Oracle9i +</small>

LECCOTECH – delivering ultimate database optimization solutions

Sample of procedures within the SYS.DBMS_STATS package

Gathering stats

- Because the CBO relies on stats to generate accurate execution plans, gather stats for all the tables and indexes used in SQL statements.
- If size and data distribution of the tables changes frequently, then regenerate the stats regularly to ensure they accurately represent the data in the tables.

Methods for gathering stats (Oracle 8i+)

- Oracle generates statistics using the following techniques:
 - Exact computation
 - Estimation based on random data sampling
 - User-defined statistics collection methods

LECCOTECH – delivering ultimate database optimization solutions

- Exact computation is the best option because it will tell Oracle exactly what data you have in your tables, indexes, etc. But this computation can use a lot of resources in large databases since it uses memory and temporary tablespace.
- If you cannot afford to perform exact computation of stats, then you can use an estimation based on a random data sampling. If the data distribution in a table is uniform, then collecting stats using a random data sampling provides a good snapshot of the data. But if the data is not uniform distributed, then the stats may not properly reflect the data.
- User-defined statistics: domain indexes.

Sample execution of dbms_stats

```
exec dbms_stats.gather_schema_stats(  
  ownname => 'SQLEXP',  
  options  => 'GATHER AUTO',  
  estimate_percent => dbms_stats.auto_sample_size,  
  method_opt => 'for all columns size repeat',  
)
```

Oracle9i +

Values for the OPTIONS parameter:

- **GATHER_** reanalyzes the whole schema
- **GATHER EMPTY_** only analyzes tables that have no existing statistics
- **GATHER STALE_** only reanalyzes tables with more than 10 percent modifications (inserts, updates, deletes)
- **GATHER AUTO_** will reanalyze objects that currently have no statistics and objects with stale statistics.

Oracle9i +

GATHER STALE and GATHER AUTO require monitoring:

```
ALTER TABLE XXX MONITORING  
  select dba_tab_modifications
```

LECCOTECH – delivering ultimate database optimization solutions

Oracle9i introduced the **dbms_stats.auto_sample_size** parameter that makes Oracle to automatically decide for you what is the most appropriate sample size to use. Oracle recommends the use of this parameter.

For Oracle8i, the sample size needs to be specified by the user.

The GATHER_STALE and GATHER_AUTO. These options allow for the computation of stats for only the tables that has more than 10 percent of modifications. GATHER_AUTO also computes stats for objects without stats. These options are more efficient since only stats for objects that changed (>10%) will be computed. GATHER_AUTO is available from Oracle9i.

You can use a DBMS_JOB to schedule the computation of the stats periodically

For more info on the options for gathering schema stats see:

Oracle8i

http://download-west.oracle.com/docs/cd/A87860_01/doc/server.817/a76992/stats.htm#25637

Oracle9i

http://download-west.oracle.com/docs/cd/B10501_01/server.920/a96533/stats.htm#26713

Stats

- DBA_ALL_TABLES
- DBA_INDEXES
- DBA_IND_PARTITIONS
- DBA_IND_SUBPARTITIONS
- DBA_OBJECT_TABLES
- DBA_PART_COL_STATISTICS
- DBA_SUBPART_COL_STATISTICS
- DBA_TABLES
- DBA_TAB_COLS
- DBA_TAB_COLUMNS
- DBA_TAB_COL_STATISTICS
- DBA_TAB_PARTITIONS
- DBA_TAB_SUBPARTITIONS

LECCOTECH – delivering ultimate database optimization solutions

Once you gather stats these views will show the stats available.

Missing Stats

Statistic Default Value Used by Optimizer

Tables

- Cardinality = 100 rows
- Average row length = 20 bytes
- Number of blocks = 100
- Remote cardinality = 2000 rows
- Remote average row length = 100 bytes

Indexes

- Levels = 1
- Leaf blocks = 25
- Leaf blocks/key = 1
- Data blocks/key = 1
- Distinct keys = 100
- Clustering factor = 800 (8 * number of blocks)

LECCOTECH – delivering ultimate database optimization solutions

When there are no stats available, the CBO uses these values when generating execution plans. It is very important that you have stats in your tables, otherwise Oracle will believe that you have 100 rows in a table that for instance has 100,000 rows. Without stats the CBO does not have the information to make good decisions when generating execution plans and the execution plans could be very inaccurate.

Example

1. Delete schema stats

```
exec dbms_stats.delete_schema_stats('SQLEXP');
```

2. Run EXPLAIN PLAN for SQL statements

3. Gather schema stats

```
exec dbms_stats.gather_schema_stats(  
    ownname      => 'SQLEXP',  
    options       => 'GATHER AUTO',  
    estimate_percent => dbms_stats.auto_sample_size,  
    method_opt    => 'for all columns size  
repeat');
```

4. Run EXPLAIN PLAN for SQL statements

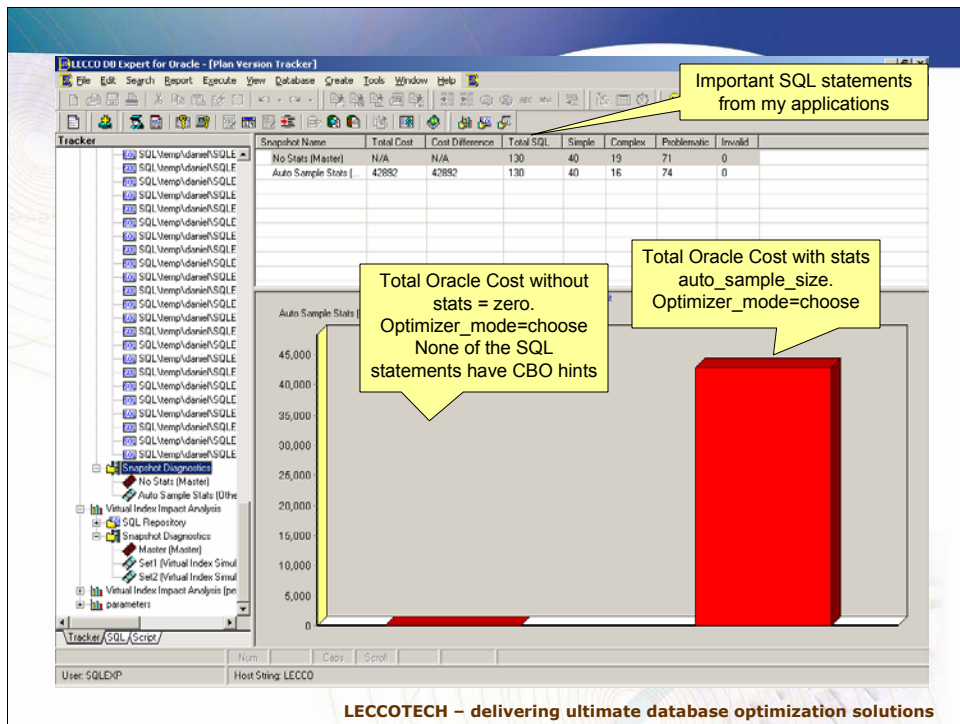
5. Compare EXECUTION PLAN snapshots

LECCOTECH – delivering ultimate database optimization solutions

This example illustrates the execution plan differences without stats and different sets of stats. I am using a test database.

Scenario:

1. I am going to delete my schema stats
2. Then I am going to run the explain plans for my important SQL statements. I am using the LECCO DB Expert product to maintain a repository of SQL statements and execution plans. If you do not have LECCO DB Expert then you can create a script that runs the explain plans for your SQL statements and manually compare the output.
3. Let's gather stats using `dbms_stats.auto_sample_size`
4. Let's get the execution plans for the SQL statements after running the stats
5. Let's compare the execution plans from step 2 and 4.



This graph illustrates the difference in accumulative Oracle Cost for the execution plan snapshots without stats and with stats computed with auto_sample_size. Whenever you gather stats, it is important that you compare the explain plan for your critical SQL statements before and after running stats. This will allow you to spot performance changes that could create performance problems.

LECCO DBI Expert for Oracle - [Plan Version Tracker]

This is an example of a SQL statement that experienced execution plan changes after running the stats

SQL	Master Cost	Snapshot Cost	Cost Diff	Changed	Master Classification	Snapshot Classification	Classifica...
SQL Vemp/insp...	N/A	1605	N/A	YES	Complex	Problematic	Degraded
SQL Vemp/insp...	N/A	N/A	N/A	NO	Complex	Complex	Unchanged
SQL Vemp/insp...	N/A	N/A	N/A	NO	Simple	Simple	Unchanged
SQL Vemp/insp...	N/A	1	N/A	NO	Simple	Simple	Unchanged
SQL Vemp/insp...	N/A	1626	N/A	YES	Complex	Problematic	Degraded
SQL Vemp/insp...	N/A	1887	N/A	YES	Complex	Problematic	Degraded
SQL Vemp/insp...	N/A	1600	N/A	YES	Complex	Problematic	Degraded
SQL Vemp/insp...	N/A	1687	N/A	YES	Complex	Problematic	Degraded
SQL Vemp/SQL...	N/A	169	N/A	NO	Problematic	Problematic	Unchanged
SQL Vemp/SQL...	N/A	169	N/A	NO	Problematic	Problematic	Unchanged
SQL Vemp/SQL...	N/A	169	N/A	NO	Problematic	Problematic	Unchanged
SQL Vemp/SQL...	N/A	173	N/A	YES	Problematic	Problematic	Unchanged
SQL Vemp/SQL...	N/A	169	N/A	NO	Problematic	Problematic	Unchanged
SQL Vemp/SQL...	N/A	14	N/A	NO	Problematic	Problematic	Unchanged

Execution plan without stats

Execution plan with stats, auto_sample_size

LECCOTECH – delivering ultimate database optimization solutions

LECCO DB Expert for Oracle - [Plan Version Tracker]

File Edit Search Report Execute View Database Create Tools Window Help

```

SELECT COUNT (*)
FROM GRADE,
EMPLOYEE,
DEPARTMENT DEPARTMENT1
WHERE GRD_ID = EMP_GRADE
AND EMP_DEPT = DPT_ID
AND EXISTS (SELECT 'X'
FROM DEPARTMENT DEPARTMENT2

```

Execution Step	Object Name	Object Owner	Est Cost
14	Select Statement [CHOOSE]		1,685
13	Sort (Aggregate)		1,685
12	Filter		1,685
6	Nested Loops		
4	Table Access... DEPARTMENT	SQLEXP	
3	Table Access... EMPLOYEE	SQLEXP	
2	Index (Ra... IDX_EMP_DEPT	SQLEXP	
5	Index (Unique Sc... GRADE_PK	SQLEXP	
11	Filter		
8	Table Access (By... DEPARTMENT	SQLEXP	
7	Index (Unique... DEPARTMENT_PK	SQLEXP	
10	Sort (Aggregate)		
9	Table Access... DEPARTMENT	SQLEXP	

Complex SQL Statement
5 Table operations found in execution plan falling in the Complex SQL Table Range.

Connection Information
User Name = sqlepx
Host String = LECCO
Connect As = Normal

Execution plan without stats

Execution Step	Object Name	Object Owner	Est Cost
11	Select Statement [CHOOSE]		1,685
10	Sort (Aggregate)		1,685
9	Nested Loops		1,685
7	Table Access (Full... EMPLOYEE	SQLEXP	
6	Index (Unique Sc... DEPARTMENT_PK	SQLEXP	
5	Table Access... DEPARTMENT	SQLEXP	
2	Index (Un... DEPARTMENT_PK	SQLEXP	
4	Sort (Aggr...		
3	Table... DEPARTMENT	SQLEXP	
8	Index (Unique Scan) [... GRADE_PK	SQLEXP	

Problematic SQL Statement
Full table scan with number of rows larger than the Problematic SQL Full Table Scan Threshold (5000 Rows).(SQLEXP.EMPLOYEE 356770)

Connection Information
User Name = sqlepx
Host String = LECCO
Connect As = Normal

Execution plan with stats, auto_sample_size

User: SQLEXP Host String: LECCO

Managing Stats (Oracle 8i+)

- Schema Stats can be imported and exported
 - Oracle 8.1.5 and above using the DBMS_STATS package
- Importing statistics to a schema will simulate the performance of one database on another
- Allows keeping different version of stats

LECCOTECH – delivering ultimate database optimization solutions

Managing Stats

“Statistics tables enable you to experiment with different sets of statistics. For example, you can back up a set of statistics before you delete them, modify them, or generate new statistics. You can then compare the performance of SQL statements optimized with different sets of statistics, and if the statistics stored in a table give the best performance, you can restore them to the data dictionary.”

--Oracle 9i, Performance and Tuning Guide

LECCOTECH – delivering ultimate database optimization solutions

Managing Stats (Oracle 8i+)

- Create stats table:

```
exec DBMS_STATS.CREATE_STAT_TABLE('SQLEXP', 'mystats');
```

“If key SQL statements experience significant performance degradation, then either gather statistics again using a larger sample size, or perform the following steps:

1. Use DBMS_STATS.EXPORT_SCHEMA_STATS to save the new statistics in a different statistics table or a statistics table with a different statistics identifier.
2. Use DBMS_STATS.IMPORT_SCHEMA_STATS to restore the old statistics. The application is now ready to run again.”

--Oracle 9i, Performance and Tuning Guide

LECCOTECH – delivering ultimate database optimization solutions

Import/Export schema stats is available from Oracle8i.

Oracle recommends that before you modify, delete, add new statistics that you keep versions of stats. Updating stats does not guarantee better system performance. If after running stats, there is performance degradation, you can recover the past performance by recovering an older version of statistics.

Export & Import stats (Oracle 8i+)

- DBMS_STATS.CREATE_STAT_TABLE(*schema_name*, *table_name*, *tablespace*);
- DBMS_STATS.EXPORT_COLUMN_STATS
- DBMS_STATS.EXPORT_INDEX_STATS
- DBMS_STATS.EXPORT_SYSTEM_STATS
- DBMS_STATS.EXPORT_TABLE_STATS
- DBMS_STATS.EXPORT_SCHEMA_STATS
- DBMS_STATS.EXPORT_DATABASE_STATS
- DBMS_STATS.EXPORT_SYSTEM_STATS — Oracle9i +
- DBMS_STATS.IMPORT_COLUMN_STATS
- DBMS_STATS.IMPORT_INDEX_STATS
- DBMS_STATS.IMPORT_SYSTEM_STATS
- DBMS_STATS.IMPORT_TABLE_STATS
- DBMS_STATS.IMPORT_SCHEMA_STATS
- DBMS_STATS.IMPORT_DATABASE_STATS
- DBMS_STATS.IMPORT_SYSTEM_STATS — Oracle9i +

NOTE: REFERENCE ORACLE DOCUMENTATION FOR PROCEDURE PARAMETERS

LECCOTECH – delivering ultimate database optimization solutions

In order to export and import statistics in Oracle, a user would first have to create a stats table to hold those statistics.

With those statistics in place, the user would then be able to export the statistics from a column, index, system, table, schema and database to that stats table.

Then by creating a dblink or by importing the stats table to the new database the user would then be able to import the new statistics to whichever column, index, system, table, schema or database he or she chooses.

http://download-west.oracle.com/docs/cd/A91202_01/901_doc/appdev.901/a89852/dbms_sta.htm#1012359

Import/Export System Statistics is available from Oracle9i.

The screenshot displays the LECCO DB Expert for Oracle interface, showing two execution plans for the same SQL query. The query is:

```
SELECT COUNT (*)
FROM GRADE,
EMPLOYEE,
DEPARTMENT DEPARTMENT1
WHERE GRD_ID = EMP_GRADE
AND EMP_DEPT = DPT_ID
AND EXISTS (SELECT 'X'
FROM DEPARTMENT DEPARTMENT2
```

Execution plan without stats (Left):

- Step 14: Select Statement [CHOOSE]
- Step 13: Sort (Aggregate)
- Step 12: Filter
- Step 6: Nested Loops
- Step 4: Nested Loops
- Step 3: Table Access (Full) DEPARTMENT SQLEXP
- Step 2: Table Access (By ... DEPARTMENT SQLEXP)
- Step 5: Index (Unique Sc... GRADE_PK SQLEXP)
- Step 11: Filter
- Step 8: Table Access (By ... DEPARTMENT SQLEXP)
- Step 7: Index (Unique... DEPARTMENT_PK SQLEXP)
- Step 10: Sort (Aggregate)
- Step 9: Table Access... DEPARTMENT SQLEXP

Execution plan with stats, auto_sample_size (Right):

- Step 11: Select Statement [CHOOSE] (Est Cost: 1,685)
- Step 10: Sort (Aggregate)
- Step 9: Nested Loops (Est Cost: 1,685)
- Step 7: Nested Loops (Est Cost: 1,685)
- Step 1: Table Access (Full) EMPLOYEE SQLEXP (Est Cost: 1,685)
- Step 6: Index (Unique Sc... DEPARTMENT_PK SQLEXP
- Step 5: Table Access... DEPARTMENT SQLEXP
- Step 2: Index (Un... DEPARTMENT_PK SQLEXP
- Step 4: Sort (Aggr...)
- Step 3: Table DEPARTMENT SQLEXP
- Step 8: Index (Unique Scan) [... GRADE_PK SQLEXP

Complex SQL Statement (Left): 5 Table operations found in execution plan falling in the Complex SQL Table Range.

Complex SQL Statement (Right): Problematic SQL Statement. Full table scan with number of rows larger than the Problematic SQL Full Table Scan Threshold (5000 Rows). [SQLEXP.EMPLOYEE 356770]

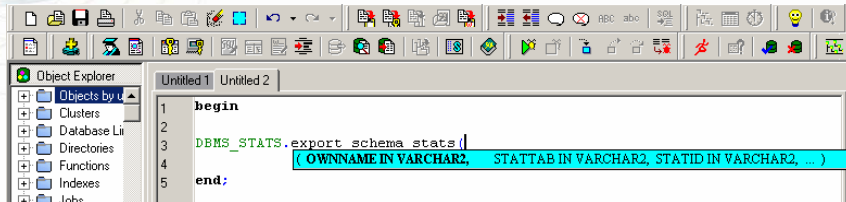
Connection Information (Both): User Name = sqlexp, Host String = LECCO, Connect As = Normal.

Let's look again at our sample that shows differences in the execution plan after gathering stats. We have already gather stats with `auto_sample_size`. Let's time this SQL statement without stats and with stats. In order to time the SQL without stats, let's keep our current stats by exporting them into a table. Then once we have backed-up the current stats, we can delete them.

Managing Stats (Oracle 8i+)

- Export Schema Stats – keeping versions

```
Exec DBMS_STATS.export_schema_stats('SQLEXP', 'mystats', '1');
```



The screenshot shows the Oracle SQL Developer interface. On the left is the Object Explorer showing a tree view of database objects. The main window, titled 'Untitled 1', contains a PL/SQL script with the following code:

```
1 begin
2
3 DBMS_STATS.export_schema_stats(
4   ( OWNNAME IN VARCHAR2, STATTAB IN VARCHAR2, STATID IN VARCHAR2, ... )
5 end;
```

LECCOTECH – delivering ultimate database optimization solutions

Exporting current stats into a user-defined table “mystats”. The STATID is a varchar parameter that we can use to name the version of the stats. In this example, let’s name them ‘1’.

The screenshot displays the Oracle SQL Developer interface. On the left, the 'Source SQL' window contains the following query:

```

SELECT COUNT (*)
FROM GRADE,
EMPLOYEE,
DEPARTMENT DEPARTMENT1
WHERE GRD_ID = EMP_GRADE
AND EMP_DEPT = DPT_ID
AND EXISTS (SELECT 'X'
FROM DEPARTMENT DEP
WHERE DPT_AVG_SALARY
AND DEPARTMENT1.DP

```

The 'SQL Information' window on the right shows the execution plan with the following steps:

Execution Step	Object Name	Object Owner	Est. Cost	Est. Row
14	Select Statement [CHOOSE]			
13	Sort (Aggregate)			
12	Filter			
6	Nested Loops			
4	Nested Loops			
3	Table Access: DEPARTMENT	SQLEXP		
2	Table Access: EMPLOYEE	SQLEXP		
5	Index (Ra... IDX_EMP_DEPT	SQLEXP		
5	Index (Unique Sc... GRADE_PK	SQLEXP		
11	Filter			
8	Table Access (By... DEPARTMENT	SQLEXP		
7	Index (Unique... DEPARTMENT_PK	SQLEXP		
10	Sort (Aggregate)			
9	Table Access: DEPARTMENT	SQLEXP		

A 'SQL Run Time' dialog box is overlaid on the execution plan, showing the following data:

Elapsed Time	Response Time	No. of Records
00:20:03.30	00:20:03.30	1

The status bar at the bottom of the window indicates 'User: SQLEXP' and 'Host String: LECCO'.

`exec dbms_stats.delete_schema_stats('SQLEXP');`

Now, since we have already backed-up the stats, let's delete them to start again without stats.

The figure shows the execution plan of the SQL statement without stats.

The elapsed time of the SQL statement is 20 minutes.

Managing Stats

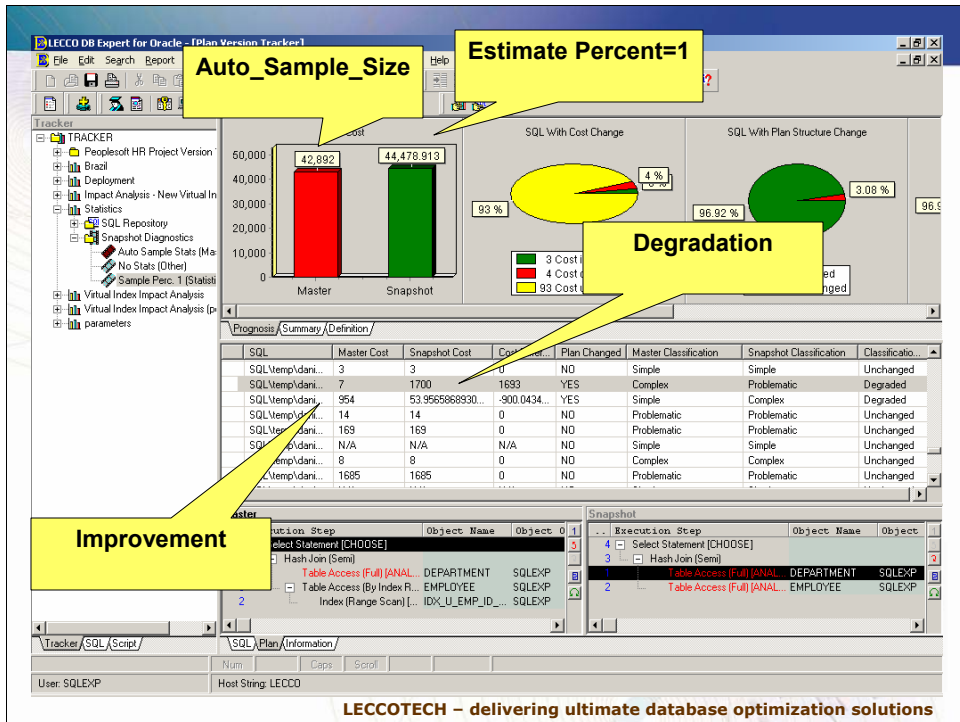
- Let's try collecting stats with a different estimate sample size

```
exec dbms_stats.gather_schema_stats(  
ownname      => 'SQLEXP',  
options      => 'GATHER AUTO',  
estimate_percent => 1  
)
```

LECCOTECH – delivering ultimate database optimization solutions

Let's collect another set of statistics and compare what happens with the execution plans. We manually specify estimate_percent=1.

GATHER_AUTO is an Oracle9i option.



This figure shows the difference in total Oracle Cost and execution plans between the snapshot that uses the stats gathered with auto_sample_size and the new stats we have just gathered with estimate_percent=1.

The figure show one SQL statement with significant Oracle Cost increase which may indicate performance degradation. But there is one SQL statement that had Oracle Cost decrease which may indicate performance improvement.

To decide which is the best set of statistics to use, you need to compare the execution plans of your important SQL under each of the sets of stats. The set auto_sample_size in general looks better than the set estimate_percent=1

Managing Stats (Oracle 8i+)

- Example: recover previous stats

begin

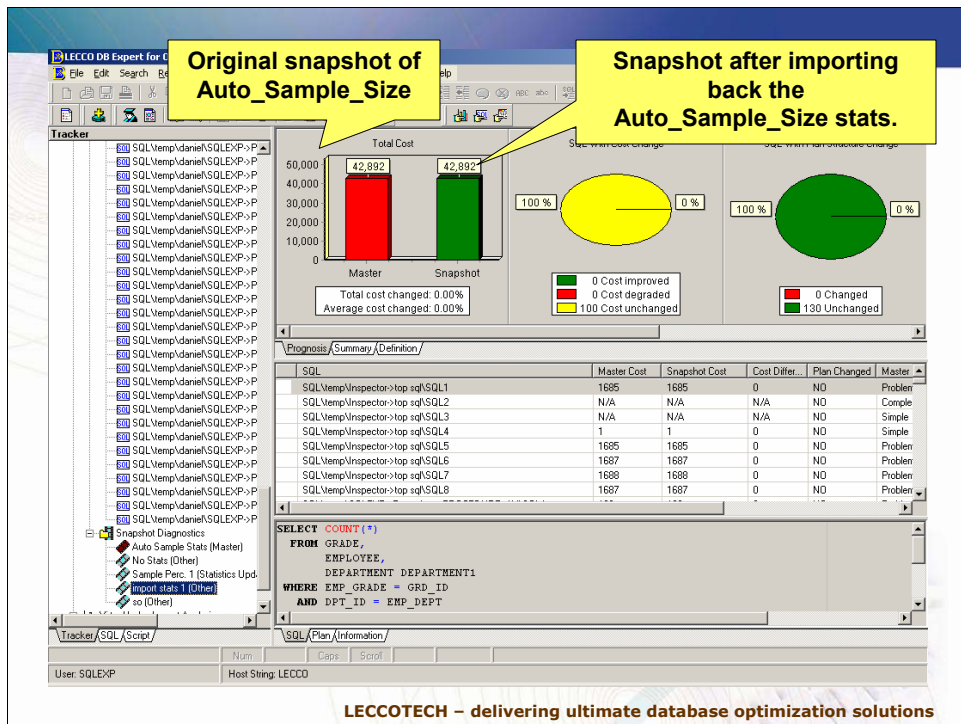
```
sys.dbms_stats.export_schema_stats('SQLEXP',  
'mystats', '2');
```

```
sys.dbms_stats.import_schema_stats('SQLEXP',  
'mystats', '1');
```

end;

LECCOTECH – delivering ultimate database optimization solutions

Following the best practice of keeping versions of stats before making changes, let's backup our current stats, estimate_percent=1 as version '2'. Let's import back the set of statistics '1', auto_sample_size.



This figure shows the total Oracle Cost and execution plans between the original snapshot of auto_sample_size and the snapshot after importing back the set of statistics 'auto_sample_size' saved in the 'mystats' table. The Oracle Cost and execution plans are the same (since nothing has changed in the database – init.ora params changes could affect execution plans.)

The screenshot displays the LECCO DB Expert for Oracle - [SQL Editor] interface. The main window shows a SQL query in the 'Source SQL' pane and its execution plan in the 'SQL Information' pane. The execution plan is a tree view showing the following steps:

Execution Step	Object Name	Object Owner	Est. Cost	Est. Row Count	Est. Byte Count
1	Select Statement [CHOOSE]		1,685	1	
2	Sort (Aggregate)			1	
3	Nested Loops		1,685	25,359	507,18
4	Table Access (Full) EMPLOYEE	SOLEXP	1,685	299,999	2,999,99
5	Index (Unique Sc... DEPARTMENT_PK	SOLEXP		15	7
6	Table Access... DEPARTMENT	SOLEXP	2	1	1
7	Index (Un... DEPARTMENT_PK	SOLEXP	1	1	1
8	Sort (Agg... DEPARTMENT	SOLEXP	2	306	1,53
9	Index (Unique Scan) [... GRADE_PK	SOLEXP		1,100	5,50

Below the execution plan, a 'SQL Run Time' dialog box is displayed with the following data:

Elapsed Time	Response Time	No. of Records
00:00:21.11	00:00:21.11	1

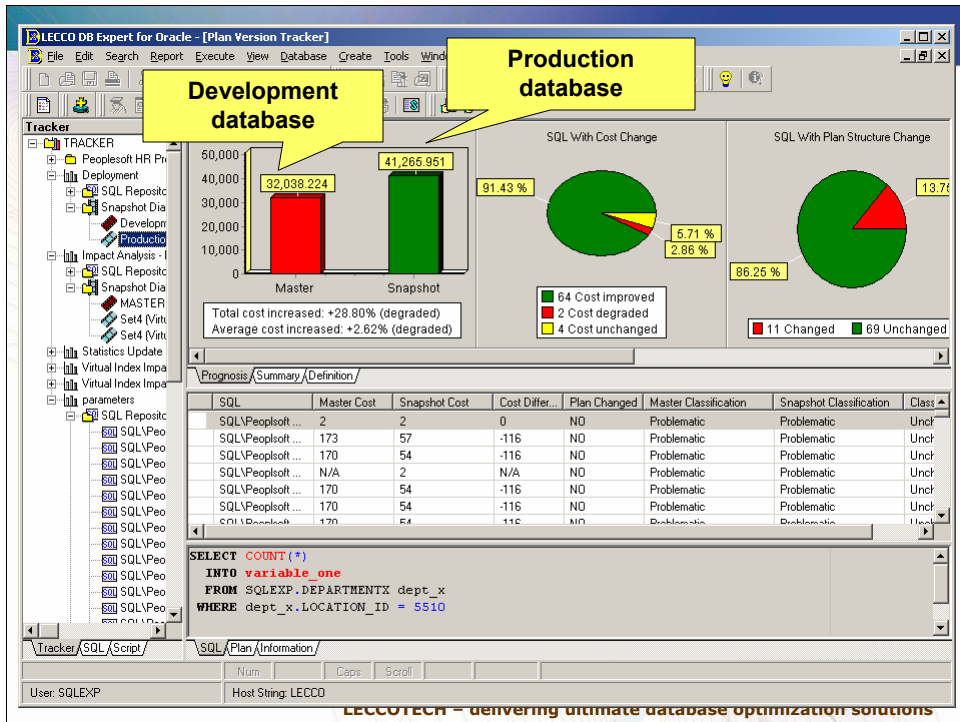
A yellow banner at the bottom of the screenshot contains the text: **Set of statistics: auto_sample_size**

The same SQL statement that took 20 minutes without stats, takes 21 seconds under auto_sample_size set of stats.

Best Practices for Managing Stats (Oracle 8i+)

- Reanalyze statistics only when necessary
- Carefully manage CBO statistics
 - Ensure that the CBO works the same in development, test and production databases
 - Migrate the production statistics into development and test environments

LECCOTECH – delivering ultimate database optimization solutions



This figure compares the Oracle Cost and execution plan for the same SQL statements under 2 databases, development and production.

Note that the total cost in development is lower than in production. The development database has less data than the production database.

Importing/Exporting the stats makes possible to have the same execution plans in development by importing the stats from production.

Note that to maintain same execution plans between 2 databases there are other factors besides stats that should be equal in the 2 databases such as init.ora params, tables/indexes/constraints, etc.

Best Practices for Managing Stats

- Ensure static execution plans
 - Re-analyzing a schema could cause thousands of SQL statements to change execution plans
 - Use Stored Outlines to freeze execution plans
 - Stored Outlines are available from Oracle8i

LECCOTECH – delivering ultimate database optimization solutions

Plan Stability (Oracle 8i+)

- Prevents database environment changes to affect execution plans
- Preserves execution plans in **Stored Outlines (SO)**
- SO is a sequence of HINTS that Oracle uses to create the execution plan
- SO are used to INFLUENCE the creation of an execution plan
- One-to-one correspondence between the SQL text and its stored outline

LECCOTECH – delivering ultimate database optimization solutions

Stored Outlines can be used to freeze execution plans. Updates to stats will no affect the execution plan of a SQL statement if a stored outline is used.

Stored Outlines (Oracle 8i+)

- SYS.USER_OUTLINES, SYS. USER_OUTLINE_HINTS: OL\$, OL\$HINTS, OL\$NODES
- Enabling Plan Stability
 - QUERY_REWRITE_ENABLED = TRUE
 - STAR_TRANSFORMATION_ENABLED = TRUE
 - OPTIMIZER_FEATURES_ENABLED > 8.1
- Privileges
 - CREATE ANY OUTLINE
- Stored Outlines are organized by CATEGORY
- PRIVATE or PUBLIC

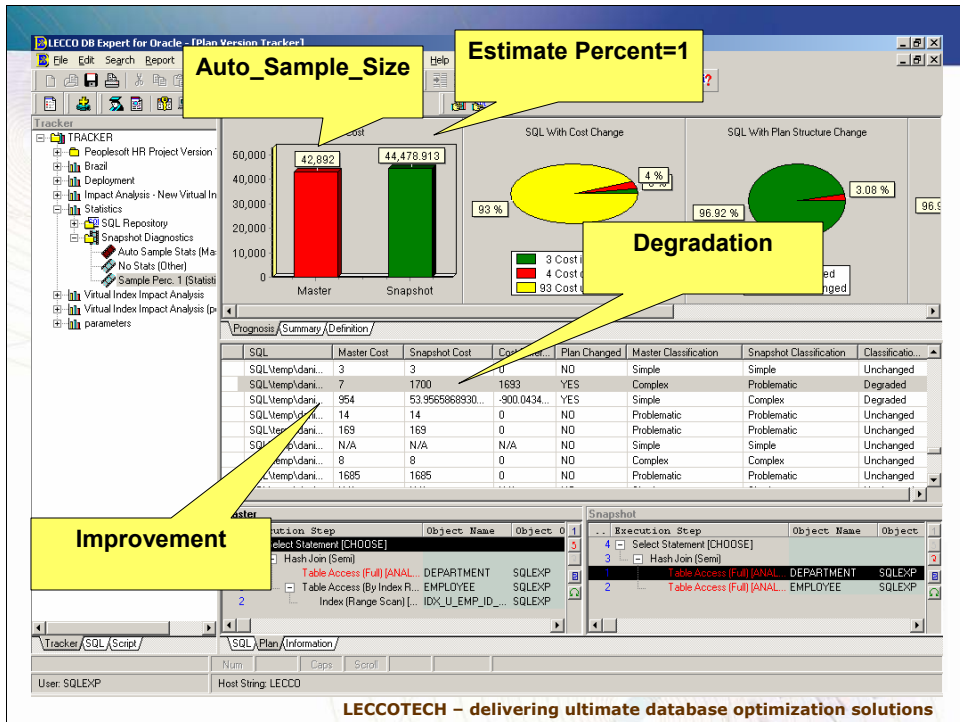
LECCOTECH – delivering ultimate database optimization solutions

PRIVATE – seen only in the current session and whose data resides in the current parsing schema

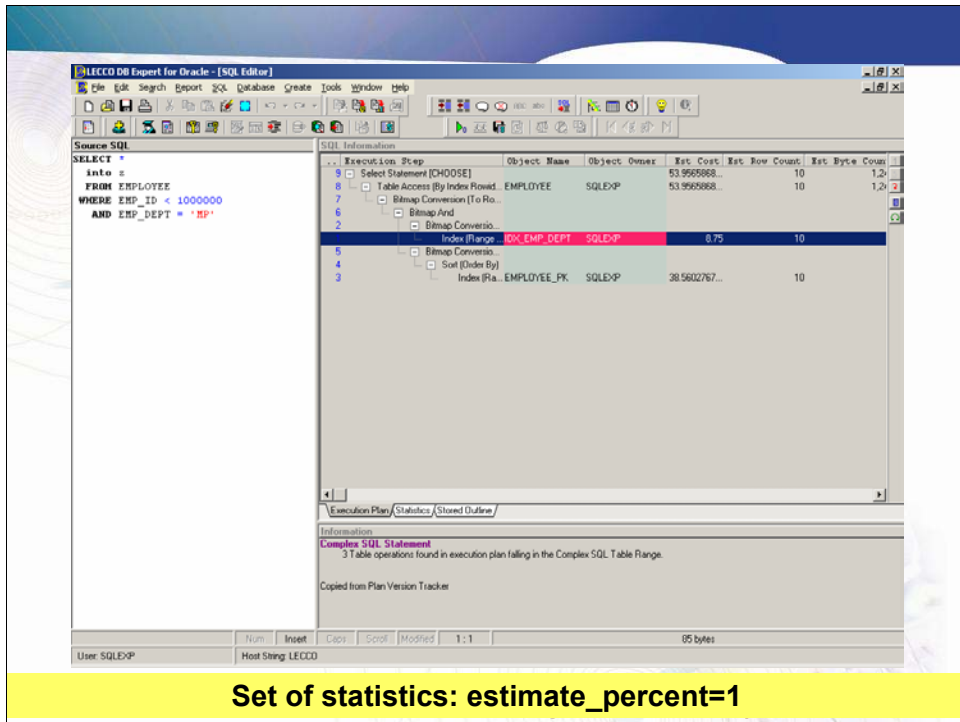
Creating Stored Outlines

- Create outline for a SQL statement
 - `CREATE OUTLINE <otl_name> FOR CATEGORY <category> ON <sql statement>`
- Automatic creation of outlines
 - `ALTER SESSION SET CREATE_STORED_OUTLINES=TRUE`
 - Use bind variables: `CURSOR_SHARING` to `SIMILAR` or `FORCE`
- Enabling outlines
 - `ALTER SESSION SET USE_STORED_OUTLINES = TRUE`
 - `ALTER SESSION SET USE_STORED_OUTLINE = CATEGORY`

LECCOTECH – delivering ultimate database optimization solutions



In the previous example where we compared the Oracle Cost and execution plans under 2 different sets of stats, auto_sample_size and estimate_percent=1 there was one SQL statement that had performance improvement under the set of stats, estimate_percent=1.



Set of statistics: estimate_percent=1

Let's recover the set of stats 2 from the mystats table. This one corresponds to the set "estimate_percent=1". The figure illustrates the execution plan for the SQL statement under this set of statistics.

Source SQL

```

SELECT *
  INTO s
FROM EMPLOYEE
WHERE EMP_ID < 1000000
AND EMP_DEPT = 'NP'

```

SQL Information

NODE	STAGE	JOIN POS	HINT
1	1	0	NOREWRITE
1	2	0	NOREWRITE
1	3	0	NO_EXPAND
1	3	0	ORDERED
1	3	0	NO_FACT(EMPLOYEE)
1	3	1	INDEX_COMBINE(EMPLO

Execution Plan / Statistics / Stored Outline

Information

Complex SQL Statement
3 table operations found in execution plan falling in the Complex SQL Table Ranges.

Copied from Plan Version Tracker

User: SQLDEV
Host String: LECCO

Set of statistics: estimate_percent=1

The figure illustrates the corresponding stored outline to the execution plan.

Save Stored Outline

Save to category: TEST
 SQL: Source
 Outline name: test
 Drop existing outline after creation
 Load min. space as default

Warning
 * The SQL text that is saved with the stored outline needs to be identical to the SQL statement in your source program, otherwise the stored outline will not be used.

SQL Text

```
SELECT *
into :v_2
FROM EMPLOYEE
WHERE EMP_ID < 1000000
AND EMP_DEPT = 'MP'
```

Stored Outline

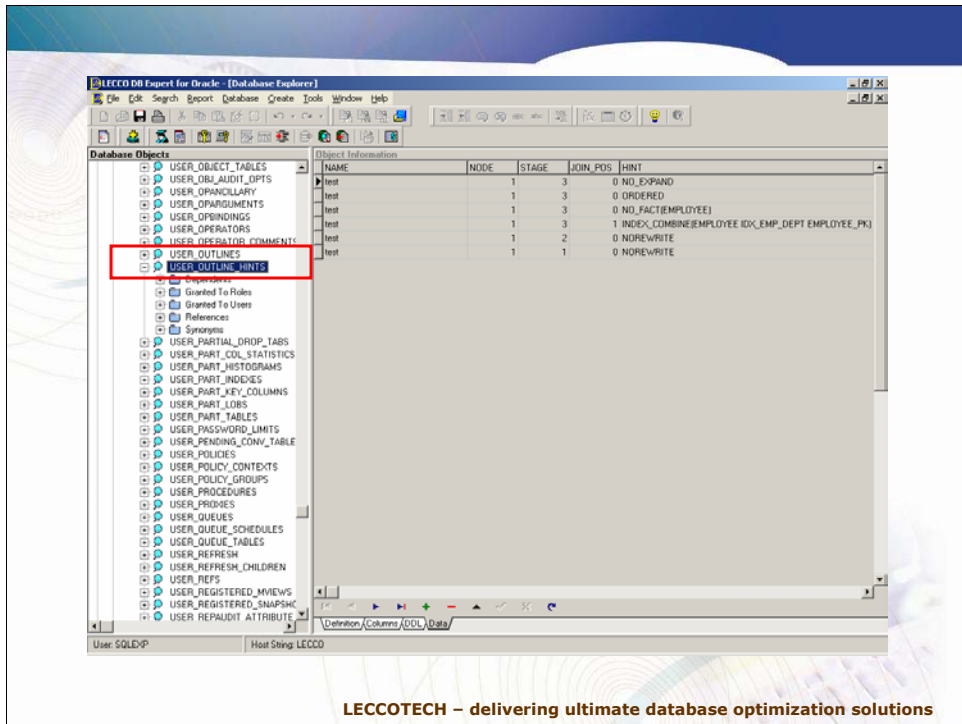
NODE	STAGE	JOIN_POS	HINT
1	1	0	NOREWRITE
1	2	0	NOREWRITE
1	3	0	NO_EXPAND
1	3	0	ORDERED
1	3	0	NO_FACT[EMPLOYEE]
1	3	1	INDEX_COMBINE[EMPLC

Reload Min. Space Reload Formatted Save Preview Cancel

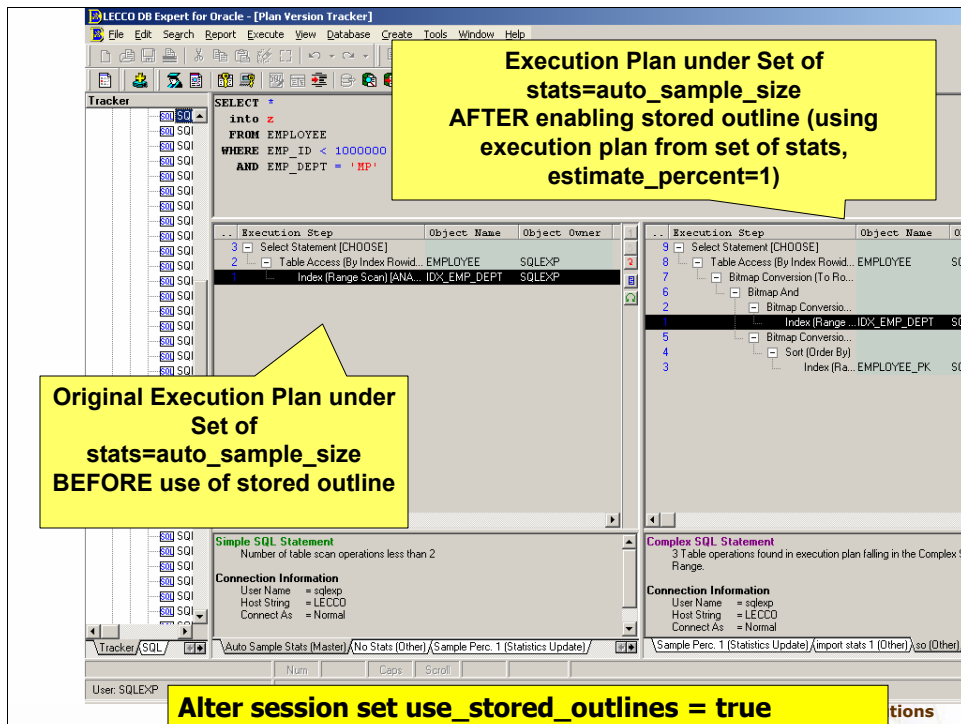
CREATE OUTLINE <otl_name> FOR CATEGORY <category> ON <sql statement>

LECCOTECH – delivering ultimate database optimization solutions

Let's save the stored outline for this SQL statement under the set of statistics, estimate_percent=1



You can see that the outline has been saved. USER_OUTLINE_HINTS view.



Let's enable the use of the stored outline, recover back the set of stats "auto_sample_size", and let's obtain again the explain plans.

The figure shows how the execution plan on the right is using the stored outline since the execution plan corresponds to the one saved under the set of stats, estimate_percent=1, while the current set of stats is "auto_sample_size".

This example shows how using different set of statistics and stored outlines it is possible to manage performance.

The background of the slide features a complex graphic. At the top, there is a dark blue horizontal bar. Below it, the background is a light, hazy blue and white. On the left side, there are several overlapping gears of different sizes and colors (green, yellow, blue). A prominent feature is a perspective view of a tunnel or a series of concentric, slightly curved lines that recede into the distance, creating a sense of depth. The overall aesthetic is technical and futuristic.

Thanks.

claudia@leccotech.com

www.leccotech.com

LECCOTECH – delivering ultimate database optimization solutions