



# Silk Performance Explorer

## 21.0

Help

**Micro Focus**  
**The Lawn**  
**22-30 Old Bath Road**  
**Newbury, Berkshire RG14 1QN**  
**UK**  
<http://www.microfocus.com>

© Copyright 1992-2020 Micro Focus or one of its affiliates.

MICRO FOCUS, the Micro Focus logo and Silk Performer are trademarks or registered trademarks of Micro Focus or one of its affiliates.

All other marks are the property of their respective owners.

2020-10-21

# Contents

<b>Performance Explorer 21.0 .....</b>	<b>4</b>
Getting Started .....	4
Performance Explorer Overview .....	4
Terminology .....	4
Time Series Data Files .....	5
Starting Performance Explorer .....	7
Tour of the UI .....	8
Results Analysis .....	9
Adding Results .....	9
Visualizing Results .....	10
Analyzing Results .....	11
Processing Results .....	13
Viewing Results .....	16
Reporting .....	18
Word Overview Reports .....	18
HTML Overview Reports .....	18
Comparison Reports .....	20
Real-Time Monitoring .....	22
Monitoring Technologies .....	22
Monitoring Data Sources .....	46
Monitoring Load Tests .....	69
Monitor Charts and Monitor Writers .....	70
Host Data .....	71
Converting Data from CSV to TSD .....	72
Built-In Measures .....	73
Summary Measures .....	73
Transaction Measures .....	77
Web Form Measures .....	78
Web Page Measures .....	79
Browser-Driven Measures .....	79
CORBA-Specific Measures .....	80
SQL Database-Specific Measures .....	80
Middleware-Specific Measures .....	81
Citrix-Specific Measures .....	81
Custom Timer Measures .....	82
Custom Counter Measures .....	82
Streaming Measures .....	82
Command Line Parameters .....	83
Available Commands .....	83
Command File .....	85
Customizing Visualization .....	86
Default Options .....	86
Customizing Chart Display .....	88
Customizing Graph Display .....	89
Displaying Measure Bounds .....	90
Displaying Performance Level Conditions .....	91

# Performance Explorer 21.0

Welcome to Performance Explorer 21.0.

Performance Explorer allows you to view measures obtained through real-time monitoring and to analyze results of completed load tests. A variety of tools and features enables exhaustive analysis, reporting, and processing of all captured data.

## Getting Started

### Performance Explorer Overview

Performance Explorer offers two main capabilities:

- Results Analysis
- Real-Time Monitoring

#### Results Analysis

With the Results Analysis functionality of Performance Explorer, you can analyze the results of a completed load test by creating charts, tables, and reports. With charts, you can visualize the data that was collected during the load test, and reports as well as tables help you to summarize important data and findings. The basis for all charts, tables, and reports is the data in the .tsd files. During each load test, Silk Performer captures a big amount of data and stores it in several time-series data (.tsd) files. When a load test is completed, you can load the data (the .tsd files) in Performance Explorer and visualize and edit them according to your requirements. All charts are fully customizable, and they can contain as many measures as required. You can open multiple charts using measures from one or multiple tests to show similarities and differences. Performance Explorer provides a variety of templates for charts, tables, and reports. Furthermore, you can place information on client and server performance in a single chart, so that you can directly view the effect of server performance on client behavior. By saving your changes as a template, you can reuse your individual settings.

#### Real-Time Monitoring

With the Real-Time Monitoring functionality of Performance Explorer, you can monitor a broad range of systems by creating and configuring charts that show the system performance in real-time. It is possible to open multiple charts at the same time, which allows you to watch the performance of two or more systems simultaneously, for example the web server performance and the operating system performance. Adding measures to a chart is easy and intuitive in Performance Explorer: You can drag a single measure or a set of measures from the tree onto the chart.

## Terminology

The following terms are used in Performance Explorer and throughout the Performance Explorer Help. To avoid misconceptions and to keep explanations clear and straightforward, these terms are declared here.

Term	Description
Chart	A chart is the drawing area, where measures are visually represented as graphs. A chart can also be described as the sum of all displayed graphs

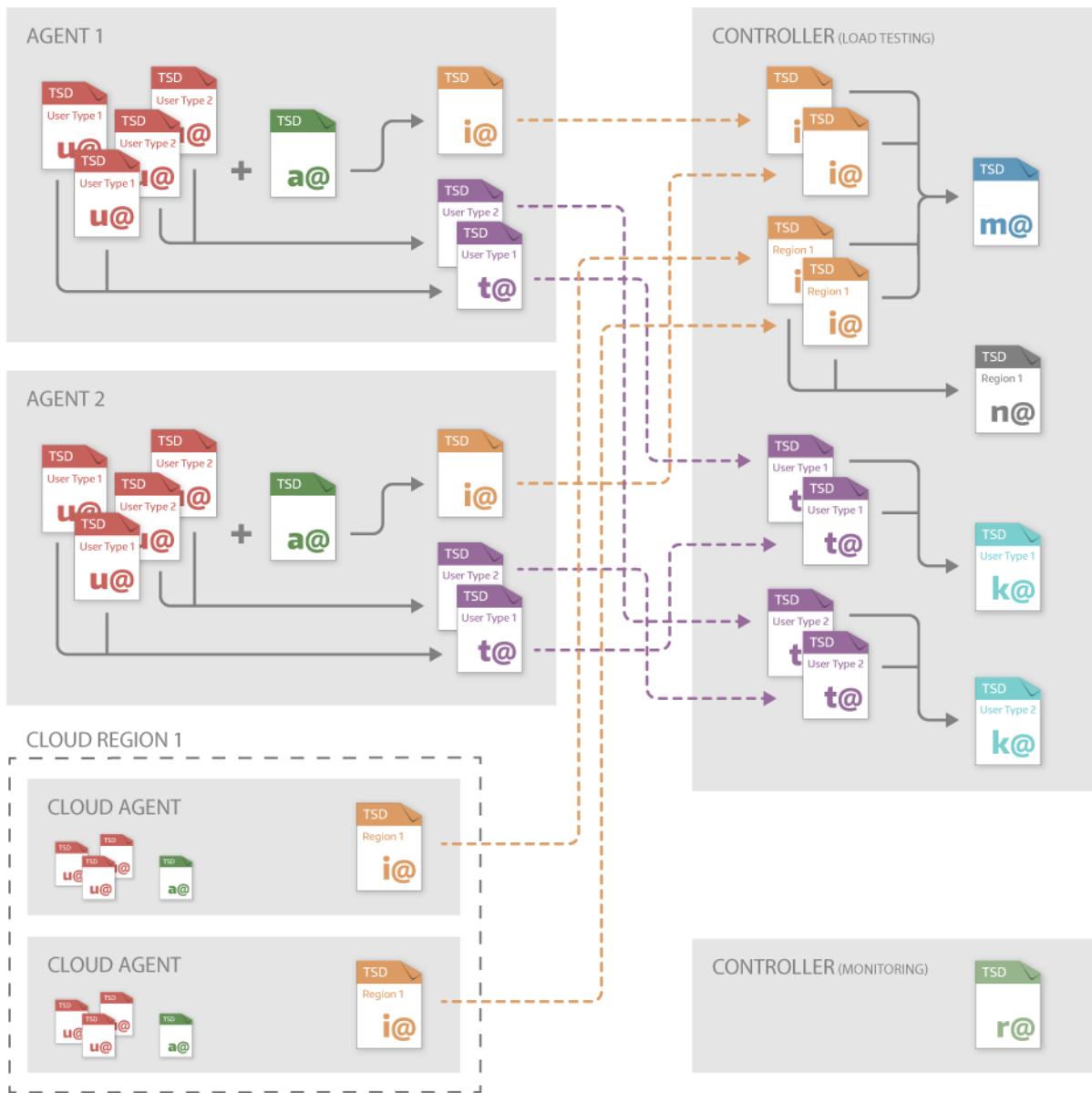
Term	Description
Graph	A graph is a graphical representation of a measure. When you drag a measure onto a chart, it appears as a graph with a particular color, style, and width.
Series	A series is a sequence of values of a certain measure. For example: The measure Active users has the values 2, 4, 6, 8. This sequence of values is a series.
Measure	A measure is the smallest grouping entity for the values that are collected during a load test or during a monitoring session for a particular measuring point. A measure has a name and type. A set of measured values that are related to the measure is stored as a series in a time series data (.tsd) file. For example: Requests sent or Trans. (busy) ok[s] are measures.
Report	In a report, you can summarize particular results and important findings of a load test. A report consists of tables and charts with descriptive text. In Performance Explorer you can create these kinds of reports: html overview reports, word overview reports, and comparison reports (region, user type, and custom comparison reports).
Table	A table shows the data that was collected during a load test or during a monitoring session in a tabular form.
Monitor Chart	A monitor chart displays graphs that are updated in real-time. The dynamic monitor charts are used for monitoring sessions, while static charts are used for the results analysis of a completed load test.
Monitor Writer	A monitor writer is a graphical and textual summary of a monitoring session.
System	A system is a machine, technology, or application that can be monitored. Performance Explorer supports a broad range of systems for monitoring.

## Time Series Data Files

TSD File	Naming	Description
User File	u@<...>.tsd	A <b>user file</b> contains the time series data from <b>one virtual user</b> . During a load test, user files are generated on the agents.
Agent File	a@<...>.tsd	An <b>agent file</b> contains the time series data from one agent. During a load test, agent files are generated on the agents.
Intermediate File	i@<...>.tsd	An <b>intermediate file</b> contains all time series data from <b>one agent</b> including the time series data from <b>all virtual users</b> running on that agent. When the load test is complete, intermediate files are generated on the agents by merging the user files and the agent files together. When merged, the intermediate files are transferred to the controller.

TSD File	Naming	Description
Merged File	m@<...>.tsd	A <b>merged file</b> contains the time series data from <b>all agents</b> . It is generated on the controller by merging the intermediate files together. The merged file can be described as the master file, which holds all the time series data of a load test.
Agent File per User Type	t@<...>.tsd	An <b>agent file per user type</b> contains the time series data from <b>one user type of one agent</b> . Do not confuse it with a user file. Several virtual users can be assigned to one user type. Therefore, the agent file per user type usually holds the time series data from several user files. When the load test is complete, the agent files per user type are generated on the agents by merging user files with the same user type together. When merged, the agent files per user type are transferred to the controller.
Merged File per User Type	k@<...>.tsd	A <b>merged file per user type</b> contains the time series data from <b>one user type of all agents</b> . It is generated on the controller by merging the agent files with the same user type together.
Region File	n@<...>.tsd	A <b>region file</b> contains the time series data from <b>one cloud region</b> . It is generated on the controller by merging the intermediate files of cloud agents of the same region together.
Real-Time File	r@<...>.tsd	A <b>real-time file</b> contains the time series data that is collected and calculated during server-side monitoring.





## Starting Performance Explorer

Before you start working with Silk Performer, make sure that you know how your application is set up and deployed. For example, find out which of your machines powers the database and application servers.

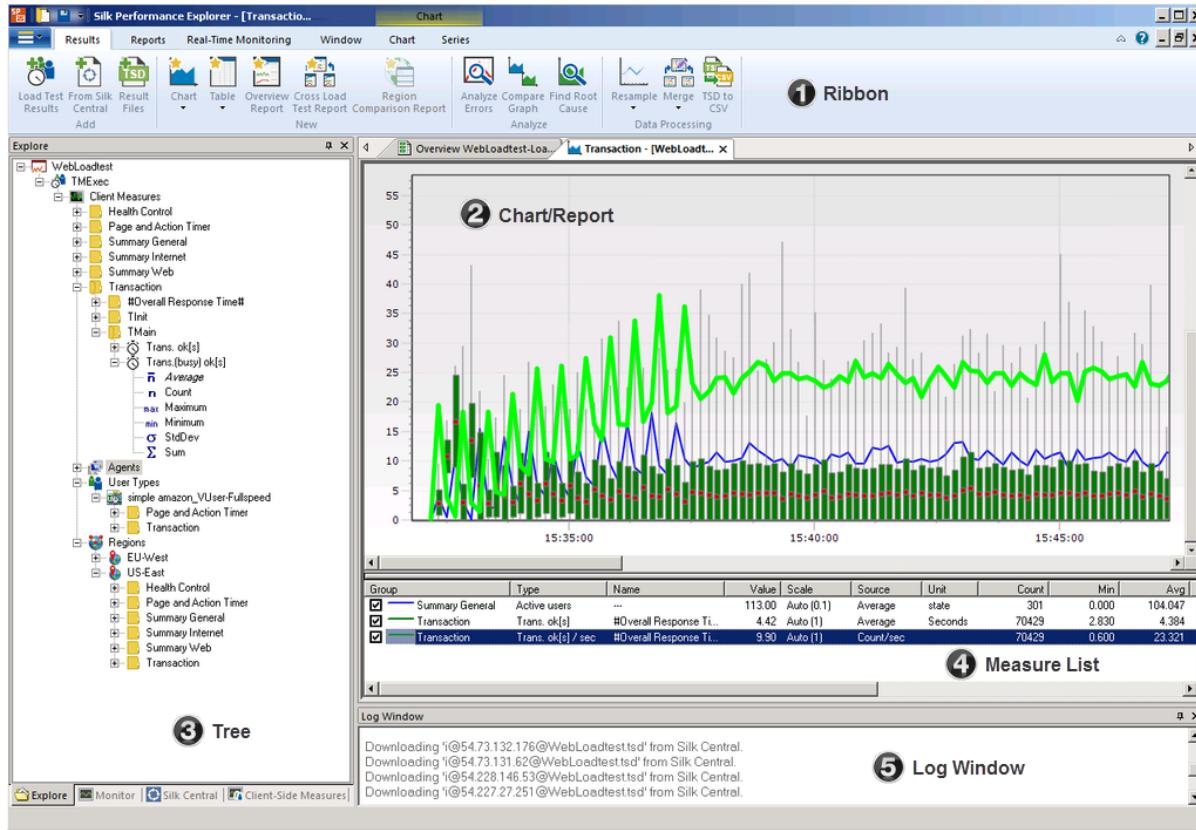
Here are some ways to start Performance Explorer:

- From the Windows start menu: Click **Start > Programs > Silk > Silk Performer <version> > Analysis Tools > Silk Performance Explorer**.
- From Silk Performer: Click **Results > Monitor Server**.
- From Silk Performer: Click **Results > Explore Time Series**.
- From Silk Performer: Click **Tools > Silk Performance Explorer**.
- From a command-line tool.

When editing a new template or launching Performance Explorer from the Silk Performer results menu, the program opens with several predefined monitor graphs and a predefined data source (your local machine).

# Tour of the UI

These are the main sections of the Performance Explorer user interface:



## (1) Ribbon

The ribbon in Performance Explorer contains 4 permanent tabs and 2 contextual tabs. While the permanent tabs are always visible, the contextual tabs are only visible if a chart or a report is open and active. These are the permanent tabs:

- Results:** This tab contains all commands around displaying and analyzing results.
- Reports:** This tab contains all commands around creating and working with reports.
- Real-Time Monitoring:** This tab contains all commands around the real-time monitoring functionality of Performance Explorer.
- Window:** This tab contains several window commands, like tiling or cascading windows, resetting the layout, and so on.

These are the contextual tabs:

- Chart:** This tab is visible when a chart is active. It contains all commands that control the appearance of charts and graphs.
- Series:** This tab is visible when a chart is active. It contains all commands that control the appearance of series.

## (2) Chart/Report

In this area the charts and reports are displayed. You can easily drag measures from the tree onto a chart. By using the commands in the contextual tabs, you can adjust the appearance of charts, graphs, and

reports. For example: In a chart, you can change colors, line width, line styles, enable scaling, display markers, null values, measure bounds, and so on.

### (3) Tree

The tree area consists of the following four trees (the tree tabs display on the bottom of the tree area): The **Explore** tree shows the measures of a load test or single .tsd file. The **Monitor** tree shows the measures of the system that is to be monitored. The **Silk Central** tree shows the measures of a load test that is managed and invoked through Silk Central. The **Client-Side-Measures** tree shows client-side measures.

You can drag measures from the tree onto a chart. You can double-click the white-space in the tree to add a load test or a system.

### (4) Measure List

The measure list is visible when a chart is open and active. It lists all measure that display in the chart. You can click the check boxes to turn the measures in the chart on and off. The list also supports multi-select functionality, so switching on and off several or all measures concurrently is possible.

### (5) Log Window

The log window lists all events that occur during performance monitoring, including errors and query states. You can switch the log window on and off in the **Window** tab and you can right-click in the log window and copy or clear the log.

## Results Analysis

With the Results Analysis functionality of Performance Explorer, you can analyze the results of a completed load test by creating charts, tables, and reports. With charts, you can visualize the data that was collected during the load test, and reports as well as tables help you to summarize important data and findings. The basis for all charts, tables, and reports is the data in the .tsd files. During each load test, Silk Performer captures a big amount of data and stores it in several time-series data (.tsd) files. When a load test is completed, you can load the data (the .tsd files) in Performance Explorer and visualize and edit them according to your requirements. All charts are fully customizable, and they can contain as many measures as required. You can open multiple charts using measures from one or multiple tests to show similarities and differences. Performance Explorer provides a variety of templates for charts, tables, and reports. Furthermore, you can place information on client and server performance in a single chart, so that you can directly view the effect of server performance on client behavior. By saving your changes as a template, you can reuse your individual settings.

Although there is no particular workflow in Performance Explorer, users usually follow this progression of steps:

- **Adding:** Add results to the Performance Explorer workspace.
- **Visualizing:** Create charts, tables, and reports based on the loaded results.
- **Analyzing:** Analyze the results (by analyzing errors, comparing graphs, finding root causes).
- **Processing:** Resample or merge the results and export the data.

## Adding Results

Before you can start visualizing, analyzing, and processing results, you have to add them to Performance Explorer. A typical way is to directly open the results from Silk Performer, by clicking the charts or buttons on the **Load Test Summary** page.

However, you can also start with a blank Performance Explorer workspace and ...

- add all results of a specified load test
- import results from Silk Central
- add single .tsd files

## Adding Load Test Results

If you want to add the results of a load test, specify a results folder and all .tsd files are added to the Performance Explorer workspace.

1. Open Performance Explorer.
2. In the **Results** tab, in the **Add** group, click **Load Test Results**.
3. Specify a project directory and a load test and click **Add**.

The added results appear in the **Explore** tree.



**Tip:** You can also double-click the white space in the tree to add results, or you right-click in the tree and use the commands in the context-menu.

## Adding Results from Silk Central

You can import results from Silk Central (as described below) or you can start Performance Explorer directly from Silk Central.

1. Open Performance Explorer.
2. In the **Results** tab, in the **Add** group, click **From Silk Central**.
3. Follow the wizard to add result files to the workspace.

The added results appear in the **Silk Central** tree.



**Tip:** You can also double-click the white space in the tree to add results, or you right-click in the tree and use the commands in the context-menu.

## Adding Single Result Files

You can choose single .tsd files (even from different load tests) and add them to the Performance Explorer workspace.

1. Open Performance Explorer.
2. In the **Results** tab, in the **Add** group, click **Result Files**.
3. Follow the wizard to add result files to the workspace.

The added results appear in the **Explore** tree.



**Tip:** You can also double-click the white space in the tree to add results, or you right-click in the tree and use the commands in the context-menu.

## Visualizing Results

When you have added results to the Performance Explorer workspace, you can visualize them with the help of charts, tables, and reports.

### Creating Charts

1. Open Performance Explorer.
2. In the **Results** tab, in the **New** group, click **Chart** to create an empty chart.
3. Select a .tsd file in the dialog.

This dialog only appears if you have not yet added results to the workspace.

4. Drag measures from the tree onto the chart.



**Tip:** You can also click the **Chart** drop-down arrow and click **Predefined Chart** to create a chart from a template or click **Empty VUser-based Chart** to create an empty chart, which shows the number of virtual users on the x-axis.

## Creating Tables

1. Open Performance Explorer.
2. In the **Results** tab, in the **New** group, click **Table** to create an empty table.
3. Select a .tsd file in the dialog.  
This dialog only appears if you have not yet added results to the workspace.
4. Drag measures from the tree onto the table.



**Tip:** You can also click the **Table** drop-down arrow and click **Predefined Table** to create a table from a template.

## Creating Reports

1. Open Performance Explorer and add results to the workspace.
2. In the **Reports** tab, in the **Overview Report** and **Comparison Report** groups, click one of the following buttons:
  - **Word**
  - **HTML**
  - **Region**
  - **User Type**
  - **Custom**

For more information on the various reports, see the respective topics in the *Reporting* section.

## Analyzing Results

Performance Explorer provides a number of features that allow you to analyze the results of a load test. You can perform a comprehensive error analysis with the **Analyze Errors** feature, you can compare measures from various load tests with the feature **Compare Graphs**, and you can use the **Find Root Cause** capability.

## Analyzing Errors

Performance Explorer includes a feature that assists you in analyzing the errors that occur during load tests. While the overview report displays some error information, the **Analyze Errors** feature displays all errors in a comprehensive error chart and lists detailed information.

Before you follow the steps below, configure and execute a load test in Silk Performer.

1. When your load test is completed, the **Load Test Summary** page displays. Click **Analyze errors** in the **Next Steps** area. Performance Explorer opens and displays the error chart.
2. You can also click the **Load Test Results** chart or you click **Analyze Load Test** in the **Next Steps** area. Performance Explorer opens. Then, on the **Results** tab, in the **Analyze** group, click **Analyze Errors**. An error chart opens. You can zoom into the chart to drill down to the specific information you need.



**Tip:** You can also start with a blank Performance Explorer workspace. On the **Results** tab, in the **Analyze** group, click **Analyze Errors** and specify a .tsd file.

## Error Analysis Graph

The Error Analysis graph contains the following information:

View	Description
Error Groups	The <b>Error Groups</b> pane displays group information for all the errors that occurred during the analyzed load test. Use the zoom and shift features to view information for specific time frames.
Error Details	The <b>Error Details</b> pane lists all occurrences of errors in the selected time span. Double-click an error or right-click an error and select <b>Show TrueLog</b> to invoke TrueLog Explorer with the respective TrueLog file that contains visualized information for the error occurrence to help you find the root cause of the selected error.
Legend	The <b>Legend</b> pane lists the displayed curves of the chart, which by default is only the error summary of the error analysis. It also displays more detailed information about occurrences, minimum and maximum values, and so on. Hold your cursor over the graph to see the number of error occurrences for a specific time frame. The <b>Value</b> column updates as you move your cursor over the graph.

## Analyzing Root Causes

Before you perform this task, specify the time-series data (TSD) file that is to be used as the data source for the graph.

Performance Explorer's automatic result-correlation feature facilitates root-cause analysis of network and server bottlenecks by correlating client-side issues with corresponding server-side measurements.

Automatic result correlation identifies the server-side measurements that are most closely associated with specific client-side errors, thereby enabling you to better identify server-side problems and expedite debugging efforts. Result correlation also works in the reverse: Server-side issues can be correlated with client-side measures.

Automatic result correlation statistically correlates key measures with dependent measures. For example, if a significant increase in server response time is detected by a client-side measure at 18:20 (6:20 p.m.), automatic result correlation can identify the server-side measures that contributed to this drop in client-side performance.

1. Click within the measurement graph that you want to analyze and drag your cursor to the right to select the time frame to analyze.

 **Note:** To slide the time line forward or backward in time, right-click the time line and drag your mouse right or left. The time line can also be moved vertically along its Y axis.

 **Note:** To select a shorter period of time for analysis, drag the time line to the right. To select a longer period of time for analysis, drag the time line to the left.

2. Click **Find Root Cause** on the workflow bar.

*Alternative:* Right-click in the graph and choose **Root Cause Analysis**.

The **Find Root Cause - Correlation Settings** dialog box opens. The base measure appears in the **Base Measure** box.

3. From the **Correlate with** list box, select the type of measurement you want to correlate with the base measurement.
4. Adjust the date- and time-selection settings by selecting a new start date, start time, end date, or end time from the appropriate list box.

Date and time selection settings are defined automatically based on the time frame selected in the time line.

- In the **Results** area, specify how to filter measure results based on how well they match.  
Choose one of the following options:
    - Click the **Best [x] correlation numbers** option button and specify the number of measures to be returned in the text box.
    - Click the **Minimum correlation of [x]%** option button and specify a minimum correlation relevance by typing a value in the text box.
  - Click **Next**.
-  **Note:** If you are correlating against client and server measures, client measures, or server measures, advance to step 10.
- The **Correlation Measures Properties** page opens.
- When correlating against a custom measure, you must define which measure groups are to be correlated. Select the groups you want to correlate against by checking the **Measures** check boxes.
  - Optional:* Add other measures by clicking **Add File** and browsing to and selecting a time-series data file. You can also remove measures by selecting them and clicking **Remove File**.
  - Click **Next** to run the correlation. The **Correlation Results** page opens.
- By default, returned measures are listed in order of degree of correlation, and all correlations are selected.
- Click the **Correlation Group** and **Name** column headers to sort returned measures.
  - Uncheck the check boxes for the measures that you do not want to include in the correlation graph.
  - Click **Finish** to generate the correlation graph.

## Comparing Graphs

When you have created and customized a chart with measures of interest, you might want to compare these measures with measures from a different user type, from different agents, different regions, or from a different load test. Performance Explorer provides the **Compare Graph** feature for this purpose. When you click **Compare Graph**, Performance Explorer creates a new chart with the selected measure and automatically adds the ones you want them to compare with.

- Create a new chart: On the **Results** tab, in the **New** group, click **Chart**.
- If you have not already loaded result files into Performance Explorer, select a .tsd file in the dialog.
- Drag the measures of your interest from the tree onto the graph.
- On the **Results** tab, in the **Analyze** group, click the drop-down arrow below **Compare Graph** and select one of the following options:
  - With other User Types:** Creates a new chart with the selected measures from the active chart and adds the same measures from all user types.
  - With other Agents:** Creates a new chart with the selected measures from the active chart and adds the same measures from all agents.
  - With other Regions:** Creates a new chart with the selected measures from the active chart and adds the same measures from all regions.
  - With other TSD File:** Creates a new chart with the selected measures from the active chart and adds the same measures from a different .tsd file.
- If you chose to compare graphs with another .tsd file, Performance Explorer prompts you to select the .tsd file.

## Processing Results

With Performance Explorer, you can process results in various ways:

- You can resample results by changing the interval and redefining the time frame.

- You can merge the results of two or more load tests.
- You can export results to process them in third-party tools like spreadsheet programs.

## Resampling Load Test Results

Specify a load test results folder and resample the data of the containing .tsd files by changing the interval and redefining the time frame.

1. On the **Results** tab, in the **Data Processing** group, click **Resample Load Test Results**.
2. Specify the **Source directory** of the load test that contains the .tsd files that are to be resampled. Specify the name of the **Target folder** and click **Next**.
3. Specify an **Output interval**. For longer running load tests, an increased output interval is useful because it results in a smaller target file.
4. Choose one of the following merge types:
  - Click **Relative** to specify that all series start at the same time.
  - Click **Absolute** to consider the individual start time of each series.

If you merge files from different load tests, the start times of those files are different, so use the **Relative** merge type.

If you click **Absolute**, you can specify a time frame. Only data within this time frame is merged into the target file.
5. Adjust the date and time settings by selecting a new start date/time and end date/time.  
Date and time settings are defined automatically based on the time frame selected in the time line.
6. Click **Next** or **Finish** to start the resampling process.

## Resampling Single Result Files

Specify a result file and resample the containing data by changing the interval and redefining the time frame. Increasing the output interval is useful to make result files smaller in size and to smoothen the corresponding graphs.

1. On the **Results** tab, in the **Data Processing** group, click **Resample Single Result File**.
2. Specify the **Source file**. This is the .tsd file that is to be resampled. Specify the name of the **Destination file** and click **Next**.  
The format of the destination file will be `m@@<sourcefilename>_r.tsd`
3. Specify an **Output interval**. For longer running load tests, an increased output interval is useful because it results in a smaller target file.
4. Choose one of the following merge types:
  - Click **Relative** to specify that all series start at the same time.
  - Click **Absolute** to consider the individual start time of each series.

If you merge files from different load tests, the start times of those files are different, so use the **Relative** merge type.

If you click **Absolute**, you can specify a time frame. Only data within this time frame is merged into the target file.
5. Adjust the date and time settings by selecting a new start date/time and end date/time.  
Date and time settings are defined automatically based on the time frame selected in the time line.
6. Click **Next** or **Finish** to start the resampling process.

## Merging Load Test Results

To run a big load test, it can be necessary to start the load test from multiple controllers. Consequently, Silk Performer will generate multiple result folders - one folder per controller. Use the **Merge Results** feature to

merge the results into one result folder. This allows you to view all metrics in a single comprehensive report.



**Note:** Results from different projects cannot be merged.

1. On the **Results** tab, in the **Data Processing** group, click **Merge Results**.
2. Drag two or more load test result folders from a file explorer onto the table or click **Add** to browse for result folders.
3. Specify a master load test in the **Master** column. Some values from the result files cannot be merged (like the project description). In such a case, Performance Explorer will use the value of the master load test.
4. Select a **Merge type**:
  - **absolute**: In a report, the results will display with their actual start time on the time line. For example: If you started a load test on Controller B 10 seconds after you started the load test on Controller A, the results will display on the time line accordingly.
  - **relative**: In a report, all results will display with the same start time on the time line, regardless of the actual start times of the load tests.
5. Specify an **Interval** for the merged .tsd files. The minimum interval (shown in the **Interval** list) is the lowest common multiple of the intervals from all .tsd files that are located in the specified results folders.
6. Specify a **Target folder**. The merged results will be stored in this folder.
7. Click **Merge**.

When the results are merged successfully, Performance Explorer displays an overview report. This newly created report has the workload type **Merged Workload**, which contains all executed usergroups.

## Merging Single Result Files

Merging single result files is useful in the following situations:

- You want to create a custom .tsd file based on selected source .tsd files.
  - You want to recreate the merged .tsd file (`m@... .tsd`) to include results from agents that have lost connection during a load test. If the controller-agent connection fails, agent files reside on the agent in the local results directory. Depending on when the connection is lost, you might find either user-only or merged agent files. If you want these results included in the intermediate file of your load-test controller, you must manually merge the files.
1. If necessary, copy all files you want to merge into a single directory.
  2. Choose **Wizard > Custom Merge Wizard**. The **Custom Merge Wizard** opens.
  3. Select the directory where all the files are stored and select all the files to merge.
  4. Click **Finish**.

A new, merged file containing all the data is generated.



**Note:** Make sure to not select .tsd files with the same or dependent data. For example: Merging User Type .tsd files and merged User Type .tsd files results in multiple aggregation of the same series, which usually is not a reasonable scenario.

## Exporting TSD Files As CSV Files

With the **TSD to CSV** feature, you can export the content of a .tsd (time series data) file to a .csv (comma separated values) file. You can then analyze and edit the data in an external spreadsheet program.

You can also use the `TSD File Export Tool`, a command-line tool that allows you to automate the export process.

1. On the **Results** tab, in the **Data Processing** group, click **TSD to CSV**.

2. Select a **Source file**, specify a name and location for your **Export file**, and click **Next**.
3. Select the measures you want to export and click **Next**.
4. Select an **Export type**:
  - Click **Standard** and specify the data source elements of the result file that must be included in the .csv file.
  - Click **Dump** to include all meta data in the .csv file.
5. You can **Include header information** in the .csv file. Select the comma or the semicolon as your separator and select how a decimal point is to be displayed in the .csv file.
6. Click **Finish** to export the data and generate the .csv file.
7. To view the output file, click **Yes** on the subsequent dialog.

## Using the TSD File Export Tool

The TSD File Export Tool is a command-line tool that allows you to automate exporting the content of .tsd files to .csv files. You can also use the **TSD to CSV** feature in Performance Explorer, which provides a user interface for the export settings.

1. Open the tool from the Microsoft Windows start menu: **Start > Programs > Silk > Silk Performer <version> > Analysis Tools > TSD File Export Tool**.
2. At the command prompt, enter specifications for the file export based on the following syntax:

```
Tsd2Csv tsdFile [csvFile] [-Delimiter char] [-DecimalPoint char] [-dump]
```

Command	Description
tsdFiles	Enter the path and name of the .tsd file that is to be exported.
[csvFile]	Enter the path and name of the .csv file that is to be created.
[-Delimiter char]	Define the delimiter character.
[-DecimalPoint char]	Define the character that signifies decimal points in the output file.
[-dump]	Ignores -DecimalPoint parameter.

3. Press **Enter** to generate the .csv file.

### Example

```
Tsd2Csv "input.tsd" "output.csv" -Delimiter "," -DecimalPoint  
". "
```

## Viewing Results

Performance Explorer can convert charts, reports, and tables into HTML documents and display these in a web browser. This enables you to include charts, reports, and tables in any kind of documents, publish them online, or print them in useful formats.

### Viewing Charts in a Web Browser

Create or open the graph that you want to view in a Web browser.

1. Create or open the chart that you want to view in a web browser.
2. On the **Chart** tab, in the **Data** group, click **View in Browser**.

3. Enter a **Caption** and a **Description** for the chart and click **OK**. The caption and description display in the browser above the chart.

## Viewing Reports and Tables in a Web Browser

Create or open the report that you want to view in a Web browser.

1. Create or open the HTML overview report or table that you want to view in a web browser.
2. On the **Reports** tab, in the **Tasks** group, click **Open in Browser**.
3. Enter a **Caption** and a **Description** for the report or table and click **OK**. The caption and description display in the browser above the chart.

## Exporting Charts

You can export charts in various formats to present them to an audience that has not access to Performance Explorer.

1. Create or open the chart that you want to export.
  2. On the **Chart** tab, in the **Chart** group, click the drop-down arrow below **Export**.
    - Click **CSV File** to export the chart data to a .csv file. On the subsequent dialog, you can select from a range of file formats, like .htm, .mht, or .jpg. The .csv format is the default value.
    - Click **Excel Document** to export the chart data to a Microsoft Excel document. A chart is automatically created in Excel.
-  **Tip:** The .mht format allows you to easily send charts through email. The .html format allows you to publish charts on the web. The .csv format allows you to further process data in external spreadsheet programs. The picture formats allow you to insert charts as pictures in other documents.

## Exporting Reports and Tables

You can export reports and tables in various formats to present them to an audience that has not access to Performance Explorer.

1. Create or open the report or table that you want to export.
  2. On the **Reports** tab, in the **Tasks** group, click **Save as**.
  3. On the subsequent dialog, select a directory, enter a file name, select a file format, and click **Save**.
-  **Tip:** The .mht format allows you to easily send reports or tables through email. The .html format allows you to publish reports or tables on the web. The .csv format allows you to further process data in external spreadsheet programs.
-  **Tip:** You can also export reports through the command line interface. This allows you to automate the export process.

## Exporting Reports Through the Command-Line Interface

HTML reports can also be exported as Web archives (MHT files) through the command-line interface. To export reports, use the command-line parameter `/EXPORTOVR:<target_file>`.

This parameter saves the active OVR as HTML or as an MHT file, depending on the file extension of the target file specification.

 **Note:** The last loaded TSD file, or the last TSD file specified through the command line (`/TSD:<tsd_file>`) is used for exporting.

The command-line parameter `/ACTION:OVERVIEWREPORT[:<template>]` allows you to specify a template file for the report.

The following example illustrates exporting a Web archive file through the command line:

# Reporting

Creating reports is one possibility to summarize and analyze the data that was collected during a load test. Learn about the report types Performance Explorer provides and how to create and work with reports.

## Word Overview Reports

Performance Explorer allows you to view your load test results in a Microsoft Word document. This is especially useful if you want to further process your results for presentation and add custom formatting.

### Creating Word Overview Reports

1. Add load test results to the Performance Explorer workspace
2. In the **Reports** tab, in the **Overview Report** group, click one of the following buttons:
  - Click **Word** or **Word Overview Report** to show the results in a Word document.
  - Click **Word Overview Report from Template**, select a .docx file and click **Create**.
  - Click **Custom Word Overview Report**, select the information that the Word overview report shall contain, and click **Create**.



**Note:** You can enable **Use template for all future reports** to always create Word overview reports based on a template. To disable this option, click the Performance Explorer application button (on the top left) and click **Options**. Click the **Reporting** tab and disable **Use template (.docx) when creating a new Word Overview Report**.

### Creating a Template for Word Overview Reports

Before you can save and use a template, create a Word overview report with all the information you want to have in the report.

1. On the **Reports** tab, in the **Overview Report** group, click **Custom Word Overview Report**.
2. Select the information that the template shall contain and click **Create**. Microsoft Word opens and the specified information displays.
3. Adapt the content and the appearance of the Word document to your needs. You can add information and comments, rearrange fields, change element styles, change the header, footer, or logo, remove unwanted content, and so on.
4. Save the Word document as .docx file and close it. Your template is created.
5. Switch back to Performance Explorer. On the **Reports** tab, in the **Overview Report** group, click **Word Overview Report from Template**.
6. Browse for the .docx file you have just defined and click **Create**.



**Note:** You can enable **Use template for all future reports** to always create Word overview reports based on a template. To disable this option, click the Performance Explorer application button (on the top left) and click **Options**. Click the **Reporting** tab and disable **Use template (.docx) when creating a new Word Overview Report**.

## HTML Overview Reports

When a load test completes, the **Load Test Summary** page displays in Silk Performer and an overview report is generated automatically. In Silk Performer, you can find the overview report in the **Results** tree. To open the overview report in Performance Explorer, double-click the report in the tree or click **Analyze load test** on the **Load Test Summary** page. You can also open a blank Performance Explorer workspace and click **HTML** in the **Overview Report** group on the **Reports** tab.

## Creating HTML Overview Reports

Before you follow the steps below, configure a load test in Silk Performer and execute it. When your load test is completed, the **Load Test Summary** page displays. Perform one of the following steps:

- Click **Analyze load test** in the **Next Steps** area.
- Click **Results** on the bottom of the tree, expand a load test node, and double-click **Overview Report**.

Performance Explorer opens and the overview report displays.

To open an HTML overview report out of Performance Explorer, perform the following steps:

1. Add load test results to the Performance Explorer workspace
2. In the **Reports** tab, in the **Overview Report** group, click one of the following buttons:
  - Click **HTML** or **HTML Overview Report** to show the results in an HTML overview report.
  - Click **HTML Overview Report from Template**, select an .ovt file (overview report template) and click **Create**.
  - Click **Custom HTML Overview Report**, select the information that the HTML overview report shall contain, and click **Create**.



**Note:** You can enable **Use template for all future reports** to always create HTML overview reports based on a template. To disable this option, click the Performance Explorer application button (on the top left) and click **Options**. Click the **Reporting** tab and disable **Use template (.ovt) when creating a new HTML Overview Report**.



**Note:** By default, overview reports are generated automatically. To disable the automatic generation, click the Performance Explorer application button (on the top left), click **Options**, click the **Reporting** tab and disable **Generate Overview Reports automatically**.

## Creating a Template for HTML Overview Reports

Templates for HTML Overview Reports are saved as .ovt files.

1. In Performance Explorer, create or open an HTML Overview Report.
2. On the **Reports** tab, in the **Overview Report** group, click **Customize**.
3. Select the information that the template shall contain and click **Customize**.
4. On the **Reports** tab, in the **Overview Report** group, click **Save As Template**.
5. Save the template as .ovt file (overview report template).



**Note:** You can instruct Performance Explorer to always use this template when an HTML Overview Report is created: Click the Performance Explorer application button (on the top left) and click **Options**. Click the **Reporting** tab, enable **Use template (.ovt) when creating a new HTML Overview Report** and specify the file.

## Assigning Overview Report Templates to Projects

You can specify an overview report template for each Silk Performer project. All automatically generated HTML overview reports will use this template and the template will be preselected when you create HTML overview reports manually.

1. In Silk Performer, expand the **Profiles** node in the menu tree.
2. Right-click the profile that you want to configure and choose **Edit Profile**.

*Alternative:* Choose **Settings > Active Profile**.

The **Profile - [<profile name>]** dialog box opens.

3. In the shortcut list on the left, click the **Results** icon.

4. Click the **Time Series** tab.
5. Browse for a template in the **Overview report template** section and click **Open**.
6. Click **OK**.

 **Note:** Performance Explorer's command line interface also offers the `/OVT:<template>` command for assigning overview report templates to Silk Performer projects.

## Comparison Reports

Performance Explorer provides three types of comparison reports:

- region comparison reports
- user type comparison reports
- custom comparison reports

A comparison report allows you to compare the results of up to eight test executions side by side. Its structure resembles the structure of a standard baseline report: Comparison reports contain summaries and statistics regarding transactions, errors, page and action timers, web forms, rankings, and many other measures.

At the bottom of a comparison report, the **Baseline Pane** displays. This pane includes a column with either the name of a specific region or user type, or a column named **Overall** if the row contains all test results. The **Agents** column displays the number of agents that were used for the test in each specified region.

Any test result can be defined as the baseline. All other results will use this baseline as reference and they will be compared to the baseline. In report rankings, the baseline result is always listed first. A **b** icon or **baseline** tag identifies the baseline result. All measures that are evaluated against the baseline are identified with heat fields (colored bars in the report). Move your cursor over any heat field to view detailed information. To set a result as the baseline, right-click it and click **Set as baseline**.

### Region Comparison Reports

Region comparison reports allow you to compare the results of cloud-based load tests. These are load tests that use cloud agents to generate load. You can compare the results from different regions side by side. The report shows one region per column. A maximum of eight regions can be included in a single region comparison report.

Both overall results and region-specific results can be compared. To add measures, drag them from the tree onto the report. To add the overall results, drag the **Overall** node onto the report.

### User Type Comparison Reports

User type comparison reports allow you to compare user types side by side. The report shows one user type per column. A maximum of eight user types can be included in a single user type comparison report.

Both overall results and user-type-specific results can be compared. To add measures, drag them from the tree onto the report. To add the overall results of a test, drag the **Overall** node onto the report. The data set is marked with **Overall** in the User Type column of the Baseline Pane.

### Custom Comparison Reports

Custom comparison reports allow you to compare the results of up to eight test executions side by side. Additionally, you can compare load test executions with user types or regions if this makes sense in your scenario.

### Creating Region Comparison Reports

Before you follow the steps below, configure a cloud-based load test in Silk Performer with cloud agents from different regions and execute the load test.

- When your cloud-based load test is completed, the **Load Test Summary** page displays. Click the **Load Test Results** chart to open it in Performance Explorer.  
The results are loaded into Performance Explorer and display in the tree.
- Expand the nodes in the tree until you see the **Regions** node. Results data (time series data) is available for each cloud region that was included in the load test.



**Note:** The data is available for the cloud regions, not for the single cloud agents.

- Select a region and click **Region** in the **Comparison Report** group on the **Reports** tab. The report displays, showing the selected region. You can now drag further regions from the tree onto the report.
  - To show all regions in the report at once, select the **Regions** node and click **Region**.
  - To show all regions and an **Overall** column in the report, select the **Client Measures** node and click **Region**.



**Tip:** You can also right-click a node in the tree and use the commands in the context menu.



**Note:** Only regions from the same load test can be added to a region comparison report.

- To define the results of a certain region as the baseline, right-click the region in the baseline pane at the bottom of the report and click **Set as baseline**.

## Creating User Type Comparison Reports

- Open Performance Explorer and add results. The results are loaded into Performance Explorer and display in the tree.
- Expand the nodes in the tree until you see the **User Types** node.
- Select a user type and click **User Type** in the **Comparison Report** group on the **Reports** tab. The report displays, showing the selected user type. You can now drag further user types from the tree onto the report.
  - To show all user types in the report at once, select the **User Types** node and click **User Type**.
  - To show all user types and an **Overall** column for the workload, select the **Client Measures** node and click **User Type**.



**Tip:** You can also right-click a node in the tree and use the commands in the context-menu.



**Note:** Only user types from the same load test can be added to a user type comparison report.

- To define the results of a certain user type as the baseline, right-click the user type in the baseline pane at the bottom of the report and click **Set as baseline**.

## Creating Custom Comparison Reports

- Open Performance Explorer and add results. The results are loaded into Performance Explorer and display in the tree.
- Select a project node in the tree and click **Custom** in the **Comparison Report** group on the **Reports** tab. The report is created with all load tests that are loaded in Performance Explorer. These are the load tests that appear under the project node in the tree.
- You can add further load test results and drag them from the tree onto the report. You can even add load tests as well as user type or region results from various projects to one custom comparison report. A maximum of eight data sets can be included in a single custom comparison report.



**Tip:** You can also right-click a node in the tree and use the commands in the context menu.

- To define the results of a certain load test as the baseline, right-click the load test in the baseline pane at the bottom of the report and click **Set as baseline**.

# Real-Time Monitoring

With the Real-Time Monitoring functionality of Performance Explorer, you can monitor a broad range of systems by creating and configuring charts that show the system performance in real-time. It is possible to open multiple charts at the same time, which allows you to watch the performance of two or more systems simultaneously, for example the web server performance and the operating system performance. Adding measures to a chart is easy and intuitive in Performance Explorer: You can drag a single measure or a set of measures from the tree onto the chart.

## Monitoring Technologies

Performance Explorer supports a broad range of monitoring technologies, which are described in this section.

### Monitoring Windows Machines

This section describes the requirements and setup of Windows machines and provides troubleshooting tips.

#### Requirements for Monitoring Windows Machines

Make sure that the following preconditions are met to be able to monitor the respective remote systems:

##### Target machines with Windows Vista and newer client operating systems

The Remote Registry service must be running on the machine that you want to monitor. This service does not run by default on Windows Vista and Windows 7 machines.

##### Troubleshooting Windows Performance Monitoring

The following error messages can occur when monitoring Windows-based machines:

Error Message	Possible Cause	Resolution
Computer Name Not Found	You might receive this error message when trying to monitor a remote computer from a non-administrator account.	Enable remote monitoring on a Windows NT computer.
Error=AGENT_ERROR_TCP_SEN D_REQUEST_FAILED	This problem can occur if your organization uses proxy auto-configuration files to direct requests to a proxy server, and you specify the automatic configuration script address rather than the registered name of the proxy server.  If your organization uses a non-passive proxy, in most cases you must also specify the port number.	Verify the following proxy settings: <ul style="list-style-type: none"><li>• Correct entry of the proxy server name, user name, and password</li><li>• Ability to ping the proxy server from the RSM computer</li><li>• Ability of the RSM computer to connect to the portal computer</li></ul>
System 5 - Access denied	When monitoring a machine with Windows 2008 or newer using CUSTOM / PERFMON option, logging in with credentials of a user added to the Administrators group	The user with which you run Performance Explorer needs to be added to the Administrators group on the remote computer that you want to monitor. If you specify different user

Error Message	Possible Cause	Resolution
	generates a System 5 - access denied error.	<p>credentials in Performance Explorer's <b>Data Source Wizard</b>, that user also needs to be added to the Administrators group on the remote computer. To monitor remote machines successfully, you need to launch Performance Explorer (<code>PerfExp.exe</code>) under the same user account that is being used to connect to the remote machine(s).</p> <p> <b>Tip:</b> To launch <code>PerfExp.exe</code> under a specific user account, press <b>shift + right-click</b> and select <b>Run as different user</b>.</p>

## JMX Monitoring

The Silk Performer support for Java Management Extensions (JMX) monitoring enables you to monitor MBean attributes that are exposed by Java application servers. MBean attributes that return numeric data are added to Performance Explorer monitors as data sources.

### Prerequisites for JMX Monitoring

The prerequisites for JMX monitoring include the following items:

- Java Runtime (JVM) 1.5 or later
- The JVM application that you want to monitor must have an open port for monitoring

### Connecting to a JMX Data Source

Establish a connection to a Java application server to monitor MBean attributes.

1. On the **Real-Time Monitoring** tab, in the **Monitor** group, click **System**.
2. Click **Select from predefined Data Sources** and then click **Next**.
3. Perform one of the following steps:
  - To explore all available MBean attributes on an application server, expand the **Application Server** folder and select a JMX recording profile from a vendor-specific folder. Profile names end with the extension `.jmxx`.
  - To explore a selection of MBeans that contain typical attributes for monitoring, expand the **Application Server** folder and select a JMX recording profile from a vendor-specific folder. Profile names end with the extension `.ejmxx`.
  - To create a generic JMX monitoring data source, expand the **Custom Data** folder and select **JMX Data**.
4. In the **Hostname** text box, specify the JMX server.
5. *Optional:* In the **Alias** text box, specify the alias name.

The alias must be a highly descriptive synonym for the monitored server. It is recommended that you group measures on a particular machine.

For example, both WebLogic and IIS might be installed on the same computer. Both servers require monitoring, but the two performance measures must appear in separate menu trees.

6. Specify the port, username, and password that are appropriate for the username.
7. If you opted to create a generic JMX monitoring data source, select a supported application configuration from the **Application** list box.
8. Click **Server Configuration**. The **JMX Connection Configuration** dialog box opens.
9. Specify the application server installation directory where the application server's communication libraries are located.

The relative classpath entries of the application configuration file, together with the specified application server install directory, form the classpath under which the JMX client runs. The resulting classpath is viewed in the **Resulting classpath** text box.



**Note:** It is recommended that you use a UNC path or copy the application library directory of the server to your local machine and specify your local copy as the application server installation directory. To complete this task, click [...] next to the **App. server install directory** text box.



**Note:** JMX monitoring of WebSphere requires that you specify an IBM JDK. If your WebSphere server has disabled administrative security or enabled administrative security and the inbound CSIV2 transport layer is TCP/IP, you can specify a SUN JDK.

10. In the **Java home directory** text box, specify the installation directory where the client's communication libraries are located.  
*Alternative:* Click [...] to navigate to the appropriate directory.
11. Specify additional libraries in the **Additional classpath** text box.
12. Specify additional virtual machine parameters in the **Additional JVM parameters** text box.  
The parameters on the **Connection** and **Visualization** pages are preconfigured for each server.
13. Click **OK**. The **Connection Parameters** page opens.
14. Click **Next** and then click **Finish**. Performance Explorer connects to the JMX server.
15. Examine the exposed beans in the **JMX Data Source Browser**.
16. Click **Finish**.

### Specifying MBeans for Monitoring

Application configuration profiles are stored as XML files at C:\Program Files\Silk\Silk Performer 21.0\include\jmx-config\. Make any changes to the default configurations in the XML file directory, and create additional profiles in this directory as well. Newly created application configuration profiles are available for use as data sources.

1. Open the **JMX Data Source** browser.

The **Type Tree** page shows all the available categories of MBeans in a tree structure. The **Type List** tab, shows all available categories in a list format. All available MBeans of the selected category are visible in the **Beans** text box. Each MBean is identified by an object name, which consists of a domain name and one or more pairs of property types and values.

2. Select an MBean of interest and expand the MBean node to view the available attributes.
3. *Optional:* Click **Numeric Attributes** to filter non-numeric attributes.  
Non-numeric attributes are not supported for JMX monitoring.
4. *Optional:* Right-click an attribute and choose **Get Value** to query the value of any attribute.
5. *Optional:* Right-click an attribute that represents an object name and choose **Follow Reference** to navigate to the referenced MBean.

Add the appropriate attributes to the bean-attribute pool.

### Adding Attributes to the Bean-Attribute Pool

Before you begin this procedure, perform one of the following steps:

- Specify the appropriate MBeans for monitoring.
  - Run an Easy JMX or custom query to identify MBeans of interest.
1. Open the **JMX Data Source Browser**.
  2. Choose the attributes to monitor in the **Beans** text box.
 

 **Note:** Select multiple beans by pressing Shift or Ctrl.

 **Note:** Certain MBeans like those based on JSR-77 might contain complex attributes that consist of several submeasures or statistics. To a certain depth of recursiveness, you can select submeasures to be monitored.
  3. Right-click the selected attributes and choose **Add As**. A submenu opens.
  4. Choose one of the following options for monitoring the attribute values:
    - **Average** – Useful for count values. By default, this option is selected.
    - **Sum** – Same as **Average**, but shows a different label in the monitor GUI.
    - **Incremental** – Useful for rising values. Instead of the attribute value, the change of the value within a monitor interval is calculated.
  5. Click **Close**. The **Select displayed measures** page opens.
  6. *Optional:* Select an MBean and click **Add**. Performance Explorer adds all visible attributes of an MBean as a data source.
  7. Check the check boxes for those measures that you want to include in the initial monitor view and then click **Finish**.

A connection to the specified host is established, and an initial view that contains the measures you selected is displayed.

## Easy JMX Queries

Easy JMX queries enable access to the most important attribute values of MBeans that are hosted on JMX MBeanServers. Easy JMX queries are preconfigured, advanced queries that are executed automatically when a connection to an application server's MBeanServer is established. All MBeans that match Easy JMX query criteria are automatically displayed in the **Beans** text box of the JMX Data Source browser. Easy JMX profiles are identifiable in the **System** menu tree on the **System Selection** dialog box by the **EJMX** tags that follow them.

### Filtering Beans Using JMX Query Language

Display only the MBeans that match specific filter criteria to narrow the JMX query.

1. Open the **JMX Data Source Browser**. All available MBeans on the selected JMX MBeanServer are visible in the **Beans** text box.
2. Click the **Simple Query** tab.
3. Select a domain from the **Domain** list box.  
Leave this field set to **<All>** to search all domains.
4. *Optional:* Select a property from the **Property field** list box.  
This list displays all the properties that are included in the loaded beans.
5. *Optional:* Select a value for the selected property from the rightmost **Property field** list box.  
The Query text box shows the resulting object name that is used for JMX queries, which are based on the JMX 1.2 (JSR-003) standard.
6. *Optional:* Type an attribute name in the **Attribute filter** text box to filter the list based on a specific attribute. Only MBeans containing the specified attributes are displayed.
7. Click **Run** to execute the query. The **Beans** text box displays only those MBeans that match the filter criteria.

## Running Advanced JMX Queries

Create a sophisticated combination of simple queries to monitor data for your environment.

1. Open the JMX Data Source Browser.
2. Click the **Advanced Query** tab.

Available query types are listed in the upper window.

For example, you might see **Object Name Query**.

3. Select one of the available operators for the query from the list box of the query.  
For example, you might select the **OR** operator.
4. Select a value for the operator from the subsequent list box.  
For example, you might select **True**.  
[...] appears, enabling you to open the JMX Object Name Query dialog box and browse through a domain list.
5. Click [...] and select the appropriate domain name from the domain list. For example, you might select **jboss.cache**.
6. Continue adding elements until your query is complete.  
For example, you might add **\*:J2EEServer =local**.
7. Click **OK**.
8. Click the root element of the **Query** menu tree.
9. Type a description of the query in the lower pane of the **Advanced Query** page.

### 10. Click **Run**.

All MBeans that meet the query's criteria are displayed.

### 11. Click **Save**.

This step allows you to specify a target destination and name for the query. The file extension for JMX Data Queries is .jdq.



**Note:** Queries that are saved to the C:\Program Files\Silk\Silk Performer 21.0\include\jmx-config directory are also listed on the **Easy JMX** page and can be run directly from there.

## Running Saved Queries

Execute an existing JMX query to access the most important attribute values of MBeans that are hosted on JMX MBeanServers.

1. Open the JMX Data Source Browser.
2. Click the **Advanced Query** tab.
3. Choose **Load > Load From File > Load**.
4. Browse through the list of pre-installed queries and any custom queries that you might have saved.  
Pre-installed query types are available for a number of server types. These types are the same query types that are available with the Easy JMX profiles.
5. Click **Run**.



**Note:** Queries that are saved to C:\Program Files\Silk\Silk Performer 21.0\include\jmx-config are also listed on the **Easy JMX** page and can be run directly from there.

Performance Explorer executes the query.

## REXEC and UNIX Data Sources

Describes how to query Unix-based systems using the Remote Execution protocol.

## REXEC and UNIX Data Sources Overview

Data sources on UNIX-based systems can often be queried by using the Remote Execution protocol. Identified data sources appear in the predefined data sources list. For example, you might want to monitor **Context switches/sec** on a Solaris system. By selecting this entry in the predefined list, you do not need to specify remote execution details. When data sources are not found in the list, create a custom data source.

### Remote Execution

Remote execution means that you execute commands on the remote host by using the command-line interface. You can also execute shell scripts. To verify shell scripts, use a telnet client, connect to your UNIX system, and run some commands or scripts.

### Single Execution

With single execution, the primary domain controller emulator (PDCE) connects to the remote machine and executes the command each time a new data point is requested. With short intervals this setting might place some load on the remote machine. However, the PDCE does not start a new command on the remote machine until the command of the last interval has finished.

Without single execution, the PDCE connects to the remote machine at the beginning of the monitoring session, starts the command, and forwards the output of the command to the parser. The parser is activated at the given monitoring interval and parses available data. The command is executed only once, and the connection to the remote machine remains active during the entire monitoring session. If the connection is lost because, for instance, the command on the remote machine exits, the PDCE attempts to reconnect after a fixed interval.

#### Single Execution Example

```
ps -ef | egrep -c ".*"
```

The preceding command counts the number of processes running on a UNIX system. Enter this command during the telnet session. The command prints a value (the number of processes running) and then closes. Such behavior is called *single execution*.

You can wrap a ‘while’ loop around such commands.

#### Multiple Execution Example

```
while [ true ]; do      // Mind the spaces
    ps -ef | egrep -c ".*";
    sleep 5;
done
```

The preceding example continuously prints the number of processes at five-second intervals. Such behavior is defined as *multiple executions*.

## Using REXEC to Query UNIX

Establish a connection to a UNIX-based system that uses the Remote Execution Protocol and add the measures that you want to monitor.

1. On the **Real-Time Monitoring** tab, in the **Monitor** group, click **System**.
2. Click **Select from predefined Data Sources** and then click **Next**.
3. Expand the **Custom Data** folder, click **Rexec Data**, and then click **Next**.
4. In the **Hostname** text box, specify the machine to be monitored.
5. *Optional:* In the **Alias** text box, specify the alias name.

The alias must be a highly descriptive synonym for the monitored server. It is recommended that you group measures on a particular machine.

For example, both WebLogic and IIS might be installed on the same computer. Both servers require monitoring, but the two performance measures must appear in separate menu trees.

6. Specify the port, username, and password that are appropriate for the username.
7. Click **Next**. The **Add Rexec Measures** page opens.
8. Specify the appropriate values to create the measure that you want to monitor.

Option	Description						
<b>Measure type</b>	For the specified command, enter a type name. All measurements with the same type name are assembled into one group.						
<b>Measure name</b>	Enter the measurement name for the specified rexec command. This name is displayed in the Performance Explorer chart.						
<b>Is an average measure</b>	Check this check box for the rexec command to be considered an average measurement. This setting affects only the generation of TSD files.						
<b>Explanation</b>	Type additional information for the specified rexec command. This information is displayed in Performance Explorer.						
<b>Command</b>	Specify the rexec command to execute on the remote machine for data collection, such as <code>ps -ef   egrep -c ".+"</code>						
<b>Single execution</b>	Check this check box if you want the PDCE to connect to the remote machine and execute the command every time a new data point is requested. With short intervals, this setting might place some load on the remote machine. However, the PDCE does not start a new command on the remote machine if the command of the last interval has not finished.  By unchecking the <b>Single execution</b> check box, multiple execution is enabled.						
<b>Regular expr</b>	Every line of data that is returned by the remote machine is filtered by using the given regular expression. If the regular expression does not match a certain line, the line is discarded. Only the remaining lines are used for further steps. By default, <code>^ . *\$</code> are entered.  For example, a command's response might appear as follows for each interval:  <table border="1"><tr><td>XXXXXXXX</td><td>XXXXXXXX</td><td>XXXXXXXX</td></tr><tr><td>123400</td><td>103093</td><td>121092</td></tr></table> Only the lines with numbers are of interest. The following regular expression allows only lines with numbers and spaces.  <code>[1234567890 ].*</code>  Any other lines are ignored.	XXXXXXXX	XXXXXXXX	XXXXXXXX	123400	103093	121092
XXXXXXXX	XXXXXXXX	XXXXXXXX					
123400	103093	121092					
<b>Field index</b>	The line used for data generation is separated into fields. The separation is accomplished by using either white-space characters or custom separator characters.  Use this text box to specify the field index that must be converted into a numerical value and used as the next measurement value.  This index is a one-based index.						
<b>Separators</b>	Use this text box to specify the custom separator characters to use to separate the lines for data generation into fields.  By default, the <b>Whitespaces are separators</b> check box is checked.						

Option	Description
<b>Whitespaces are separators</b>	Check this check box to insert a separator for each empty space.
<b>Lines to skip</b>	Use this text box to specify the number of lines the PDCE ignores after the start of the command on the remote machine. This option might be useful to filter out garbage data that the command returns at the beginning. Only lines that match the provided regular expression use this setting. If more than one valid line is provided for an interval, the last line is used for data calculation.

**9.** Click **Add**.

**10.** Repeat the previous steps until you have entered all the commands that you require for measure collection, then click **Close**. The **Select displayed measures** page opens.

**11.** Check the check boxes for those measures that you want to include in the initial monitor view and then click **Finish**.

A connection to the specified host is established, and an initial view that contains the measures you selected is displayed.

### Example

Enter the command, in slightly modified form, that was used when defining multiple executions as follows:

```
while [true]; do
    ps -ef | egrep -c ".*";
    sleep {%Interval};
done
```



**Note:** The command `sleep {%Interval}` is a placeholder for a number provided by Performance Explorer. This number is the collection interval. For example, Performance Explorer might collect new data every five seconds.

A command's response might appear as follows for each interval:

```
XXXXXXXXX XXXXXXXXX XXXXXXXXX
123400    103093   121092
```

The **Field index** value specifies the column that is relevant for your defined measure. If **Field index 1** is specified, the value 123400 is returned from the table. If **Field index 2** is specified, the value 103093 is returned.

Columns in the preceding example are separated by white spaces. However, other characters, such as colons, can be used. Such values are specified in the **Separators** text box. White spaces are the default separators.

### Memory Free Custom Example

Consider this command for Linux machines:

```
cat /proc/meminfo
```

It returns the following table.

	<b>total:</b>	<b>used:</b>	<b>free:</b>	<b>shared:</b>	<b>buffers:</b>	<b>cached:</b>
Mem:	13108428	54214656	76869632	26947584	7938048	28471296
	8					
Swap:	70148096	3387392	66760704			

	<b>total:</b>	<b>used:</b>	<b>free:</b>	<b>shared:</b>	<b>buffers:</b>	<b>cached:</b>
MemTotal:	128012					
	kB					
MemFree:	75068	kB				
MemShared:	26316	kB				
Buffers:	7752	kB				
Cached:	27804	kB				
SwapTotal:	68504	kB				
SwapFree:	65196	kB				

The following table identifies the field values to use to obtain the value for **MemFree** from the preceding table:

<b>Command</b>	cat /proc/meminfo
<b>Single execution</b>	Checked
<b>Regular expr</b>	^MemFree:.*\$
<b>Field index</b>	2
<b>Separators</b>	Check <b>Whitespaces are separators</b>
<b>Lines to skip</b>	0

## SNMP Support

Standard SNMP support includes a generic interface for all SNMP data sources in which MIBs are present. Specific measures are accessed using a combination of object ID (OID) and instance ID (when objects have more than one instance). The challenge with this approach is that you must know exactly which counter instance is to be applied. Instances are only distinguished by instance ID, not by instance name. Compounding this challenge is the fact that when servers reboot, certain instance numbers can change due to dynamic monitoring-object creation. Therefore monitoring scripts must be able to cope with dynamic monitoring targets.

The value of Silk Performer's enhanced SNMP support is that it allows you to access values not by OID and instance ID, but by OID and instance name.

The following example illustrates the enhanced SNMP support that is available with Silk Performer. This example shows how predefined ESNMP measures are defined, how they are used, and how additional measures can be added to the list of pre-defined measures.

### Identifying instance IDs and OIDs

The first step in working with a predefined ESNMP measure is to identify the OID, instance ID, and instance name of a WebLogic counter. The JDBC Connection Pool Runtime Active Connections Current Count for a certain instance is accessible with the OID:

```
1.3.6.1.4.1.140.625.190.1.25 + <Instance ID>
```

Without enhanced SNMP support, you have to provide the OID of the table containing the values and the correct instance ID.

JDBC Connection Pool Runtime Object Name and JDBC Connection Pool Runtime Name contain the human readable names of the instances and are accessible with the OIDs:

```
1.3.6.1.4.1.140.625.190.1.5 + <Instance ID> for "Object Names"
```

and...

1.3.6.1.4.1.140.625.190.1.15 + <Instance ID> for "Names"

ESNMP support provides you with the option of specifying an instance name, retrieving an instance ID using the (object-) name table, and then getting the value for the correct counter. With ESNMP support, the following must be provided:

- OID of the table containing the values
- Instance name
- OID of the table containing the instance names

### Monitoring Server Performance Through SNMP

Performance Explorer offers you the possibility of selecting the object identifiers (OIDs) of management information bases (MIBs) from several sources for use as data sources for server monitoring. For example, database systems like Oracle offer extensive performance data for collection through an SNMP protocol.

1. Click **Monitor Server** on the Performance Explorer workflow bar. The **Data Source Wizard** opens in the **Select Data Sources** dialog box.
2. Click the **Select from predefined Data Sources** option button and click **Next**. The **System Selection** dialog box opens.
3. In the **System** menu tree, expand the **Custom Data** folder.
4. Select **SNMP Data**, then click **Next**. The **Connection Parameters** dialog box opens.
5. In the **Connection parameters** area, specify connection parameters such as the host name or IP address of the appropriate server system, the port number, and other data required to connect to the data source.

You must possess access rights to the specified server, and the SNMP service must be configured properly on the server system.

6. Click **Next**. The **Add SNMP Measures** dialog box opens
7. Select one of the precompiled MIBs delivered with Performance Explorer or click **Compile New MIB** to use an MIB that was shipped with the server system.  
To use the contents of an MIB with Silk Performer, the MIB must be compiled first. A number of common MIBs are delivered in a precompiled format. You can compile a new MIB if necessary.  
After you select a compiled MIB, its contents are displayed in a menu tree.
8. Click the OID in which you are interested and click **Get Value(s)**. Connection to the server is established, and the actual data item is shown.
9. Click the corresponding data item and then click **Add**.
10. Repeat the previous two steps until you select all the OIDs you want.
11. Click **Close**. The **Select Displayed Measures** dialog box opens
12. Select the factors to be monitored and click **Finish**. A Monitor graph displays the specified elements in a real-time, color-coded, display of server performance.

### Compiling a New MIB

Vendors of hardware and software systems provide information about the nature of their systems by way of management information bases (MIBs). This information can be retrieved with the help of the SNMP protocol by specifying the appropriate object identifier (OID). MIB browsers assist you in selecting the appropriate OID stored in an MIB.

1. In the **Add SNMP Measure** dialog box, click **Compile New MIB** for an MIB that has been shipped with the server system. The **Compile New MIB** dialog box opens.
2. Browse for the MIB file you want to compile and click **Open**.
3. Click **Compile**. If the compilation succeeds, a new name is added to the list of compiled MIBs.
4. If the message **Could not IMPORT from <name>-MIB** appears, compile the MIB to which error message refers.

- Click **Close**. The new MIB can now be used to monitor server performance through SNMP.

### Defining an SNMP Data Source

To define a new data source, do not edit the `REALTIME.INI` file. If you do, you will lose your changes when you next upgrade Silk Performer. To avoid this, create a new file in the sub-folder `<installation path>\Silk\Silk Performer 15.0\Include\DataSourceWzD`, for example `myESNMP_realtime.ini`. This file must contain the `S=` and the `M=` lines of your data source definition.

By searching for “WebLogic” and “ESNMP” in the `REALTIME.INI` file, you will find the following entries:

```
S= Application Server\BEA WebLogic preconfigured\SNMP, ESNMP:BEA-WEBLOGIC-MIB,  
  
M= ESNMP:1.3.6.1.4.1.140.625.340.1.25, BEA WebLogic\JVM Runtime  
\HeapFreeCurrent, eAvgOnlyCounter, 0, kbytes, 0, 1, 2, 3, 4, 5, Current Heap  
Free, measureMatchName=Runtime Object  
Name;measureMatchOID=1.3.6.1.4.1.140.625.340.1.5;  
  
M= ESNMP:1.3.6.1.4.1.140.625.340.1.30, BEA WebLogic\JVM Runtime  
\HeapSizeCurrent, eAvgOnlyCounter, 0, kbytes, 0, 1, 2, 3, 4, 5, Current Heap  
Size, measureMatchName=Runtime Object  
Name;measureMatchOID=1.3.6.1.4.1.140.625.340.1.5;  
  
M= ESNMP:1.3.6.1.4.1.140.625.180.1.25, BEA WebLogic\Queue Runtime  
\ExecuteThreadCurrentIdleCount, eAvgOnlyCounter, 0, , 0, 1, 2, 3, 4, 5, Thread Idle Count, measureMatchName=Queue Runtime Object  
Name;measureMatchOID=1.3.6.1.4.1.140.625.180.1.5;  
  
M= ESNMP:1.3.6.1.4.1.140.625.180.1.35, BEA WebLogic\Queue Runtime  
\PendingRequestCurrentCount, eAvgOnlyCounter, 0, , 0, 1, 2, 3, 4, 5, Current Requests, measureMatchName=Queue Runtime Object  
Name;measureMatchOID=1.3.6.1.4.1.140.625.180.1.5;  
  
M= ESNMP:1.3.6.1.4.1.140.625.180.1.40, BEA WebLogic\Queue Runtime  
\ServicedRequestTotalCount, eAvgOnlyCounter, 0, , 0, 1, 2, 3, 4, 5, Total Requests, measureMatchName=Queue Runtime Object  
Name;measureMatchOID=1.3.6.1.4.1.140.625.180.1.5;  
  
M= ESNMP:1.3.6.1.4.1.140.625.430.1.50, BEA WebLogic\WebApp Component Runtime  
\OpenSessionsCurrentCount, eAvgOnlyCounter, 0, , 0, 1, 2, 3, 4, 5, Open sessions, measureMatchName=Runtime Object  
Name;measureMatchOID=1.3.6.1.4.1.140.625.430.1.25;  
  
M= ESNMP:1.3.6.1.4.1.140.625.430.1.55, BEA WebLogic\WebApp Component Runtime  
\OpenSessionsHighCount, eAvgOnlyCounter, 0, , 0, 1, 2, 3, 4, 5, Session High Count, measureMatchName=Runtime Object  
Name;measureMatchOID=1.3.6.1.4.1.140.625.430.1.25;  
  
M= ESNMP:1.3.6.1.4.1.140.625.550.1.25, BEA WebLogic\Execute Queue  
\ThreadCount, eAvgOnlyCounter, 0, , 0, 1, 2, 3, 4, 5, Thread Count, measureMatchName=Queue Object  
Name;measureMatchOID=1.3.6.1.4.1.140.625.550.1.5;
```

The first line, which begins with `S=`, is the data source definition header. You only need one definition header per data source. So if you want to enhance the existing WebLogic ESNMP data source, you can leave the data source definition header as is.

Subsequent lines are for the individual measures:

- Following `M=` is the OID that is to be queried, without the instance ID (for example, `1.3.6.1.4.1.140.625.190.1.25`).
- Following that is the name of the measure as it will appear in the GUI. Note that back slashes are used to create the hierarchy (for example, `BEA\JDBC Connection Pool\Active Connections`).
- Following that are standard entries: `measure class`, `scale`, `unit`, `scale factor`, and five unused fields (for example, `eAvgOnlyCounter, 0, , 0, 1, 2, 3, 4, 5`).

- The next field is a description for tool-tip help in Performance Explorer. You might write, Current count of active connections in the JDBC connection pool runtime. The measure name itself can be shorter.
- The last entry is divided into two parts:
  - omeasureMatchName= This is used in the data source wizard to describe the instances (for example, JDBC Connection Pool Runtime Object Name).
  - omeasureMatchOID= This is the OID of the table that contains the instance names (for example, 1.3.6.1.4.1.140.625.190.1.15).



**Note:** The final measure name consists of the measure name given here (BEA WebLogic\JDBC Connection Pool\Active Connections) and the instance name in parenthesis.

So the key for the measure should appear as follows:

```
M= ESNMP:1.3.6.1.4.1.140.625.190.1.25, BEA WebLogic\JDBC Connection Pool
\Active Connections, eAvgOnlyCounter, 0, , 0, 1, 2, 3, 4, 5, Current count of
active connection in the JDBC connection pool runtime, measureMatchName=JDBC
Connection Pool Runtime Object
Name;measureMatchOID=1.3.6.1.4.1.140.625.190.1.15;
```

## Secure Shell Performance Monitoring

Use Performance Explorer to retrieve performance metrics by using a secure shell. For example, retrieve CPU-utilization or memory-utilization statistics from a SunOS system. Performance Explorer is shipped with a highly configurable performance-monitoring project that allows for the scripting of any monitors by using the secure shell. This process is done by driving a Silk Performer project inside Performance Explorer. This capability also allows you to keep track of system performance in real time.

The Silk Performer project is located in a SilkEssential package named Ssh.sep, which is located at C:\Program Files\Silk\Silk Performer 21.0\Monitors.

### Monitoring Performance by Using a Secure Shell

The system that you want to monitor by using a secure shell must be running a secure shell daemon. Make sure the monitored user has permission to log on multiple times.

- On the **Real-Time Monitoring** tab, in the **Monitor** group, click **System**. The **Data Source** wizard opens.
- Click the **Select from predefined Data Sources** option button.
- Click **Next**. The **System selection** page opens.
- Expand the **Miscellaneous** folder and click **Secure Shell**.
- Click **Next**.

The **Connection parameters** page opens.

- In the **Hostname** text box, specify the host that you want to monitor.
- Click **Next**. The **Attributes Configuration** page opens.
- Define the following monitoring specific attributes:
  - Username** – The user to be used when logging on to a remote system by using a secure shell.
  - Password** – The user's password.
  - Command(x)** – The preconfigured package allows for retrieving up to five performance measures by using secure shell as the following table shows:

Attribute	Type	Value
Command1	string	ps -ef   egrep -c \".*\"

Attribute	Type	Value
Command1.Active	boolean	true
Command1.Column	number	1
Command2	string	mpstat   head -n 2   tail -1
Command2.Active	boolean	true
Command2.Column	number	13

9. Click **OK**. The **Select displayed measures** page opens.

10. Check the check boxes for those measures that you want to include in the initial monitor view and then click **Finish**.

A connection to the specified host is established, and an initial view that contains the measures you selected is displayed.

### Example Command2

This example works for a SunOs system. The following command is sent to the server.

```
mpstat | head -n 2 | tail -1
```

This command returns a one-line response in the following format.

```
0 0 0 0 228 25 106 1 0 0 0 219 0 0 0 99
```

The attribute `Command2.Column` specifies which column to pass on to Performance Explorer for keeping track of the performance counter. In this case, it is column 13, which is the percentage of user time on a SunOs system.

`Command2.Active` specifies only whether this measure must be collected.

### Example Command1

The following command is sent to the server.

```
ps -ef | egrep -c "\.*\"
```

This command counts the number of processes running on a SunOs system and could return a one-line response in the following format.

```
87
```

Therefore, one line is returned with exactly one column.

The attribute `Command1.Column` specifies the column to forward to Performance Explorer. In this case, it is column 1.

`Command1.Active` specifies only whether this measure must be collected.

## Generating BDL Monitoring Projects

Silk Performer projects, or more precisely BDL scripts, can be used to collect performance data for monitored systems. This section outlines the steps that are required to assemble BDL monitoring projects for use with Performance Explorer and thereby enable real-time display of collected data.

### Silk Performer BDL Monitoring Projects

The Performance Data Collection Engine (PDCE) processes performance data collected by BDL scripts. Performance data collected with such scripts can be displayed in Performance Explorer in real-time. Entities collected by the PDCE are called measures, which may be specified in BDL using the following functions:

- MeasureInc
- MeasureIncFloat
- MeasureStart & MeasureStop
- MeasureSetTimer
- MeasureSet

These functions are called within transactions. One-to-many measures are allowed within each transaction. Furthermore, monitoring projects can use exactly one script defining exactly one usergroup. This usergroup may define one `init` transaction, one `end` transaction and one `main` transaction.

Wrapper functions, defined in `bdlMonitor.bdh`, are used to deliver additional functionality that is necessary for BDL realtime monitoring. See the Supplemental tutorials for information on using these functions:

- MonitorInc
- MonitorIncFloat
- MonitorStart and MonitorStop
- MonitorSet
- MonitorSetFloat
- MonitorSetTimer

## Creating a BDL Monitoring Project

1. Start Silk Performer and create a BDL monitoring project.
  - a) Click the **Outline Project** button on the Workflow bar. The **Workflow - Outline Project** dialog box opens.
  - b) Enter a name for the project in the **Name** text box and an optional description in the **Description** text box.
  - c) From the **Application Type** list, select **Monitoring > Bdl Monitor for Performance Explorer**.
  - d) Click **Next**.
2. On the **Model Script** dialog box, select the **Open existing script** option button.
3. Select the script `BdlMonitorSample.bdf` and click **OK**.

This brings up a pre-configured monitoring script that can be used as a template.

4. Select **Project > Project Attributes** to define the project attributes.

Each measure exported to Performance Explorer requires a fixed set of project attributes.

To be viewed with Performance Explorer, each measure requires at least these three project attributes:

- **Name:** Name to be shown in Performance Explorer
- **Type:** An average value or a cumulative value.
- **Enabled:** Reserved. Always set to true.

5. Review the sample monitor script

As specified in the project attributes, the project exports two measures that can be viewed in Performance Explorer in real-time.

Look for `MonitorInc` and `MonitorIncFloat` in the `TMon` transaction. This is where the last snapshot is handed over to Performance Explorer.

```
use "bdlMonitor.bdh"

const
    nMeasure1 := 1;
    nMeasure2 := 2;

dclrand
    rRand    : RndExpN  (1..10000: 100.000000);
    rRandf   : RndExpF  (5.500000);
```

```

var
  fScale : float;

dcluser
  user
    VMon
  transactions
    TInit          : begin;
    TMon           : 1;
    TEnd           : end;

dcltrans
  transaction TInit
begin
  fScale := AttributeGetDouble("scale");
end TInit;

transaction TMon
var
  n : number;
  f : float;
begin
  n := rRand;
  MonitorInc(1, n);

  f := rRandf;
  f := f * fScale;
  MonitorIncFloat(2, f);
end TMon;

```

**6.** Edit the .conf file.

The sample .conf file is available in the **Project** tree under the **Data Files** node. Double-click the .conf file to open it and change the value of the **Type** entry. This indicates where in the Performance Explorer hierarchy the project is to be located. For example, use <Type>Monitoring \Sample Project</Type> to locate the monitoring project in Performance Explorer under **Monitor > Add DataSource > Predefined data sources** at the location **Monitoring\Sample Project**.

**7.** Export the project to a single ZIP archive.

Choose **File > Export Project**. Check the **Zip to single archive file** check box.

For the **Export location**, set C:\Program Files\Silk\Silk Performer <version>\Monitors \BdlMonitorSample.sep.



**Note:** Save the file with an .sep extension.

**8.** Start a Realtime BDL Monitoring Project

- Start Performance Explorer. On the **Monitoring** tab, in the **Monitor** group, click **System**. Select the **Select from predefined Data Sources** option button and click **Next**.
- Select the newly created monitoring project (for example, **Monitoring > Sample Project**) and click **Next**.
- Enter the name of the host that is to be monitored (select **localhost** or the host recommended by Performance Explorer and click **Next**).  
This brings up a dialog on which project attributes can be modified (only one attribute can be modified in this instance).
- Enter a value greater than 0. This value will be used in the monitoring project. See the sample script for details.
- Click **OK**. This brings up a choice of measures specified in the BDL monitoring project.
- Select both measures and click **Finish**. This launches the monitoring project. The values can now be seen in real-time using a monitoring graph.

## **Creating a Silk Performance Manager Infrastructure Monitor**

1. Click **File > New Project** or click **Start here** on the workflow bar if no project is open. The **Workflow - Outline Project** dialog opens.
2. Enter a **Name** and optionally a **Description**.
3. From the list of application types, select **Monitoring > Performance Manager - Infrastructure Monitor** and click **Next**. Performance Explorer opens and the **Data Source Wizard** displays.
4. Follow the wizard to select a data source. A monitor chart displays and the monitoring automatically starts.
5. On the **Real-Time Monitoring** tab, in the **Export** group, click **As Project**.
6. Follow the **Reuse Monitor Wizard** to export the monitor chart.
7. Specify the settings on the **Upload Project** dialog and click **OK**.

## **Performance Data Collection Engine (PDCE)**

The Performance Data Collection Engine (PDCE) is a module inside the Performance Explorer engine that is used for querying monitoring data in real-time. Its functionality can alternately be accessed through the Benchmark Description Language (BDL). The BDL interface for the PDCE is described in the `pdce.bdh` include file.

Each of the measurements that Performance Explorer monitors can alternately be monitored via Silk Performer using BDL scripts. After registering measurements of interest, a monitoring data set can be retrieved from the PDCE via the provided polling functions (PDCE API). These functions provide data, status information, and descriptions of the measurements (if available).

## **Tutorial**

This tutorial consists of two use cases: (1) a simple use case to get you started and (2) an advanced use case. The first use case involves a monitor project that consists of one user group with a single transaction and a single measure.

The second use case illustrates how to have several user groups, transactions, and measures in a single project.

### **Basic Use Case**

In this use case you will create a project that keeps track of the number of processes running on a SunOs. SunOs can usually be accessed through remote execution. Compare with the Win2000 command line tool rexec. To count the number of processes running on a SunOs, execute `'ps -ef | egrep -c ".*"'` within an X-Terminal, Telnet session, rexec, or other preferred tool. For example, at a DOS prompt type: `c:\>rexec yourSunHost -l username "ps -ef | egrep -c \".*\""`

This returns the number of processes running on your SunHost. The goal here is to continuously track and display this value in a Performance Explorer real-time chart.

### **Creating a New Project**

Create a new project of application type **Monitoring/Bdl Monitor** for Performance Explorer as explained earlier in this section. Enter a simple project description that reflects the overall purpose of the monitoring project to be displayed in Performance Explorer - for example, "A powerful project used to collect the number of running processes on SunOs systems."

### **Planning Project Attributes**

Begin by entering a name for the measure in the project attributes, for example `CountNrOfProcesses`. This name will also be used in the Performance Explorer hierarchy. A hierarchy is introduced with '`\`'.

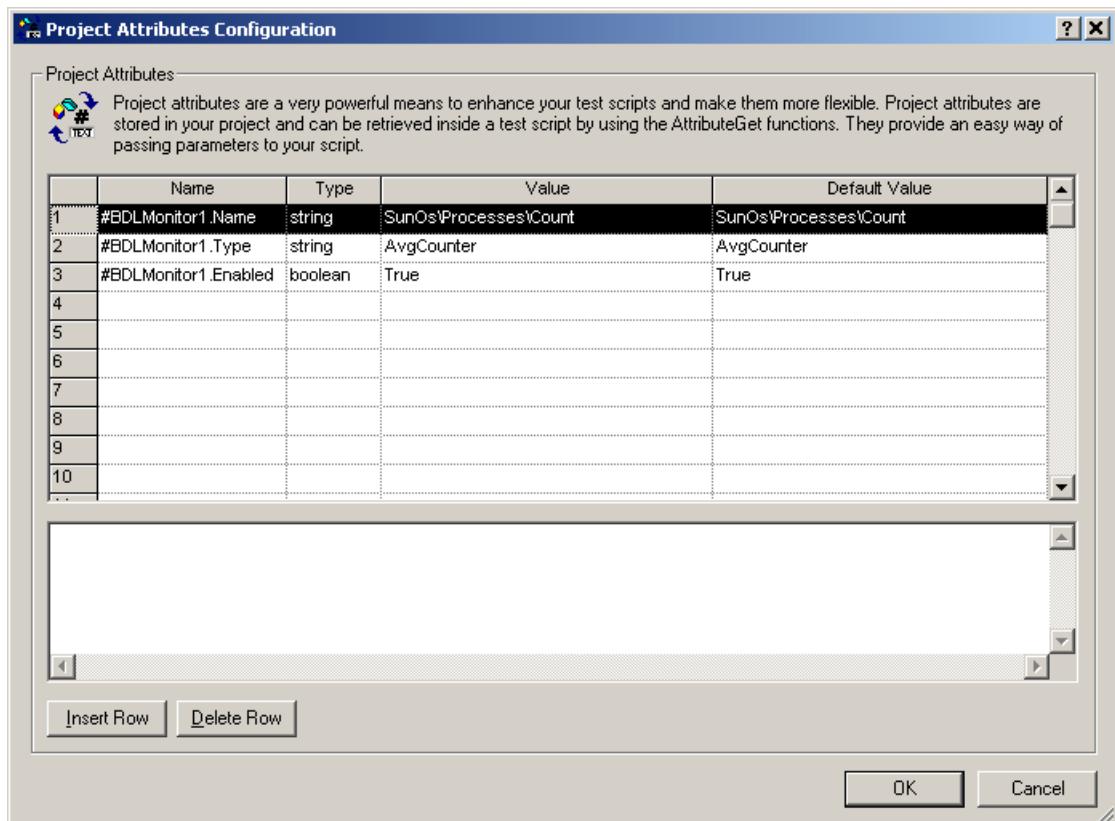
For example, to create this hierarchy:

```
OS
  Processes
    CountNrOfProcesses
    Another Counter
```

Specify the name of the measure as SunOs\Processes\CountNrOfProcesses.

This will be an AvgCounter (for average) counter and it should be collected by default.

These values must be transferred into the project's attributes. Open the attribute editor by choosing **Project > Project Attributes**.



The values in the project attributes identify a single measure. Several attributes (lines in the dialog) may be required to create a measure however. For example, all entries beginning with #BDLMonitor1 in the above example. This is how settings are defined for a measure.

The measure is now defined with a name, a type, and the setting Enabled. A second measure will begin with #BDLMonitor2 (covered in the Advanced Use Case).

### *Creating a BDL Monitoring Script*

To remotely execute certain command line tools in BDL, for example "ps" on a SunOs, three functions are required:

- Connect to a remote machine's "exec" server.
- Send the command, to be executed on the remote machine.
- Close the connection

```
// hCon will be assigned the connection handle
function Connect(/*inout*/hCon : number; sHost : string)
begin
```

```

    WebTcpipConnect(hCon, sHost, 512);
end Connect;

// Send a request to the remote execution server
// remote execution protocol:
// What does a request look like in binary:
// 0username0password0shellCommandToExecute00
// What does the response look like
// 0responseData
// sample request:
//    0root0labpass00ps -ef | egrep -c ".*"00
function Request(hCon: number; sUser: string; sPwd: string;
                  sCmd: string):number
var
  sSend : string;
  sBuf  : string;
  nSize : number;
  nRec  : number;
begin
  sSend := "\h00";
  SetString(sSend, 2, sUser);
  SetString(sSend, Strlen(sUser) + 3, sPwd);
  SetString(sSend, Strlen(sUser) + Strlen(sPwd) + 4, sCmd);

  nSize := 3 + Strlen(sUser) + Strlen(sPwd)
           + Strlen(sCmd) + 1;

  WebTcpipSendBin(hCon, sSend, nSize);
  WebTcpipRecvExact(hCon, NULL, 1);
  WebTcpipRecv(hCon, sBuf, sizeof(sBuf), nRec);
  Request := number(sBuf);
end Request;

// Closes the connection to the remote exec server
function Close(hCon : number)
begin
  WebTcpipShutdown(hCon);
end Close;

```

A function wrapper is needed around the Silk Performer MeasureInc functions. This function can be used in all monitoring projects. A function named MonitorInc is created to access project attributes. This function accesses the attributes you specified earlier.

The MonitorInc function can also be imported from an existing bdh, bdlMonitor.bdh.

```

function MonitorInc(nMon : number; nVal : number)
var
  sMeasure : string;
begin
  // This will check whether the attribute
  // "#BDLMonitor1.Enabled" was set to true
  if AttributeGetBoolean("#BDLMonitor" + string(nMon)
                           + ".Enabled") then
    // If yes then let's read the name of the measure.
    // To do this we read the the project attribute
    // "#BDLMonitor1.Name" and store it
    // to a local variable named sMeasure.
    // sMeasure will have the value:
    // "SunOs\Processes\CountNrOfProcesses"
    AttributeGetString("#BDLMonitor" + string(nMon)
                           + ".Name", sMeasure, sizeof(sMeasure));

    // Set a new value for
    // "SunOs\Processes\CountNrOfProcesses"

```

```

    MeasureInc( sMeasure, nVal, MEASURE_KIND_AVERAGE );
end;
end MonitorInc;

```

Now the transaction that will take the snapshot using all the functions that have been defined can be coded. This transaction also accesses the project file attributes. The goal is to later have these attributes set in Performance Explorer. For now however, to ensure that the script works, four attributes need to be added to the project attributes.



**Note:** Attribute names are case sensitive.

- host: Assign it a sample value, for example sunserver
- command: Assign it a sample value, for example ps -ef | egrep -c ".\*"
- user: Assign a sample value, for example root
- password: Assign a sample value for testing purposes

Open the project attributes editor by choosing **Project > Project Attributes** and add these additional attributes. All are of type string except for the attribute password which is type password. Assign values to the attributes for testing purposes. Choose a description for each attribute that conveys the purpose of the attribute.

```

const
  nMeasure := 1;

dcluser
  user
    VMonitor
  transactions
    TSnap : 1;

dclfunc
.... // your functions here

dcltrans
  transaction TSnap
  var
    hCon : number init 0;
    sHost : string;
    sCmd : string;
    sUser : string;
    sPwd : string;
    nVal : number;
begin
  AttributeGetString( "host", sHost, sizeof(sHost));
  AttributeGetString( "command", sCmd, sizeof(sCmd));
  AttributeGetString( "user", sUser, sizeof(sUser));
  AttributeGetString( "password", sPwd, sizeof(sPwd));

  Connect(hCon, sHost);
  nVal := Request(hCon, sUser, sPwd, sCmd);
  MonitorInc(nMeasure, nVal);
  Close(hCon);
end TSnap;

```

The project now consists of the newly created script. Save the project and verify that it works by initiating a TryScript run. Have nVal printed or written to a log file to verify that the script works. If the script works, save and close the project.

### *Packaging the BDL Monitoring Script*

To export the project as a single ZIP file, go to **File > Export Project**. Check the **zip to single archive** check box and export the file to C:\Program Files\Silk\Silk Performer 10.0\Monitors \CountProcess.sep. (Note the .sep extension).

Before the .sep file can be used with Performance Explorer the .conf file for the .sep must be modified. The .conf file looks like this:

```
<?xml version='1.0' encoding='UTF-8'?>
<Project>
  <Type>Sample\Remote</Type>
  <Copyright>Borland</Copyright>
  <Author>Borland</Author>
  <Version>5.1</Version>
</Project>
```

For the type setting, specify a hierarchy separated by "\". This hierarchy will be reflected in Performance Explorer.

### *Using the Monitoring Project inside Performance Explorer*

1. Launch Performance Explorer. On the **Real-Time Monitoring** tab, in the **Monitor** group, click **System**. Select the **Select from predefined Data Sources** option button and click **Next**.
2. Select the newly created monitoring project and click **Next**.
3. Specify the host that is to be monitored, preferably a machine running SunOS. Click **Next** to bring up a dialog for setting project attributes. This dialog displays the attributes that you set (except for the host which was specified previously).

Therefore, the dialog displays the following attributes:

- user
- password
- command

4. Select the measures (in this case only a single measure) and click **OK**. The measures are then immediately handed over to the data collection mechanism.

### *Advanced Use Case*

This use case includes a project with two user groups, the user group from the previous use case and a new user group that measures HTTP requests. The two URLs and host are configurable through project attributes.

### *Creating a Silk Performer Project*

Enter a simple project description that is to be displayed in Performance Explorer. The description should describe the overall purpose of the monitoring project. For example, a powerful project used for collecting the number of running processes on SunOS systems and measuring two configurable HTTP requests.

### *Planning Project Attributes*

For this tutorial you will plan three measures. Pick measure names that indicate the hierarchy and purpose of the measures. In this example, a measure counts the number of processes on a SunOS system and includes two measures that measure HTTP requests (Performed using WebURL):

- SunOs\Processes\CountNrOfProcesses
- SunOs\Http Server\URI1
- SunOs\Http Server\URI2

The intended hierarchy will then look like the following:

```

SunOs
  Processes
    CountNrOfProcesses
  Http Server
    URI1
    URI2

```

A BDL script with two user groups and two transactions is written. See the table below to see how measures are assigned to transactions and how transactions relate to user groups.

User Group	Transaction	Measure
VMonitor	TSnap	CountNrOfProcesses
VWebMon	TWebSnap	URI1
VWebMon	TWebSnap	URI2

With projects that include numerous transactions and/or user groups, measures must be specified with the corresponding user group and assigned measure. This differs from the basic use case outlined previously, which involved only one transaction and one user group.

Open the project attributes editor at **Projects > Project Attribute** and enter the following data:

Name	Type	Value
#BDLMonitor1.Name	string	SunOs\Processes\\CountNrOfProcesses
#BDLMonitor1.Type	string	AvgCounter
#BDLMonitor1.Enabled	boolean	True
#BDLMonitor1.Script	string	remote.bdf
#BDLMonitor1.Usergroup	string	VMonitor
#BDLMonitor1.Transaction	string	TSnap
#BDLMonitor2.Name	string	SunOs\Http Server\URI1
#BDLMonitor2.Type	string	AvgCounter
#BDLMonitor2.Enabled	boolean	true
#BDLMonitor2.Script	string	remote.bdf
#BDLMonitor2.Usergroup	string	VWebMon
#BDLMonitor2.Transaction	string	TWebSnap
#BDLMonitor3.Name	string	SunOs\Http Server\URI2
#BDLMonitor3.Type	string	AvgCounter
#BDLMonitor3.Enabled	boolean	True
#BDLMonitor3.Script	string	remote.bdf
#BDLMonitor3.Usergroup	string	VWebMon
#BDLMonitor3.Transaction	string	TWebSnap
host	sunserver	sunserver
command	string	ps -ef   egrep -c ".*"
user	string	root
password	password	*****

Name	Type	Value
URI1	string	/
URI2	string	/manual/ibm/index.html

### *Creating a BDL Monitoring Script*

To remotely execute certain command line tools in BDL, for example "ps" on a SunOs, three functions are required:

- Connect to a remote machine's "exec" server.
- Send the command, to be executed on the remote machine.
- Close the connection

```
// hCon will be assigned the connection handle
function Connect(/*inout*/hCon : number; sHost : string)
begin
    WebTcpipConnect(hCon, sHost, 512);
end Connect;

// Send a request to the remote execution server
// remote execution protocol:
// What does a request look like in binary:
// 00username00password00shellCommandToExecute00
// What does the response look like
// 00responseData
// sample request:
//    00root00labpass00ps -ef | egrep -c ".*"00
function Request(hCon: number; sUser: string; sPwd: string;
                  sCmd: string):number
var
    sSend : string;
    sBuf  : string;
    nSize : number;
    nRec  : number;
begin
    sSend := "\h00";
    SetString(sSend, 2, sUser);
    SetString(sSend, Strlen(sUser) + 3, sPwd);
    SetString(sSend, Strlen(sUser) + Strlen(sPwd) + 4, sCmd);

    nSize := 3 + Strlen(sUser) + Strlen(sPwd)
           + Strlen(sCmd) + 1;

    WebTcpipSendBin(hCon, sSend, nSize);
    WebTcpipRecvExact(hCon, NULL, 1);
    WebTcpipRecv(hCon, sBuf, sizeof(sBuf), nRec);
    Request := number(sBuf);
end Request;

// Closes the connection to the remote exec server
function Close(hCon : number)
begin
    WebTcpipShutdown(hCon);
end Close;
```

A function wrapper is needed around the Silk Performer MeasureInc functions. This function can be used in all monitoring projects. A function named MonitorInc is created to access project attributes. This function accesses the attributes you specified earlier.

The MonitorInc function can also be imported from an existing bdh, bdlMonitor.bdh.

```
function MonitorInc(nMon : number; nVal : number)
var
```

```

    sMeasure : string;
begin
    // This will check whether the attribute
    // "#BDLMonitor1.Enabled" was set to true
    if AttributeGetBoolean( "#BDLMonitor" + string(nMon)
        + ".Enabled" ) then
        // If yes then let's read the name of the measure.
        // To do this we read the project attribute
        // "#BDLMonitor1.Name" and store it
        // to a local variable named sMeasure.
        // sMeasure will have the value:
        // "SunOs\Processes\CountNrOfProcesses"
        AttributeGetString( "#BDLMonitor" + string(nMon)
            + ".Name", sMeasure, sizeof(sMeasure) );

        // Set a new value for
        // "SunOs\Processes\CountNrOfProcesses"
        MeasureInc( sMeasure, nVal, MEASURE_KIND_AVERAGE );
    end;
end MonitorInc;

```

Now the transaction that will take the snapshot using all the functions that have been defined can be coded. This transaction also accesses the project file attributes. The goal is to later have these attributes set in Performance Explorer. For now however, to ensure that the script works, four attributes need to be added to the project attributes.



**Note:** Attribute names are case sensitive.

- host: Assign it a sample value, for example sunserver
- command: Assign it a sample value, for example ps -ef | egrep -c "./\*"
- user: Assign a sample value, for example root
- password: Assign a sample value for testing purposes

Open the project attributes editor by choosing **Project > Project Attributes** and add these additional attributes. All are of type string except for the attribute password which is type password. Assign values to the attributes for testing purposes. Choose a description for each attribute that conveys the purpose of the attribute.

```

const
    nMeasure := 1;

dcluser
    user
        VMonitor
    transactions
        TSnap : 1;

dclfunc
.... // your functions here

dcltrans
    transaction TSnap
    var
        hCon : number init 0;
        sHost : string;
        sCmd : string;
        sUser : string;
        sPwd : string;
        nVal : number;
begin
    AttributeGetString( "host", sHost, sizeof(sHost));
    AttributeGetString( "command", sCmd, sizeof(sCmd));
    AttributeGetString( "user", sUser, sizeof(sUser));

```

```

AttributeGetString( "password", sPwd, sizeof(sPwd) );

Connect(hCon, sHost);
nVal := Request(hCon, sUser, sPwd, sCmd);
MonitorInc(nMeasure, nVal);
Close(hCon);
end TSnap;

```

The project now consists of the newly created script. Save the project and verify that it works by initiating a TryScript run. Have `nVal` printed or written to a log file to verify that the script works. If the script works, save and close the project.

### *Packaging the Project*

Export the project to a single ZIP file. Go to **File > Export Project**. Check **Zip to single archive** and export the file to `C:\Program Files\Silk\Silk Performer\Monitors\Advanced.sep` (note the SEP file extension).

Before the SEP file can be used with Performance Explorer, the CONF file for the SEP must be modified. The CONF file appears as:

```

<?xml version='1.0' encoding='UTF-8'?>
<Project>
  <Type>Sample\Advanced\HttpHit</Type>
  <Copyright>Borland</Copyright>
  <Author>Borland</Author>
  <Version>5.1</Version>
</Project>

```

For the type setting, specify a hierarchy that is to be reflected in Performance Explorer, separated by backward slashes ("\").

### *Using the Monitoring Project in Performance Explorer*

1. Start Performance Explorer. On the **Real-Time Monitoring** tab, in the **Monitor** group, click **System**. Choose **Select from predefined Data Sources**.
2. Select the newly created monitoring project and click **Next**.
3. Enter the name of the server that is to be monitored. For example, enter `sunserver` and click **Next**. A dialog box appears. Define the following attributes:
  - **command**: As with the basic test case you set up earlier in this tutorial, a command that counts the number of processes.
  - **user**: A valid user account for accessing the remote machine.
  - **password**: The password.
  - **URI1**: The URI where response time is to be measured.
  - **URI2**: A second URI where response time is to be measured.
4. Select the measures (in this case there is only one) and click **OK**. The measures are immediately handed over to the data collection mechanism. The choice corresponds to the hierarchy defined in the project attributes.

### **Best Practices**

This section outlines suggestions for working with real-time monitoring projects as relates to the writing of monitoring scripts.

#### *Loops*

Performance Explorer executes transactions that run snapshots at regular intervals. For example, a snapshot transaction might be executed every 10 seconds. In such a case, Performance Explorer assumes

that the entire transaction is executed within this one interval. Otherwise Performance Explorer returns a warning stating that it can not collect the interval's snapshot.

Do not use endless loops such as the following:

```
// Forbidden
transaction TSnap
begin
    while true do
        Snap();
    end;
end TSnap;
```

#### *ThinkTimes and Wait*

Be careful with wait statements. Think-times are ignored. A monitoring transaction must be executed within a given interval. Wait statements block execution and transactions may not return on time.

#### **Example**

```
// Forbidden
transaction TSnap
begin
    wait 500000.0;
end TSnap;
```

#### *Initialization*

Do not place function calls in TInit transactions that may stop runtime. For example, OraLogon may generate a ProcessExit on a failed logon. Performance Explorer will not catch this message and will assume that the project is still running. Of course, Performance Explorer will report that it cannot gather data, but it will not be able to report why this is happening.

#### *Other Best Practices*

##### **Name Length**

Attribute value lengths must not exceed 79 characters.

##### **Performing a Try Script run for the Project**

Execute a Try Script run to verify that the project is working before you create an SEP file.

##### **Special Project Attributes**

The monitoring-project attribute host can be accessed within BDL scripts using the following function call:  
AttributeGetString("host", sHost, sizeof(sHost));

Instead of host, an attribute called #MonitorHost may also be queried. Those two attributes are treated equally.

In Performance Explorer the value for this attribute is set using the following dialog:

## **Monitoring Data Sources**

Before you can start monitoring a system with Performance Explorer, you need to specify the system and the measures. There are two ways to do so:

- You can select a predefined data source.

Performance Explorer provides predefined data sources for the most popular web servers, application servers, database servers, and operating systems. You are not restricted to using only these data

sources. These data sources are included to speed the selection process for measures that are to be monitored.

- You can use the data source scanner to detect data sources.

The Data Source Scanner checks your system for a variety of data sources that are commonly found on a range of platforms and returns a list of available data sources.

To specify a system, click **System** in the **Monitor** group on the **Real-Time Monitoring** tab and follow the wizard.

## Predefined Data Sources

Performance Explorer provides a wizard that guides you through the setup process for each server type that Performance Explorer can monitor.

### Enabling Monitoring for Apache Servers

With Apache Web servers, you have the option of using the built-in status report functionality. This functionality requires that a `mod_status` module be built in, which is done by default.

1. Click the **ExtendedStatus On** option button to view more detailed information.
2. Add code to the configuration file `access.conf` to enable status reports only for browsers from the `.foo.com` domain.  
The Apache configuration is maintained in a set of files typically located in `/usr/local/apache/etc` or `/usr/local/apache/conf` (Solaris) or `/etc/httpd/conf` (Linux).
3. Add the following code to the main Apache configuration file, `httpd.conf`.

```
<Location /server-status> SetHandler server-status  
order deny, allow deny from all allow from .foo.com  
</Location>
```

If this code already exists but is commented out, un-comment it.



**Note:** For monitoring to function appropriately, all `allow` statements must be correct.

4. Restart your server.
5. To verify the server's performance module, type the URL `http://<hostname>/server-status?auto` into your browser. Monitoring is enabled if you receive a response like the following message:  
Total Accesses: 210 Total kBytes: 94 CPULoad: .000278279 Uptime: 366539 ReqPerSec: .000572927 BytesPerSec: .262608 BytesPerReq: 458.362 BusyServers: 1 IdleServers: 6 Scoreboard:  
W.....



**Note:** The default Apache monitor works only if Telnet is enabled. To monitor an Apache server with Telnet disabled, use the `wGet` command instead of the default `Rexec` commands, as the following example shows.

```
wget -q -O - http://<hostname>/server-status | grep  
"requests/sec" | cut -c5- | cut -d\ -f1
```

### Enabling SNMP Support for Silverstream Servers

1. Run SilverStream designer.
2. Choose **SilverMaster3 database > Objects**.
3. Open `AgSNMPGetStats` in design mode.
4. Double-click the trigger icon. The **Properties** page opens.
5. Set the **Enabled** option to **true**.
6. Save the changes and close the file.

## Monitoring Silk Central Performance

Use Performance Explorer to monitor performance and reliability metrics of a Silk Central application server and front-end server.

1. On the **Real-Time Monitoring** tab, in the **Monitor** group, click **System**. The **Data Source** wizard opens.
2. Click the **Select from predefined Data Sources** option button.
3. Click **Next**. The **System selection** page opens.
4. Expand the **Silk** folder and the **Silk Central** folder with the version that you want to monitor, then select application server or front-end server.
5. Click **Next**.

The **Connection parameters** page opens.

6. Specify the following settings:

- In the **Hostname** text box, specify the host name or IP address on which Silk Central is running.
- *Optional:* In the **Alias** text box, specify an alias for the server which is monitored.
- If Silk Central does not listen on the default port, specify the JMX port number in the **Port** text box.
- If JMX authentication is enabled, enter the **Username** and **Password**. Per default JMX authentication is disabled and **Username** and **Password** should be empty.



**Note:** Make sure that your local firewall is turned off or that ports 19140 and 19142 are allowed to communicate through the firewall.

7. Click **Next**. The **JMX Data Source Browser** opens.

8. Select the measures that you want to include in the initial monitor view and click **Add**. When you are done adding the measures, click **Close**.



**Tip:** Select the **Simple Query** tab and filter the measures by `borland.com` to quickly access the relevant Silk Central measures.

The **Select displayed measures** page opens.

9. Check the check boxes for those measures that you want to include in the initial monitor view and then click **Finish**.

A connection to the specified host is established, and an initial view that contains the measures you selected is displayed.

## IBM DB2 Performance Monitoring

Use Performance Explorer to retrieve performance metrics from IBM DB2 database systems no matter the database platform. For example, the database can run on a SunOs, Linux, or a Windows system.

With Performance Explorer, you can monitor DB2 database systems in a platform-independent way. To accomplish this task, execute a DB2-specific command for retrieving a database's performance and utilization measures and drive a Silk Performer project inside Performance Explorer. This approach also allows Performance Explorer to keep track of the database system's performance in real time.

The Silk Performer project is located in a Essential package named `DB2Monitor.sep`, which is located at `C:\Program Files\Silk\Silk Performer 21.0\Monitors`.

### Prerequisites for DB2 Monitoring

To monitor DB2, ensure that your environment meets the following requirements:

- IBM DB2 client software (DB2 Connect) must be installed on the machine on which you intend to run Performance Explorer.
- The DB2 command line processor, `db2cmd.exe`, must be installed as part of the client installation. This prerequisite is met by default.

- The appropriate DB2 database user must possess one of the following authorizations:
  - sysadm
  - sysctrl
  - sysmaint
- DB2 snapshot monitoring must be set up correctly.

### *Testing DB2 Snapshot Monitoring*

#### 1. Connect to a DB2 database.

db2 command:

```
=> connect to sample user <dbusername> using <password>
```

#### 2. Attach to a DB2 instance.

db2 command:

```
=> attach to <db2instancename> user <dbusername> using <dbusername>
```

#### 3. Check if monitor switches are on.

db2 command:

```
=> get monitor switches
```

### **Monitor Recording Switches**

Switch list for node 0

```
Buffer Pool Activity Information (BUFFERPOOL) = ON 02-06-2002
18:27:48.722132
Lock Information (LOCK) = ON 02-06-2002
18:28:00.095212
Sorting Information (SORT) = ON 02-06-2002
18:28:12.263183
SQL Statement Information (STATEMENT) = ON 02-06-2002
18:28:24.323446
Table Activity Information (TABLE) = OFF
Unit of Work Information (UOW) = OFF
```

#### 4. Turn on switches.

db2 commands:

```
=> update monitor switches using bufferpool on
=> update monitor switches using lock on
=> update monitor switches using sort on
```

#### 5. Retrieve data from the DB2 UDB system monitor snapshot.

db2 command:

```
=> get snapshot for all databases
```

### **Database Snapshot:**

```
Database name = SAMPLE
Database path = /home/db2inst1/db2inst1/
NODE0000/SQL00001/
Input database alias =
Database status = Active
Catalog node number = 0
Catalog network node name =
Operating system running at database server= LINUX
Location of the database = Remote
First database connect timestamp = 02-06-2002 18:01:31.198883
Last reset timestamp =
Last backup timestamp =
Snapshot timestamp = 02-06-2002 18:37:39.254985
```

```
High water mark for connections = 1
Application connects = 1
Secondary connects total = 0
Applications connected currently = 1
Appls. executing in db manager currently = 0
Agents associated with applications = 1
Maximum agents associated with applications= 1
Maximum coordinating agents = 1
Locks held currently = 0
Lock waits = 0
Time database waited on locks (ms) = 0
Lock list memory in use (Bytes) = 792
Deadlocks detected = 0
Lock escalations = 0
Exclusive lock escalations = 0
Agents currently waiting on locks = 0
Lock Timeouts = 0
Total sort heap allocated = 0
Total sorts = 0
Total sort time (ms) = 0
Sort overflows = 0
Active sorts = 0
Buffer pool data logical reads = 0
Buffer pool data physical reads = 0
Asynchronous pool data page reads = 0
Buffer pool data writes = 0
Asynchronous pool data page writes = 0
Buffer pool index logical reads = 0
Buffer pool index physical reads = 0
Asynchronous pool index page reads = 0
Buffer pool index writes = 0
Asynchronous pool index page writes = 0
Total buffer pool read time (ms) = 0
Total buffer pool write time (ms) = 0
Total elapsed asynchronous read time = 0
Total elapsed asynchronous write time = 0
Asynchronous read requests = 0
LSN Gap cleaner triggers = 0
Dirty page steal cleaner triggers = 0
Dirty page threshold cleaner triggers = 0
Time waited for prefetch (ms) = 0
Direct reads = 0
Direct writes = 0
Direct read requests = 0
Direct write requests = 0
Direct reads elapsed time (ms) = 0
Direct write elapsed time (ms) = 0
Database files closed = 0
Data pages copied to extended storage = 0
Index pages copied to extended storage = 0
Data pages copied from extended storage = 0
Index pages copied from extended storage = 0
Host execution elapsed time = 0.000000
Commit statements attempted = 1
Rollback statements attempted = 0
Dynamic statements attempted = 0
Static statements attempted = 1
Failed statement operations = 0
Select SQL statements executed = 0
Update/Insert/Delete statements executed = 0
DDL statements executed = 0
Internal automatic rebinds = 0
Internal rows deleted = 0
Internal rows inserted = 0
```

```

Internal rows updated = 0
Internal commits = 1
Internal rollbacks = 0
Internal rollbacks due to deadlock = 0
Rows deleted = 0
Rows inserted = 0
Rows updated = 0
Rows selected = 0
Rows read = 9
Binds/precompiles attempted = 0
Log space available to the database (Bytes)= 20400000
Log space used by the database (Bytes) = 0
Maximum secondary log space used (Bytes) = 0
Maximum total log space used (Bytes) = 0
Secondary logs allocated currently = 0
Log pages read = 0
Log pages written = 0
Appl id holding the oldest transaction = 0
Package cache lookups = 0
Package cache inserts = 0
Package cache overflows = 0
Package cache high water mark (Bytes) = 51824
Application section lookups = 0
Application section inserts = 0
Catalog cache lookups = 0
Catalog cache inserts = 0
Catalog cache overflows = 0
Catalog cache heap full = 0
Number of hash joins = 0
Number of hash loops = 0
Number of hash join overflows = 0
Number of small hash join overflows = 0

```

#### *Retrieving IBM DB2 Performance Metrics*

Establish a connection to a data source to monitor IBM DB2 DBMS systems and to create an initial view that contains the measures you want to monitor.

1. On the **Real-Time Monitoring** tab, in the **Monitor** group, click **System**. The **Data Source** wizard opens.
2. Click the **Select from predefined Data Sources** option button.
3. Click **Next**. The **System selection** page opens.
4. Expand the **Database System** folder and select **IBM Universal Database DB2 (snapshot dll)**.
5. Click **Next**.

The **Connection parameters** page opens.

6. In the **Hostname** text box, specify the host on which the DB2 database is running.
7. Click **Next**. The **Attributes Configuration** page opens.
8. Define the following monitoring-specific attributes:

- **Alias** – The database alias is used by the monitoring project in the following DB2-specific command:  
"connect to 'alias' user 'user' using 'password'".
- **Instance** – The instance name is used by the monitoring project for executing the following DB2-specific command: "attach to 'Instance' user 'user' using 'password'".
- **User** – Specify a user with the following DB2 authorizations:
  - sysadm
  - sysctrl
  - sysmaint

- **Password** – The password for the user.
9. Click **OK**. The **Select displayed measures** page opens.
10. Check the check boxes for those measures that you want to include in the initial monitor view and then click **Finish**.

A connection to the specified host is established, and an initial view that contains the measures you selected is displayed.

### **IBM WebSphere Application Server Monitoring**

With Performance Explorer, you can monitor the following IBM WebSphere application servers:

- IBM WebSphere Application Server 6.1 via JMX
- IBM WebSphere Application Server 7.0 via JMX
- IBM WebSphere Application Server 8.0 via JMX
- IBM WebSphere Application Server 8.5 via JMX

#### *Typical Performance Measures on WebSphere Application Server 6.1*

The following table displays typical performance measures and definitions for WebSphere application server.

Performance Measure	Description
BeanModule	Specific data about deployed EJBs, such as how many beans were created.
ConnectionPoolModule	Contains performance counters about JDBC connections, like how many JDBC connections are currently in the pool.
JvmRuntimeModule	Java Virtual Machine specific performance counters.
ServletSessionModule	Contains performance counters about servlet sessions, such as how many sessions are active or the average session live time.
TransactionModule	Contains performance counters providing information about how many transactions are in progress or the average time required to execute one transaction.
WebApplicationModule	Provides information about deployed servlets and JSPs, such as the average time required for the servlet to process requests.

#### *Monitoring IBM WebSphere Application Server*

1. On the **Real-Time Monitoring** tab, in the **Monitor** group, click **System**. The **Data Source wizard** opens.
2. Click the **Select from predefined Data Sources** option button.
3. Click **Next**. The **System selection** page opens.
4. Expand the **Application Server** folder and the **IBM WebSphere Application Server** folder.
5. Click the **IBM WebSphere <version> (JMX MBean Server)** node and then click **Next**. The **Connection parameters** page opens.
6. In the **Hostname** field, specify the host on which WebSphere is running.
7. In the **User** and **Password** fields, specify a user with administrator permissions for the above specified host.

8. **Optional:** In the **Alias** field, specify an alias for the monitored server.
9. Click **Server Configuration...** The **JMX Connection Configuration** window opens.
10. Specify the **Java Home directory** and the **Application Server install directory**. Use a UNC path (for example, `\\\c$\IBM\WebSphere\AppServer\java`).



**Note:** If you cannot specify a UNC path, copy the application server installation directory to your local machine and specify a local path.

11. Specify additional JVM parameters based on WebSphere security settings.

JVM vendor	IBM	SUN
<b>Security settings</b>		
Disabled administrative security	Without additional JVM parameters	Without additional JVM parameters
Enabled administrative security	Additional JVM parameter: – <code>Dcom.ibm.CORBA.ConfigURL</code>	Additional JVM parameter: – <code>Dcom.ibm.CORBA.ConfigURL</code>
Inbound CSIV2 Transport Layer = TCP/IP	(already predefined)	(predefined)
Enabled administrative security	Additional JVM parameters: – <code>Dcom.ibm.CORBA.ConfigURL</code> (predefined) – <code>Dcom.ibm.SSL.ConfigURL</code>	Does not work
Inbound CSIV2 Transport Layer = SSL-supported		
Enabled administrative security	Additional JVM parameters: – <code>Dcom.ibm.CORBA.ConfigURL</code> (already predefined)	Does not work
Inbound CSIV2 Transport Layer = SSL-required		
	– <code>Dcom.ibm.SSL.ConfigURL</code>	



**Note:** If your WebSphere server has enabled administrative security and the inbound CSIV2 transport layer is SSL-supported (or SSL-required), copy the application server installation directory to your local machine. Specify **Additional JVM parameters** according to the table above. The **Additional JVM parameters** field holds the predefined parameter –

`Dcom.ibm.CORBA.ConfigURL`.

The value for the `-Dcom.ibm.CORBA.ConfigURL` parameter must be the URL path to the file `sas.client.props`. The default `sas.client.props` file is located in the folder `%INSTALL_DIR%\Include\jmx-config\WebSphere%MAJOR_VERSION%`. For example: For WebSphere 7, the URL path is `C:\Program%20Files%20(x86)\Silk\Silk%20Performer%2010.0\Include\jmx-config\WebSphere7`. If your server has specific settings, you can use the `sas.client.props` file from the properties folder of the WebSphere profile. For Example: file: `/C:\Program%20Files%20(x86)\IBM\WebSphere\AppServer1\profiles\AppSrv01\properties\sas.client.props`

Pay attention to the following two variables and their values in your `sas.client.props` file:  
`com.ibm.CSI.performTransportAssocSSLTLSRequired=false` and  
`com.ibm.CSI.performTransportAssocSSLTLSSupported=true`

The value for the `-Dcom.ibm.SSL.ConfigURL` parameter must be the URL path to the file `ssl.client.props`, which is found in the properties folder of the WebSphere profile. For example: file: `/C:\Program%20Files%20(x86)\IBM\WebSphere\AppServer1\profiles\AppSrv01\properties\ssl.client.props`.

Pay attention to the variable `user.root` in the file `ssl.client.props`. It must contain the valid path to the profile folder with the `etc` directory that contains the key- and trust-files. Blanks in the URL path to the files `sas.client.props` and `ssl.client.props` must be replaced with `%20`.

12. Click **OK**.

13. Click **Next**. The **JMX Data Source Browser** opens.

14. Choose MBeans for those measures that you want to include in the initial monitor view. Select the type of measures (**Average**, **Sum**, or **Incremental**) and click **Add**.
15. Click **Close**. The **Select displayed measures** page opens.
16. Check the checkboxes for those measures that you want to include in the initial monitor view.
17. Click **Finish**.

## **Internet Information Server (IIS) Performance Monitoring**

*Monitoring Internet Information Services (IIS)*

1. On the **Real-Time Monitoring** tab, in the **Monitor** group, click **System**. The **Data Source wizard** opens.
2. Click the **Select from predefined Data Sources** option button.
3. Click **Next**. The **System selection** page opens.
4. Expand the **Web Server** folder and the **IIS <version>** folder.
5. Click the **System Statistics (PERFMON)** node and then click **Next**. The **Connection parameters** page opens.
6. In the **Hostname** text box, specify the host on which IIS is running.
7. In the **User** and **Password** text boxes, specify a user with administrator permissions on the above-specified host.
8. *Optional:* In the **Alias** text box, specify an alias for the monitored server.
9. Click **Next**. The **Select displayed measures** page opens.
10. Check the check boxes for those measures that you want to include in the initial monitor view.
11. Click **Finish**.

*Available IIS Performance Measures*

### **Performance Measures on IIS 6.0**

Measures can be divided into several groups according to perfmon objects.

#### **Web Service Measures**

- **Web Service(\_Total)\Bytes Received/sec**  
Rate of total bytes transferred by service (received)
- **Web Service(\_Total)\Bytes Sent/sec**  
Rate of total bytes transferred by service (sent)
- **Web Service(\_Total)\Bytes Total/sec**  
Rate of total bytes transferred by service (sum of bytes sent and received)
- **Web Service(\_Total)\Current Connections**  
Current number of connections to the service
- **Web Service(\_Total)\Get Requests/sec**  
Total number of HTTP GET requests received by WWW service
- **Web Service(\_Total)\Post Requests/sec**  
Number of HTTP requests using POST method

#### **Web Service Cache Measures**

- **Web Service Cache(\_Total)\Current Files Cached**  
Current number of files whose content is in the user-mode cache

- **Web Service Cache(\_Total)\Current Metadata Cached**  
Current number of metadata information blocks currently in the user-mode cache.
- **Web Service Cache(\_Total)\Current URIs Cached**  
URI information blocks currently in the user-mode cache
- **Web Service Cache(\_Total)\File Cache Hits %**  
The ratio of user-mode file cache hits to total cache requests (since service startup). Note: This value might be low if the Kernel URI cache hits percentage is high.
- **Web Service Cache(\_Total)\Metadata Cache Hits**  
The ratio of user-mode metadata cache hits to total cache requests (since service startup)
- **Web Service Cache(\_Total)\URI Cache Hits %**  
The ratio of user-mode URI Cache Hits to total cache requests (since service startup)
- **Web Service Cache(\_Total)\Kernel:URI Cache Hits %**  
Applies to static unauthenticated content and dynamic content that is marked as cacheable

### Performance Measures on IIS 7.0, 7.5

All IIS 6.0 measures are available for IIS 7.0/7.5, but in IIS7 there are additional performance counters that give you insight into worker processes level at runtime.

**Web Service Measures** - Same as IIS 6.0.

**Web Service Cache Measures** - Same as IIS 6.0.

### W3SVC\_W3WP Cache Measures

- **W3SVC\_W3WP(\_Total)\Requests / Sec**  
HTTP requests/sec being processed by the worker process
- **W3SVC\_W3WP(\_Total)\Active Requests**  
Current number of requests being processed by the worker process
- **W3SVC\_W3WP(\_Total)\Active Threads Count**  
Number of threads actively processing requests in the worker process
- **W3SVC\_W3WP(\_Total)\Current File Cache Memory Usage**  
Current number of bytes used by user-mode file cache
- **W3SVC\_W3WP(\_Total)\Current Files Cached**  
Current number of files whose contents are present in user-mode cache
- **W3SVC\_W3WP(\_Total)\Current URIs Cached**  
URI information blocks currently in the user-mode cache
- **W3SVC\_W3WP(\_Total)\Current Metadata Cached**  
Number of metadata information blocks currently present in user-mode cache
- **W3SVC\_W3WP(\_Total)\Metadata Cache Hits**  
Total number of successful lookups in the user-mode metadata cache (since service startup)
- **W3SVC\_W3WP(\_Total)\Metadata Cache Misses**  
Total number of unsuccessful lookups in the user-mode metadata cache (since service startup)
- **W3SVC\_W3WP(\_Total)\Metadata Cache Flushes**  
Total number of user-mode metadata cache flushes (since service startup)
- **W3SVC\_W3WP(\_Total)\File Cache Hits / sec**  
Rate of successful lookups in file cache during last sample interval
- **W3SVC\_W3WP(\_Total)\File Cache Misses / sec**

- Rate of unsuccessful lookups in file cache during last sample interval
- **W3SVC\_W3WP(\_Total)\Metadata Cache Hits / sec**
- Rate of successful lookups in metadata cache during last sample interval
- **W3SVC\_W3WP(\_Total)\Metadata Cache Misses / sec**
- Rate of unsuccessful lookups in metadata cache during last sample interval
- **W3SVC\_W3WP(\_Total)\Uri Cache Hits / sec**
- Rate of successful lookups in URI cache during last sample interval
- **W3SVC\_W3WP(\_Total)\Uri Cache Misses / sec**
- Rate of unsuccessful lookups in URI cache during last sample interval

#### *Troubleshooting IIS Monitoring*

#### **Issues With x64/x86 Counters**

For support, please refer to <http://support.microsoft.com/kb/891238>.

#### **Issues With W3SVC\_W3WP**

Note that this group of counters monitors worker processes at runtime. So when no workers are running, this group of counters returns NULL values.

### **Microsoft SQL Server Performance Monitoring**

#### *Monitoring Microsoft SQL Server*

1. On the **Real-Time Monitoring** tab, in the **Monitor** group, click **System**. The **Data Source wizard** opens.
2. Click the **Select from predefined Data Sources** option button.
3. Click **Next**. The **System selection** page opens.
4. Expand the **Database System** folder and the **SQL Server <version>** folder.
5. Click the **System Statistics (PERFMON)** node and then click **Next**. The **Connection parameters** page opens.
6. In the **Hostname** text box, specify the host on which SQL Server is running.
7. In the **User** and **Password** text boxes, specify a user with administrator permissions on the above-specified host.
8. *Optional:* In the **Alias** text box, specify an alias for the monitored server.
9. Click **Next**. The **Select displayed measures** page opens.
10. Check the check boxes for those measures that you want to include in the initial monitor view.
11. Click **Finish**.

#### *Available Microsoft SQL Server Performance Measures*

#### **Performance Measures on Microsoft SQL Server 2005**

The following metrics can be monitored on Microsoft SQL Server 2005 systems:

- **Process(sqlservr)\% Processor Time**
- % Processor Time is the percentage of elapsed time that all of the threads of this process used the processor to execute instructions.
- **SQLServer:Access Methods\Full Scans/sec**

Number of unrestricted full scans. These can either be base table or full index scans.

- **SQLServer:Access Methods\Index Searches/sec**  
Number of index searches. Index searches are used to start range scans, single index record fetches, and to reposition within an index.
- **SQLServer:Access Methods\Table Lock Escalations/sec**  
The number of times locks on a table were escalated.
- **SQLServer:Buffer Manager\Buffer cache hit ratio**  
Percentage of pages found in the buffer cache without having to read from disk.
- **SQLServer:Buffer Manager\Checkpoint pages/sec**  
Number of pages flushed by checkpoint or other operations that require all dirty pages to be flushed.
- **SQLServer:Buffer Manager\Lazy writes/sec**  
Number of buffers written by buffer manager's lazy writer.
- **SQLServer:Buffer Manager\Page lookups/sec**  
Number of requests to find a page in the buffer pool.
- **SQLServer:Buffer Manager\Page reads/sec**  
Number of physical database page reads issued.
- **SQLServer:Buffer Manager\Page writes/sec**  
Number of physical database page writes issued.
- **SQLServer:Buffer Manager\Readahead pages/sec**  
Number of pages read in anticipation of use.
- **SQLServer:Cursor Manager by Type\Active cursors**  
Number of active cursors.
- **SQLServer:Cursor Manager by Type\Cursor memory usage**  
Amount of memory consumed by cursors in kilobytes (KB).
- **SQL Server:Databases\ Active Transactions**  
Number of active update transactions for the database.
- **SQL Server:Databases\ Shrink Data Movement Bytes/sec**  
The rate data is being moved by Autoshrink, DBCC SHRINKDATABASE or SHRINKFILE.
- **SQL Server:Databases\ Transactions/sec**  
Number of transactions started for the database.
- **SQLServer:General Statistics\User Connections**  
Number of users connected to the system.
- **SQLServer:Locks(\_Total)\Average Wait Time (ms)**  
The average amount of wait time (milliseconds) for each lock request that resulted in a wait.
- **SQLServer:Locks(\_Total)\Lock Waits/sec**  
Number of lock requests that could not be satisfied immediately and required the caller to wait before being granted the lock.
- **SQLServer:Locks(\_Total)\Number of Deadlocks/sec**  
Number of lock requests that resulted in a deadlock.
- **SQLServer:Memory Manager\Target Server Memory (KB)**  
Total amount of dynamic memory the server is willing to consume.
- **SQLServer:Memory Manager\Total Server Memory (KB)**  
Total amount of dynamic memory the server is currently consuming.
- **SQLServer:SQL Statistics\Batch Requests/sec**  
Number of SQL batch requests received by server.

- **SQLServer:SQL Statistics\SQL Compilations/sec**  
The number of SQL compilations per second.
- **SQLServer:SQL Statistics\SQL Re-Compilations/sec**  
The number of SQL re-compiles per second.
- **SQLServer:Transactions\Free Space in tempdb (KB)**  
The amount of space (in kilobytes) available in tempdb. There must be enough free space to hold both the snapshot isolation level version store and all new temporary objects created in this instance of the database engine.
- **SQLServer:Transactions\Transactions**  
The number of currently active transactions of all types.

### Performance Measures on Microsoft SQL Server 2008, 2008 R2

All Microsoft SQL Server 2005 counters are still available in Microsoft SQL Server 2008 and 2008 R2 systems. Additional metrics are also available:

- **SQL Server:Databases\Tracked transactions/sec**  
Number of committed transactions recorded in the commit table for the database.
- **SQL Server: Databases\Write Transactions/sec**  
Number of transactions which wrote to the database in the last second.
- **SQL Server: General Statistics\Connection reset/sec**  
Total number of connection resets per second.
- **SQL Server: General Statistics\Tempdb rowset id**  
Number of duplicate tempdb rowset id generated
- **SQL Server: SQL Statistics\Misguided plan executions/sec**  
Number of plan executions per second in which a plan guide could not be honored during plan generation
- **SQL Server: SQL Statistics\Guided plan executions/sec**  
Number of plan executions per second in which the query plan has been generated by using a plan guide.

### Oracle Forms Performance Monitoring

Use Performance Explorer to retrieve performance metrics from Oracle Forms Dynamic Monitoring System (DMS). Performance Explorer can monitor the DMS performance Web page that is provided through the Oracle Forms HTTP server when the DMS module is installed.

Web page requests and the parsing of performance metrics are achieved by driving a Silk Performer project through Performance Explorer. This approach allows you to keep track of application server performance in real time.

The Silk Performer project is located in a SilkEssential package named `OraFormsDMS.sep`, which is located at `C:\Program Files\Silk\Silk Performer 21.0\Monitors`.

The Oracle Forms application architecture includes three tiers:

- Oracle database
- Oracle application server
- Client applet

Performance Explorer should be used to gather server-side measures from the database and application servers.

To monitor the database server a set of predefined Oracle database measures is available.

## *Prerequisites for Oracle Forms Monitoring*

To monitor Oracle Forms, ensure that your environment meets the following requirements:

- Oracle Forms Dynamic Monitoring System must be installed on the Oracle Forms application server.
- The performance page must be accessible from the monitoring machine.

## *Monitoring Oracle Forms*

 **Note:** In addition to default measures for the application server, Performance Explorer offers a BDL monitor that allows querying of performance data from the Oracle Forms Dynamic Monitoring System (DMS). Please review Oracle Technology Network instructions (<http://otn.oracle.com>) for setting up DMS on your Oracle Application Server.

1. On the **Real-Time Monitoring** tab, in the **Monitor** group, click **System**. The **Data Source** wizard opens.
2. Click the **Select from predefined Data Sources** option button.
3. Click **Next**. The **System selection** page opens.

*Optional:* To monitor the application server you can use measures that are provided by the platform upon which the application server runs. Choose from predefined sets or use PERMON, SNMP or REXEC and manually define the measures that are of interest to you.

4. Expand the **Application Server** folder and the **Oracle** folder.
5. Select **Oracle Forms AS** and click **Next**. The **Connection parameters** page opens.
6. In the **Hostname** text box, specify the host on which the Oracle application server is running. You may optionally add an **Alias** name for the application server.
7. Click **Next**. The **Attributes Configuration** page opens.
8. Specify the following settings:
  - URL of the **DMS-Site**. The standard format is `http://server:port/dms0`
  - **Proxy** (if you need to connect using an HTTP proxy)
  - **Proxy-Port** of the HTTP proxy
  - **Username** (if you need to authenticate against the Web server to access the DMS site)
  - **Password** for accessing the DMS site

This monitor provides interesting measures that are provided by the DMS including:

- Common Apache measures
- Module information for HTTP, OC4J, and PLSQL
- Database connection statistics

9. Click **OK**. The **Select displayed measures** page opens.
10. Check the check boxes for those measures that you want to include in the initial monitor view and then click **Finish**.

A connection to the specified host is established, and an initial view that contains the measures you selected is displayed.

## *Oracle Forms Client-Side Measures*

Client-side measures are automatically generated by load test agents during Oracle Forms load tests. They are aggregated into a single result file by the load test controller.

Agents create measures for the following:

- Time taken by an action on a control (for example, button press or control edit)
- Number of bytes received for an action
- Number of bytes sent for an action

- Number of messages received for an action
- Number of messages sent for an action
- Number of round-trips for an action

## **Oracle Performance Monitoring**

Use Performance Explorer to retrieve performance metrics from Oracle database systems no matter the database platform. For example, the database can run on a SunOs, Linux, or a Windows system.

*Monitoring Oracle Database via Perfmon*

*Oracle Database Monitoring Prerequisites*

*Installing Oracle Client and Oracle Counters for Windows*

Before you monitor Oracle Database, ensure that the following requirements have been met:

- SQL\*NET client software must be installed on the machine on which you run Performance Explorer.
- SQL\*Plus must be available on the machine on which you run Performance Explorer.
- Oracle Counters for Windows Performance Monitor must be available.
- The Oracle client must have DB administrator permissions and be able to recognize the targeted database server.



**Note:** Oracle Counters for Windows Performance Monitor is not installed with default installation options (for example, Instant Client and Administrator installations). It is only available via custom installation options.

*Adding Oracle Counters to Windows Performance Monitor*

1. Open file <Oracle Home>\network\admin\tnsnames.ora and add the connection string to the database you want to monitor.
2. Start the command line and test the connection with the command `sqlplus <user>/<password>@<database>`
3. Configure which database is to be monitored using the `operfcfg` command: `<Oracle Home>\bin\operfcfg [-U username] [-P password] [-D TNS_Alias_for_database]`  
If successful, you will receive the message `operfcfg: New registry values have been successfully set.`
4. **Start Windows Performance Monitor Start > Run > Perfmon.**
5. Oracle Counters for Windows Performance Monitor can now be added via the **Add Counters** dialog box.

*Monitoring Oracle Performance (Perfmon)*

1. Choose **Monitor > Add Data Source**. The **Data Source wizard** opens.
2. Click the **Select from predefined Data Sources** option button.
3. Click **Next**. The **System selection** page opens.
4. Expand the **Database System** folder and the **Oracle 9/10/11 (perfmon)** folder.
5. Click the **System Statistics (PERFMON)** node and then click **Next**. The **Connection Parameters** page opens.
6. In the **Hostname** text box, specify the host on which the Oracle Client and Counters for Windows Performance Monitor are running.
7. In the **User** and **Password** text box, specify the user with administrator permissions on the above specified host.

**8.** *Optional:* In the **Alias** text box, specify an alias for the host where the Oracle database is running.

**9.** Click **Next**.

**10.** The **Select displayed measures** page opens.

**11.** Check the check boxes for those measures that you want to include in the initial monitor view and then click **Finish**.

### *Performance Counters*

#### **Oracle Database Buffer Cache**

The counter is `phyrds/gets %`. The percentage of phyrds/gets is calculated as a miss ratio. The lower the `Miss` counter, the better. To improve performance, increase the number of buffers in the buffer cache, if memory is available on the computer. This value is not time-derived.

#### **Oracle Database Redo Log Buffer**

The counter is `redo log space requests`. The value of this counter must be near zero. If this value increments consistently, then processes have had to wait for space in the redo log buffer. In this case, it may be necessary to increase the size of the redo log buffer.

#### **Oracle Database Data Dictionary Cache**

The counter is `getmisses/gets %`. The value of this counter must be less than 10-15% for frequently accessed data dictionary caches. If the ratio continues to increase over this threshold while your application is running, then increase the amount of memory available to the data dictionary cache.

To increase the memory available to the cache, increase the value of initialization parameter `SHARED_POOL_SIZE`. This value is not time-derived.

#### **Oracle Database Library Cache**

The counter is `reloads/pins %`. This is the percentage of SQL statements, PL/SQL blocks, and object definitions that required reparsing. Total Reloads must be near zero. If the ratio of Reloads to Pins is greater than 1%, then reduce the library cache misses. This value is not time-derived.

#### **Oracle Database DBWR stats1**

The two counters available, buffers scanned/sec and LRU scans/sec, are helpful in tuning Buffer Cache. Buffers scanned/sec is the number of buffers DBWR scanned in each second. The buffers scanned are on the LRU (Least Recently Used) list. LRU scans/sec is the number of times DBWR scanned the (Least Recently Used) buffer list in each second.

#### **Oracle Database DBWR stats2**

The two counters available, timeouts/sec and checkpoints/sec, are helpful in determining how much work DBWR has been requested to perform. Timeouts/sec is the number of times DBWR timed-out in each second. DBWR is on a three second timeout interval. If DBWR has not been posted within a three second interval, then it times out.

Checkpoints/sec is the number of checkpoint messages processed by the database writer each second. Whenever a checkpoint occurs, DBWR must be messaged (posted) to "write dirty buffers to disk."

#### **Oracle Database Dynamic Space Management**

The counter is `recursive calls/sec`. Dynamic extension causes Oracle Database to execute SQL statements in addition to those SQL statements issued by user processes. These SQL statements are called recursive calls.

## Oracle Database Free List

The counter is free list waits/requests %. Contention for free lists is reflected by contention for free data blocks in buffer cache. You can determine if contention for free lists is reducing performance by querying V\$WAITSTAT.

If the number of free list waits for free blocks is greater than 1% of the total number of requests, then consider adding more free lists to reduce contention.

## Oracle Database Sorts

The available counters are sorts in memory/sec and sorts on disk/sec. The default sort area size is adequate to hold all data for most sorts. However, if your application often performs large sorts on data that do not fit into the sort area, then you may increase sort area size.

### *Known Issues*

Silk Performer can monitor both 32bit and 64bit Oracle Database performance, but only the 32bit Oracle client version (Oracle Counters for Windows Performance Monitor) is supported.



**Note:** The 32bit Oracle client works fine on a 64bit OS and it is not an issue to monitor a 64bit database from a client machine running a 64bit OS.

### *Monitoring Oracle Database via V\$SYSSTAT*

You can monitor DB2 database systems in a platform-independent way by fetching performance information from an Oracle database's V\$SYSSTAT table.

### *Prerequisites for Oracle Monitoring*

Before you monitor Oracle, ensure that the following requirements have been met:

- Oracle Net (part of Oracle Net Services) client software is installed on the machine on which you run Performance Explorer.
- The Oracle client must be a 32-bit version, must have DB administrator permissions and be able to recognize the targeted database server.
- The relevant project attributes have been changed accordingly for your Oracle installation (choose **Menu Project > Project Attributes** ).
- SQL\*Plus is available on the machine on which you want to run Performance Explorer.
- Log on using the specified user name, password, and server information and perform the query `SELECT * FROM V$SYSSTAT`.

If the logon attempt is successful, the environment is most likely set up correctly.

If the logon attempt fails, you can judge from the error messages generated by SQL\*Plus what caused the problem.

### *Monitoring Oracle Performance*

Drive a Silk Performer project from inside Performance Explorer to keep track of the database system's performance in real time. The Silk Performer project is located a Essential package named OracleMonitoring.sep, which is located at C:\Program Files\Silk\Silk Performer 21.0\Monitors.

1. On the **Real-Time Monitoring** tab, in the **Monitor** group, click **System**. The **Data Source** wizard opens.
2. Click the **Select from predefined Data Sources** option button.
3. Click **Next**. The **System selection** page opens.
4. Expand the **Database System** folder and the **Oracle (v\$sysstat)** folder.

5. Click the **System Statistics (v\$sysstat)** node and then click **Next**. The **Connection parameters** page opens.
6. In the **Hostname** text box, specify the host on which the Oracle database is running.
7. *Optional:* In the **Alias** text box, specify an alias for the host where the Oracle database is running.
8. Click **Next**. The **Attributes Configuration** page opens.
9. If necessary, change the attribute values.  
Typically, the Oracle TNS name, UserId, and Password are included in these settings.
10. Click **OK**. The **Select displayed measures** page opens.
11. Check the check boxes for those measures that you want to include in the initial monitor view and then click **Finish**.

A connection to the specified host is established, and an initial view that contains the measures you selected is displayed.

## SAP Performance Monitoring

Use Performance Explorer to monitor performance and reliability metrics of a SAP system. Performance Explorer is shipped with two monitors specially designed for monitoring SAP installations.

1. On the **Real-Time Monitoring** tab, in the **Monitor** group, click **System**. The **Data Source** wizard opens.
2. Click the **Select from predefined Data Sources** option button.
3. Expand the **Application Server** folder and the **SAP** folder.
4. Depending on the type of SAPGUI monitor that you want to use, select the appropriate monitor type.

The following types of SAPGUI monitors are available:

- SAPGUI Monitoring (ST02): Monitors buffer related metrics. SAP transaction ST02 is executed when running this monitor.
- SAPGUI Monitoring (ST03N): Monitors application specific metrics. SAP transaction ST03n is executed when running this monitor.
- SAPGUI Monitoring (ST04): Monitors database related metrics. SAP transaction ST04 is executed when running this monitor.
- SAPGUI Monitoring (ST07): Monitors user distribution on the SAP system. SAP transaction ST07 is executed when running this monitor.
- SAPGUI OS-Monitoring (ST06): Monitors operating system specific metrics. SAP transaction ST06 is executed when running this monitor.

5. Click **Next**.

The **Connection parameters** page opens.

6. In the **Hostname** text box, specify the host of the machine to be monitored  
This value is for display purposes only. It is not used in the monitor itself.
7. Click **Next**. The **Attributes Configuration** page opens.
8. Define the following monitoring-specific attributes:

- **ConnectionString** – Complete connection string to the SAP server. If you are not sure, record the logon sequence with Silk Performer. The connection string is the first parameter of SapGuiOpenConnection.
- **Username** – SAP username with access rights to the monitoring transaction.
- **Password** – Password for the SAP user.
- **ClientNum** – Client number for the logon procedure, such as 850.
- **Language** – Language to use for logon, such as EN.
- **Entity** – The data entity that should be monitored, such as Dialog, RFC, or Background.



**Note:** The **Entity** attribute does not have to be provided for the SAP OS-Monitor.

- **TimeFrame** – SAP provides the average values of the past interval that you define.



**Note:** The **Timeframe** attribute does not have to be provided for the SAP OS-Monitor.

- **Server - (ST03N only)** The server that should be monitored. Its the name of the tree node that is to be selected in ST03N. Total will return measures from the overall SAP System

9. Click **OK**. The **Select displayed measures** page opens.

10. Check the check boxes for those measures that you want to include in the initial monitor view and then click **Finish**.

A connection to the specified host is established, and an initial view that contains the measures you selected is displayed.

## WebLogic (SNMP) Monitoring

This section explains how to monitor BEA WebLogic using Performance Explorer. It also explains important WebLogic performance measures.

By default, the WebLogic SNMP agent is not running. You can enable SNMP support by using the WebLogic Administration Console. Open the console by typing the URL `http://weblogichost:port/console` into your browser. In the menu tree on the left, select the SNMP node. This selection opens the **SNMP configuration** page. Within this page, verify that the SNMP agent is enabled. You must restart WebLogic to enable SNMP support.

### *Monitoring BEA WebLogic (SNMP)*

Establish a connection to a data source to query BEA WebLogic through SNMP and to create an initial view that contains the measures you want to monitor.

1. On the **Real-Time Monitoring** tab, in the **Monitor** group, click **System**. The **Data Source** wizard opens.
2. Click the **Select from predefined Data Sources** option button.
3. Click **Next**. The **System selection** page opens.
4. Expand the **Application Server** folder and the **BEA WebLogic (SNMP)** folder.
5. Select the **SNMP** node and then click **Next**. The **Connection parameters** page opens.
6. In the **Hostname** text box, specify the host on which WebLogic is running.
7. Click **Next**. The **Add SNMP Measures** page opens. **BEA-WEBLOGIC-MIB** is selected in the list box of available MIBs.

The following entries for monitoring the performance of your WebLogic Server are available:

- `bea.wls.jdbcConnectionPoolRuntimeTable`
- `bea.wls.servletRuntimeTable`
- `bea.wls.ejbEntityHomeRuntimeTable`
- `bea.wls.ejbStatelessHomeRuntimeTable`
- `bea.wls.executeQueueRuntimeTable`

8. Select **jdbcConnectionPoolRuntimeEntry** and click **Get value(s)** to retrieve all available JDBC connection pools.

For example, when monitoring the Pet Store sample application, a connection pool named `ServerRuntime:petstoreServer` is returned.

Interesting performance counters exposed by connection pools include the following counters:

- **JdbcConnectionPoolRuntimeActiveConnectionsCurrentCount** – Returns the number of connections that are currently being used.

- **JdbcConnectionPoolRuntimeWaitingForConnectionCurrentCount** – Returns the number of waiters for connections. This number is greater than zero when all connections are in use and additional requests for connections are being sent to the connection pool.
  - **JdbcConnectionPoolRuntimeWaitSecondsHighCount** – Returns the maximum number of seconds a client waits to check out a connection from the connection pool.
9. To monitor the number of active connections, search for the **jdbcConnectionPoolRuntimeActiveConnectionsCurrentCount** column in the table that was previously returned and select the cell.



**Note:** Connection pools provide a large number of performance counters. You might need to resize some columns to view the columns in which you are interested.

#### 10. Click **Add**.

11. Repeat the previous steps until you have entered all commands you require for measure collection, then click **Close**. The **Select displayed measures** page opens.

12. Check the check boxes for those measures that you want to include in the initial monitor view and then click **Finish**.

A connection to the specified host is established, and an initial view that contains the measures you selected is displayed.

## WebLogic (JMX) Monitoring

This section explains prerequisites for monitoring WebLogic (JMX) Server, including configuring IIOP and JMX Management Server.

### *Enabling JMX Management Server*

Enable the JMX Management Server in the Oracle WebLogic console at **Configuration > General > Advanced Settings**. Set both **Compatibility Mbean Server Enabled** and **Management EJB Enabled**. This enables both the legacy and the new JMX interface.

You must restart your system for these changes to go into effect.

### *Enabling and Configuring IIOP*

You must enable and configure the IIOP protocol in Oracle WebLogic 11g R1 before you can monitor WebLogic (JMX).

1. In the Change Center of the Oracle WebLogic administration console, click **Lock & Edit**.
2. In the left pane of the console, expand **Environment** and select **Servers**.
3. Select the **Protocols** tab, then select **IIOP**.
4. Check the **Enable IIOP** check box to enable the IIOP protocol.
5. To modify the default configuration, click **Advanced**.
6. To specify a default IIOP user name and password:
  - a) In the **Default IIOP User** field, enter a user name.
  - b) In **Default IIOP Password** field, enter a password.
7. To activate these changes, in the Change Center of the administration console, click **Activate Changes**.

You must restart your system for these changes to take effect.

### *Monitoring WebLogic (JMX)*

1. On the **Real-Time Monitoring** tab, in the **Monitor** group, click **System**. The **Data Source** wizard opens.

2. Click the **Select from predefined Data Sources** option button.
  3. Click **Next**. The **System selection** page opens.
  4. Expand Application Server and select the WebLogic server version to be monitored.  
Select either EJXM or JMX.
  5. Click **Next**. The **Connection parameters** page opens.
  6. In the **Hostname** text box, specify the host on which WebLogic is running.
  7. Specify the WebLogic **Port** (Default is 7001).
  8. Do not specify a user or password.
  9. Click **Server Configuration**.
  10. Specify the **Java Home directory** and the **Application Server install directory**.  
Use a UNC path, including the folder lib. Example: \\<server name>\Oracle\Middleware\wlserver\_10.3\server\lib.
-  **Note:** If you can not specify a UNC path, copy wljmxclient.jar from the folder <WL\_HOME>\server\lib to your local machine and add it to the classpath.
11. Click **OK**.
  12. Click **Next**. The **JMX Data Source Browser** opens.
  13. Select measures that you want to include in the initial monitor view.
  14. Click **Finish**.

## Custom Data Sources

In addition to predefined data sources, Performance Explorer offers the following monitoring interfaces:

- PerfMon – Provides access to all Windows Performance Monitoring data sources.
- REXEC – Provides access to data sources on UNIX-based systems through the Remote Execution Protocol.
- SNMP – Provides access to monitoring data that is exposed by Simple Network Management Protocol services.

### Monitoring PerfMon Data Sources

PerfMon is a Microsoft-specific monitoring interface that queries performance counters. Establish a connection to a PerfMon data source and create an initial view that contains the measures you want to monitor.

1. On the **Real-Time Monitoring** tab, in the **Monitor** group, click **System**.
2. Click **Select from predefined Data Sources** and then click **Next**.
3. Expand the **Custom Data** folder, click **NT Performance Monitor Data**, and click **Next**.
4. In the **Hostname** text box, specify the machine to be monitored.  
The host to be monitored might be located on a different NT Domain.
5. *Optional:* In the **Alias** text box, specify the alias name.  
The alias must be a highly descriptive synonym for the monitored server. It is recommended that you group measures on a particular machine.  
For example, both WebLogic and IIS might be installed on the same computer. Both servers require monitoring, but the two performance measures must appear in separate menu trees.
6. In the **Username** text box, specify a user who has administrative security rights.
7. In the **Password** text box, specify the appropriate password for the username.
8. Click **Next**.
9. Examine your system for available performance counters and add them to the monitoring template.  
To select multiple counters from the list, press Ctrl or Shift.

- a) Select a counter.
- b) Select the amount of instances.
- c) Click **Add**. The **Counter Usage** page opens.
- d) Ensure that **Is an average measure** is enabled.
- e) Click **Next**.
- f) Add more counters as described above.
- g) When you are done, click **Close**.

**10.** Check the check boxes for those measures that you want to include in the initial monitor view and then click **Finish**.

A connection to the specified host is established, and an initial view that contains the measures you selected is displayed.

### Querying Through SNMP

Performance Explorer provides a set of predefined data sources that are queried through SNMP. However, you can add an entry for ORADB-MIB in an SNMP datasource.

Establish a connection to a data source to query through SNMP and create an initial view that contains the measures you want to monitor.

SNMP involves an SNMP agent, which is typically provided by the application vendor. For example, Oracle databases expose data sources using their SNMP agents. The client-side counterpart is Performance Explorer, which collects performance data.

1. On the **Real-Time Monitoring** tab, in the **Monitor** group, click **System**.
2. Click **Select from predefined Data Sources** and then click **Next**.
3. Expand the **Custom Data** folder, click **Snmp Data**, and then click **Next**.
4. In the **Hostname** field, specify the host running the SNMP agent.
5. *Optional:* In the **Alias** text box, specify the alias name.

The alias must be a highly descriptive synonym for the monitored server. It is recommended that you group measures on a particular machine.

For example, both WebLogic and IIS might be installed on the same computer. Both servers require monitoring, but the two performance measures must appear in separate menu trees.

6. Specify the port, community, and version that are appropriate and click **Next**. If you select version 3, the community field disappears.
7. If you have selected version 3, specify the authentication details for SNMPv3 and click **Next**.
8. Specify the appropriate values to create the measure that you want to monitor as follows:
  - a) Select an MIB from the **Select a MIB, and object, and get the object's value(s)** list box. After you select an MIB, the objects and the object values display in a hierarchical tree below the list box.
  - b) Select the object that you want to measure. The object's properties are displayed in the text boxes adjacent to the list box.
  - c) Click **Get value(s)** to check the object's availability.
- If the performance counter is queried successfully, the object's value is displayed in the grid below the list box.
9. Click **Add** to add the SNMP measure.
- 10.** Repeat the previous steps until you have entered all the commands that you require for measure collection, then click **Close**. The **Select displayed measures** page opens.
- 11.** Check the check boxes for those measures that you want to include in the initial monitor view and then click **Finish**.

A connection to the specified host is established, and an initial view that contains the measures you selected is displayed.

## **Adding MIB Files to Browse SNMP Agents**

To browse SNMP agents for available performance counters, your MIB files must contain textual descriptions of performance counters.

1. On the **Real-Time Monitoring** tab, in the **Monitor** group, click **System**.
2. Click **Select from predefined Data Sources** and then click **Next**.
3. Expand the **Custom Data** folder, click **Snmp Data**, and then click **Next**.
4. In the **Hostname** field, specify the host running the SNMP agent.
5. *Optional:* In the **Alias** text box, specify the alias name.

The alias must be a highly descriptive synonym for the monitored server. It is recommended that you group measures on a particular machine.

For example, both WebLogic and IIS might be installed on the same computer. Both servers require monitoring, but the two performance measures must appear in separate menu trees.

6. Specify the port, community, and version that are appropriate and click **Next**.
7. Select **Import new Mib...** from the **Select a MIB, and object, and get the object's value(s)** list.
8. Type the MIB file name that you want to compile in the **MIB source file** field.  
*Alternative:* Click [...] to navigate to and select the file.
- Commonly used MIB files are located in the <public user documents>\Silk Performer 21.0\Data\Mibs\VendorMibs directory.
9. Click **Import**. The MIB file is added to the **Compiled MIBs** list.

## **Automatic Detection of Data Sources**

The data source scanner queries your machine for data sources that can be monitored. This eliminates the need for you to do the search manually. The automatic detection might not find every possible data source, since it is based on a configuration that closely mimics the predefined data sources. It goes through the list of the predefined data sources and checks for their availability.

For example: The data source scanner queries your machine for the number of processors, the number of network interfaces, the number of database instances, and how these components can be monitored.

Details such as the platform on which your system runs (NT, AIX, or Linux) do not matter because the monitored components, such as CPU or hard disk utilization, are platform-independent.

## **Using the Data Source Scanner**

The data source scanner queries your machine for data sources that can be monitored.

1. On the **Real-Time Monitoring** tab, in the **Monitor** group, click **System**.
2. Click **Have Data Sources detected** and then click **Next**.
3. In the **Hostname** field, specify the machine that is to be scanned.
4. *Optional:* In the **Alias** text box, specify the alias name.

The alias must be a highly descriptive synonym for the monitored server. It is recommended that you group measures on a particular machine.

For example, both WebLogic and IIS might be installed on the same computer. Both servers require monitoring, but the two performance measures must appear in separate menu trees.

5. Click **Next**.

The data source scanner now queries your system for the following data sources on various platforms:

- .NET
- Active Server Pages
- Apache

- Coldfusion
- IBM Universal Database DB2
- Internet Information Servers
- iPlanet Directory Servers
- Microsoft Proxy Servers
- Microsoft SQL Servers
- Netscape Directory Servers
- Netscape Enterprise 3.0
- Netscape/iPlanet Directory Server
- Operating System Data, such as hard disk, CPUs, and network interfaces
- Oracle Databases
- Rstat which often can be queried on UNIX-based systems and provides large amounts of operating system-related data.
- SilkPerformer Agent
- SilkPerformer Controller
- Silverstream

While your system is scanned, the data source scanner might display dialog boxes that request credentials for specific data sources (for a remote execution daemon, for example).

The scanner also checks whether your system resides within a remote NT domain. In such instances, you must provide the domain name of the specified host.



**Note:** The specified user must possess administration security rights, which are necessary for monitoring NT based systems.

6. When the **Progress** section indicates Done, click **Next**.
7. Check the check boxes for those measures that you want to include in the initial monitor view and then click **Finish**.

A connection to the specified host is established, and an initial view that contains the measures you selected is displayed.

## Monitoring Load Tests

With Performance Explorer, you can collect data for client-side monitoring in addition to server data.

### Configuring Load Test Monitoring

To use real-time client-side measures, such as in built-in charts and monitor reports, add a connection entry for the client-side data source.

1. On the **Real-Time Monitoring** tab, in the **Monitor** group, click **Load Test**.
2. In the **Hostname** field, specify the host name of the load test controller.
3. In the **Alias** field, specify the alias for the controller that provides the client-side measures. The alias will display in the **Client-Side Measures** tree.
4. Click **Finish**.

The host appears in the tree. The client-side measures that are available on this data source are listed in folders in the tree.



**Note:** Check the **Enable real-time measures** check box in the **Workload Configuration** dialog, which instructs the runtime system to supply host measures. A green icon on in the **Client-Side Measures** tree indicates a successful connection to the data source. A red icon indicates that the data source is not connected.

## Removing Client-Side Measures

You can remove a client-side measure node from the tree. However, Silk Performer replaces the measure node if it receives updated information for the measure.

1. Click the **Client-Side Measures** tab in the tree area.
2. Right-click the measure node that you want to delete and click **Remove Client-Side Data Source**.
3. Click **Yes**.

## Monitor Charts and Monitor Writers

Performance Explorer provides monitor charts and monitor reports to display and store measured values.

- A monitor chart displays the measured values of a data source in real-time. You can adjust the polling interval of a monitor chart.
- A monitor writer records the measured values of a data source and stores them in a .tsd file. Monitor writers are also useful to compare server-side and client-side measures when a load test is completed.



**Note:** A short polling interval can reduce the performance of your computer which, consequently, can influence test results. For large load tests, define intervals greater than 60 seconds.



**Note:** The values that display in a monitor chart cannot be written to a .tsd file afterwards.

### Creating Monitor Charts

Monitor charts are useful for providing a graphical overview of the state of a system.

1. On the **Real-Time Monitoring** tab, in the **New** group, click **Monitor Chart**.
2. Drag measures from the **Monitor** tree onto the chart.
3. Right-click a measure in the measure list and click **Properties**.
4. On the **General** tab, you can adapt the caption and description.
5. On the **Monitor** tab, you can adapt monitor and view options: The **Interval** defines the refresh rate for retrieving new values. The **Page size** defines how many intervals display on each page. The **History size** defines how many intervals are stored in the chart history.
6. Click **OK**.



**Tip:** You can also create a **Transaction Rate Monitor**. Performance Explorer will automatically populate this chart with transaction measures. Make sure to **Enable real-time measures** on the **Workload Configuration** dialog before you start monitoring.

### Creating Monitor Writers

A monitor writer can be defined to capture measures during load tests. Monitor writes write a time series data (.tsd) file that you can include in an overview report or use as the basis for a chart.

1. On the **Real-Time Monitoring** tab, in the **New** group, click **Monitor Writer**.
2. Drag measures from the **Monitor** tree onto the writer.
3. On the **Real-Time Monitoring** tab, in the **Control** group, click **Recording**.
4. Specify a directory for the .tsd file and click **Save**. Performance Explorer will write all captured values into this .tsd file.



**Note:** If you are running a load test and the workspace you have assigned to your profiles contains a monitor writer, all monitor writers begin writing automatically when the test starts and stop when the test finishes.

All monitor writers are stored in the result directory of the load test.



**Note:** For SAPGUI data sources, it is recommended that you change the monitor interval to 20 seconds or higher. To change the interval, click **Properties** in the **Report** group, on the **Report** tab. Then click the **TSD-Writer** tab and change the interval.

## Host Data

You can change, rename, and duplicate hosts and update the credentials that allow you to access a host.

### Changing Hosts

You can change a host that has already been specified or you can change just the alias of the host. In both cases, the specified measures, which display in the tree, will not be lost.

1. Select a host (a top-level node) in the **Monitor** tree.
2. On the **Real-Time Monitoring** tab, in the **Host** group, click **Change**.
3. To change the host, enter the **Hostname** of the new host, enter an **Alias**, and click **Next**.
4. To change the alias of the host (the name that displays in the **Monitor** tree), enter the **Hostname** of the current host, enter a new **Alias**, and click **Next**.
5. Enter the **Username** and **Password** and click **Finish**.

### Duplicating Hosts

When you duplicate a host, Performance Explorer creates a new host node in the **Monitor** tree and adds all measures that are specified for the selected host.

1. Select a host (a top-level node) in the **Monitor** tree. This host will be duplicated.
2. On the **Real-Time Monitoring** tab, in the **Host** group, click **Duplicate**.
3. Specify the **Hostname** of the new host and an **Alias** and click **Next**. The alias will display as name of the new host in the **Monitor** tree. Depending on the specified data source, fields with various connection parameters appear. These fields are automatically populated with the original values.  
If there are configurable attributes for the specified data source, a **Change Attributes** button appears.
4. Enter the credentials of the host and click **Finish**. The measures are checked to confirm validity. If invalid measures are discovered, you have the option of changing the connection parameters or ignoring the measures.



**Note:** It is not possible to duplicate or rename invalid measures.



**Note:** Measures are grouped into measure groups when they use the same data source and connection parameters. For example: If you have PerfMon measures with different credentials, you have different measure groups.

### Editing Host Credentials

The security policies of many companies require that user passwords expire at regular intervals. Such policies can create in automatic access of tools like Performance Explorer. For such environments, Performance Explorer enables you to change user names and passwords for existing hosts.

1. Select a host (a top-level node) in the **Monitor** tree.
2. On the **Real-Time Monitoring** tab, in the **Host** group, click **Edit Credentials**.
3. The **Hostname** and the **Alias** of the selected host are already prefilled. Click **Next**.
4. If necessary, click **Change Attributes**.
5. Change the **Username** and **Password** as required and click **Finish**.

# Converting Data from CSV to TSD

Silk Performer offers a command-line tool (`csv2tsd.exe`) that enables you to convert .csv files (comma-separated values) into .tsd files (time series data).

Converting data to the .tsd format is useful when you monitor external Windows data that, for security reasons, cannot be monitored by Silk Performer. By converting .csv data into the .tsd format, external data can be analyzed alongside internal Silk Performer monitoring data within Performance Explorer.

For example, if a performance counter measures the behavior of a remote server and the results of that monitor are saved in .csv files, those results can be imported into Performance Explorer for analysis alongside the results of a Silk Performer load test involving the same server. Using Silk Performer, you then have the opportunity to look for correlations between the two data sets.

## Generating a CSV File on a Remote Server for Windows 7/2008 R2

Any performance counter log (CSV) file you specify in the command line tool must follow the format generated by the Windows Performance Monitor tool ( **Administrative Tools > Performance** ). So, before using `csv2tsd.exe` you must create a performance counter with settings that will allow you to import its data.

1. Open the Windows Performance Monitor tool ( **Administrative Tools > Performance** ).
2. Expand the tree **Performance > Data Collector Sets > User Defined**.
3. Right-click and select **New > Data Collector Set** from the context menu.
4. Enter a name and select **Create manually (Advanced)** and click **Next**.
5. Create data logs and select **Performance counter**. Click **Next**.
6. Specify the location of the CSV file and click **Next**.
7. *Optional:* Specify a different user in **Run as**. Click **Save** and close the dialog box.
8. Right-click on the defined Data Collector Set and select **New > Data Collector** from the context menu.
9. Enter a name and select **Performance counter data collector**. Click **Next**.
10. Click **Add** to add the performance counters.
11. Specify the computer (can be both local and remote) and performance counter instances for which you want to track results.
12. Select **Open properties for this data collector** and click **Next**.
13. On the **Performance Counters** tab specify the log format to be **Comma Separated**.
14. Select the **File** tab and specify a file name.
15. Click **OK** to close the dialog.
16. Close the Performance Monitor.

Once you have configured a performance counter so that its log results can be converted to TSD format using `csv2tsd.exe`, you can start capturing behavioral data using the performance counter.

## Capturing Performance Counter Log Data

Once you have configured a performance counter so that its log results can be converted to TSD format using `csv2tsd.exe`, you can start capturing remote-server log data using the performance counter.

1. Open the Windows Performance Monitor tool ( **Administrative Tools > Performance** ).
  2. Expand the **Performance Logs and Alerts** tree node.
  3. Click the **Counter Logs** node.
  4. Right-click a counter log in the right-hand window and select **Start** from the context menu.
- Active counters are shown in green. Inactive counters are shown in red.

Generated CSV counter log files have the following format:

```
"(PDH-CSV 4.0) (W. Europe Standard Time)(-60)", "\\\JONES\Processor(_Total)\% Interrupt Time", "\\\JONES\Processor(_Total)\% Privileged Time", "\\\JONES\Processor(_Total)\% Processor Time"  
"02/25/2008 23:24:23.826", " ", " ", "  
"02/25/2008 23:24:38.841", "0", "0.10406093659265107", "0.10150087105497141"  
"02/25/2008  
23:24:53.841", "0.052084666700800877", "1.8229633345280307", "2.445419269399962  
1"  
"02/25/2008  
23:25:08.840", "0.052084666700800877", "3.0729953353472514", "5.466329938046410  
3"  
"02/25/2008  
23:25:23.840", "0.052084666700800877", "0.57293133370880955", "1.03913326847834  
28"
```

## Converting a Properly Formatted CSV File to TSD Format

1. Launch `csv2tsd.exe` (**Start > All Programs > Silk > Silk Performer 21.0 > Analysis Tools > CSV to TSD Converter**).
2. Enter file-import and export settings using the following format: `Csv2Tsd <csvFile> <tsdfile> [ [+/-]<time offset>]`.  
`<csvFile>` is the full path of the CSV file to be imported. `<tsdFile>` is the full path of the TSD file that is to be generated. `<time offset>` is the time difference in seconds (the default is 0).
3. Hit **Enter** to execute the conversion.

Example:

```
Csv2Tsd C:\\perflogs\\Test1.csv C:\\perflogs\\Test1.tsd
```

## Built-In Measures

A measure is the smallest grouping entity for the values that are collected during a load test or during a monitoring session for a particular measuring point. A measure has a name and type. A set of measured values that are related to the measure is stored as a series in a time series data (.tsd) file.

Measures can be either timers or counters. Timers collect data related to response times, counters collect data on throughput and load test events.

Performance Explorer provides a big amount of measures, which are divided into groups.

## Summary Measures

The group *Summary Measures* contains summarized measurements on a global level. It contains measures that aggregate individual measures from other measure groups as well as measure types that represent information on a global level that are not included in other measure groups.

This group contains only counters, it does not contain timers. It represents a fixed set of global measures that are collected only during the measurement period, thus excluding the warm-up and shutdown period of the load test. The *Summary Measures* group contains measure types that are available as real-time data and time-series data (.tsd).

## General Measures

Displays the measures that are shown at the beginning of the overview report.

Measure	Description
Active users	<p>The number of active virtual users. A virtual user is regarded as active if it has been started and is currently in one of the following states:</p> <ul style="list-style-type: none"> <li>• executing</li> <li>• warm up</li> <li>• wait database</li> <li>• loading documents</li> <li>• wait resource</li> <li>• rendezvous</li> <li>• thinktime</li> <li>• shutdown</li> </ul> <p>A virtual user is regarded as inactive if it currently is in one of the following states:</p> <ul style="list-style-type: none"> <li>• starting</li> <li>• wait queue</li> <li>• wait user</li> <li>• wait stop</li> <li>• pacing time</li> <li>• suspended</li> </ul>
Transactions	<p>The number of Silk Performer transactions that have completed, regardless of whether or not they were successful.</p> <p>The overview report shows the following information:</p> <ul style="list-style-type: none"> <li>• A graph indicating how many transactions ended in a given second.</li> <li>• Average number of ended transactions per second.</li> <li>• Total number of ended transactions.</li> </ul>
Errors	<p>The number of API errors, including Internet, database, and middleware APIs. An API error is counted if the severity is <code>error</code> or higher (<code>transaction exit</code> or <code>process exit</code>). An error is ignored if the severity is informational or warning.</p>
Running users	<p>The number of running virtual users. A virtual user is regarded as running if it has been started and not been stopped. In contrast to the measure <b>Active users</b>, it also includes virtual users that are in one of the following states:</p> <ul style="list-style-type: none"> <li>• wait queue</li> <li>• wait user</li> <li>• wait stop</li> <li>• pacing time</li> </ul>

## Agent Health Control Measures

Describes the measures that are used for analyzing an agent's health.

Measure	Description
CPU utilization	<p>The percentage of time that a processor is busy executing tasks. If the processor utilization is equal to or near 100%, the computer has run out of available processing capacity.</p>
Memory	<p>The percentage amount of memory allocated (committed) by the driver machine in relation to the total amount of physical RAM. High memory utilization indicates that the driver machine is low on available memory, and this situation can cause disk swapping, which degrades the overall performance of the computer.</p>

Measure	Description
Responsiveness	 <b>Note:</b> Task Manager may show different memory statistics depending on your operating system. For most Silk Performer project types the responsiveness measure is determined directly by the internal measure <i>System Health</i> . For Silk Performer projects of type SAP GUI, Citrix, or browser driven load testing (BDLT), the responsiveness measure also takes into account the health of the application under test (SAP GUI or <code>perfBrowserHost.exe</code> ). This health is indicated by the internal <i>App Health</i> value. The overall <i>Responsiveness</i> measure corresponds to the worse of these two health values.

## Internet-Related Measures

Describes the measures that are related to the internet connection.

Measure	Description
Concurrent connections	The number of currently open TCP/IP connections.
Requests sent	The number of HTTP requests sent by low-level web functions, the number of data packets sent by TCP/IP-level functions, or the number of commands sent by POP3, SMTP, FTP, or LDAP function calls. Silk Performer also includes LDAP traffic in this measure; it reports the number of LDAP requests sent. IIOP uses other counters.
Requests failed	The number of the requests listed above that are not sent or executed successfully. IIOP uses other counters.
Request data sent	The amount of data (in KB) sent to the server, including header and body content information as well as all TCP/IP-related traffic (HTTP, native TCP/IP, IIOP, POP3, SMTP, FTP, and LDAP), and secure traffic over SSL/TLS. It does not include data overhead caused by encryption using SSL/TLS or other API-related traffic like TUXEDO ATMI, ODBC, and Oracle OCI.
Responses received	The number of HTTP responses received by low-level web functions, the number of data packets received by TCP/IP-level functions, and the number of responses received by POP3, SMTP, FTP, and LDAP functions. IIOP uses other counters.
Responses failed	The number of the expected responses that are not received successfully. IIOP uses other counters.
Response data received	The amount of data (in KB) received from the server, including header and body content information as well as TCP/IP-related traffic (HTTP, native TCP/IP, IIOP, POP3, SMTP, FTP, and LDAP), and secure traffic over SSL/TLS. It does not include API-related traffic like TUXEDO ATMI, ODBC, and Oracle OCI.
Connects successful	The number of successful TCP/IP connects to the remote hosts established by low-level, TCP/IP-level, IIOP, SMTP, POP3, FTP, or LDAP functions.
Connects failed	The number of TCP/IP connection failures caused by invalid host names, invalid IP addresses, invalid ports, unavailable remote hosts, network-related problems, or authentication failures. This counter is used by low-level, TCP/IP-level, IIOP, SMTP, POP3, FTP, and LDAP functions. By default, low-level web functions perform up to three connect attempts (up to two retries). Only if all attempts fail, a failure is recorded.
Connects retries	The number of attempts to connect to the server if the initial connection fails.

Measure	Description
Page data	The amount of data (in KB) for all requests and responses related to the request and retrieval of an entire Web page, including all embedded objects.
Embedded objects data	The amount of data (in KB) for all requests and responses related to the request and retrieval of objects embedded in a Web page.

## Web-Specific Measures

Describes the measures that are specific for the Web.

Measure	Description
HTTP request retries	The number of times a client is forced to retry a request. This may happen when the server closes a persistent connection (available with Keep-Alive or HTTP/1.1) because of a timeout occurring while the client tries to send a request. This counter is used only by low-level web functions.
HTTP redirections	The number of HTTP redirections performed due to a Web server response with status code 3xx. This measure is incremented only by Web functions.
HTTP re-authentications	The number of re-authentications necessary because of a status code 401 or 407 returned by the Web server. This counter is used only by Web functions that perform basic authentication.
HTTP hits	The number of HTTP requests that arrive at the Web server. This measure is incremented only by Web functions.
Hits failed	The number of failed HTTP requests. This measure is incremented only by Web functions.
HTTP 1xx responses	The number of HTTP responses containing status codes indicating that their content has the status informational. This measure is incremented by Web functions.
HTTP 2xx responses	The number of HTTP responses containing status codes indicating successful requests. This measure is incremented by Web functions.
HTTP 3xx responses	The number of HTTP responses containing status codes indicating redirected requests. This measure is incremented by Web functions.
HTTP 4xx responses	The number of HTTP responses containing status codes indicating client errors. This measure is incremented by Web functions.
HTTP 5xx responses	The number of HTTP responses containing status codes indicating server errors. This measure is incremented by Web functions.
HTTP cache conditional reloads	The number of requests sent to the server to download an item if its content has changed since it was last placed in the cache.
HTTP cache hits	The number of times requests that are fulfilled from the cache.
HTTP cookies sent	The number of cookies sent by low-level web functions. If the header line of an HTTP cookie request contains multiple cookies, each cookie in the header is counted.
HTTP cookies received	The number of cookies received by low-level web functions. If the HTTP response includes multiple cookie header lines, each cookie of these header lines is counted.
HTTP pages	The number of HTML pages requested by the page-level Web API.

## IIOP-Related Measures

The following table describes the scripted measures which relate to IIOP operations.

Measure	Description
IIOP messages	The number of IIOP messages sent by Silk Performer.
IIOP requests	The number of IIOP request messages sent by Silk Performer.
IIOP loc requests	The number of IIOP locate request messages sent by Silk Performer.
IIOP fragments sent	The number of fragmented IIOP messages sent by Silk Performer.
IIOP replies OK	The number of IIOP reply messages with status OK received by Silk Performer.
IIOP replies exception	The number of IIOP reply messages with the status exception raised received by Silk Performer. This count includes system exceptions as well as user exceptions.
IIOP replies obj moved	The number of IIOP reply messages with the status object has moved. This status message causes Silk Performer to reissue the request to the new location.
IIOP request timed out	The number of times the time limit for receiving data (specified in the load-testing script) was exceeded while waiting for a pending reply. Whenever the time limit is exceeded, no reply is received and Silk Performer sends an IIOP Cancel Request message to inform the server about the aborted operation.
IIOP loc replies obj here	The number of IIOP Locate Reply messages with status object here received by Silk Performer.
IIOP loc replies obj unknown	The number of IIOP Locate Reply messages with status object unknown received by Silk Performer.
IIOP loc replies obj moved	The number of IIOP Locate Reply messages with status object moved received by Silk Performer.
IIOP message errors	The number of IIOP error messages received by Silk Performer.
IIOP fragments received	The number of fragmented IIOP messages received by Silk Performer.
IIOP loc request timed out	The number of times that the time limit for receiving data (specified in the load-testing script) was exceeded while waiting for a pending locate reply. Whenever the time limit is exceeded, no locate reply is received and Silk Performer sends an IIOP Cancel Request message to inform the server about the aborted operation.

## Database-Specific Measures

The following table describes the measures that are specific to the database.

Measure	Description
SQL commands	The number of SQL commands sent to the DBMS for execution.
Open database logins	Counts the currently open database sessions. This counter is relevant for the OCI, ODBC, and ODBC high-level interfaces (This measure is not available in report files).
Open database cursors	Counts the currently opened database cursors (handles). This counter is relevant for the OCI, ODBC, and ODBC high-level interfaces. For ODBC, only OdbcClose calls with the SQL_DROP option decrease this measure value. (This measure is not available in report files.)

## Transaction Measures

The Transaction measure group contains response-time data on the transaction level. For each transaction defined in your load-test script, a measure of this group type is automatically created, using the name of

the transaction as the key for the measures. Only measures that contain a counted value are displayed. This fact means that Trans.failed is not displayed if none of the transactions fail.



**Note:** The transactions TInit and TEnd do not count towards the load-test measures because beginning and ending transactions should not contain measures that relate to the application under test.

The following table describes the corresponding measure types.

Measure	Description
Trans. ok	The response time for successful transactions, in seconds. A transaction time is reported in this measure type if no API function call performed within the transaction returns an error message.
Trans.(busy) ok	The busy time for successful transactions, in seconds. A transaction busy time is the transaction response time without any think time, wait time, and time necessary for function transformations. A transaction time is reported in this measure type if no API function call performed within the transaction returns an error message.
Trans. failed	The response time for failed transactions, in seconds. A transaction time is reported in this measure type if at least one error of severity error or higher occurs within the transaction.
Trans.(busy) failed	The busy time for failed transactions, in seconds. A transaction busy time is the transaction response time without any think time, wait time, and time necessary for function transformations. A transaction time is reported in this measure type if at least one error of severity error or higher occurs within the transaction.
Trans. canceled	The response time for canceled transactions, in seconds. A transaction time is reported in this measure type if a return statement in the script is issued with a return value of 1.

## Web Form Measures

The Web Form measure group contains the measure types related to forms declared in the `dclform` section of a load-test script. The Web Form measure group provides response time and throughput rates for form submissions with the POST, GET, and HEAD command sent by scripting function calls. For each form declared in the `dclform` section, a measure group is created automatically with the name of the form as the key. The following table describes the corresponding measure types:

Measure	Description
Server busy time	Measure, in seconds, that starts after the user has sent the complete request and ends when the user receives the first part of the response.
Round trip time	The time, in seconds, that it takes to send a request and receive the appropriate response. It also includes the connect and shutdown time when the connection is not persistent (the connection can persist if the Keep-Alive option is enabled or HTTP/1.1 is used).
Hits	The number of successful form submissions.
Request data sent	The amount of data (in KB) sent to the Web server during submission of a form. This measure also includes HTTP header and body content information. The count specifies the number of Web server requests sent.

Measure	Description
Response data received	The amount of data (in KB) received from the Web server in response to the submission of a form. This measure also includes HTTP response header and body content information. The count specifies the number of Web server responses received.

## Web Page Measures

The Web Page measure group contains the measure types related to Web pages. For each Web page that is downloaded during a load test, a measure group is created automatically with the name of the Web page as the key. The following table describes the corresponding measure types:

Measure	Description
Page time	The time, in seconds, that it takes to retrieve an entire Web page from the server, that is, for retrieving all HTML documents and all embedded documents, such as images, videos, and audio files.
Document download time	The time, in seconds, that it takes to retrieve all HTML documents of a Web page.
Server busy time	The time, in seconds, that the server takes to process a Web page request. This measure starts after a virtual user has sent the last byte of the request and ends when the user retrieves the first byte of the response.
Page data	The amount of data (in KB) for all requests and responses related to the request and retrieval of an entire Web page including all embedded objects.
Embedded objects data	The amount of data (in KB) for all requests and responses related to the request and retrieval of objects embedded on a Web page.

## Browser-Driven Measures

The following table describes the corresponding measure types:

Measure	Description
Dom interactive	Reflects the time span from the request until the browser has completed parsing all HTML elements and constructing the DOM.
Dom complete	Reflects the time span from the request until the browser has completed downloading and processing all resources (images, stylesheets, scripts, and so on).
Load End	Reflects the time span from the request until the browser has completed executing the <code>onLoad</code> function. As a final step in every page load process the browser sends an <code>onLoad</code> event, which triggers the <code>onLoad</code> function. Once the <code>onLoad</code> functions are executed, additional application logic might be executed.
First paint	Reflects the time span from the request until the page begins to display. This measure is available for Internet Explorer only.
Time to interact	Reflects the time span from the request until all TTI-relevant elements are available on the page. At this point in time, the user can interact with the web page.

Measure	Description
Action time	Reflects the time span from the request until the browser has completed downloading and processing all resources. The end of this time span varies, depending on the defined synchronization mode: If the synchronization mode <b>HTML</b> is defined, the action time ends when the <code>onLoad</code> function is completed. If the synchronization mode <b>AJAX</b> is defined, the action time ends during the <i>Asynchronous Application Logic</i> phase.

 **Important:** When the TTI feature is being used, the other measures listed here might not be available or might show incorrect values.

## CORBA-Specific Measures

The IIOP measure group contains the measure types related to IIOP operations that are called within a load-testing script. It provides response time and throughput ratings for the IIOP operation calls issued by the API function `IiopRequest`. With the `IiopObjectSetMeasures` function you can specify a prefix to the name of the measure groups associated with the operations of a certain CORBA object. If you decide not to do so, the name of the operation is used as the name of the corresponding measure group. The following table describes the corresponding measure types:

Measure	Description
Round trip time	The time measured, in seconds, from when the processing of the <code>IiopRequest</code> function call begins to when the processing of the reply of the CORBA object ends.
Server busy time	Measure, in seconds, that starts after a user has sent the last byte of the CORBA operation call and ends when the user retrieves the first byte of the response.   <b>Note:</b> Depending on the content of the application and the request, server busy time might begin before the last byte is sent and continue after the user has received the first byte. Therefore, server busy time is, in many cases, only an approximation.

For non-blocking IIOP requests, these measures are performed in the same way. This approach can lead to a situation where the sum of all your measured times exceeds the total time of your load test.

If a CORBA object decides to redirect a request, the `IiopRequest` function has to reissue this request to a different location. The result is two request-reply message pairs, where both are measured separately. As in the case of the measure times for the `IiopRequest` function, the sum of both (separately measured) request-reply pairs is supplied.

## SQL Database-Specific Measures

The following table describes the corresponding measure types:

Measure	Description
Parse	The time, in seconds, that it takes to prepare (Oracle terminology: parse) a SQL command. Preparing a SQL command usually includes a server round-trip during which the SQL command is sent to the server. The SQL command is then parsed and transformed into an internal format that can be reused on subsequent calls of that SQL command.

Measure	Description
	 <b>Note:</b> If you use the OraParse function with the ORA_DEFERRED option, the parsing of the SQL command is deferred to the execute step, and the server round-trip time is produced for the parse step. In this case, this measure type is meaningless.
Exec	The time, in seconds, in which a SQL command is executed. Executing a SQL command includes at least one server round-trip for sending the command and returning the result to enable the SQL command to be executed. Because of the considerable amount of data processing required on the server for a SQL command, executing SQL commands usually generates a major part of the load on the database server. A separate measure type in the SQL measure group is used for fetching data. Fetching data can either be included as part of the execute step or in a separate step. In the latter case, results are not included in the SQL measure group.
ExecDirect	The time, in seconds, in which a SQL command is prepared and executed.

## Middleware-Specific Measures

The TUXEDO measure group contains the measure types related to TUXEDO service calls performed in a load-testing script. It provides response time and throughput ratings for the TUXEDO service calls issued by the functions `Tux_tpccall`, `Tux_tpacall`, `Tux_tpsend`, `Tux_tprecv`, `Tux_tpgetrply`, `Tux_tpenqueue`, and `Tux_tpdequeue`. For each TUXEDO service used in these TUXEDO functions, a measure group is created automatically with the name of the service as the key. The following table describes the corresponding measure types:

Measure	Description
Response time	The response time, in seconds, for a TUXEDO service call. For synchronous service calls, the response time includes the time the server needs to process the call. For asynchronous service calls, Silk Performer reports the time that elapses between sending the request and receiving the reply, plus the time spent waiting for the request. Asynchronous service calls do not measure the time a server needs to execute the service. For send and request calls, Silk Performer measures the time for sending data across an open connection as well as the time for receiving data from an open connection. For queuing communication, both the time required for sending the service call to the queue and the time for retrieving the results of a service call are measured.
Request data sent	The amount of data (in KB) sent to the TUXEDO server for submission of a given service. Silk Performer measures the application data buffer size used by the service calls rather than the transferred bytes on the network.
Response data received	The amount of data (in KB) received from the TUXEDO server by a given service. Silk Performer measures the application data buffer size used by the service calls rather than the transferred bytes on the network.

## Citrix-Specific Measures

The Citrix Synchronization Points measure group states how many synchronization functions already have been executed. The main purpose of this measure is to visualize the progress of a transaction.

Measure	Description
Citrix Synchronization points	The number of executed Citrix synchronization functions (CitrixWaitForXXX).

## Custom Timer Measures

The Custom Timer measure group contains the timers that are defined in the load test script with the `MeasureStart` and the `MeasureStop` functions, as the following table shows.

Measure	Description
Response time	The time, in seconds, measured between <code>MeasureStart</code> and <code>MeasureStop</code> function calls.
Busy time	The busy time, in seconds, measured between <code>MeasureStart</code> and <code>MeasureStop</code> function calls.
	A busy time is the response time without any think time, wait time, and time necessary for function transformations.



**Note:** You can pause and resume timers with the `MeasurePause` and the `MeasureResume` function.

## Custom Counter Measures

The Custom Counter measure group contains the counters that are defined in the load-testing script with the `MeasureInc` function, as the following table shows.

Measure	Description
Custom counter	Counts values that are set with the <code>MeasureInc</code> function.

## Streaming Measures

The following table describes the corresponding measure types:

Measure	Description
Actual video duration	The play time of all downloaded media segments (the complete stream) in seconds.
Overall download time	The download time of all media segments in seconds.
Segments	The number of all downloaded media segments.
Segments critical D/P ratio	The number of media segments with a D/P ratio (download time/play time ratio) of more than 0.9.
Stream D/P ratio	The ratio of the download time to the play time (D/P ratio). A value over 0.9 indicates potential lagging/stuttering/buffering.
Stream size	The size of all downloaded media segments (the complete stream) in kilobytes.
Time to first segment	The time elapsed from loading the page to the complete download of the first media segment.

# Command Line Parameters

Performance Explorer offers a number of parameters that you can execute from a command line interface.

## Available Commands

The following table lists the Performance Explorer commands that you can use in a command line interface.

Command	Description
/TSD:<file>	Opens the specified .tsd file.
/ADDLOADTEST:<file>	Opens the load test results that contain the specified .tsd file.
/OVT:<template>	Sets the specified template as preselection of the overview report template when manually generating an overview report.
/MONITORDIR:<dir>	Sets the output directory for monitor writer files.
/MONCREATE:<dir>	Starts in Monitoring mode (BDL Script Generation).
/ACTION:STDGRAPH	Opens a predefined chart of type <i>General</i> . Uses the specified .tsd file (/TSD:<file>).
/ACTION:SELGRAPH	Opens a wizard that allows you to select a template for the chart. Uses the specified .tsd file (/TSD:<file>).
/ACTION:MONSERVER	Starts the server monitoring wizard.
/ACTION:STARTALLMONITORS	Starts all monitor charts.
/ACTION:STOPALLMONITORS	Stops all monitor charts.
/ACTION:OVERVIEWREPORT [<template>]	Generates an overview report for the specified .tsd file (/TSD:<file>). Can apply a template file. Make sure to specify a merged .tsd file (@m).
/ACTION:WORDOVERVIEWREPORT	Generates a Word overview report for the specified .tsd file (/TSD:<file>). Make sure to specify a merged .tsd file (@m). The parameters /WORDTEMPLATE and /WORDTARGET can be set optionally. Performance Explorer will be executed in GUI-less mode.
/WORDTARGET:<file>	The absolute path of the Word document (.docx) that will be generated. This parameter must be added before the parameter /ACTION:WORDOVERVIEWREPORT (see example below).
/WORDTEMPLATE:<file>	The absolute path of the Word document (.docx) that will be used as a template for the generated Word overview report. This parameter must be added before the parameter /ACTION:WORDOVERVIEWREPORT (see example below).

Command	Description
/EXPORTOVR :<file>	Exports a recently generated overview report view to <file>. Use file.htm for HTML or file.mht for WebArchive export.
/NOGUI	Specifies whether Performance Explorer is executed visibly or invisibly (GUI-less mode). /NOGUI functions appropriately only when /NOEXIT is used in a DOS-based command line.
/SINGLEINSTANCE [ :NOSTARTUP ]	Searches for an existing instance. When used with NOSTARTUP, no new instances are created.
/PEW:<workspace file>	Opens or creates the specified workspace. When used with /SINGLEINSTANCE, an existing instance is selected only when it uses the given workspace file. Otherwise, a new instance is created.
/COMMANDFILE:<xml file>	Load commands from the specified XML file.
/EXIT	Saves the .tsd file, then closes Performance Explorer without saving changes to the workspace.
/NOEXIT	Performance Explorer stays open after the command line execution.
/?	Displays Performance Explorer with allowed command line parameters.

## Examples

The following snippets illustrate how some of the above listed commands can be used.

If you add the parameter /TSD to the command PerfExp.exe, as shown in the first snippet, Performance Explorer opens and shows the specified .tsd file. Additionally, the parameters /ACTION:STDGRAPH and /ACTION:STARTALLMONITORS ensure that a predefined chart (of type *General*) is opened and all monitors are started automatically.

```
PerfExp.exe /TSD:"c:\test.tsd" /ACTION:STDGRAPH /
ACTION:STARTALLMONITORS
```

This second snippet starts Performance Explorer and opens the specified .pew file (/PEW). Then the monitor directory is set (/MONITORDIR) and all monitors contained in the .pew file are started automatically (/ACTION:STARTALLMONITORS). All these actions are performed without showing a user interface (/NOGUI). Finally, the parameter /SINGLEINSTANCE causes Performance Explorer to search for an existing instance and the parameter /NOEXIT ensures that Performance Explorer stays open when the command line is fully executed.

```
PerfExp.exe /PEW:"c:\test.pew" /MONITORDIR:"c:\temp\perf" /
ACTION:STARTALLMONITORS /NOGUI /SINGLEINSTANCE /NOEXIT
```

The third snippet causes the following behavior: Performance Explorer starts and the specified .pew file is opened (/PEW). While starting-up, Performance Explorer searches for an existing instance. If an instance is found, no new instance is started /SINGLEINSTANCE. The monitor directory is set (/MONITORDIR) and all monitors are stopped (/ACTION:STOPALLMONITORS). No user interface displays (/NOGUI). Finally, the .tsd file is saved and Performance Explorer closes without saving any changes to the workspace (/EXIT).

```
PerfExp.exe /PEW:"c:\test.pew" /MONITORDIR:"c:\temp\perf" /
ACTION:STOPALLMONITORS /NOGUI /SINGLEINSTANCE:NOSTARTUP /EXIT
```

The fourth snippet generates a Word overview report: First, Performance Explorer starts without displaying any user interface (caused by the parameter /ACTION:WORDOVERVIEWREPORT) and opens the specified .tsd file (/TSD). Make sure to specify a merged .tsd file (@m). Then, the path and name of the Word document that will be generated are specified (/WORDTARGET) as well as the path and name of the Word template that is to be applied (/WORDTEMPLATE). Finally, the parameter /ACTION:WORDOVERVIEWREPORT causes the report to be generated.

```
PerfExp.exe /TSD:"c:\test.tsd" /WORDTARGET:"c:\report.docx" /WORDTEMPLATE:"c:\template.docx" /ACTION:WORDOVERVIEWREPORT
```

## Command File

Besides submitting commands directly within the command line, you can also submit a command file in XML format. Specify the commands by using the following syntax.

```
Perfexp.exe /COMMANDFILE:<CmdFile>
```

The command file is designated for batch operation.

The structure of the XML file to use as a command file consists of a root node called `PerfExpCommandFile` and two optional sub-nodes called `OverviewReport` and `Command`.

### PerfExpCommandFile

Possible specifications in the root node `PerfExpCommandFile` include the following examples:

- `NoGui="True" / NoGui="False"`  
Specifies whether Performance Explorer launches visibly or invisibly.
- `LogFile="<filename>"`  
Specifies the name and location of the log file.

Different `Command` and `OverviewReport` nodes can reside below the root node.

### OverviewReport

The `OverviewReport` node specifies which .tsd file Performance Explorer uses. The following attributes are available:

- `File`  
Specifies which .tsd file to use. If omitted, the last file loaded by Performance Explorer is used.
- `Export`  
Specifies the output file to which the overview report is saved. Available formats include the following options:
  - `HTML (.htm)`
  - `WebArchive (.mht)`

- `Template`  
Specifies an optional template OVT file to use for the overview report.

### Command

The text of a `Command` node is interpreted as a command line parameter. Nodes are executed sequentially.

The following code shows an XML command file:

```
<?xml version="1.0" encoding="UTF-8" ?>
<PerfExpCommandFile NoGui="True"
   LogFile="c:\TEMP\Results\PerfExp.log">
```

```
<OverviewReport  
    File="c:\TEMP\Results\m@host@Simple.tsd"  
    Export="c:\TEMP\Results\Simple.mht"  
/>  
<Command>PEW: "c:\test.pew"</Command>  
<Command>SINGLEINSTANCE</Command>  
<Command>ACTION:STARTALLMONITORS</Command>  
<Command>EXIT</Command>  
</PerfExpCommandFile>
```

### Help command

Use the help command /? with the following syntax.

```
Perfexp.exe /?
```

Performance Explorer opens and displays the command line help.

## Customizing Visualization

Performance Explorer offers numerous possibilities to customize the appearance of charts, graphs, and reports. You can set default values for a variety of visualization settings and you can set specific options for individual charts, graphs, and reports.

## Default Options

You can change the default options for how charts, graphs, and reports are visualized.

### Setting Default Chart Options

1. Click the Performance Explorer Application Button (on the top left) and click **Options**.
2. Check the **Highlight Selected Series** check box to display the currently selected series with an emphasized, thicker line.  
 **Note:** While this option might be useful as long as you are working with charts, uncheck the check box before you export or print charts.
3. Check the **Overlay series** check box to assume that all time series that are displayed in a chart start at the same time.  
Use this option when you are viewing a chart that contains time series from tests with different starting times.
4. From the **Background colors** lists, select the colors that you want to use to enhance your charts.  
To use a custom color, click [...] to the right of the list box.  
Performance Explorer displays a vertical color transition from the first to the second color as the background image.
5. From the **Text and grid color** list box, select the color that you want to use for the axis information and the grid in your charts.  
To use a custom color, click [...] to the right of the list box.
6. In the **Scaling** area, you can enable the following features:
  - Enable **Scale values between 0 and 100** to display scaled values rather than real time series data values.
  - Enable **Use same scale factor for similar series** to set the same scale factor for series of the same type.
7. Click **OK**.

## Setting Default Time Series Data Options

1. Click the Performance Explorer Application Button (on the top left) and click **Options**.
2. Click the **Series** tab.
3. If you want to display null values by default, check the **Display NULL values** check box.  
On the **Series** tab, in the **Display** group, you can click **Null Values** to enable or disable null values separately for single measures. The check box in the **Options** dialog enables or disables null values for all measures.
4. From the **Show summary and counter values per** list box, select the time unit for which you want time series data to be displayed.
  - Select **Interval** if you want to display data for the computation interval contained in the .tsd file.
  - Select **Second** to scale displayed data to one-second intervals.

To change the default computation interval, use the **Time Series** tab of the **Profile Settings – Results** dialog.

5. From the **Default line width** list, select a line width to use whenever you add a new measure to a chart.
6. To display a chart marker for each measure value of each series, indicating the exact measure value, check the **Show chart markers** check box.
7. In the **Decimal digits** field, type the number of digits to display in the chart markers.  
Use chart markers to display exact measure values for each measure within a chart.
8. Click **OK**.

## Setting Default Chart Exporting Options

1. Click the Performance Explorer Application Button (on the top left) and click **Options**.
2. Click the **HTML View** tab.
3. In the **Chart picture resolution** area, select an option to determine the resolution of the image that is generated when you export a chart as an HTML document.  
The size of an image file is directly proportional to its resolution. As a result, images with higher resolutions take longer to transfer over the web than images with lower resolutions.
4. Use the **Picture type** area to select the picture format of your chart in an HTML document.
  - Click **JPEG** to include the chart in an HTML document as a JPEG file.
  - Click **BMP** (only with Internet Explorer) to include the chart in an HTML document as a BMP file.
5. Click **OK**.

## Setting Default Candle Graph Options

1. Click the Performance Explorer Application Button (on the top left) and click **Options**.
2. Click the **Candles** tab.
3. From the **Border color** list box, select the color you want to use for the candle borders in your current graph.  
To use a custom color, click [...] to the right of the list box.
4. In the **Candle width** spin box, enter the width for the candles in your current graph.
5. From the **Marker color** list box, select the color you want to use for the markers.  
To use a custom color, click [...] to the right of the list box.
6. In the **Marker height** spin box, specify the height for the markers.
7. Click **OK**.

## Setting Default Monitoring Options

1. Click the Performance Explorer Application Button (on the top left) and click **Options**.
2. Click the **Monitor** tab.
3. In the **Interval** field, set the length of the intervals at which the chart is updated.



**Note:** When modifying the monitor interval to a value lower than the default minimum interval during a run, the interval will be reset to the minimum monitor interval upon reopening the saved workspace.

4. In the **Page size** field, set the number of measured values currently displayed per graph on the horizontal axis.
5. In the **History size** field, set the number of measured values stored for the entire graph on the horizontal axis.
6. From the **Background colors** lists, select the colors that you want to use to enhance your charts. To use a custom color, click [...] to the right of the list box.  
Performance Explorer displays a vertical color transition from the first to the second color as the background image.
7. From the **Text and grid color** list box, select the color that you want to use for the axis information and the grid in your charts.  
To use a custom color, click [...] to the right of the list box.
8. Click **OK**.

## Setting Default TSD Writer Options

1. Click the Performance Explorer Application Button (on the top left) and click **Options**.
2. Click the **TSD-Writer** tab.
3. In the **Interval** field, select a time interval and a time unit for the time series data that is written.
4. Click **OK**.

## Setting Default Reporting Options

1. Click the Performance Explorer Application Button (on the top left) and click **Options**.
2. Click the **Reporting** tab.
3. To automatically display an overview report when you start exploring the results of a test, check the **Generate overview reports automatically** check box.  
An overview report contains the most important results of a test, such as the number of virtual users that were active, the number of transactions that were executed, and the number of errors that occurred.
4. To display null values in charts, check the **Show NULL values in charts** check box.  
If you uncheck this check box, Performance Explorer replaces each null value with a direct line from the preceding to the following value.
5. To use a previously stored template for the creation of overview reports, check the **Use template when creating a new overview report** check box, click [...], and select the template. The selected template is used to create a new overview report for all projects.
6. Click **OK**.

## Customizing Chart Display

Performance Explorer displays charts in a standard format but offers numerous possibilities for customization. You can enable scaling, overlay graphs, highlight graphs, export charts, and more. These display options can be found on the **Chart** tab, which is a contextual tab. It displays when you create or select a chart.

1. Create a chart and drag measures from the tree onto the chart.
2. On the **Chart** tab, in the **Chart** group, you have the following options:
  - **Scaling:** The scaling feature allows you to visually equalize graphs in a chart. This means that measures with big values and measures with small values can be displayed appropriately and a meaningful chart can be drawn. Without scaling, measures with small values would display as flat lines, while measures with big values would display as steep lines. To set a specific scaling factor for individual graphs, select a factor from the **Scaling** list, in the **Series** group on the **Series** tab.
  - **Overlay Graphs:** When you overlay graphs, all graphs are shifted to a common start time, regardless of the actual start time. Use this option when you are viewing a chart that contains measures from tests with different starting times.
  - **Logarithmic Y-Axis:** Changes the linear y-axis into a logarithmic y-axis.
  - **Highlight:** Emphasizes the selected graph with a thicker line.

 **Tip:** Highlighting graphs is useful while you work with and view charts in Performance Explorer. However, make sure to uncheck this option before you export or print a chart.
3. In the **Display** group, you have the following options:
  - **Series Points:** Displays the series points of all graphs.
  - **Zoom Out:** Resets the zoom level.
  - **Zoom & Scale:** Zoom & Scale is enabled by default. It allows you to zoom into the chart, which means that both the x- and y-axis are adjusted. When you disable Zoom & Scale, only the x-axis will be adjusted.
  - **Measure List:** Shows or hides the measure list. The measure list can also be regarded as the legend of the chart. However, it provides lots of additional information for each measure.
4. In the **Generate** group, you have the following options:
  - **Monitor Writer:** Creates a new monitor writer with the real-time measures from a monitor graph.
  - **VUser-based Chart:** Creates a new virtual user-based chart using the same measures as the active time-based chart. A time-based chart displays the measured values on the y-axis and the time on the x-axis. A VUser-based chart displays the number of virtual users on the x-axis.
5. In the **Data** group, you have the following options:
  - **Clear Monitor Chart:** Clears all graphs from the active monitor chart.
  - **View in Browser:** Shows the active chart as HTML document in a browser.

## Customizing Graph Display

Performance Explorer displays graphs in a standard format but offers numerous possibilities for customization. You can change colors, line width, and line styles. You can change scaling options, insert markers, and more. These display options can be found on the **Series** tab, which is a contextual tab. It displays when you create or select a chart.

1. Create a chart, drag measures from the tree onto the chart, and select a graph or a measure from the measure list.

2. On the **Series** tab, in the **Line Style** group, you have the following options:
  - **Style:** Select from various line styles.
  - **Color:** Select a color.
  - **Width:** Select a width.
3. In the **Series** group, you have the following options:
  - **Scaling:** The scaling factor allows you to visually equalize graphs in a chart. This means that measures with big values and measures with small values can be displayed appropriately and a meaningful chart can be drawn. Without scaling, measures with small values would display as flat lines, while measures with big values would display as steep lines. To enable auto scaling for all graphs in the chart, click **Scaling** in the **Chart** group on the **Chart** tab.
  - **Display as:** Select a display type for the selected graph. This can be **Line**, **Candle**, **Candle with sticks**, and **Area**.
  - **Source Type:** Select one of the following source types: **Average**, **Count**, **Sum**, **Maximum**, **Minimum**, **StdDev**.
4. In the **Display** group, you have the following options:
  - **Markers:** Markers show the measured values in the chart.
  - **Null Values:** Displays the null values for the selected graph.
  - **Measure Bounds:** Displays the bounds for the selected graph. You can configure these bounds in the **Automatic Threshold Generation** dialog before you start a load test.
  - **Performance Level Conditions:** Displays the performance levels for the selected graph. You can define and assign performance levels on the **Reports** tab.
  - **Normalized Values:** Displays normalized values for the selected graph. Normalized values are values per second.
  - **Standard Deviation Area:** Displays the standard deviation for the selected graph.
  - **Visible:** Click this check box to show or hide the selected graph. You can also use the check boxes in the measure list to show or hide graphs.

## Displaying Measure Bounds

Before you start a load test, you must define a lower and upper bound within the **Automatic Threshold Generation** dialog or directly within the script. The values of the bounds display in the Overview Report.

To display measure bounds in a chart:

1. Create or open a chart.
2. Drag measures that have defined bounds onto the chart.
3. Click a measure in the measure list to active it.
4. On the **Series** tab, in the **Display** group, enable **Measure Bounds**.

The chart shows a red and a yellow colored area. The red colored area is the area above the upper bound. The yellow colored area is the area between the upper bound and the lower bound.



**Note:** The colored areas only display if the measure exceeds the configured thresholds. Performance Explorer only shows graphical measure bounds for one measure at a time. When you display measure bounds, the scale of the chart may adjust automatically.

Performance Explorer shows graphical bounds for the **Average**, **Maximum**, and **Minimum** values of the following measures:

- Page and Action Timer - Action Time
- Page and Action Timer - Page Time
- Timer - Response Time[s]
- Transaction - Trans. (busy) Ok

# Displaying Performance Level Conditions

Before you can display performance levels in charts, you have to define and assign them. You can do so before a load test execution (proactive approach) or after an execution (reactive approach).

To display performance level conditions in a chart:

1. Create or open a chart.
2. Drag measures that have performance levels assigned onto the chart.
3. Click a measure in the measure list to activate it.
4. On the **Series** tab, in the **Display** group, enable **Performance Level Conditions**.

The chart shows colored boundaries for the conditions of the assigned performance level and a small legend describing them. For example: If the current performance level contains a condition like  $P90 < 0.2$ , the condition is fulfilled if 90 % of the graph lie below the shown boundary.



**Note:** Performance Explorer only shows performance level conditions for one measure at a time.

# Index

## A

adding  
  attributes 24  
adding results  
  from results folder 10  
  from Silk Central 10  
  overview 9  
  single result files 10  
agents  
  agent health  
  responsiveness 74  
files 5  
  health 74  
Apache server 47  
attribute pools 24

## B

BDL monitoring projects  
  advanced use case 41  
  basic use case 37  
  creating 35  
  creating new project 37  
  creating script 38, 43  
  in Performance Explorer 45  
initialization 46  
loops 45  
more best practices 46  
overview 34  
packaging the script 41  
planning project attributes 37, 41  
ThinkTimes and Wait 46  
tutorial 37  
using projects 41  
BEA WebLogic  
  monitoring 64

## C

charts  
  creating 10  
  creating monitors 70  
  customize appearance 88  
  exporting 17  
  setting default options 86, 87  
  viewing in web browser 16  
client-side measures  
  configuring 69  
  removing 70  
command line interface  
  command files 85  
  commands 83  
command-line interface  
  overview 83  
comparison reports  
  creating custom 21  
  creating region 20

creating user type 21  
custom 20  
region 20  
types of 20  
user type 20  
Connection Parameters dialog 31  
contact lost 15  
controller loses contact 15  
converting CSV to time-series data format 73  
CPU utilization  
  agent health 74  
CSV  
  automating file export 16  
  capturing remote-server performance data 72  
  conversion to time-series data 72  
  converting remote-server performance log data 73  
  exporting results in CSV format 15  
  generating performance-counter log on a remote  
    server 72  
customize  
  charts 88  
  graphs 89

## D

data source tree menu 9  
data sources  
  automatic detection 68  
  custom 66  
  defining 46  
  JMX 23  
  PerfMon 66  
  predefined 47  
  scanning 68  
  UNIX 27  
  wizards 46  
dialog boxes  
  System Selection 25

## E

error analysis graphs 12  
errors  
  analyzing 11  
executions 27  
exporting  
  charts 17

## F

files  
  CSV conversion to time-series data 72

## G

graphs  
  comparing 13

- customize appearance 89
- default options 86
- error analysis 12
- setting candle options 87
- GUI**
  - Performance Explorer tour 8
- H**
- hosts
  - changing 71
  - duplicating 71
  - editing credentials 71
- HTML
  - exporting reports 17
- HTML overview reports
  - creating 19
  - creating templates 19
  - templates 19
- I**
- IBM
  - DB2
    - monitoring 48
    - retrieving performance metrics 51
    - snapshot monitoring 49
  - WebSphere
    - performance measures 52
- IIOP 65
- IIS performance monitoring
  - available performance measures 54
  - troubleshooting 56
- intermediate files 5
- J**
- JMX
  - advanced queries 26
  - connecting 23
  - filtering beans 25
  - monitoring prerequisites 23
  - queries 25
- JMX management server 65
- L**
- launching product 7
- Linux commands 29
- load test results
  - merging 14
- M**
- Management Information Base (MIB) 31
- MBeans specifying 24
- measure bounds
  - displaying 90
- measurements
  - correlating 12
- measures
- browser-driven 79
- Citrix 81
- CORBA 80
- custom counter 82
- custom timer 82
- database 77
- general 73
- IOP 76
- Internet 75
- middleware 81
- overview 73
- SQL database 80
- summary 73
- transaction 77
- TUXEDO 81
- Web 76
- Web form 78
- Web page 79
- MemFree 29
- memory utilization
  - agent health 74
- merged files 5
- MIB 31
- MIB files 68
- Microsoft SQL Server performance monitoring
  - available performance measures 56
- monitor
  - charts 70
- monitoring
  - IBM WebSphere application servers 52
  - interfaces 66
  - monitor charts 70
  - monitor writers 70
  - real-time 22
    - real-time monitoring
    - overview 22
  - setting default options 88
  - windows machines 22
- monitoring projects
  - infrastructure monitor 37
- multiple executions 27
- O**
- OIDs 31
- Oracle
  - monitoring 60, 62
- Oracle database monitoring
  - adding Oracle counters 60
  - Installing Oracle Client and Oracle Counters 60
  - known issues 62
  - monitoring Oracle performance (Perfmon) 60
  - performance counters 61
- Oracle Forms
  - client-side performance measures 59
  - monitoring 58, 59
- overview reports
  - HTML 18
  - word 18
- P**
- PerfMon 66

performance  
monitoring 31  
Performance Data Collection Engine 37

Performance Explorer  
overview 4  
real-time monitoring 4  
results analysis 4  
terminology 4  
performance levels  
display in charts 91  
performance measures  
WebSphere 52  
product  
starting 7

## **Q**

queries  
JMX 26  
REXEC 27  
running 26  
UNIX 27

## **R**

real-time  
files 5  
remote executions 27  
remote servers  
capturing performance data 72  
converting performance log data 73  
generating performance-counter log 72  
reporting  
types of 18  
reports  
comparison 20  
creating 11  
custom comparison 20, 21  
default options 86  
exporting command-line 17  
Performance Explorer 11  
region comparison 20  
setting default options 88  
templates 19  
user type comparison 20, 21  
viewing in web browser 17  
resampling  
load test results 14  
result files 14  
responsiveness  
agent health 74  
results  
analysing 11  
exporting 15  
processing 13  
REXEC  
monitoring 66  
overview 27  
REXEC queries 27  
root cause analysis 12

## **S**

SAP  
monitoring 63

saved queries 26  
secure shells 33  
server performance  
monitoring 31  
Silk Central  
integration 86  
Silverstream 47  
single executions 27  
SNMP  
adding datasource 67  
monitoring 64, 66  
querying 67  
SNMP support  
data source definition 32  
starting product 7  
System Selection  
dialog box 25

## **T**

tables  
viewing in web browser 17  
technologies  
monitoring 22  
template  
creating a word overview report 18  
templates  
assigning 19  
test results  
exploring 9  
time series  
data writer  
setting default options 88  
setting data options 87  
time-series  
file types 5  
time-series data  
conversion from CSV 72  
tour  
user interface, Performance Explorer 8

## **U**

UNIX queries 27  
user files 5  
user interface  
Performance Explorer overview 8

## **W**

Web browsers  
viewing results 16  
WebLogic (JMX) monitoring  
IIOP 65  
JMX management server 65  
welcomePE 4  
windows machines  
monitoring requirements 22  
word overview reports  
template 18  
viewing 18

workflow bar

    Performance Explorer 8  
writers

creating monitors 70