176
DBMS_SERVER_ALERT

The DBMS_SERVER_ALERT package enables you to configure the Oracle Database server to issue an alert when a threshold for a specified server metric has been violated. You can configure both warning and critical thresholds for a large number of predefined metrics.

If a warning threshold is reached, the server generates a severity level 5 alert. If a critical threshold is reached, the server generates a severity level 1 alert.

The chapter contains the following topics:

- Security Model
- Object Types
- Relational Operators
- Supported Metrics
- Summary of DBMS_SERVER_ALERT Subprograms

DBMS_SERVER_ALERT Security Model

The user needs DBA or IMP_FULL_DATABASE roles to use the DBMS_SERVER_ALERT package.

DBMS_SERVER_ALERT Object Types

You qualify the metric by an individual object for the listed object types.

Table 176-1 Object Types Defined as Constants

Constant	Description
OBJECT_TYPE_SYSTEM	Metrics collected on the system level for each instance.
OBJECT_TYPE_FILE	Metrics collected on the file level. These are used for AVG_FILE_READ_TIME and AVG_FILE_WRITE_TIME metrics.
OBJECT_TYPE_SERVICE	Metrics collected on the service level. Currently ELAPSED_TIME_PER_CALL and CPU_TIME_PER_CALL are collected.
OBJECT_TYPE_TABLESPACE	Metrics collected on the tablespace level. **Note**: Dictionary managed tablespaces are not supported.
OBJECT_TYPE_EVENT_CLASS	Metrics collected on wait event class level. Currently supported metrics are AVG_USERS_WAITING and DB_TIME_WAITING.
OBJECT_TYPE_SESSION	Metrics collected on the session level. Currently only BLOCKED_USERS is collected. The threshold can only be set at the instance level, which means that no object name should be specified when setting the threshold for this type of metric.
OBJECT_TYPE_WRCLIENT	Refers to a group of metrics (WCR_...) used during replay to monitor the replay clients' performance

DBMS_SERVER_ALERT Relational Operators

You can specify a relational comparison operator to determine whether or not a given metric's value violates the threshold setting. The server supports the following operators.

Table 176-2 Relational Operators Defined as Constants

Constant	Description
`OPERATOR_CONTAINS`	A metric value matching an entry in a list of threshold values is considered a violation.
`OPERATOR_DO_NOT_CHECK`	The metric value is not compared to the threshold value, and no alerts are generated. Use this operator to disable alerts for a metric.
`OPERATOR_EQ`	A metric value equal to the threshold value is considered a violation.
`OPERATOR_GE`	A metric value greater than or equal to the threshold value is considered a violation.
`OPERATOR_GT`	A metric value greater than the threshold value is considered a violation.
`OPERATOR_LE`	A metric value less than or equal to the threshold value is considered a violation.
`OPERATOR_LT`	A metric value less than the threshold value is considered a violation.
`OPERATOR_NE`	A metric value not equal to the threshold value is considered a violation.

DBMS_SERVER_ALERT Supported Metrics

These metrics are supported. All internal metric names are supplied as package constants.

Table 176-3 List of Supported Metrics

Metric Name (Internal)	Metric Name (External)	Units
`AVG_FILE_READ_TIME`	Average File Read Time	Microseconds
`AVG_FILE_WRITE_TIME`	Average File Write Time	Microseconds
`AVG_USERS_WAITING`	Average Number of Users Waiting on a Class of Wait Events	Count of sessions
`BLOCKED_USERS`	Number of Users blocked by some Session	Number of Users
`BRANCH_NODE_SPLITS_SEC`	Branch Node Splits (for each second)	Splits for each Second
`BRANCH_NODE_SPLITS_TXN`	Branch Node Splits (for each transaction)	Splits for each Transaction
`BUFFER_CACHE_HIT`	Buffer Cache Hit (%)	% of cache accesses
`CONSISTENT_CHANGES_SEC`	Consistent Changes (for each second)	Changes for each Second
`CONSISTENT_CHANGES_TXN`	Consistent Changes (for each transaction)	Changes for each Transaction
`CONSISTENT_GETS_SEC`	Consistent Gets (for each second)	Gets for each Second

Table 176-3 (Cont.) List of Supported Metrics

Metric Name (Internal)	Metric Name (External)	Units
CONSISTENT_GETS_TXN	Consistent Gets (for each transaction)	Gets for each Transaction
CR_BLOCKS_CREATED_SEC	CR Blocks Created (for each second)	Blocks for each Second
CR_BLOCKS_CREATED_TXN	CR Blocks Created (for each transaction)	Blocks for each Transaction
CR_RECORDS_APPLIED_SEC	CR Undo Records Applied (for each second)	Records for each Second
CR_RECORDS_APPLIED_TXN	CR Undo Records Applied (for each transaction)	Records for each Transaction
CURSOR_CACHE_HIT	Cursor Cache Hit (%)	% of soft parses
DATABASE_WAIT_TIME	Database Wait Time (%)	% of all database time
DATABASE_CPU_TIME	Database CPU Time (%)	% of all database time
DB_BLKGETS_SEC	DB Block Gets (for each second)	Gets for each Second
DB_BLKGETS_TXN	DB Block Gets (for each transaction)	Gets for each Transaction
DB_TIME_WAITING	Percent of Database Time Spent Waiting on a Class of Wait Events	% of Database Time
DBWR_CKPT_SEC	DBWR Checkpoints (for each second)	Checkpoints for each Second
DISK_SORT_SEC	Sorts to Disk (for each second)	Sorts for each Second
DISK_SORT_TXN	Sorts to Disk (for each transaction)	Sorts for each Transaction
ELAPSED_TIME_PER_CALL	Elapsed time for each user call for each service	Microseconds for each call
ENQUEUE_DEADLOCKS_SEC	Enqueue Deadlocks (for each second)	Deadlocks for each Second
ENQUEUE_DEADLOCKS_TXN	Enqueue Deadlocks (for each transaction)	Deadlocks for each Transaction
ENQUEUE_REQUESTS_SEC	Enqueue Requests (for each second)	Requests for each Second
ENQUEUE_REQUESTS_TXN	Enqueue Requests (for each transaction)	Requests for each Transaction
ENQUEUE_TIMEOUTS_SEC	Enqueue Timeouts (for each second)	Timeouts for each Second
ENQUEUE_TIMEOUTS_TXN	Enqueue Timeouts (for each transaction)	Timeouts for each Transaction
ENQUEUE_WAITS_SEC	Enqueue Waits (for each second)	Waits for each Second
ENQUEUE_WAITS_TXN	Enqueue Waits (for each transaction)	Waits for each Transaction
EXECUTE_WITHOUT_PARSE	Executes Performed Without Parsing	% of all executes
FULL_INDEX_SCANS_SEC	Fast Full Index Scans (for each second)	Scans for each Second
FULL_INDEX_SCANS_TXN	Fast Full Index Scans (for each transaction)	Scans for each Transaction
GC_AVG_CR_GET_TIME	Global Cache CR Request	Milliseconds
GC_AVG_CUR_GET_TIME	Global Cache Current Request	Milliseconds
GC_BLOCKS_CORRUPT	Global Cache Blocks Corrupt	Blocks
GC_BLOCKS_LOST	Global Cache Blocks Lost	Blocks
HARD_PARSES_SEC	Hard Parses (for each second)	Parses for each Second

Table 176-3 (Cont.) List of Supported Metrics

Metric Name (Internal)	Metric Name (External)	Units
HARD_PARSES_TXN	Hard Parses (for each transaction)	Parses for each Transaction
LEAF_NODE_SPLITS_SEC	Leaf Node Splits (for each second)	Splits for each Second
LEAF_NODE_SPLITS_TXN	Leaf Node Splits (for each transaction)	Splits for each Transaction
LIBRARY_CACHE_HIT	Library Cache Hit (%)	% of cache accesses
LIBRARY_CACHE_MISS	Library Cache Miss (%)	% of cache accesses
LOGONS_CURRENT	Current Number of Logons	Number of Logons
LOGONS_SEC	Cumulative Logons (for each second)	Logons for each Second
LOGONS_TXN	Cumulative Logons (for each transaction)	Logons for each Transaction
LONG_TABLE_SCANS_SEC	Scans on Long Tables (for each second)	Scans for each Second
LONG_TABLE_SCANS_TXN	Scans on Long Tables (for each transaction)	Scans for each Transaction
OPEN_CURSORS_SEC	Cumulative Open Cursors (for each second)	Cursors for each Second
MEMORY_SORTS_PCT	Sorts in Memory (%)	% of sorts
NETWORK_BYTES_SEC	Network Bytes, for each second	Bytes for each Second
OPEN_CURSORS_CURRENT	Current Number of Cursors	Number of Cursors
OPEN_CURSORS_TXN	Cumulative Open Cursors (for each transaction)	Cursors for each Transaction
OS_SCHED_CPU_WAIT_TIME	Operating System Scheduler CPU Wait (by time)	Microseconds
PARSE_FAILURES_SEC	Parse Failures (for each second)	Parses for each Second
PARSE_FAILURES_TXN	Parse Failures (for each transaction)	Parses for each Transaction
PGA_CACHE_HIT	PGA Cache Hit (%)	% bytes processed in PGA
PHYS_DESGN_WAIT_SCT	Physical Design Wait (by session count)	Count of sessions
PHYSICAL_READS_SEC	Physical Reads (for each second)	Reads for each Second
PHYSICAL_READS_TXN	Physical Reads (for each transaction)	Reads for each Transaction
PHYSICAL_WRITES_SEC	Physical Writes (for each second)	Writes for each Second
PHYSICAL_WRITES_TXN	Physical Writes (for each transaction)	Writes for each Transaction
PHYSICAL_READS_DIR_SEC	Direct Physical Reads (for each second)	Reads for each Second
PHYSICAL_READS_DIR_TXN	Direct Physical Reads (for each transaction)	Reads for each Transaction
PHYSICAL_WRITES_DIR_SEC	Direct Physical Writes (for each second)	Writes for each Second
PHYSICAL_WRITES_DIR_TXN	Direct Physical Writes (for each transaction)	Writes for each Transaction
PHYSICAL_READS_LOB_SEC	Direct LOB Physical Reads (for each second)	Reads for each Second
PHYSICAL_READS_LOB_TXN	Direct LOB Physical Reads (for each transaction)	Reads for each Transaction
PHYSICAL_WRITES_LOB_SEC	Direct LOB Physical Writes (for each second)	Writes for each Second

Table 176-3 (Cont.) List of Supported Metrics

Metric Name (Internal)	Metric Name (External)	Units
PHYSICAL_WRITES_LOB_TXN	Direct LOB Physical Writes (for each transaction)	Writes for each Transaction
PROCESS_LIMIT_PCT	Process Limit Usage (%)	% of maximum value
PX_DOWNGRADED_SEC	Downgraded Parallel Operations (for each second)	Operations for each Second
PX_DOWNGRADED_25_SEC	Downgraded to 25% and more (for each second)	Operations for each Second
PX_DOWNGRADED_50_SEC	Downgraded to 50% and more (for each second)	Operations for each Second
PX_DOWNGRADED_75_SEC	Downgraded to 75% and more (for each second)	Operations for each Second
PX_DOWNGRADED_SER_SEC	Downgraded to serial (for each second)	Operations for each Second
RB_RECORDS_APPLIED_SEC	Rollback Undo Records Applied (for each second)	Records for each Second
RB_RECORDS_APPLIED_TXN	Rollback Undo Records Applied (for each transaction)	Records for each Transaction
REDO_ALLOCATION_HIT	Redo Log Allocation Hit	% of redo allocations
REDO_GENERATED_SEC	Redo Generated (for each second)	Redo Bytes for each Second
REDO_GENERATED_TXN	Redo Generated (for each transaction)	Redo Bytes for each Transaction
REDO_WRITES_SEC	Redo Writes (for each second)	Writes for each Second
REDO_WRITES_TXN	Redo Writes (for each transaction)	Writes for each Transaction
RECURSIVE_CALLS_SEC	Recursive Calls (for each second)	Calls for each Second
RECURSIVE_CALLS_TXN	Recursive Calls (for each transaction)	Calls for each Transaction
RESPONSE_TXN	Response (for each transaction)	Seconds for each Transaction
ROWS_PER_SORT	Rows Processed for each Sort	Rows for each Sort
SESS_LOGICAL_READS_SEC	Session Logical Reads (for each second)	Reads for each Second
SESS_LOGICAL_READS_TXN	Session Logical Reads (for each transaction)	Reads for each Transaction
SESSION_CPU_SEC	Database CPU (for each second)	Microseconds for each Second
SESSION_CPU_TXN	Database CPU (for each transaction)	Microseconds for each Transaction
SESSION_LIMIT_PCT	Session Limit Usage (%)	% of maximum value
SHARED_POOL_FREE_PCT	Shared Pool Free(%)	% of shared pool
SOFT_PARSE_PCT	Soft Parse (%)	% of all parses
SQL_SRV_RESPONSE_TIME	Service Response (for each execution)	Seconds
TABLESPACE_PCT_FULL	Tablespace space usage	% full
TABLESPACE_BYT_FREE	Tablespace bytes space usage	Kilobytes free
TOTAL_TABLE_SCANS_SEC	Total Table Scans (for each second)	Scans for each Second
TOTAL_TABLE_SCANS_TXN	Total Table Scans (for each transaction)	Scans for each Transaction
TOTAL_INDEX_SCANS_SEC	Total Index Scans (for each second)	Scans for each Second

Table 176-3 (Cont.) List of Supported Metrics

Metric Name (Internal)	Metric Name (External)	Units
TOTAL_INDEX_SCANS_TXN	Total Index Scans (for each transaction)	Scans for each Transaction
TOTAL_PARSES_SEC	Total Parses (for each second)	Parses for each Second
TOTAL_PARSES_TXN	Total Parses (for each transaction)	Parses for each Transaction
USER_COMMITS_SEC	User Commits (for each second)	Commits for each Second
USER_COMMITS_TXN	User Commits (for each transaction)	Commits for each Transaction
USER_ROLLBACKS_SEC	User Rollbacks (for each second)	Rollbacks for each Second
USER_ROLLBACKS_TXN	User Rollbacks (for each transaction)	Rollbacks for each Transaction
USER_CALLS_SEC	User Calls (for each second)	Calls for each Second
USER_CALLS_TXN	User Calls (for each transaction)	Calls for each Transaction
USER_CALLS_PCT	User Calls (%)	% of all calls
USER_LIMIT_PCT	User Limit Usage (%)	% of maximum value
WCR_AVG_IO_LAT	Average IO response time (for a WRC client)	Milliseconds
WCR_PCPU	Percentage of replay threads on CPU (for a WRC client)	% of total replay threads
WCR_PIO	Percentage of replay threads doing IOs (for a WRC client)	% of total replay threads

Summary of DBMS_SERVER_ALERT Subprograms

This table lists the DBMS_SERVER_ALERT subprograms and briefly describes them.

Table 176-4 DBMS_SERVER_ALERT Package Subprograms

Subprogram	Description
EXPAND_MESSAGE Function	Expands alert messages
GET_THRESHOLD Procedure	Gets the current threshold settings for a specified metric
SET_THRESHOLD Procedure	Sets the warning and critical thresholds for a specified metric

EXPAND_MESSAGE Function

This function expands alert messages.

Syntax

```
DBMS_SERVER_ALERT.EXPAND_MESSAGE(
   user_language             IN   VARCHAR2,
   message_id                IN   NUMBER,
   argument_1                IN   VARCHAR2,
   argument_2                IN   VARCHAR2,
   argument_3                IN   VARCHAR2,
   argument_4                IN   VARCHAR2,
   argument_5                IN   VARCHAR2)
  RETURN VARCHAR2;
```

Parameters

Table 176-5 *EXPAND_MESSAGE Function Parameters*

Parameter	Description
`user_language`	The language of the current session.
`message_id`	Id of the alert message
`argument_1`	The first argument in the alert message.
`argument_2`	The second argument in the alert message.
`argument_3`	The third argument in the alert message.
`argument_4`	The fourth argument in the alert message.
`argument_5`	The fifth argument in the alert message.

GET_THRESHOLD Procedure

This procedure gets the current threshold settings for the specified metric.

Syntax

```
DBMS_SERVER_ALERT.GET_THRESHOLD(
   metrics_id                 IN   BINARY_INTEGER,
   warning_operator           OUT  BINARY_INTEGER,
   warning_value              OUT  VARCHAR2,
   critical_operator          OUT  BINARY_INTEGER,
   critical_value             OUT  VARCHAR2,
   observation_period         OUT  BINARY_INTEGER,
   consecutive_occurrences    OUT  BINARY_INTEGER,
   instance_name              IN   VARCHAR2,
   object_type                IN   BINARY_INTEGER,
   object_name                IN   VARCHAR2);
```

Parameters

Table 176-6 GET_THRESHOLD Procedure Parameters

Parameter	Description
`metrics_id`	The internal name of the metric. See "Supported Metrics".
`warning_operator`	The operator for the compa3ring the actual value with the warning threshold.
`warning_value`	The warning threshold value.
`critical_operator`	The operator for the comparing the actual value with the critical threshold.
`critical_value`	The critical threshold value.
`observation_period`	The period at which the metric values are computed and verified against the threshold setting.
`consecutive_occurrences`	The number of observation periods the metric value should violate the threshold value before the alert is issued.

Table 176-6 (Cont.) GET_THRESHOLD Procedure Parameters

Parameter	Description
`instance_name`	The name of the instance for which the threshold is set. This is `NULL` for database-wide alerts. In cases in which this parameter is not `NULL`, this should be set to one of the `INSTANCE_NAME` values found in the `GV$INSTANCE` View.
`object_type`	Either `OBJECT_TYPE_SYSTEM` or `OBJECT_TYPE_SERVICE`.
`object_name`	The name of the object.

Usage Notes

Note that this subprogram does not check if the value of the `instance_name` parameter is meaningful or valid.

SET_THRESHOLD Procedure

This procedure sets the warning and critical thresholds for a specified metric.

Syntax

```
DBMS_SERVER_ALERT.SET_THRESHOLD(
   metrics_id                IN   BINARY_INTEGER,
   warning_operator          IN   BINARY_INTEGER,
   warning_value             IN   VARCHAR2,
   critical_operator         IN   BINARY_INTEGER,
   critical_value            IN   VARCHAR2,
   observation_period        IN   BINARY_INTEGER,
   consecutive_occurrences   IN   BINARY_INTEGER,
   instance_name             IN   VARCHAR2,
   object_type               IN   BINARY_INTEGER,
   object_name               IN   VARCHAR2);
```

Parameters

Table 176-7 SET_THRESHOLD Procedure Parameters

Parameter	Description
`metrics_id`	The internal name of the metric. See "Supported Metrics".
`warning_operator`	The operator for the comparing the actual value with the warning threshold (such as `OPERATOR_GE`). See "Relational Operators".
`warning_value`	The warning threshold value. This is `NULL` if no warning threshold is set. A list of values may be specified for `OPERATOR_CONTAINS`.
`critical_operator`	The operator for the comparing the actual value with the critical threshold. See "Relational Operators".
`critical_value`	The critical threshold value. This is `NULL` if not set. A list of values may be specified for `OPERATOR_CONTAINS`.
`observation_period`	The period at which the metric values are computed and verified against the threshold setting. The valid range is 1 to 60 minutes.
`consecutive_occurrences`	The number of observation periods the metric value should violate the threshold value before the alert is issued.

Table 176-7 (Cont.) SET_THRESHOLD Procedure Parameters

Parameter	Description
`instance_name`	The name of the instance for which the threshold is set. This is `NULL` for database-wide alerts.
`object_type`	See "Object Types".
`object_name`	The name of the object. This is `NULL` for `SYSTEM`.

Usage Notes

Note that this subprogram does not check if the value of the `instance_name` parameter is meaningful or valid. Passing a name that does not identify a valid instance will result in a threshold that is not used by any by any instance although the threshold setting will be visible in the `DBA_THRESHOLDS` view. The exception is the lower-case string 'database_wide' which is semantically equivalent to passing `NULL` for the instance name, the latter being the preferred usage.