41

JDBC DMS Metrics

DMS metrics are used to measure the performance of application components.

This chapter discusses the following topics:

- Overview of JDBC DMS Metrics
- About Determining the Type of Metric to Be Generated
- About Generating the SQLText Metric
- About Accessing DMS Metrics Using JMX

> **Note:**
>
> There is another kind of metrics called end-to-end metrics. End-to-end metrics are used for tagging application activity from the entry into the application code through JDBC to the database and back.
>
> JDBC supports the following end-to-end metrics:
>
> - `Action`
> - `ClientId`
> - `ExecutionContextId`
> - `Module`
> - `State`
>
> For earlier releases, to work with the preceding metrics, you could use the `setEndToEndMetrics` and `getEndToEndMetrics` methods of the `oracle.jdbc.OracleConnection` interface. However, starting from Oracle Database 12*c* Release 1 (12.1), these methods have been deprecated. Oracle recommends to use the `setClientInfo` and `getClientInfo` methods instead of the `setEndToEndMetrics` and `getEndToEndMetrics` methods.
>
> In Oracle Database 10*g*, Oracle Java Database Connectivity (JDBC) supports end-to-end metrics. In Oracle Database 12*c* Release 1 (12.1), an application can set the end-to-end metrics directly only when it does not use a DMS-enabled JAR files. But, if the application uses a DMS-enabled JAR file, the end-to-end metrics can be set only through DMS.

> **WARNING:**
>
> Oracle strongly recommends using DMS metrics, if the application uses a DMS-enabled JAR file.

> **See Also:**
>
> *Oracle Database JDBC Java API Reference* for more information about end-to-end metrics

41.1 Overview of JDBC DMS Metrics

DMS metrics enable application and system developers to measure and export customized performance metrics for specific software components.

Starting with Oracle Database 23ai Release, Oracle does not provide the `ojdbc<n>dms.jar` and `ojdbc<n>dms_g.jar` files, for example, the `ojdbc8dms.jar`, `ojdbc8dms_g.jar`, `ojdbc11dms.jar`, `ojdbcdms_g.jar` files. This release includes only the `ojdbc8.jar`, `ojdbc11.jar`, and `ojdbc17.jar` files, which acts as the `ojdbc<n>dms.jar` file, when the `dms.jar` file is present in the `CLASSPATH` environment variable.

The metrics generated in Oracle Database Release 23ai are different from the metrics generated for the earlier versions of Oracle JDBC as it makes no attempt to retain compatibility with earlier versions. There are also no compatibility modes. A system that is dependent on the exact details of the DMS metrics generated by earlier versions of JDBC may have unexpected behavior, when processing the metrics generated by Oracle JDBC 23ai. This is by design and cannot be changed.

Statement metrics can be reported in a consolidated manner for all statements in a connection or individually for each statement. All DMS metrics, except those related to individual statements, are enabled at all times.

> **Note:**
>
> You can enable or disable the `SQLText` statement metric. It is disabled by default. If enabled, it is enabled for all the statements. See About Generating the SQLText Metric for more information.

41.2 About Determining the Type of Metric to Be Generated

To determine whether to use consolidated or individual metrics, JDBC checks the DMSConsole sensor weight. If the sensor weight is less than or equal to `DMSConsole.NORMAL`, then JDBC generates consolidated statement metrics. If the sensor weight is greater than `DMSConsole.NORMAL`, then JDBC generates individual statement metrics.

JDBC checks the DMSConsole sensor weight when creating a Prepared or Callable statement and depending on the sensor weight at the time the statement is created, the metrics are generated. Changing the value of the sensor weight, after the statement has been created, does not cause a statement to switch between consolidated and individual metrics.

> **Note:**
>
> In the presence of Statement caching, it may appear that changing sensor weight has no impact as statements are retrieved from the cache rather than created anew.

There is only one list of statement metrics that is generated for both consolidated and individual statement metrics. The only difference between these two lists is the aggregation of the statements. When individual statement metrics are generated, one set of metrics is generated for each distinct statement object created by the JDBC driver. On the other hand, when consolidated statement metrics are generated, all statements created by a given connection use the same set of statement metrics.

For example, consider an '`execute`' phase event. If individual statement metrics are used, then each statement created will have a distinct '`execute`' phase event. So, from two such statements, it will be possible to distinguish the execution statistics for the two separate statements. If one has an execution time of 1 second and the other an execution time of 3 seconds, then there will be two distinct 'execute' phase events, one with a total time and average of 1 second and the other with a total time and average of 3 seconds. But, if consolidated statement metrics are used, all statements will use the single '`execute`' phase event common to the connection. So, from two such statements created by the same connection, it will not be possible to distinguish the execution statistics for the two statements. If one has an execution time of 1 second and the other an execution time of 3 seconds, then the common '`execute`' phase event will report a total execution time of 4 seconds and an average of 2 seconds.

41.3 About Generating the SQLText Metric

Depending on the version of DMS, there are two mechanisms for determining the generating of the `SQLText` statement metrics:

- If the 12*c* version of the DMS JAR file is present in the `CLASSPATH` environment variable, then JDBC checks the DMS update SQL text flag. If this flag is `true`, then the `SQLText` metric is updated.
- If the 12*c* version of the DMS JAR file is not present in the `CLASSPATH` environment variable, then JDBC uses the value of the `DMSStatementMetrics` connection property. If this statement property is `true`, then the `SQLText` metric is updated. The default value of this connection property is `false`.

The generation of the `SQLText` metric is independent of the type of statement metrics used, that is, it does not matter whether you are using individual statement metrics or consolidated statement metrics.

41.4 About Accessing DMS Metrics Using JMX

JMX (Java Management Extensions) is a Java technology that supplies tools for managing and monitoring applications, system objects, devices, service-oriented networks, and the JVM (Java Virtual Machine). You can easily access DMS metrics at run time using a management application that supports JMX. For more information about using JMX to access DMS data, go to the following URL `http://www.oracle.com/technetwork/middleware/toplink/overview/index.html`

See Also:

Oracle Database Java Developer's Guide for more information about JMX