DBPLUS
better performance

DBPLUS
Performance Monitor for Microsoft SQL Server
description of changes in version 2022.2

Date: July 10, 2022

DBPLUS
better performance

Table of contents

Below is a list of changes to the DBPLUS Performance Monitor system for Microsoft SQL Server database monitoring.

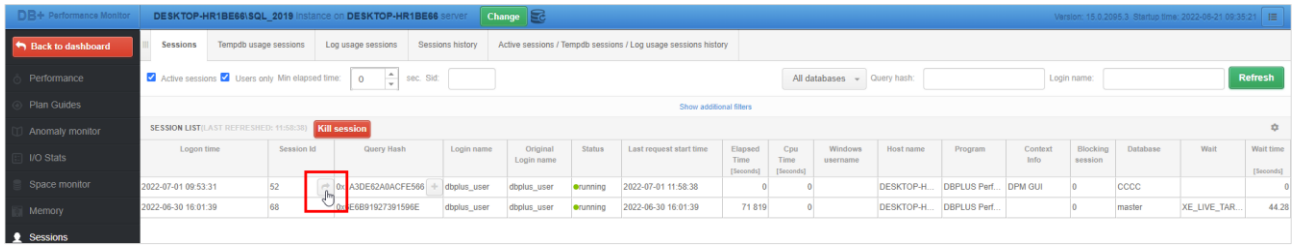
New in 2022.2 version

1 Session Trace profiler

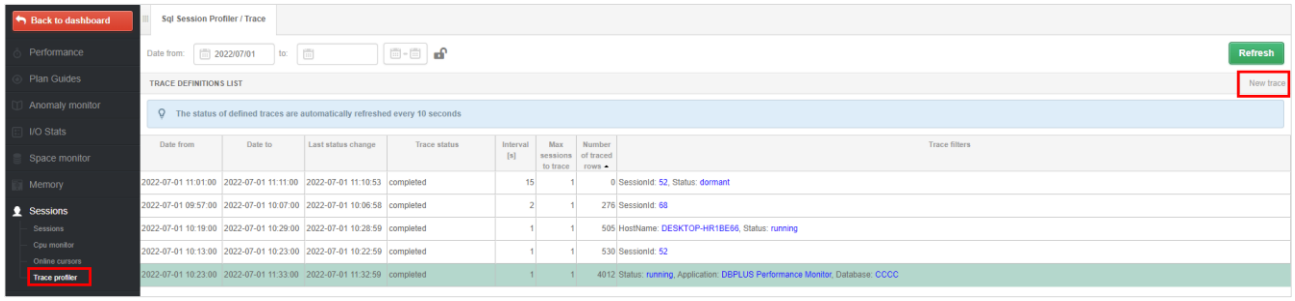
In the latest version of the application, we have added the functionality of session monitoring using Trace Profiler. This option is available from the level of each monitored instance from the ***Sessions*** menu. The functionality allows you to monitor the users' session regardless of the functionality available in the application by default.

Session monitoring can be started in two ways:

- by clicking the button in the ***Session id*** column for a dedicated session on the online session screen:



- from the **Sessions> Trace profiler** menu by clicking **[New trace]:**



In both cases, after clicking, a dedicated window will appear in which we can set the conditions with which the session monitoring is to be started.

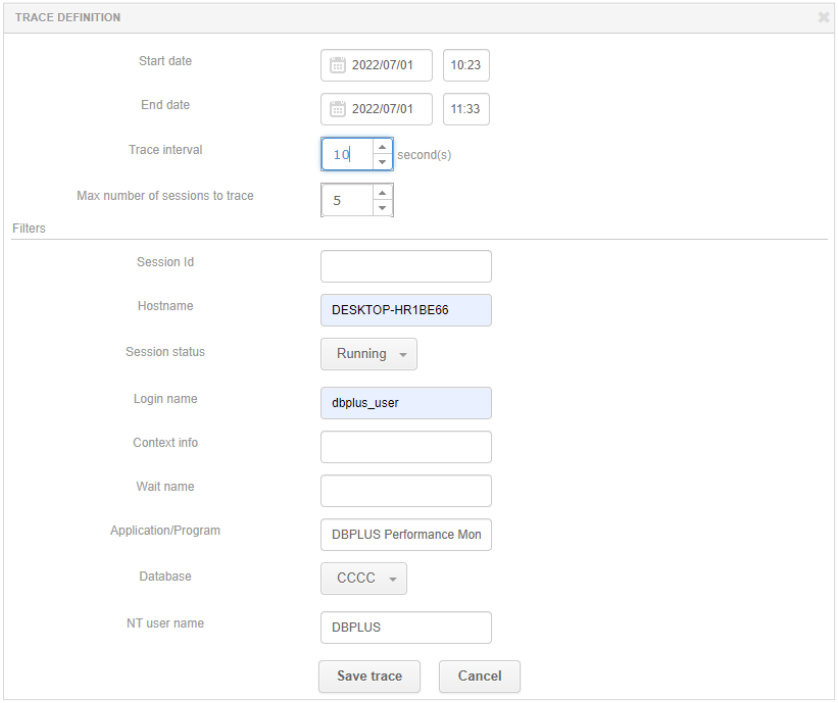
The basic settings include:

- Start date, End date - start and end time of session monitoring,
- Trace interval - the interval of retrieving information about sessions,
- Max number of session to trace - the maximum number of monitored sessions.

The minimum interval for session monitoring is 1 second. It means that every second a command with given conditions will be executed checking information about the session.

By setting up session monitoring, we can configure filters for:

- Session Id – session ID
- Hostname
- Session status – session status
- Login name – login name
- Context Info
- Wait name – name of the wait
- Application/Program – the name of the application / program
- Database – database name
- Nt user name – Windows user name

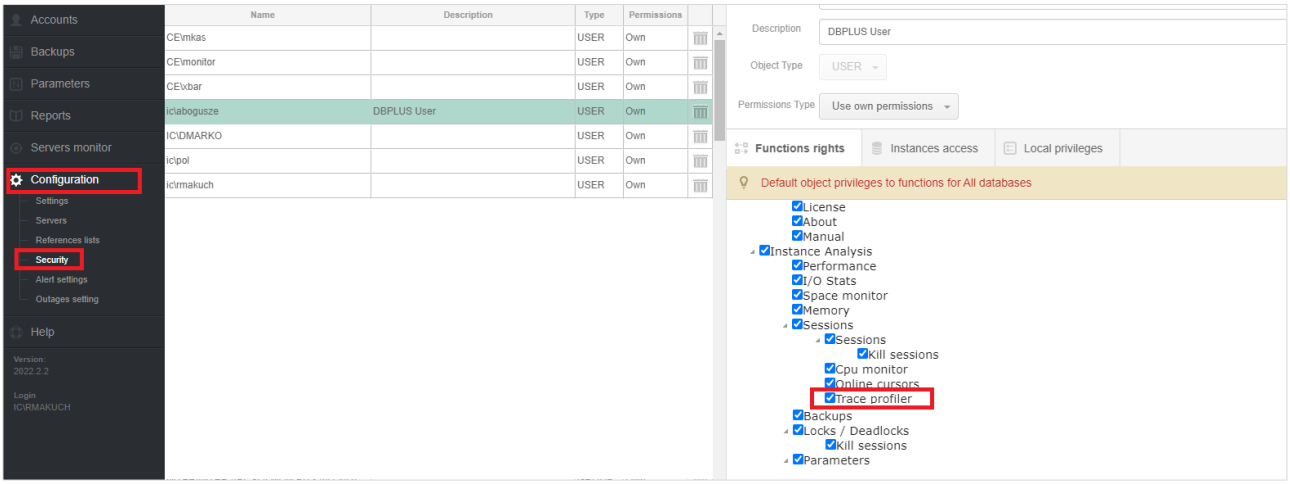


After starting the session monitoring, the session data will be presented at the bottom of the screen after clicking on the selected row. The monitoring screen refreshes automatically once every 10 seconds. During the session monitoring, the user has the option to modify the filters or stop the monitoring currently being performed.
When the session monitoring is completed, the (Trace status) will change to completed.

The user can delete previously performed monitoring sessions, or wait for them to be deleted automatically based on the parameter set for the length of data storage in the repository in the menu ***Configuration> Settings> History settings section, Session / Locks statistics***.

Access to the Trace Profiler menu

If the **Trace profiler** option is not visible in the **Sessions** menu, please verify that the access to this menu has been granted (by default the Trace profiler menu is invisible). To grant access, on the Dashboard screen in the main menu, select Configuration> Security and then grant appropriate permissions.

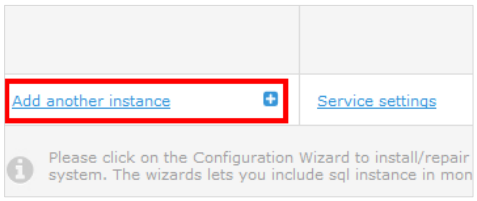


2 Google Cloud support

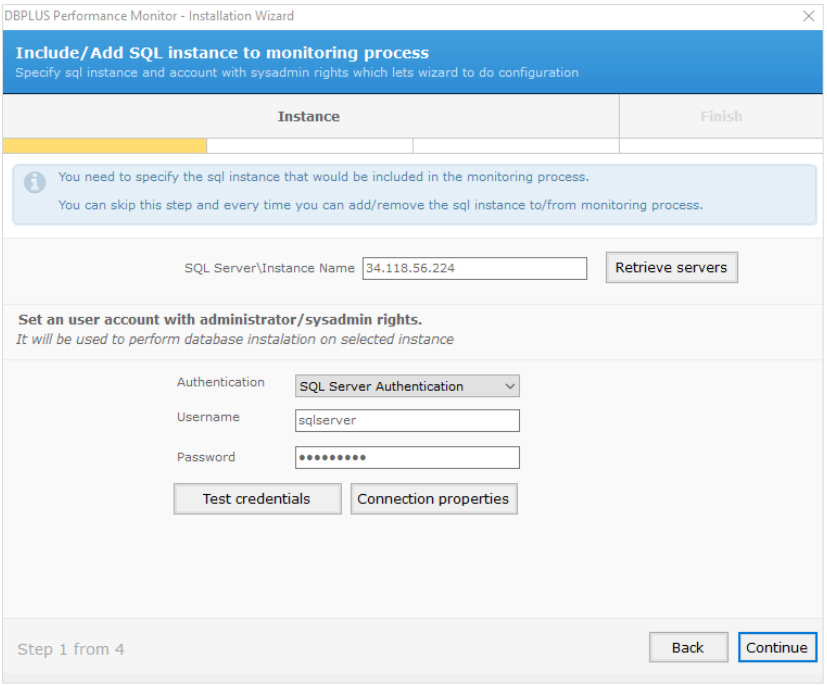
In the latest version, we have added support for monitoring SQL instances on the Google Cloud platform. The range of functionalities available in the DBPLUS Performance Monitor application is the same as for the on premise version. Below is a short scenario of adding a SQL instance to DBPLUS monitoring.

Adding a SQL instance on the Google Cloud platform

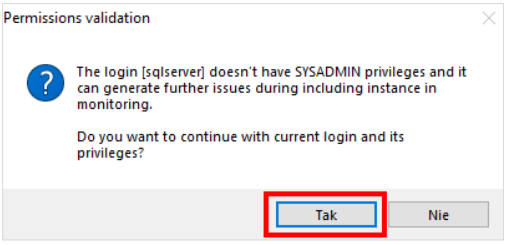
SQL instances installed on the Google Cloud platform for monitoring are added by clicking the **[Add another instance]** button from the DBPLUS Configuration Wizard program.



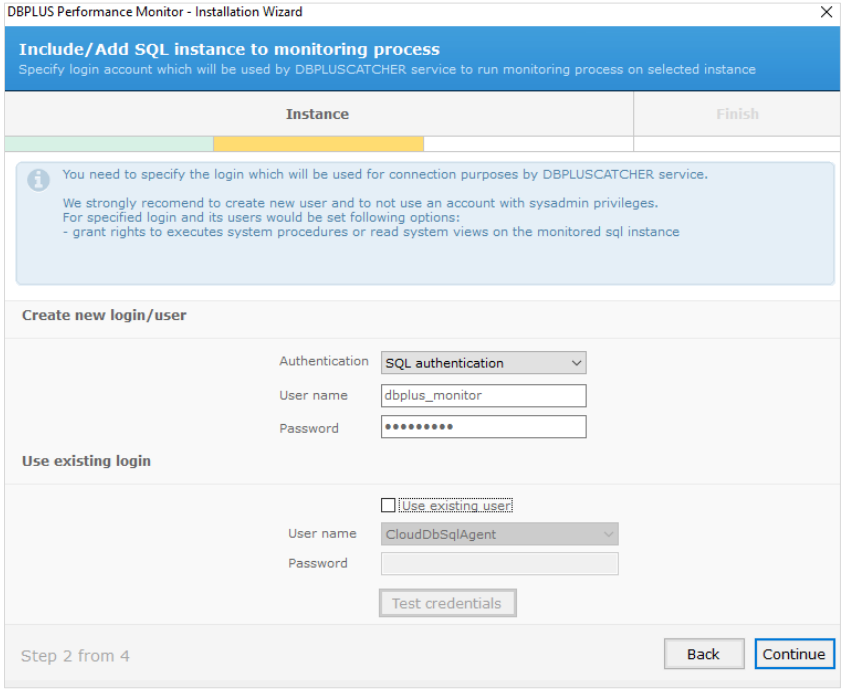
On the next screen, complete the connection data of the SQL instance and the user name. In the SQL instance on the Google Cloud platform, we do not have access to a user with **SYSADMIN** privileges, so on the screen we complete the data of the user who is to create a new user as well as to grant the privileges needed to monitor the instance.



After clicking the **[Continue]** button, accept the selection and proceed to the next stage of configuration.



In the next step, we choose the option to create a new one (recommended option) or to select an existing user for monitoring. Then we receive information about the recommended functionalities that will be turned on by monitoring.



Finally, we accept the configuration and finish the installation by clicking the **[Finish]** button.

3 REST API – Performance Monitor

In the latest version of the application, we have added new methods to the REST API:
- get PerfCounters performance statistics information,
- get information about IO Stats disk array statistics.

3.1. Performance Counters

Method	GET
Database platform	Oracle, MS SQL (from version 2022.2)
URL	/perfcounters
Action	Gets information about performance statistics (Performance Counters)
Input data:	
view	▪ last_snapshot ▪ history * in the case of the history option, additional filters must be completed (group_time, date_from, date_to)
performance_counter	Counter name * supports condition like '% name%' (returns max. 3 statistics that meet the condition)
server_id	Server identifier in the DBPLUS repository
group_time	Grouping of returned data (return date format): ▪ year (YYYY) ▪ month (YYYY-MM) ▪ day (YYYY-MM-DD) ▪ hour (YYYY-MM-DD HH24) ▪ snap (YYYY-MM-DD HH24:MI:SS)
date_from	Date from which statistics will be downloaded *In format YYYY-MM-DD HH24:MI:SS
date_to	Date by which statistics will be downloaded *In format YYYY-MM-DD HH24:MI:SS
Output data:	
PerfCounterList	Counters list
PerfCounterRecord	Statistic record
ClassGroup	Statistics class

Name	Statistics name
Value	The value of the statistics
Logdate	Date for the given statistic

Example data output [xml]:

```
<?xml version="1.0" encoding="utf-16"?>
<Root xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <PerfCounterList>
    <PerfCounterRecord>
      <ClassGroup>Debug</ClassGroup>
      <Name>background timeouts</Name>
      <Value>0</Value>
      <Logdate>2022-06-02 08</Logdate>
    </PerfCounterRecord>
    </PerfCounterList>
  <Response>
    <Status>OK</Status>
    <Message />
  </Response>
</Root>
```

Example data output [JSON]:

```
{
    "PerfCounterList": [
        {
            "ClassGroup": "SQL Statistics",
            "Name": "Batch Requests/sec",
            "Value": 2227,
            "Logdate": "15.06.2022 10:21:33"
        }
    ],
    "Response": {
        "Status": "OK",
        "Message": ""
    }
}
```

3.2.IO Stats

Method	GET
Database platform	PostgreSQL, Oracle, MS SQL (from version 2022.2)
URL	/iostats
Action	Gets information about IO statistics

Input data:

View	▪ last_snapshot ▪ history * in the case of the history option, additional filters must be completed (group_time, date_from, date_to)
server_id	Server identifier in the DBPLUS repository
group_time	Grouping of returned data (return date format): ▪ year (YYYY) ▪ month (YYYY-MM) ▪ day (YYYY-MM-DD) ▪ hour (YYYY-MM-DD HH24) ▪ snap (YYYY-MM-DD HH24:MI:SS)
group_type	Group type:

	▪ Database ▪ Tablespace ▪ File * no field completed means options without grouping
date_to	Date by which statistics will be downloaded *In format YYYY-MM-DD HH24:MI:SS
date_from	Date from which statistics will be downloaded *In format YYYY-MM-DD HH24:MI:SS
database	Database name * used in MSSQL, PostgreSQL
tablespace	Tablespace name *used in Oracle
file_name	File name *used in MSSQL, Oracle

Output data:

IODataList	List of IO statistics
IOStatRecord	Record of IO statistics
Database	Database name
Tablespace	Tablespace name
File	File name
Logdate	Date for the given statistic
NumberOfReads	Number of reads
NumberOfWrites	Number of writes
BytesReads	The number of bytes read
BytesWrites	Number of bytes written
MBytesReads	The number of reads in [MB]
MBytesWrites	The number of writes in [MB]
BlockReads	The number of blocks read
BlockWrites	Number of blocks written
ReadTime	Reading time
WriteTime	Writing time
SingleMByteReadTime	Single MB read time
SingleMByteWriteTime	Single MB write time
SingleBlockReadTime	Single block read time
SingleBlockWriteTime	Single block write time

Example output data [xml]:

```
<?xml version="1.0" encoding="utf-16"?>
<Root xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <IODataList>
        <IOStatRecord>
            <Database>All databases</Database>
            <Tablespace>N/A</Tablespace>
            <File>All files</File>
            <Logdate>2022-06-15 11:28:50</Logdate>
            <NumberOfReads>739</NumberOfReads>
            <NumberOfWrites>637</NumberOfWrites>
            <BytesReads>48234496</BytesReads>
            <BytesWrites>5242880</BytesWrites>
            <MBytesReads>46</MBytesReads>
            <MBytesWrites>5</MBytesWrites>
            <BlockReads>5888</BlockReads>
            <BlockWrites>640</BlockWrites>
            <ReadTime>1.61</ReadTime>
            <WriteTime>2.112</WriteTime>
```

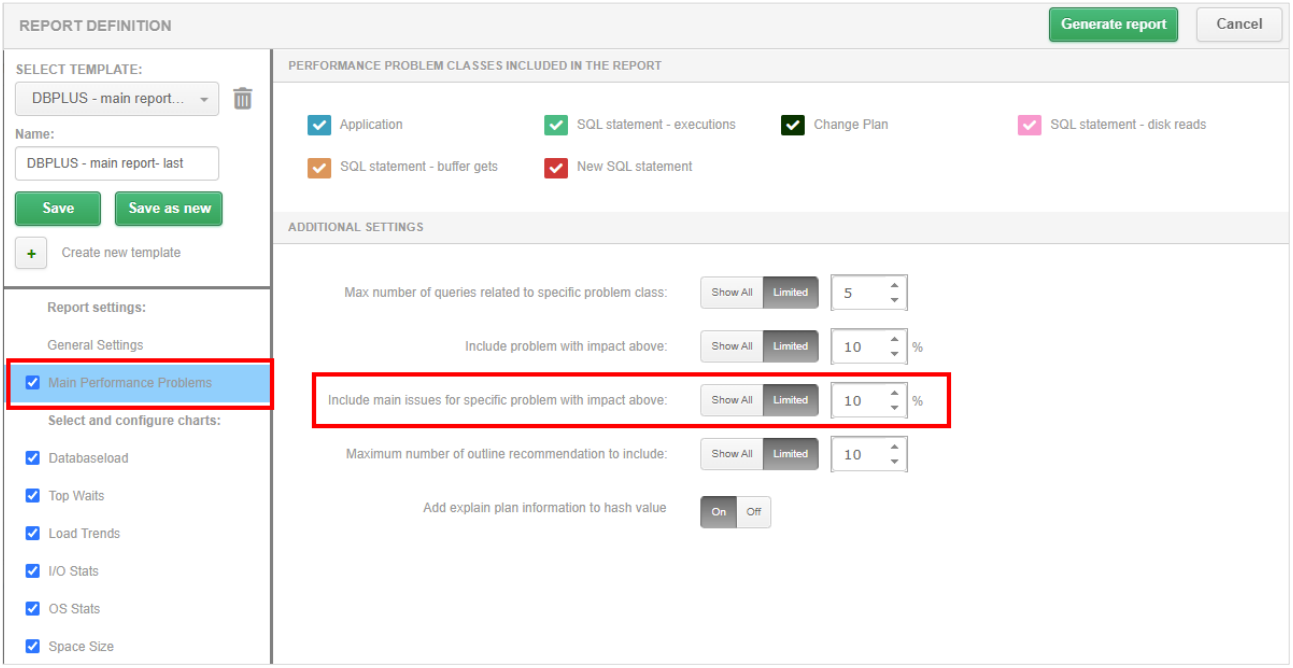
```
            <SingleMByteReadTime>0.035</SingleMByteReadTime>
            <SingleMByteWriteTime>0.4224</SingleMByteWriteTime>
            <SingleBlockReadTime>0.000273</SingleBlockReadTime>
            <SingleBlockWriteTime>0.0033</SingleBlockWriteTime>
        </IOStatRecord>
    </IODataList>
    <Response>
        <Status>OK</Status>
        <Message />
    </Response>
</Root>
```

Example output data [JSON]:

```
"IOStatList": [
        {
            "Database": "All databases",
            "Tablespace": "N/A",
            "File": "%C:\\Program Files\\Microsoft SQL Server\\MSSQL15.SQL_2019\\MSSQL\
\DATA\\ABCD%%",
            "Logdate": "2022-06-28 14:17:10",
            "NumberOfReads": 85,
            "NumberOfWrites": 25875,
            "BytesReads": 4194304,
            "BytesWrites": 238026752,
            "MBytesReads": 4,
            "MBytesWrites": 227,
            "BlockReads": 512,
            "BlockWrites": 29056,
            "ReadTime": 0.343,
            "WriteTime": 69.303,
            "SingleMByteReadTime": 0.08575,
            "SingleMByteWriteTime": 0.305300,
            "SingleBlockReadTime": 0.000670,
            "SingleBlockWriteTime": 0.002385
        }
    ],
    "Response": {
        "Status": "OK",
        "Message": ""
    }
}
```

4 Anomaly monitor improvements

In the latest version of the application, we have made changes to the process of generating the Anomaly Monitor report containing the performance anomalies detected by the Performance Monitor applications. The changes consist in including in the report only the largest cases of a given problem in the analyzed period of time. The change consists in adding a dedicated parameter which is responsible for filtering out the occurrences of a given problem in the period for which the report is generated and taking into account only those occurrences that exceed the threshold indicated in the parameter. The parameter value is set to **10%** by default. This will allow the Anomaly Monitor report not to include information about problem occurrences that are not the main performance problem in the monitored instance.



If the user would like the report to describe all performance problems and all occurrences of the problem, it is enough to change the value for a given parameter to **Show All**, which will mean that all problems will be described in the report.

5 Bug fixes and improvements

5.1. Improved SQL Parser

In the latest version we have added a fix pack related to SQL Parser parsing and highlighting of objects in queries. DBPLUS SQL Parser is a functionality that allows the user to analyze queries more easily by selecting the analyzed objects in the plan and the query content.

5.2. Remove depreciated data types from repository database objects

In the lates version, we have removed depreciated data types like text, image, ntext in repository database and change to appropriate types like varchar(max), nvarchar(max), varbinary(max).

5.3. Implementation of TLS1.1 and TLS1.2 support

Support for TLS 1.1 and TLS1.2 (Transport Layer Security) has been added to the latest version of the application. The latest version of the application has been coded in .Net 4.7.2.