Oracle Gather statistics

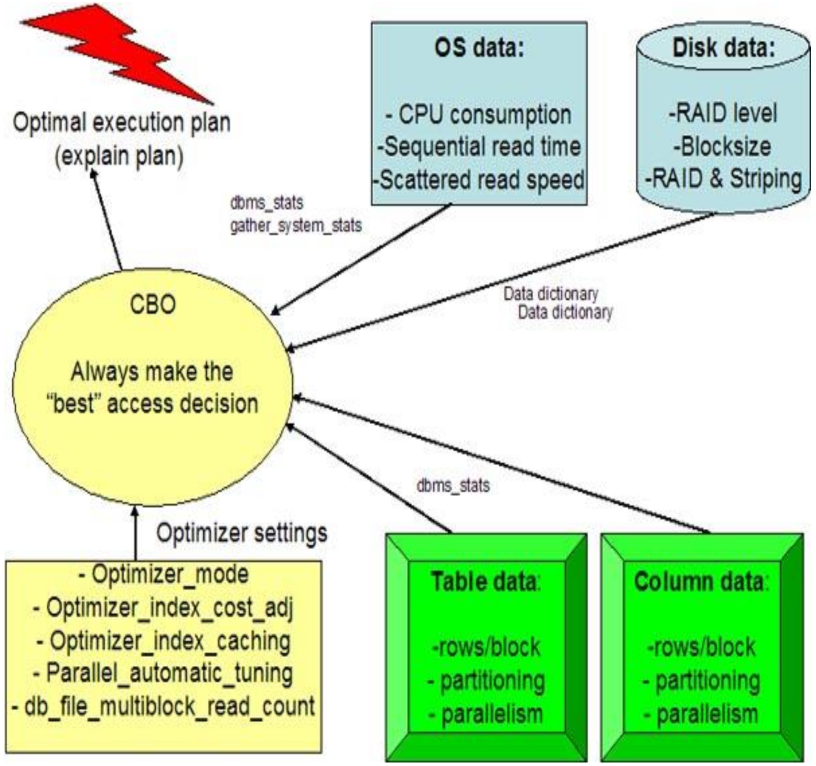
Agenda

- Introduction
- Optimizer statistics
- When to Gathering Statistics
- How to Gathering Statistics
- How to speed up gather statistics
- Gathering other types of statistics
- Conclusion
- References
- Q&A

Introduction:

- In Oracle Database 7, the Cost Based Optimizer (CBO) was introduced to deal with the enhanced functionality being added to the Oracle Database at this time, including parallel execution and partitioning, and to take the actual data content and distribution into account.
- The Cost Based Optimizer examines all of the possible plans for a SQL statement and picks the one with the lowest cost, where cost represents the estimated resource usage for a given plan. The lower the cost the more efficient an execution plan is expected to be. In order for the Cost Based Optimizer to accurately determine the cost for an execution plan it must have information about all of the objects (tables and indexes) accessed in the SQL statement, and information about the system on which the SQL statement will be run.
- This necessary information is commonly referred to as **Optimizer statistics**.

Optimizer statistics:



Optimal execution plan
(explain plan)

OS data:

- CPU consumption
-Sequential read time
-Scattered read speed

Disk data:

-RAID level
-Blocksize
-RAID & Striping

dbms_stats
gather_system_stats

Data dictionary
Data dictionary

CBO

Always make the
“best” access decision

dbms_stats

Optimizer settings

- Optimizer_mode
- Optimizer_index_cost_adj
- Optimizer_index_caching
- Parallel_automatic_tuning
- db_file_multiblock_read_count

Table data:

-rows/block
- partitioning
- parallelism

Column data:

-rows/block
- partitioning
- parallelism

Optimizer statistics:

- **Table Statistics**: Table statistics include information
  - Number of rows(NUM_ROWS)
  - Number of data blocks(BLOCKS)
  - Average row length(AVG_ROW_LEN)
- **Column Statistics**: The Optimizer uses the column statistics information in conjunction with the table statistics (number of rows) to estimate the number of rows that will be returned by a SQL operation.
  - Number of distinct values in a column (NUM_DISTINCT)
  - Number of nulls (NUM_NULLS)
  - Avg column length(AVG_COL_LEN)
  - Minimum value(LOW_VALUE)
  - Maximum value(HIGH_VALUE)
  - Histograms (HISTOGRAM)and Number of buckets(NUM_BUCKETS)
  - Density(DENSITY)
- **Additional column statistics**: histograms, column groups, and expression statistics. (Discuss later)

Optimizer statistics:

- **Index Statistics**: Index statistics provide information on
  - ➢ Number of distinct values in the index (DISTINCT_KEYS)
  - ➢ The depth of the index (BLEVEL)
  - ➢ The number of leaf blocks in the index (LEAF_BLOCKS)
  - ➢ The clustering factor(CLUSTERING_FACTOR)
- **System Statistics**: Needed for I/O and CPU costing(sys.aux_stats$)
  - ➢ Collect statistics after any new installation and/or after any hardware changes .
  - ➢ CPU Speed
  - ➢ Time taken for single block IO
  - ➢ Time taken for multi block IO
- **Fixed Object Statistics**:
  - ➢ These are statistics on the in-memory "dynamic performance" objects, the x$ and similar tables (what V$ views sit on).
  - ➢ Need to gather "once" and Re-gather after upgrade etc.
- **Data Dictionary Statistics:** SYS and some other schema part of the dictionary(if disabled auto stats job then DD stats will be outdated).

When to Gathering Statistics:

- **Automatic Statistics Gathering Job :**
- ➢ Oracle will automatically collect statistics for all database objects, which are missing statistics or have stale statistics by running an Oracle AutoTask task during a predefined maintenance window (10pm to 2am weekdays and 6am to 2am at the weekends). Oracle internally prioritizes the database objects that require statistics, so that those objects, which most need updated statistics, are processed first.
- **Gather stats if stats are stale:**
- ➢ If data is changed more than 10%, then stats will go stale.
- ➢ The STALE_STATS column in USER_TAB_STATISTICS to determine if statistics are stale. This information is updated on a daily basis only.
- ➢ If you need more timely information on what DML has occurred on your tables you will need to look in USER_TAB_MODIFICATIONS, which lists the number of INSERTS, UPDATES, and DELETES that occurs on each table.
- ➢ this information is automatically updated, from memory, periodically. If you need the latest information you will need to manual flush the information using the DBMS_STATS.FLUSH_DATABASE_MONITORING_INFO function.
- **For volatile tables**, gather stats between a load and Consumption(best).
- Delete and lock statistics(rely on Dynamic Sampling)
- Dynamic sampling collects additional statement-specific object statistics during the optimization of a SQL statement.

When to Gathering Statistics

Balance statistics:

- ➢ Sample Size
- ➢ Quality of the statistics
- ➢ Frequency of the gathering
- **Sample Size:**
- ➢ Sample size is OK, when data is not skewed or there are no histograms
- ➢ When data is very skewed you may miss low cardinality buckets, when using small sample size or AUTO sample size, then oracle assumes the number of rows for missing bucket is smallest bucket/2. so some times query works fine and some times not well.
- **Frequency :**
- ➢ If the data changes TOO quickly(i.e temp table), then delete and lock statistics. So it will use dynamic sampling.
- If your data changes frequently(Volatile Tables) :
- ➢ If you have plenty of resources , then gather statistics often and with a very large sample size
- ➢ If your resources are limited (most cases) , then gather often using new 11g features like AUTO_SAMPLE_SIZE/Incremental/Concurrent (recommended)
- If your data doesn’t change frequently(Non-volatile Tables) :
- ➢ Gather statistics less often and with a very large sample size.

How to Gathering Statistics:

- DBMS_STATS package used to gather statistics.

- Gather Schema statistics:

```
BEGIN
  DBMS_STATS.GATHER_SCHEMA_STATS (
    ownname            => 'SCHEMA_NAME_IN_CAPS',
    estimate_percent   => DBMS_STATS.AUTO_SAMPLE_SIZE,
    degree             => DBMS_STATS.AUTO_DEGREE,
    cascade            => TRUE,
    options            => 'GATHER',
    granularity        => 'AUTO',
    method_opt         => 'FOR ALL COLUMNS  SIZE AUTO');
END;
/
```

How to Gathering Statistics:

- Gather table statistics:

```
BEGIN
  DBMS_STATS.gather_table_stats (
    ownname          => 'SCHEMA_NAME_IN_CAPS',
    tabname          => 'TABLE_NAME_IN_CAPS',
    CASCADE          => TRUE,
    estimate_percent => DBMS_STATS.AUTO_SAMPLE_SIZE,
    DEGREE           => DBMS_STATS.AUTO_DEGREE);
END;
/
```

How to Gathering Statistics:

- ESTIMATE_PERCENT : The ESTIMATE_PERCENT parameter determines the percentage of rows used to calculate the statistics. Oracle Database 11g introduced a new sampling algorithm that is hash based and provides deterministic statistics. This new approach has the accuracy close to a 100% sample(accuracy for number of rows and not NDV) but with the cost of, at most, a 10% sample. By default, it will calculate 10% sample size of the table. The AUTO_SAMPLE_SIZE algorithm often chose too small a sample size when an extreme skew was present. So If table having large skewed data, then use 100% sample size.
- DEGREE :The DEGREE parameter controls the number of parallel server processes that will be used to gather the statistics. By setting the parameter DEGREE to AUTO_DEGREE, Oracle will automatically determine the appropriate number of parallel server processes that should be used to gather statistics, based on the size of an object. The value can be between 1 (serial execution) for small objects to DEFAULT_DEGREE (PARALLEL_THREADS_PER_CPU X CPU_COUNT) for larger objects.
- CASCADE : The CASCADE parameter determines whether or not statistics are gathered for the indexes on a table. By default, AUTO_CASCADE, Oracle will only re-gather statistics for indexes whose table statistics are stale.
- OPTIONS:

➢ GATHER: analyzes 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. Using GATHER AUTO is like combining GATHER STALE and GATHER EMPTY.

- METHOD_OPT : The METHOD_OPT parameter controls the creation of histograms during statistics collection. Histograms are a special type of column statistic created to provide more detailed information on the data distribution in a table column.

How to Gathering Statistics:

- GRANULARITY : The GRANULARITY parameter dictates the levels at which statistics are gathered on a partitioned table. The possible levels are table (global), partition, or sub-partition. By default Oracle will determine which levels are necessary based on the table's partitioning strategy.
- The NO_INVALIDATE parameter determines if dependent cursors (cursors that access the table whose statistics are being re-gathered) will be invalidated immediately after statistics are gathered or not. With the default setting of DBMS_STATS.AUTO_INVALIDATE, cursors (statements that have already been parsed) will not be invalidated immediately. They will continue to use the plan built using the previous statistics until Oracle decides to invalidate the dependent cursors based on internal heuristics. The invalidations will happen gradually over time to ensure there is no performance impact on the shared pool or spike in CPU usage as there could be if you have a large number of dependent cursors and all of them were hard parsed at once.

How to speed up gather statistics:

- Using parallelism:
- ➢ Intra object parallelism(DEGREE to AUTO_DEGREE)
- ➢ Inter object parallelism(CONCURRENT)
- ➢ A combination of both intra and inter object parallelism

Gather table statics and create indexes, if possible as create index will take care of gather index statistics.

Gathering other types of statistics

- **Online statistics gathering:**
  - In Oracle Database 12*c,* online statistics gathering as part of a direct-path data loading operation such as, create table as select (CTAS) and insert as select (IAS) operations.
  - Gathering statistics as part of the data loading operation, means no additional full data scan is required to have statistics available immediately after the data is loaded.
  - Online statistics gathering does not gather histograms or index statistics, as these types of statistics require additional data scans, which could have a large impact on the performance of the data load.
  - To gather the necessary histogram and index statistics without re-gathering the base column statistics use the DBMS_STATS.GATHER_TABLE_STATS procedure with the new options parameter set to GATHER AUTO.
- **Incremental statistics :** Gathering statistics on partitioned tables consists of gathering statistics at both the table level (global statistics) and (sub)partition level. If the INCREMENTAL preference for a partitioned table is set to TRUE, the DBMS_STATS.GATHER_*_STATS parameter GRANULARITY includes GLOBAL, and ESTIMATE_PERCENT is set to AUTO_SAMPLE_SIZE, Oracle will accurately derive all global level statistics by scanning only those partitions that have been added or modified, and not the entire table.
- **Partition Exchange:** By setting DBMS_STATS table preference INCREMENTAL_LEVEL to TABLE (default is PARTITION), Oracle will automatically create a synopsis for the table when statistics are gathered on it. This table level synopsis will then become the partition level synopsis after the load the exchange.

Gathering other types of statistics

- **Dictionary statistics:** Statistics on the dictionary tables are maintained via the automatic statistics gathering job run during the nightly maintenance window. It is highly recommended that you allow Oracle to automatic statistics gather job to maintain dictionary statistics even if you choose to switch off the automatic statistics gathering job for your main application schema.
- **Extended statistics:** Extended statistics help in getting a more accurate cardinality estimation when
- ▪ There is a correlation between columns.
- ▪ There is an expression(function based indexes) on column/s
- ➢ Column group are not created automatically by default.
- ➢ Below statement will show whether extended stats are required or not
- ➢ select ***dbms_stats.report_col_usage***(‘USERNAME', 'TABLE_NAME') FROM DUAL;

Customize the automatic statistic job

- **Customize the automatic statistic job:** The package DBMS_STAT allow you to configure statistic gathering with a better granularity by setting preferences for statistics at TABLE, SCHEMA, DATABASE or GLOBAL level.

➢ For example if you know that for a particular table in your schema, you need to compute statistics reading the whole table (ESTIMATE_PERCENT=100) you can configure it using:

```
dbms_stats.set_table_prefs('DBO', 'CASHRECORDS', 'ESTIMATE_PERCENT', '100');
```

➢ Another example, if you want to toggle the automatic re-analysis of a table if 5% of rows has changed instead of the default 10% you can configure the automatic gathering job by the following command:

```
begin dbms_stats.set_table_prefs('DBO', 'CASHRECORDS','STALE_PERCENT', '5'); end; /
```

Conclusion

- Most common reason to have a bad plan is having bad statistics.
  - ➢ Poor quality of the statistics.
- Second most common reason is poor rows estimate by the CBO even with good statistics.
  - ➢ Limitations of the statistics model like multiple predicated.
- Statistics are not the solution to all poorly performing plans but are the foundation to achieve better plans.
  - ➢ Good statistics have better chances to produce good plans.
  - ➢ Bad statistics have better chances to produce bad plans.
  - ➢ Good statistics sometimes produce bad plans(defect?).
  - ➢ Bad statistics sometimes produce good plans(By accident!)

TPT recommendations

- Gather statistics to with auto_sample_size when scheduled.
- Gather Statistics with Gather Stale option every day.
- Gather full statistics over the weekend.
- Notes: When performance is not as per expectations, one can consider collecting stats with 100 percent estimate for tables involved in the query performing sub optimally.

References:

- https://docs.oracle.com/cd/B28359_01/server.111/b28274/stats.htm
- https://docs.oracle.com/database/121/TGSQL/tgsql_stats.htm#TGSQL389
- http://www.oracle.com/ocom/groups/public/@otn/documents/webcontent/1354477.pdf
- http://www.oracle.com/technetwork/database/bi-datawarehousing/twp-bp-optimizer-stats-04042012-1577139.pdf
- https://oracle-base.com/articles/misc/cost-based-optimizer-and-database-statistics
- https://docs.oracle.com/cd/B28359_01/appdev.111/b28419/d_stats.htm
- http://www.oracle.com/technetwork/database/bi-datawarehousing/twp-bp-for-stats-gather-12c-1967354.pdf
- http://docs.oracle.com/cd/B28359_01/appdev.111/b28419/d_stats.htm#BEIBJJHC
- https://blogs.oracle.com/optimizer/entry/understanding_dbms_statsset__prefs_procedures