CA DataMinder

Database Statistics Reference Guide

Release 14.6

CA Technologies Product References

This document references the following CA Technologies products:

- CA DataMinder™

Contact CA Technologies

Contact CA Support

For your convenience, CA Technologies provides one site where you can access the information that you need for your Home Office, Small Business, and Enterprise CA Technologies products. At http://ca.com/support, you can access the following resources:

- Online and telephone contact information for technical assistance and customer services
- Information about user communities and forums
- Product and documentation downloads
- CA Support policies and guidelines
- Other helpful resources appropriate for your product

Providing Feedback About Product Documentation

If you have comments or questions about CA Technologies product documentation, you can send a message to techpubs@ca.com.

To provide feedback about CA Technologies product documentation, complete our short customer survey which is available on the CA Support website at http://ca.com/docs.

Contents

Chapter 1: About Database Statistics

This section contains the following topics:

About the Wgn_Stats Package

To provide a standard framework to gather Oracle cost based optimizer (CBO) statistics, a standard pl/sql package 'wgn_stats' has been developed and is now shipped with CA DataMinder.

The package is automatically installed in the CA DataMinder schema, and is available for a DBA to call as part of the normal operational procedures for maintaining the database, along with other activities such as partition maintenance, purging backups and so on.

This standard package removes some of the burden from DBAs of having to design and set up a statistics gathering process for CA DataMinder. It also allows CA to improve customer support by having a single point of reference when discovering how a customer is gathering CBO statistics.

Our experience has shown that different customers have different volumes and patterns of data, and even a single customer's data volume changes over time. The package is therefore flexible enough to allow these differences to be accommodated, while at the same time providing ease of use for simple installations.

Note: The statistics gathered by this package are specific to CA DataMinder and do not replace the gathering of system statistics (that is, Workload Statistics for system performance) or the Oracle Data Dictionary statistics.

Which Versions of CA DataMinder and the Database Schema?

The wgn_stats package can be used with version 5.0 or later of the CA DataMinder product. We expect it to also work with versions 4.5 and 4.7, but these versions have not been tested.

The wgn_stats package is dependent on the CA DataMinder database schema 3.50.

Note: Before r12.0, the CA DataMinder product was previously called 'Orchestria APM'.

What Is in this Document?

In this document, Package Components section describes the main package components while Package Usage section uses example scenarios to illustrate alternative strategies for gathering statistics.

Section Package Detail describes the main methods (procedures) that the package exposes, with particular emphasis on the GATHER_ALL_STATS procedure and the arguments that it supports.

Section Package Customization describes how to customize your statistics gathering using wgn_stats_cust. This section describes the supported attributes for the user-defined types and the customizable interfaces.

More information:

Package Components (see page 9)
Package Customization (see page 21)
Package Detail (see page 15)
Package Usage (see page 11)

Chapter 2: Package Components

The main components of the wgn_stats package are:

wgn_stats

This is the main pl/sql package as shipped by CA.

wgn_stats_cust

This is a pl/sql package that permits customizations to be made to wgn_stats.

wgn_stats_param

This is an abstract data type that is used to define how customized statistics are applied to an object.

The following components are installed in the database:

Component	Description
wgn_stats	Package specification
wgn_stats	Package body
wgn_stats_cust	Package specification
wgn_stats_param	Object type
wgn_stats_param_tab	Objects type (table of wgn_stats_param)

The wgn_stats package provides comprehensive configuration options, which reduce the need for package customizations. However, provision has still been made for customization if required.

Customization is handled via the wgn_stats_cust package, for which only a package specification is installed, defining the interfaces that are available for customization purposes. It is the responsibility of the DBA or CA professional services to define and implement the corresponding wgn_stats_cust package body, if customizations are required.

The user defined types wgn_stats_param and wgn_stats_param_tab are only of interest when implementing customizations.

Chapter 3: Package Usage

It is impractical to give strict guidelines on gathering optimizer statistics that cover all situations. This is because there are too many factors to consider. This section considers a number of example environments and how wgn_stats might be used in each.

In practice, the best approach is to choose the scenario that best matches your environment, measure how long the statistics gathering takes, and consider how frequently to execute this based on your business needs and the impact the gathering may have on other operations.

More information:

Methods (see page 15)

This section contains the following topics:

Example Scenarios (see page 11)

Example Scenarios

Scenario 1: Fresh Install, Low Data Volumes

In this scenario, it is generally advisable to collect full statistics on a daily basis. All parameters can be left at their defaults except for P_def_cap_samp_pct, which we explicitly set to 100 to ensure that full statistics are gathered on all tables.

```
BEGIN
        WGN_STATS.GATHER_ALL_STATS(P_def_cap_samp_pct=>100);
END;
```

Scenario 2: Non-captured Data

The previous example only dealt with gathering statistics on the captured data and audit data tables, as these are the tables that in general are constantly changing and subject to searching and reporting. However many large customers perform a periodic synchronization process with external user and group information. When these synchronizations occur it is advisable to gather statistics on the non-captured tables. This could be combined with the previous example if it occurs at say the same weekly frequency, or if not it could be done separately as follows:

```
BEGIN
WGN_STATS.GATHER_ALL_STATS(
  P_def_audit_samp_pct=>0,
  P_def_non_cap_samp_pct=>100,
  P_def_cap_samp_pct=>0
  );
END;
```

Interpreting the above parameters means:

- Gather no statistics on the audit tables (P_def_audit_samp_pct=>0)
- Gather no statistics on the captured data tables (P_def_cap_samp_pct =>0)
- Gather full statistics on other tables (P_def_non_cap_samp_pct =>100)

Scenario 3: High Volumes, Daily and Weekly Statistics

This scenario covers a mature installation that has been capturing data for many months or years. The database is partitioned on a daily basis and most new data is captured in the current daily partition. In this case, we might define two different configurations: one for daily use and one for use at the weekend (when we assume there is more spare resource).

Daily configuration

This gathers statistics at a low percentage for just the current day's data of the main captured data tables; to gather statistics at a higher sample rate for the audit tables; and to exclude all other tables from statistics gathering. So an example usage for daily statistics gathering in this scenario might be:

```
BEGIN
WGN_STATS.GATHER_ALL_STATS(
    P_def_audit_samp_pct=>50,
    P_def_non_cap_samp_pct=>0,
    P_def_cap_samp_pct=>5,
    P_num_partitions=>1);
END;
```

Interpreting the above parameters means:

- Gather statistics on the audit tables at 50% (P_def_audit_samp_pct=>50)
- Gather statistics on the captured data tables at 5% (P_def_cap_samp_pct =>5)
- Gather no statistics on other tables (P_def_non_cap_samp_pct =>0)
- Gather statistics on the most recent partition only (P_num_partitions=>1)

Weekend configuration

Once a week, when more resource is available on the server, we may want to gather statistics at a higher sample rate on the most recent 7 partitions and, assuming we are using a large multiprocessor server, also exploit higher degrees of parallelism.

```
BEGIN
WGN_STATS.GATHER_ALL_STATS(
    P_def_audit_samp_pct=>50,
    P_def_non_cap_samp_pct=>0,
    P_def_cap_samp_pct=>25,
    P_num_partitions=>7,
    P_degree=>32);
END;
```

Interpreting the above parameters means:

- Gather statistics on the audit tables at 50% (P_def_audit_samp_pct=>50)
- Gather statistics on the captured data tables at 25% (P_def_cap_samp_pct =>25)
- Gather no statistics on other tables (P_def_non_cap_samp_pct =>0)
- Gather statistics on the most recent partition only (P_num_partitions=>7)
- Use a degree of parallelism of 32 (P_degree=>32)

Scheduling

As outlined previously, it is not possible to give completely hard and fast rules on what the exact frequency and sampling percentage should be used to gather statistics, as so much will depend on the volumes of data, the rate at which new data arrives, and the available resources on the database server at various times during the working day/week.

As an initial recommendation we suggest gathering statistics on a daily basis until the data volumes grow to a point where this becomes impractical.

At the point this becomes impractical, we recommend that statistics are still gathered on a daily basis, but with the number of partitions analyzed, and the sampling percentage reduced.

In addition to this reduced daily statistics gathering, we recommend that fuller statistics can be gathered across more partitions at a more convenient interval, such as the weekend. This could also be combined with gathering statistics on user and group data if this corresponds to the synchronization process.

One final consideration is the purge process. In general, the purge process is run out of normal hours, once this is complete, we highly recommend that statistics are collected.

Chapter 4: Package Detail

WGN_STATS is the main CBO statistics gathering package supplied by CA. This section describes the main methods that the package exposes for use and details about associated logging and error reporting.

This section contains the following topics:

Methods

Note: All methods are described here for completeness, but only GATHER_ALL_STATS should ever be used. The other lower-level methods are called by GATHER_ALL_STATS and may be useful on when testing customizations, as described later.

More information:

Method: GATHER_ALL_STATS

This is the main procedure (entry point) into the package. Ordinarily this should be the only procedure that a customer is interested in using. Its purpose is to gather CBO statistics on all the relevant objects in the CA DataMinder database schema.

It has a long list of arguments, each of which has a default value. The intention is that for a new installation it should be sufficient to invoke this procedure on a scheduled basis, allowing all the parameters to default.

```
PROCEDURE Gather_all_stats(p_pn  IN NUMBER DEFAULT  1,
        p_indexes               IN NUMBER DEFAULT  1,
        p_num_partitions        IN NUMBER DEFAULT -1,
        p_process               IN NUMBER DEFAULT  1,
        p_Degree                IN NUMBER DEFAULT  4,
        p_lock                  IN BOOLEAN DEFAULT TRUE,
        p_delete_gtt_stats      IN BOOLEAN DEFAULT TRUE,
        p_set_rls_stats         IN BOOLEAN DEFAULT TRUE,
        p_custom_call           IN BOOLEAN DEFAULT TRUE,
        p_def_cap_samp_pct      IN NUMBER DEFAULT  15,
        p_def_non_cap_samp_pc   IN NUMBER DEFAULT  100,
        p_def_audit_samp_pct    IN NUMBER DEFAULT  100,
        p_debug                 IN NUMBER DEFAULT  0);
```

The procedure operates by gathering statistics on all tables in the current schema that start with the prefix 'Wgn'. Then, depending on the supplied parameters and/or customizations, it will gather statistics on zero or more partitions for each table. It will optionally gather statistics on all indexes, and again for zero or more index partitions.

The number of partitions that have statistics gathered (for partitioned tables and indexes) depends on the p_num_partitions parameter, which directs it to gather statistics starting from the current partition and working backwards to the specified number of partitions. By default, all partitions have statistics collected.

By default, the procedure removes any statistics that have been collected on global temporary tables that start with the prefix 'TMP'. Additionally it will also set some user statistics on selected columns in the TMP_Wgn3AddrRLS and Wgn3EventParticipant tables that have been known to give inaccurate cardinality estimates without these statistics being set.

The following table describes each of the parameters in detail.

Parameter name	Values		Default	Description
p_pn	0 1	Global Global & partition	1	Defines whether statistics are to be collected at global or global and partition level.
P_indexes	0 1	No indexes Indexes	1	Defines if statistics are to be collected on indexes as well as tables.
P_num_partitions	Positive number		-1	Specifies the number of partitions to collect statistics for. Defaults to -1, meaning all partitions.

P_process	Positive number	1	A user defined process number. For example, 1 might be used for a daily process and 2 for a weekly process.
P_degree	Positive number	4	Sets the degree of parallelism to use.
P_lock	Boolean	True	Locks statistics on completion.
P_delete_gtt_stats	Boolean	True	Deletes statistics on global temporary tables.
P_set_stats_rls	Boolean	True	Sets user-defined column cardinality statistics on some key columns
P_custom_call	Boolean	True	Makes a call to a user defined method in wgn_stats_cust. Only applicable if using customizations.
P_def_cap_samp_pct	Positive number	15	Default sampling percentage used for main captured data tables (see Note 1 below).
P_def_non_cap_samp_pct	Positive number	100	Default sampling percentage used for non captured data tables (see Note 3 below).
P_def_audit_samp_pct	Positive number	100	Default sampling percentage used for audit data tables (see Note 2 below).
P_debug	0 No debug 1 Sparse 2 Verbose	0	Defines the level of debug information written by DBMS_OUTPUT.

Notes:

The captured data tables are:

- Wgn3Event
- Wgn3EventParticipant
- Wgn3Trigger
- Wgn3EA
- Wgn3Address

The audit data tables are:

- Wgn3EventAudit
- Wgn3AuditIssue
- Wgn3IssueParticipant

The non-captured data tables are all the others not listed above.

Method: GATHER_STATS

Performs a subset of the gather_all_stats functionality but does **not** do the following:

- Lock or unlock statistics
- Remove GTT stats
- Set user-defined column cardinality statistics on some key columns

All the other parameters behave like the corresponding parameters of the same name on gather_all_stats.

```
PROCEDURE Gather_Stats(p_pn                     IN NUMBER DEFAULT 1,
                       p_indexes                        IN NUMBER DEFAULT  1,
                       p_num_partitions         IN NUMBER DEFAULT  -1,
                       p_process                        IN NUMBER DEFAULT  1,
                       p_Degree                 IN NUMBER DEFAULT  4,
                       p_def_cap_samp_pct               IN NUMBER DEFAULT  15,
                       p_def_non_cap_samp_pct   IN NUMBER DEFAULT  100,
                       p_def_audit_samp_pct     IN NUMBER DEFAULT  100 );
```

Method: GATHER_TABLE_STATS

Performs a subset of the gather_stats functionality but only gathers statistics on the table specified by the supplied parameter p_table.

All the other parameters behave like the corresponding parameters of the same name on gather_all_stats.

```
PROCEDURE Gather_Stats(  p_pn                   IN NUMBER DEFAULT 1,
                       p_indexes                        IN NUMBER DEFAULT  1,
                       p_num_partitions         IN NUMBER DEFAULT  -1,
                       p_process                        IN NUMBER DEFAULT  1,
                       p_Degree                 IN NUMBER DEFAULT  4,
                       p_def_cap_samp_pct               IN NUMBER DEFAULT  15,
                       p_def_non_cap_samp_pct   IN NUMBER DEFAULT  100,
                       p_def_audit_samp_pct     IN NUMBER DEFAULT  100 );
```

Method: DELETE_GTT_STATS

Removes statistics from all global temporary tables whose names start with the prefix 'TMP'.

```
PROCEDURE DELETE_GTT_STATS;
```

Method: SET_STATS_RLS

Sets user-defined statistics on selected columns in TMP_Wgn3AddRLS and Wgn3EventParticipant. This is to overcome some poor cardinality estimates that have been found in the past when joining these two tables.

```
PROCEDURE Set_Stats_RLS (p_StatType IN NUMBER DEFAULT NULL );
```

The argument p_StatType defines which tables to set the user-defined statistics on:

NULL Both tables (default)

1 Wgn3EventParticipant only

2 TMP_Wgn3AddRLS only

Method: LOCK_STATS

Locks or unlocks statistics on all temporary tables that start with the prefix 'TMP' and all permanent tables that start with the prefix 'WGN'.

The Boolean argument p_lock is False to unlock the statistics and True to lock them.

```
PROCEDURE LOCK_STATS(p_lock in Boolean);
```

Method: ENABLE_DEBUG

Turns on debugging using DBMS_OUTPUT. Argument p_level has these values:

0 None

1 Normal

2 Verbose

This method performs the same function as the p_debug argument to gather_all_stats.

```
PROCEDURE Enable_Debug(p_level in number default 1);
```

Progress Logging and Error Reporting

Every step that wgn_stats performs is logged to a table called Wgn3StatsState:

Wgn3StatsState table		
Column	**Datatype**	**Description**
Step	Number	A unique step number indicating the sequence of the step.
Created	Date	The date and time the entry was made.
Statement	Varchar	The actual statement submitted to the RDBMS.
Status	Varchar	The status of the step values STARTED or COMPLETED. If a step fails unexpectedly, the error message and code are written into this column.

Each step that wgn_stats performs has two entries written to this table, one before it starts and the other on completion (or failure). As the date and time is recorded this permits fine-grained tracking of how long each step takes, and where possible performance improvements may be made.

This table is transient in nature; the next time wgn_stats runs, it will remove the table entries from the previous run so the table only ever contains entries for the current (most recent) run.

Chapter 5: Package Customization

Important! This section contains advanced customization information. But for most installations, this will not be necessary.

As noted earlier, wgn_stats_cust is the mechanism that allows wgn_stats to be customized to meet specific customer requirements. To do this without requiring modifications to wgn_stats itself, the customizable interfaces are defined in the package specification for wgn_stats_cust, which is installed without a corresponding package body.

When statistics are gathered, wgn_stats attempts to execute any appropriate customized methods defined in wgn_stats_cust. If the customized method has been implemented (via the package body) then it gets executed; if it has not, wgn_stats executes the default implementation contained in wgn_stats.

The customizable interfaces are described on Type: WGN_STATS_PARAM_TAB section. First, it is important to know what can be customized. These customizations fall into three categories:

1. Simple customization, allowing the default sampling percentages to be customized on a per process basis.
2. Allow a user-defined procedure to be defined that can be called to do an arbitrary operation as the last step performed by wgn_stats.gather_all_stats.
3. Allow individual and specific customization of the method used to gather statistics at the granularity of a specific table or index, at either a global or partition level.

Of the three types, the first two are self-explanatory; the third requires more detail. To provide the ability to customize statistics gathering at the object level, we need a way to encapsulate this information so it can be created and returned to wgn_stats in a well-defined form. This is done through two user-defined types:

- WGN_STATS_PARAM
- WGN_STATS_PARAM_TAB

These user-defined types are described in the next section.

This section contains the following topics:

Types

This section describes the two user-defined types, WGN_STATS_PARAM and WGN_STATS_PARAM_TAB.

Type: WGN_STATS_PARAM

This is a user-defined type that defines the significant statistics attributes that may need to be customized on a per object basis. Defined as:

```
CREATE OR REPLACE TYPE WGN_STATS_PARAM AS OBJECT (
                        l_process               number,
                        l_object_name   varchar2(30),
                        l_partition_level       number(1),
                        l_estimate_percent      number,
                        l_block_sample  number(1),
                        l_method_opt    varchar2(2000),
                        l_degree        number)
```

Each user-defined type has the following attributes:

Attribute	Description
l_process	A user-defined number to allow different process to be defined for example it may be required to have different attribute values for daily versus weekly process for a specified object.
l_object_name	Specifies the name of the object (a table or index).
l_partition_level	Allows different attributes to be specified at a global or partition level for the specified object. 0 Global level 1 Partition level
l_estimate_percent	Specifies the percentage sampling rate. In addition, a value of zero or -1 has a special meaning: 0 Do not gather statistics on the specified object -1 Use the default sampling rate
l_block_sample	Specifies whether or not to use random block sampling: 0 Do not use block sampling 1 Use block sampling
l_method_opt	Specifies the method_opt parameter passed to dbms_stats for the specified object. This corresponds to the parameter in the GATHER_TABLE_STATS method. It controls whether and how histograms are constructed for the statistics.

l_degree	Allows the default DOP to be overridden for the specified object. A value of -1 means 'adopt the default'.

Returning objects of this type to wgn_stats is the mechanism that allows statistics to be customized down to the level of individual objects, at either a global or partition level. In addition, using a process identifier allows this to be varied across different processes (for example, daily or weekly).

Note: If specific values (other than -1) are set for l_estimate_percent or l_degree, then these values are used in preference to any global defaults that may exist.

Example

The pseudo code below defines an object of this type to represent a customized statistic for the 'WGN3USER' table for process 2, at a global level, sampled at 50%, using block sampling, with a method_opt to make Oracle automatically compute relevant histograms, and with a degree of parallel of 32:

```
BEGIN
   My_stats_rec WGN_STATS_PARAM := WGN_STATS_PARAM
(2,
'WGN3USER',
0,
50,
1,
'for all columns size auto',
32);
END;
```

Type: WGN_STATS_PARAM_TAB

A collection of wgn_stats_param types allowing a table of wgn_stats_param to be returned from an SQL cursor.

Methods

This section describes the interfaces contained in wgn_stats_cust. By definition, these are the elements that can be customized.

Method: GET_PARAMS – Returns stats Type for Specific Object

Returns wgn_stats_param for the requested object and process. If p_partition is supplied as 0, then this function returns a global wgn_stats_param, corresponding to global statistics for the specified object. Otherwise, it returns wgn_stats_param, corresponding to partition-level statistics for the specified object.

Note: If no customization exists for the supplied object /process/partition, it returns null.

```
FUNCTION GET_PARAMS(     p_object_name    IN VARCHAR2,
                         p_process                IN NUMBER DEFAULT 1,
                         p_partition              in number DEFAULT 0)
RETURN wgn_stats_param
```

Method: GET_PARAMS – Returns All Statistic Types

Returns a table (wgn_stats_param_tab) of all wgn_stats_param defined by the package. This allows all specific object level customizations to be retrieved from this single method call, and it makes them directly accessible as an SQL cursor.

```
FUNCTION GET_PARAMS
RETURN wgn_stats_param_tab
```

Method: GET_CAPTURED_DEFAULT_PCT

Returns the default sampling percentage to be used for the specified process for tables and indexes that are classified as captured data tables.

```
FUNCTION GET_CAPTURED_DEFAULT_PCT (p_process IN NUMBER DEFAULT 1)
RETURN NUMBER
```

Method: GET_NON_CAPTURED_DEFAULT_PCT

Returns the default sampling percentage to be used for the specified process for tables and indexes that are classified as non-captured data tables.

```
FUNCTION GET_NON_CAPTURED_DEFAULT_PCT (p_process IN NUMBER DEFAULT 1)
RETURN NUMBER
```

Method: GET_AUDIT_DEFAULT_PCT

Returns the default sampling percentage to be used for the specified process for tables and indexes that are classified as audit data tables.

```
FUNCTION GET_AUDIT_DEFAULT_PCT (p_process IN NUMBER DEFAULT 1)
RETURN NUMBER
```

Method: CUSTOM_CALL

Returns a user-defined procedure called as the final step in wgn_stats.gather_all_stats.

```
PROCEDURE CUSTOM_CALL(p_process IN NUMBER)
```