Creating a Monitored Query

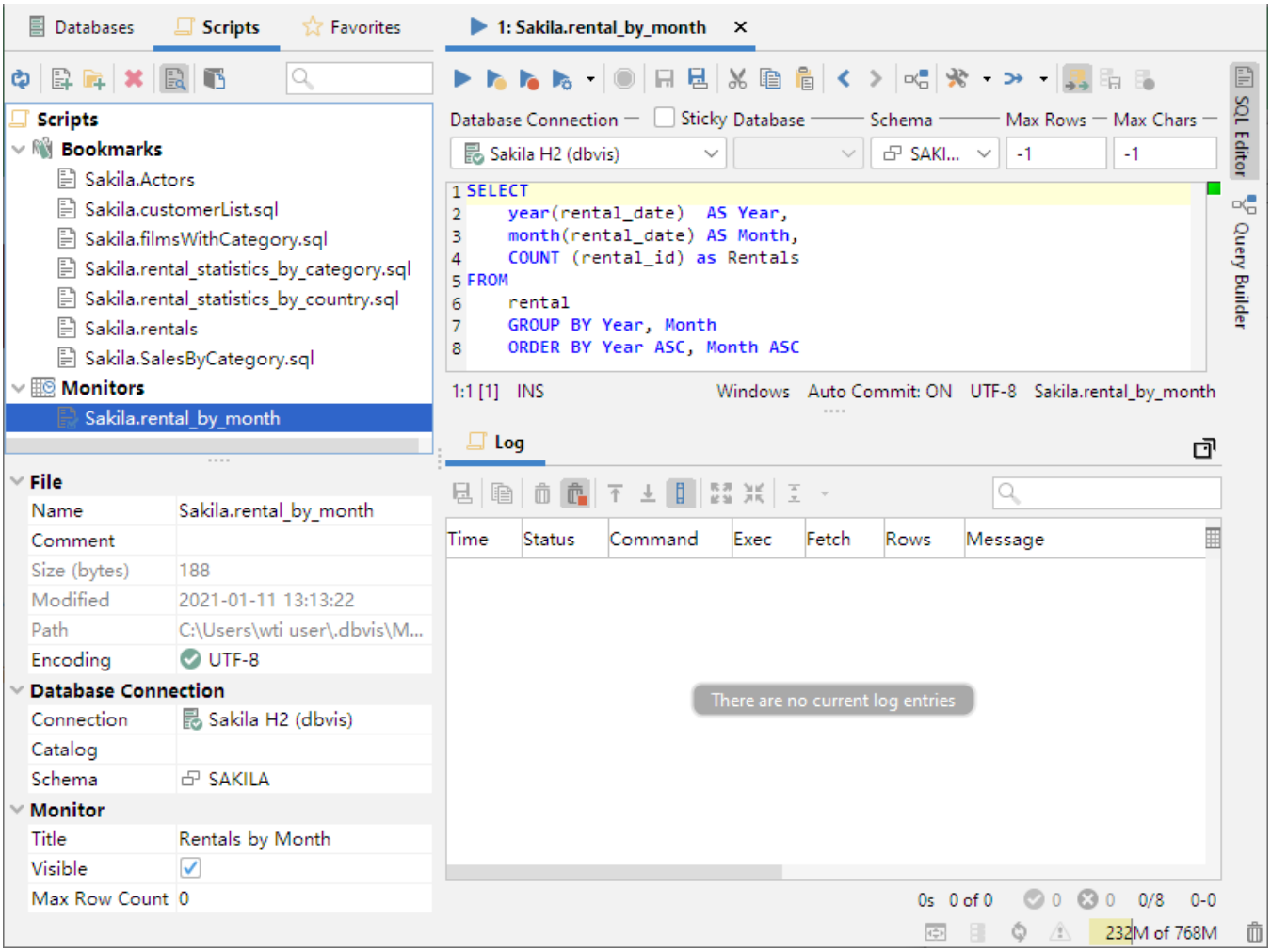
Monitored SQL statements are managed under the **Monitors** node in the **Scripts** tab in the tree area to the left in the main DbVisualizer window.

- Monitor table row count
- Monitor table row count difference

A monitor is basically a regular SQL script with some additional information and controls. You create it like any other script but place it in the **Monitors** folder.

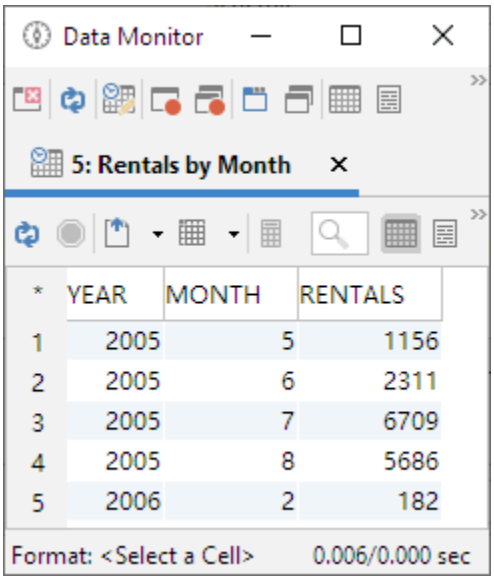
Note that the monitor is not supposed to run scripts with multiple statements, and especially not client-side commands (@). Only statements returning a single result set (i.e. select statements) are supported.

A monitored SQL script is associated with information about the target database connection and (optionally) the catalog (the JDBC term which translates to a database for some databases, like Sybase, MySQL, SQL Server, etc) and schema. It also has a title, a maximum row count (how many results to keep track of) and a visibility status (whether the monitored statement result should be included in the Monitors windows, discussed below). This information is displayed, and can be edited, in the lower part of the **Scripts** tab, along with information about the file that holds the monitored statement. If you don't want to see these details, you can disable it with the **Show Details** toggle control in the right-click menu for a node.



The figure above shows the Incidents/Day monitored statement and the SQL that is associated with it.

The following is an example of the result set produced by the statement:



The interesting columns in the result are the **Month** and **Count**. The **Year** and **MonthNum** are there just to get the correct ascending order of the result.

You can create and work with monitored statements in the same way as with a Bookmark. The main difference is how they are used and a couple of additional ways monitored statements can be created. For information about how to manually create, manage and share monitored statements, please see the Managing Frequently Used SQL page. The following sections describe how you can get help creating the bookmarks for a couple of cases that are commonly used for monitoring.

Monitor table row count

It is very common to want to keep track of how the number of rows in a table varies over time. The right-click menu in the grid for a table or result set therefore has a **Create Row Count Data Monitor** operation that creates a monitored statement for you automatically.

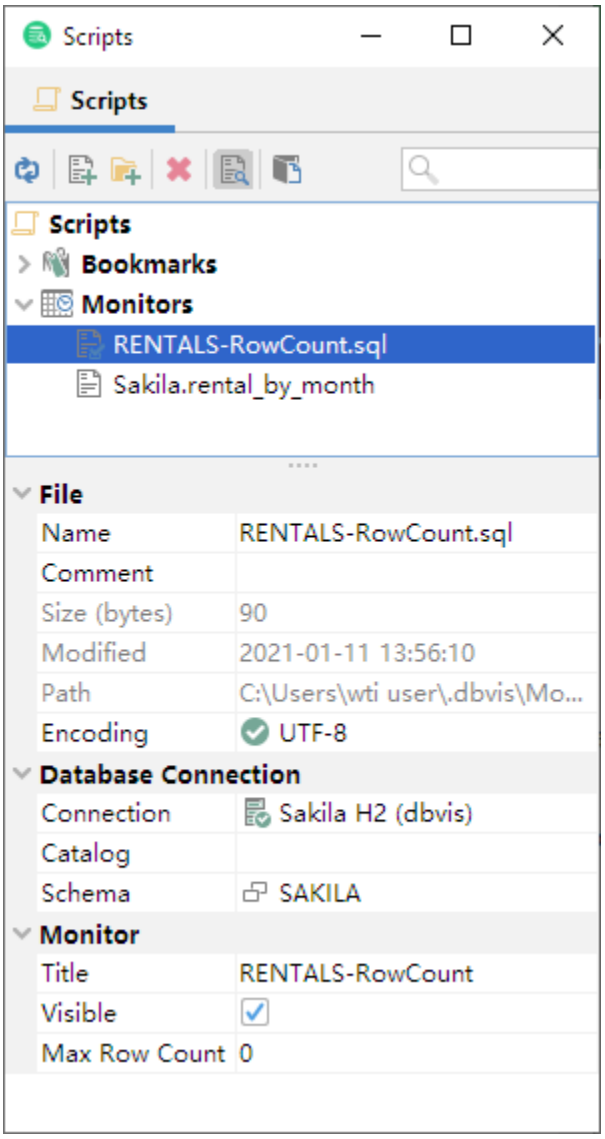
It creates a monitor with SQL for returning a single row with the timestamp for when the monitor was executed and the total number of rows in the table at that time. Every time the monitor is executed, a new row is added to the grid, up to a specified maximum number of rows. When the maximum row limit is reached, the oldest row is removed when a new row is added. Example:

PollTime	NumRows
2016-01-23 12:19:10	43123
2016-01-23 12:11:40	43139
2016-01-23 12:21:10	43143
2016-01-23 12:22:40	43184
…	…

The SQL for this monitor uses two variables, **dbvis-date** and **dbvis-time**. These variables are substituted with the current date and time, formatted according to the corresponding Tool Properties settings. The reason for using these variables instead of using SQL functions to retrieve the values is simply that it is almost impossible to get the values in a database-independent way. Another reason is that we want to see the client machine time rather than the database server time. You can, of course, modify the SQL any way you see fit, as long as the **PollTime** and **NumRows** labels are not changed.

```
SELECT '${dbvis-date}$ ${dbvis-time}$' AS PollTime,
       COUNT(*) AS NumRows
FROM RENTAL
```

DbVisualizer keeps the result for previous executions from the Monitor, up to the specified maximum number of rows, so that you can see how the result changes over time. You define the maximum number of rows in the **Max Row Count** field in the details area at the bottom of the Scripts tab. This property is initially set to 100 when you use **Create Row Count Data Monitor** to create the monitor.



You can change the value to limit or extend the number of rows that DbVisualizer should keep. Setting it to 0 or a negative number tells DbVisualizer to always clear the grid between executions of monitors.

Monitor table row count difference

In addition to tracking the number of rows in a table over time, you may want to see by how many rows the value changes. You can create a monitor for this purpose with the **Create Row Count Diff Data Monitor** operation, available in the right-click menu for the grid.

In addition to the **Row Count Monitor**, the **Row Count Diff Monitor** reports the difference between the number of rows in the last two executions:

PollTime	NumRows	NumRowsChange
2016-01-23 12:19:10	43123	0
2016-01-23 12:11:40	43139	16
2016-01-23 12:21:10	43143	4
2016-01-23 12:22:40	43184	41
…	…	…

The SQL for this monitor adds a third column, named **NumRowsChange**. It utilizes the fact that DbVisualizer automatically creates variables for the columns in a monitor result set, holding the values from the previous execution. The **NumRowsChange** column is set to the value returned by the count(*) aggregate function for the current execution minus the value from the previous execution, held by the **NumRows** variable. All columns in a monitor result set can be used like this to reference values from the previous execution of the monitor.

```
SELECT '${dbvis-date}$ ${dbvis-time}$' AS PollTime,
       COUNT(*) AS NumRows,
       COUNT(*) - ${NumRows||count(*)}$ AS NumRowsChange
from RENTAL
```