

DBPLUS™
better performance

DBPLUS Performance Monitor™
for PostgreSQL®
User's Manual

March 2020

Table of contents

1	Introduction	4
1.1	DBPLUS Technical Support	5
1.2	System architecture	5
1.3	System requirements	6
1.4	Installation of DBPLUS Performance Monitor	7
2	System configuration	10
2.1	The main configurator screen	10
2.2	Setting up DBPLUSPOSTGRESCATCHER monitoring service	12
2.3	System Repository configuration	14
2.3.1	PostgreSQL Instances Configuration	14
2.3.2	Selection of the Repository database	15
2.3.3	Creating a monitoring user	16
2.4	IIS service configuration	18
2.4.1	Configuration of SSL in the IIS (additional option)	18
2.5	User application configuration	23
2.6	Configuration summary	24
3	Additional functionalities	26
3.1	Add PostgreSQL Instance for monitoring	26
3.1.1	Add an PostgreSQL instance without a Superuser (AWS)	31
3.2	Modifying existing user's privileges	32
4	System Upgrade	34
4.1	Upgrade to the latest version	34
5	License	36
6	Working with program	37
6.1	„Dashboard“	37
6.1.1	Information bar	37
6.1.2	The Summary area	38
6.1.3	Servers and instances area	39
6.1.4	Details of PostgreSQL database performance	40
6.1.5	Dashboard – various forms of presentation	45
6.1.6	Additional functionalites	46
6.2	„Instance Analysis“ Menu	52
6.2.1	„Performance“ Menu – Instance Analysis	52
6.2.2	Anomaly Monitor Menu	80
6.2.3	„I/O Stats“ Menu	84
6.2.4	„Space Monitor“ Menu	85
6.2.5	„Bg Writer Stats“ Tab	87
6.2.6	„Sessions“ Menu	87
6.2.7	„Locks“ Menu	92
6.2.8	„Parameters“ Menu	93
6.2.9	„Logs“ Menu	95
6.2.10	„Reports“ Menu	97
6.3	Space monitor Menu	98
6.4	Parameters Menu	98
6.5	Reports Menu	99
6.6	Servers Monitor Menu	99
6.6.1	Application architecture	99
6.6.2	Schedules outages	100
6.6.3	Scheduled works	101

6.6.4	Logs.....	102
6.7	Menu Configuration	103
6.7.1	Settings	103
6.7.2	Instances	103
6.7.3	Reference lists	104
6.7.4	Security.....	104
6.7.5	Alert Settings	107
6.8	Help Menu	121

1 Introduction

What is DBPLUS Performance Monitor?

DBPLUS Performance Monitor™ tool is the software used for monitoring and analyzing the Oracle, MS SQL Server and PostgreSQL.

Using DBPLUS Performance Monitor, you can:

- track trends of database server load and the individual components: Wait, I/O and other,
- identify performance issues of PostgreSQL databases,
- track trends of performance SQL queries,
- analyze data and present them in graphical form,
- watch in real time active user sessions,
- observe the status of full and incremental databases backups,
- troubleshoot a non-optimal SQL queries,
- report database problems

and many, many more...

Question:

"Why do database work too slow in any specified period of time?"

will never be left without an answer!

1.1 DBPLUS Technical Support

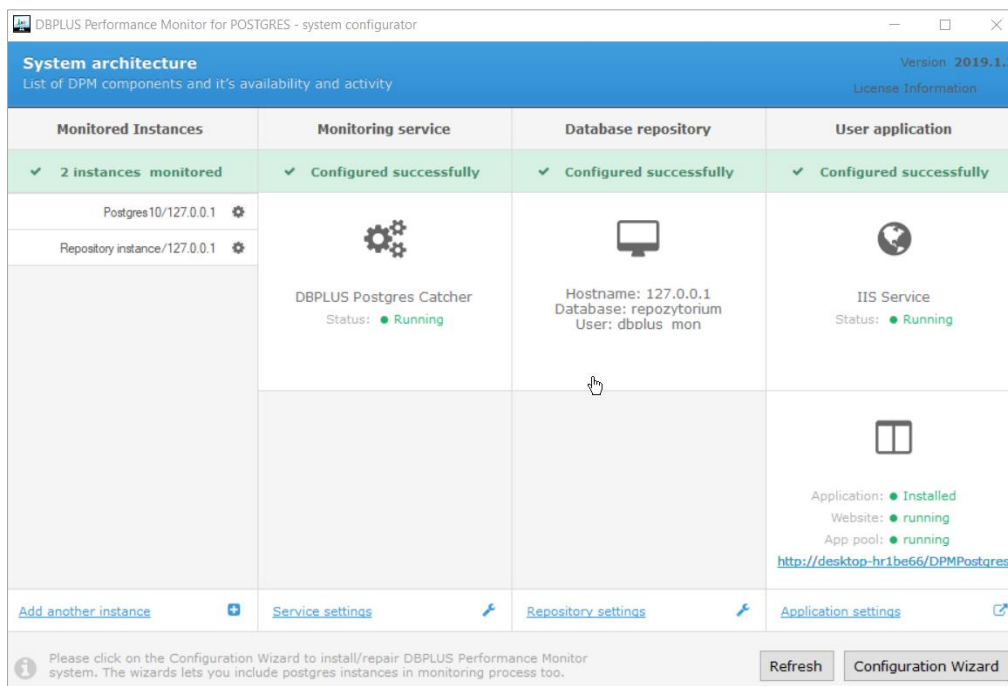
Technical support provides the access to new software updates published 4 times a year as well as to engineers' - help in PostgreSQL database diagnosis (by using **DBPLUS Performance Monitor** software).

1.2 System architecture

The system is designed in client-server architecture and in the presented solution we can distinguish the following components:

- **SQL Instances** - a list of SQL instance covered by the monitoring,
- **Server program** - an application running as a windows service, which consists of a set of procedures performed on individual SQL Instances. The aim of the program is to run periodically procedures, which are responsible for collecting basic data about SQL servers' performance. According to the DBPLUS nomenclature, program is called **DBPLUSPOSTGRESCATCHER** and one-up cycle within the service **DBPLUSPOSTGRESCATCHER** is called "a snap".
- **Repository** - selected SQL instances stores performance statistics of monitored SQL instances. Collected statistics are the result of the work of **DBPLUSCATCHER** service.
- **Application** - this is a client of the system, which implements user interface which allows to implement functionality of the system, i.e. monitoring review, performance analysis, query execution statistics reports, the current sessions of database, chart of server load, etc. The application is made in web technology using IIS application server and it is accessible from a web browser.

DBPLUS Performance Monitor requires the installation and configuration of each of the elements to ensure full functionality of the solution. Below we present a general model of the system:



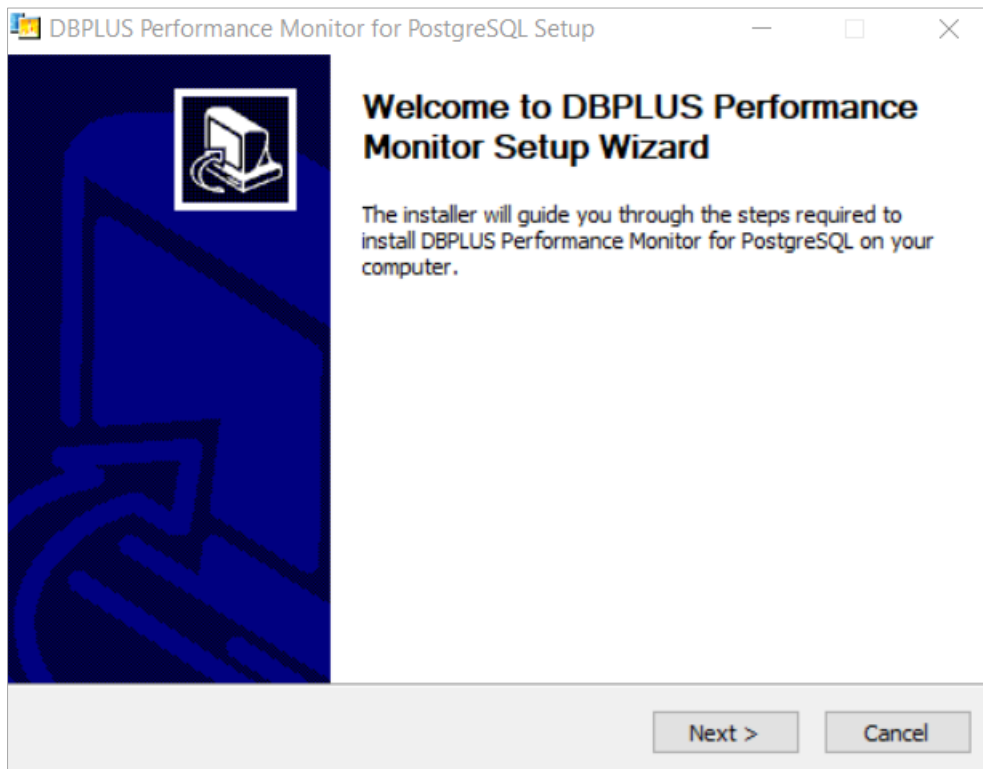
IMPORTANT: Application DBPLUS Performance Monitor requires the installation and configuration on any given server / computer in the company. During normal use of application, system does not require any installation on the user's local computers.

1.3 System requirements

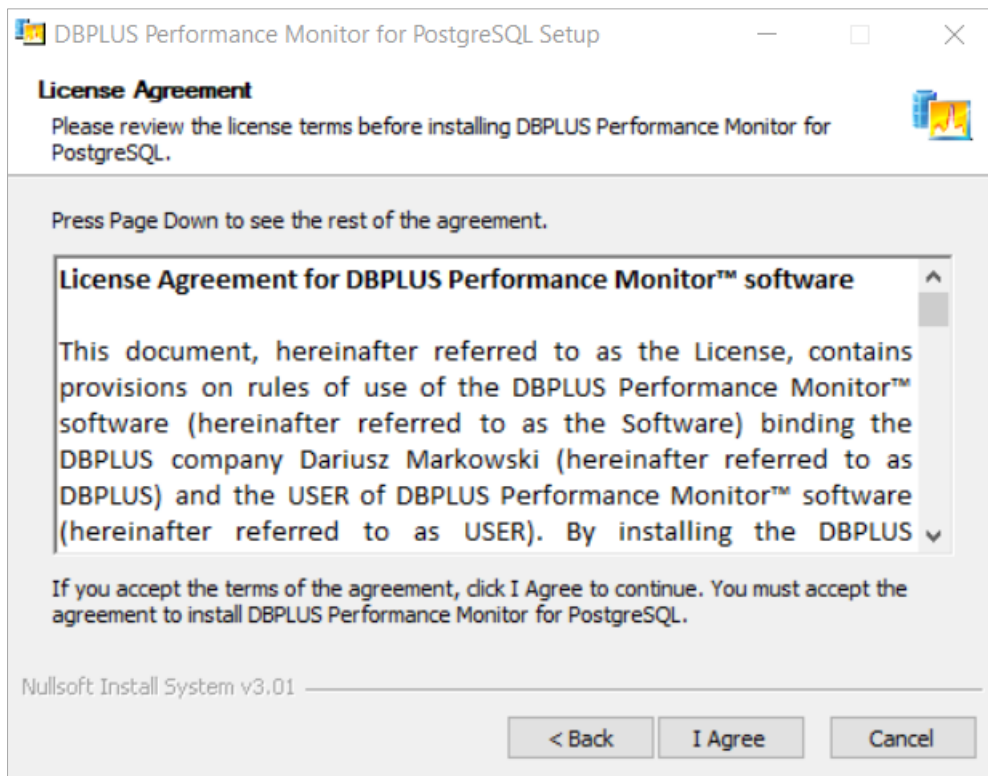
Parameter	Description
Monitoring of PostgreSQL instances	Supported types of monitored PostgreSQL instances: <ul style="list-style-type: none"> All versions of PostgreSQL are monitored (starting from version 9.4 and above)
Server operating system with installed DBPLUS PERFORMANCE MONITOR software	<p>Servers:</p> <ul style="list-style-type: none"> Windows Server 2008 and above <p>Also:</p> <ul style="list-style-type: none"> Windows 7 and above <p>Additional requirements</p> <ul style="list-style-type: none"> .NET Framework 4.0 (installer on a server) User with Administrator privileges <p>Scale and layout: Screen resolution: 800x600 and above Text size 100%</p> <p>On the server / computer with DBPLUS PERFORMANCE MONITOR software is not required to install PostgreSQL components.</p>
Server's hardware requirements with installed DBPLUS Monitor (20 PostgreSQL instances)	<ul style="list-style-type: none"> 4 CPU 8 GB RAM HD – no requirements <p>When monitoring 20 PostgreSQL instances:</p> <ul style="list-style-type: none"> DBPLUSPOSTGRESCATCHER Monitoring Service consumes at a level 2 GB RAM, IIS do 500 MB RAM. Assign 4 CPU due to the multithreading services, monitoring a number of instances, plus user applications. DBPLUS Software up to 30 MB.
The impact of the system to PostgreSQL servers	<p>The system generates an average load of less than 1% dependent on generally accepted "quality" of databases</p> <p>Repository Instance: As a result of the installation of repository on a selected database, the system sets up:</p> <ul style="list-style-type: none"> Database with DBPLUS objects – tables, functions. User with permissions to read system views and execute query plan. <p>Attention! The DBPLUS database user doesn't have permission to read data from the schemas of other database users.</p> <p>Monitoring instance: As a result of inclusion in the monitoring process a specific instance it is set up the user used only to connect with a given instance</p>
User interface	<p>The user application is accessible from a web browser. Supported browsers include:</p> <ul style="list-style-type: none"> Internet Explorer (ver. 9 and above) Google Chrome Mozilla Firefox Opera Edge

1.4 Installation of DBPLUS Performance Monitor

DBPLUS Performance Monitor is available on DBPLUS server through the provided link. User can install DBPLUS Performance Monitor by double-clicking downloaded EXE file: [dpmPostgresInstaller.exe](#)

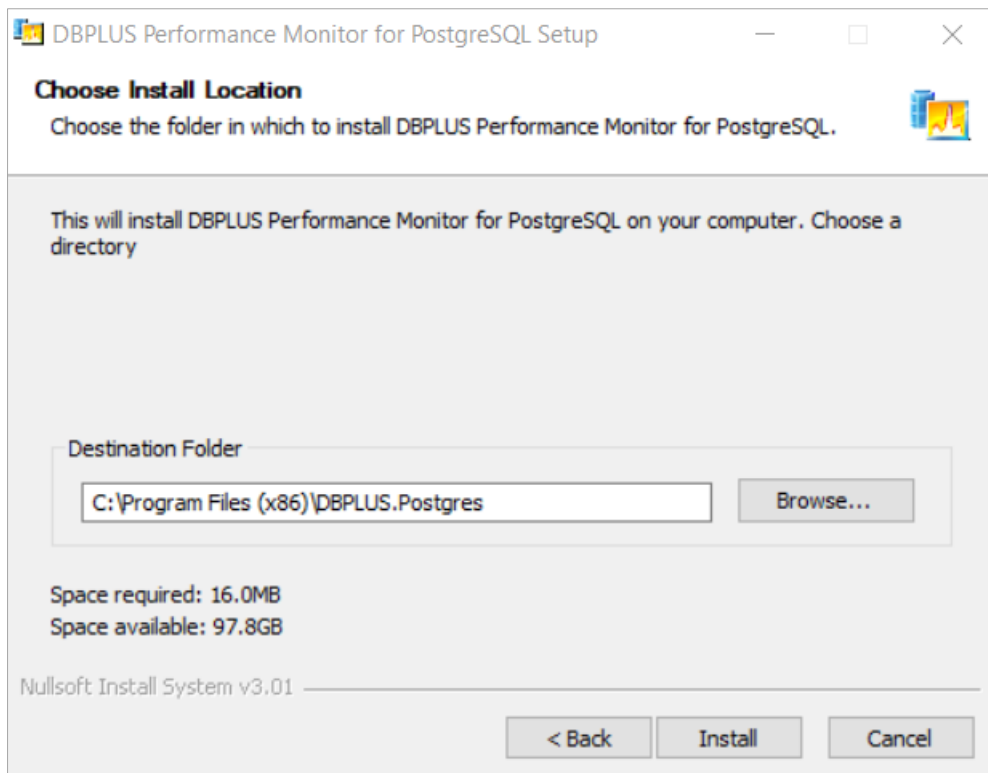


By clicking "Next" we get information about the license:

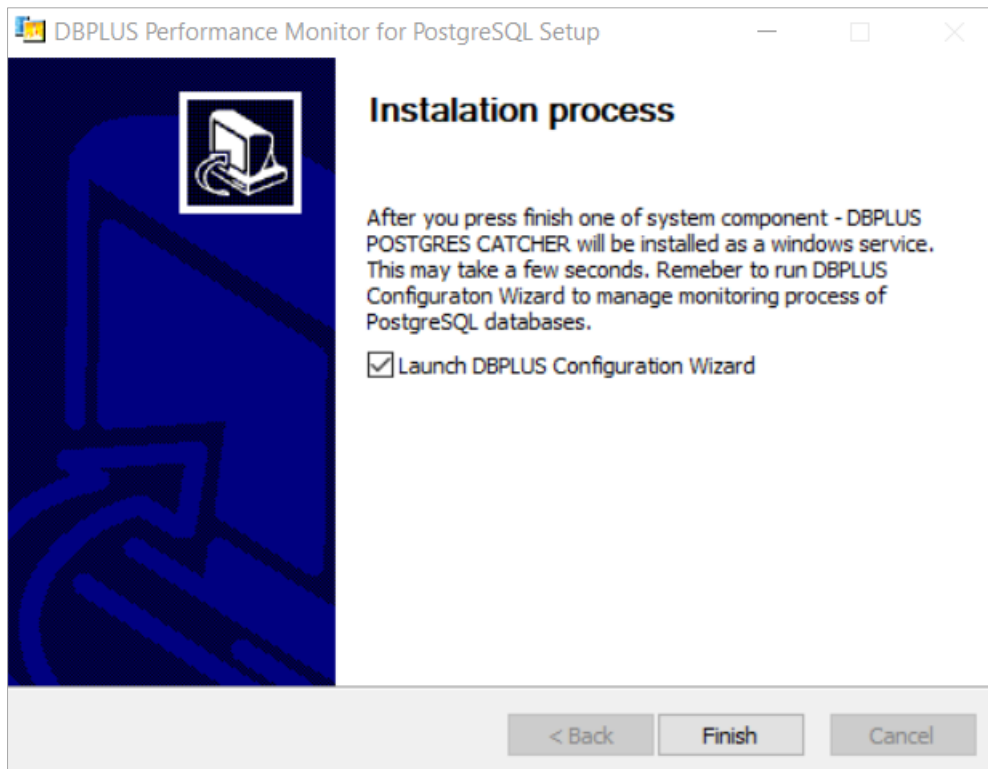


In order to continue installation, you should read and accept the terms of the license.

The next step is to select the directory, where DBPLUS Performance Monitor will be installed. Default directory is „C:\Program Files (x86)\DBPLUS.Postgres”



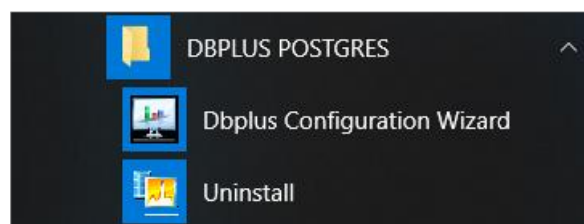
After the correct installation User will receive the following information:



The installation process is completed by pressing "Finish" button. By default, we start system configurator, which will carry out the further process of installing individual components of the system or configuration.

Installed DBPLUS Performance Monitor program is in the menu (view for Windows 10 Pro):

„Start” >”→”DBPLUS POSTGRES folder”



The following tools are available after the correct installation:

1. DBPLUS Configuration Wizard.
2. Uninstall.

2 System configuration

In the first stage you must set up a system on the server with **DBPLUS Performance Monitor™** installed, in order to:

- Create a DBPLUS repository in the selected PostgreSQL instance, which will store all the information about PostgreSQL instances performance,
- Add PostgreSQL servers in the monitoring process,
- Configuration monitoring service DBPLUSPOSTGRESCATCHER responsible for gathering information about individual instances performance,
- User Application configuration.

To perform the above tasks, the system requires the rights of the database user with **superuser** role.

This is required to configure the Repository database. To this end, we indicate one PostgreSQL instance, where a new DBPLUS database user (name to be chosen) will be created. Technical tables will be created there in the same database schema.

When a PostgreSQL instance is added for monitoring, an existing database user is created (or indicated). A user with superuser permissions is required. High powers are required to:

- collecting statistics on a monitored PostgreSQL instance,
- estimation of the execution plan for monitored queries,
- visibility of query texts in the pg_stat_activity view.

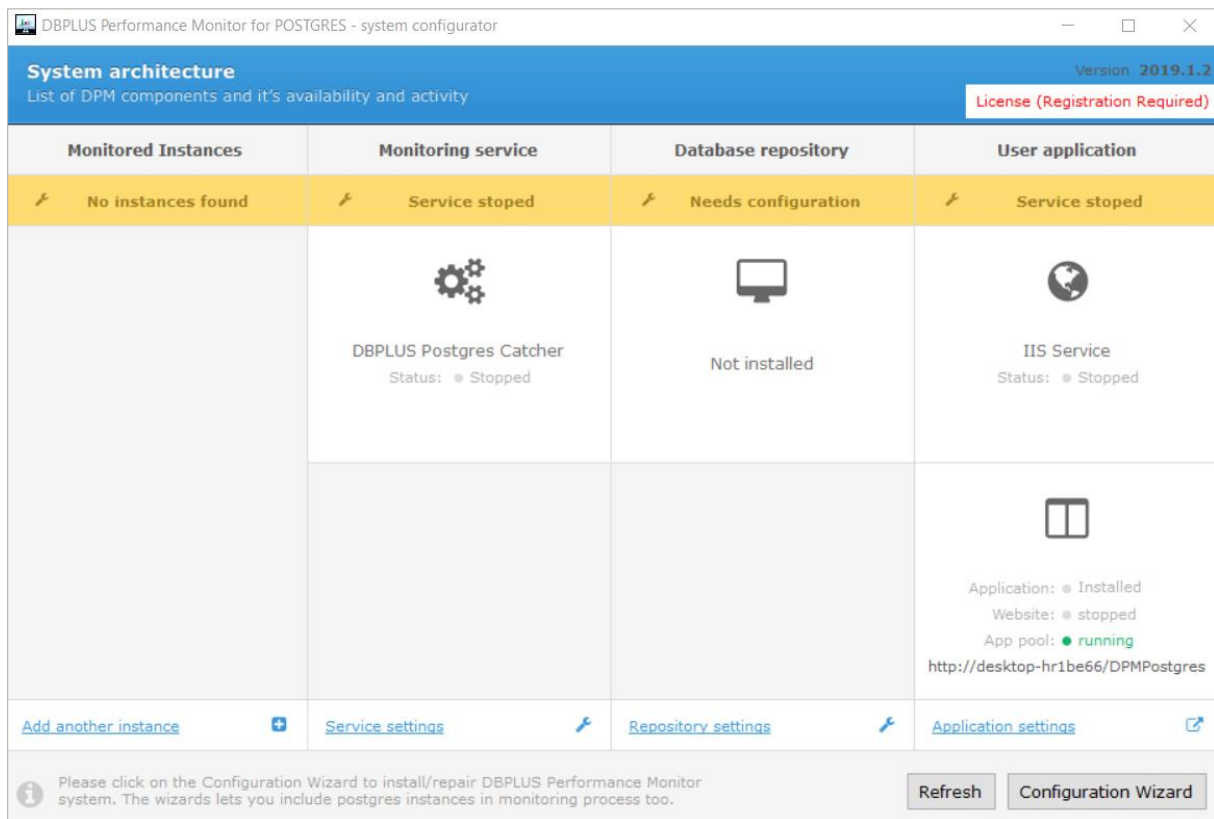
After completing these steps, the application will be available to user from a web browser.

2.1 *The main configurator screen*

On the server, where software has been installed, by clicking Start> DBPLUS POSTGRES>DBPLUS Configuration Wizard we open a window with the system management tool:

DBPLUS Performance Monitor for POSTGRES - system configurator.

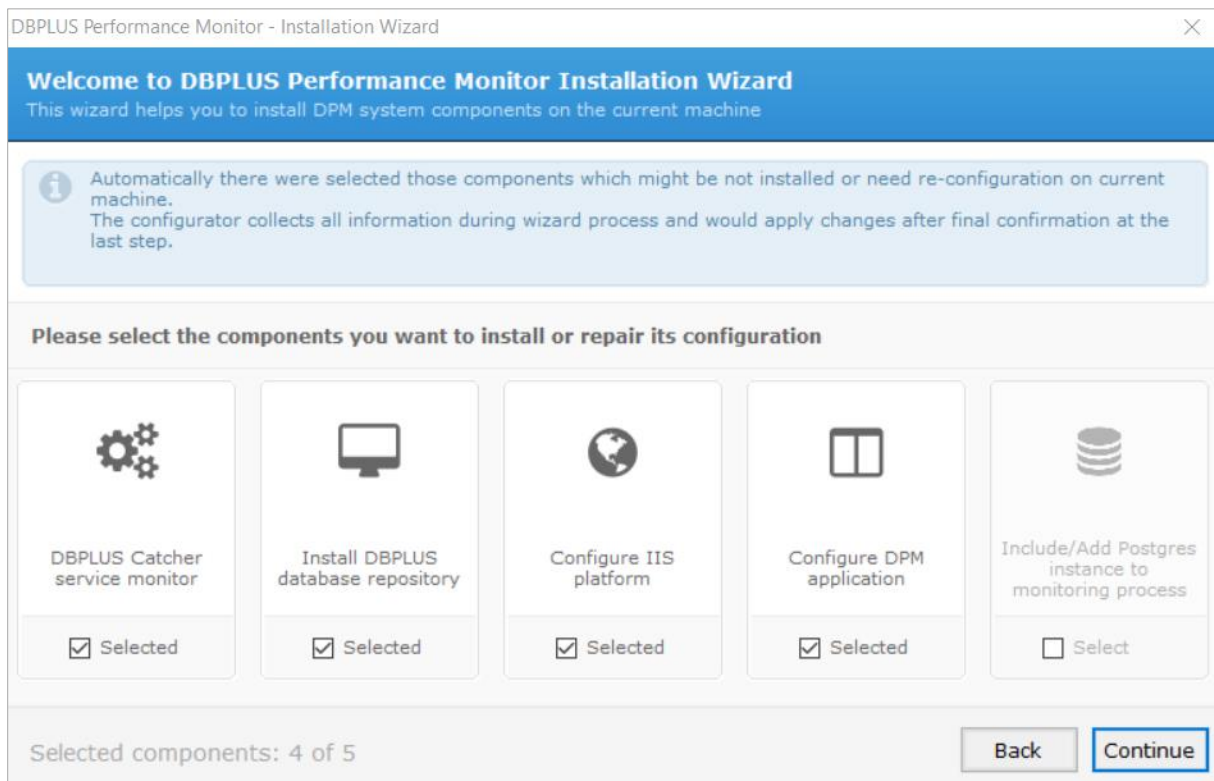
Below is a view of the configurator after installing the application and starting the configurator.



The main window shows the system architecture and informs i.a.:

- number of monitored PostgreSQL instances (Monitored Instances),
- place in which monitoring data is stored (Database Repository),
- installation / configuration of individual components of system, for example:
 - lack of monitored PostgreSQL instances
 - user application installed or not, application services status (IIS website, application pool),
 - if the monitoring service is enabled.

In order to perform basic system configuration, click **[Configuration Wizard]** button and - as a result - user get the window to configure individual components.



By default, Application selects components that require configuration by default. User can always reconfigure e.g. a monitoring service or add another (not included so far) PostgreSQL Instances to monitoring.

As part of the configuration, the application will perform:

- Create the DBPLUS database repository
- Run IIS role/service on the current machine
- Configure [DBPLUSPOSTGRESCATCHER](#) monitoring
- Configure user application [DPM Application](#).

2.2 *Setting up DBPLUSPOSTGRESCATCHER monitoring service*

DBPLUSPOSTGRESCATCHER is a program that runs as a service of Windows. In the current version, the service can run using a local account.

The DBPLUSPOSTGRESCATCHER service configuration screen during installation:

DBPLUS Performance Monitor - Installation Wizard
✕

DBPLUS POSTGRES Catcher - windows service responsible for postgres instances monitoring
Specify if service should be ran in context of windows/domain account or using local system account

Catcher	Repository	App	Finish

i For DBPLUSPOSTGRESCATCHER service it can be used:

- Local system account
- Windows/Domain account.

On the database level system uses only internal postgres users.

Remarks:

- Please do not use account with administrator privileges, it's not required

Set an user account which will be used by the DBPLUSPOSTGRESCATCHER service

Login type Local System Account ▾

Username

Password

Step 1 from 6

Back
Continue

Click on the **[Continue]** button to advance to the next configuration stage.

NOTE: All settings - made in the components of system - are ultimately confirmed in the final step of the configuration creator.

2.3 System Repository configuration

The DBPLUS PERFORMANCE MONITOR system repository is a set of tables that must be created on the selected database in the PostgreSQL instance. The repository configuration consists of three steps:

- indicate of the PostgreSQL instance the repository will be installed,
- select the database the DBPLUS tables will be installed,
- create a monitoring user.

2.3.1 PostgreSQL Instances Configuration

For this purpose, please provide:

- **Connection name** (any)
- Host name,
- TCP Port,
- Default maintenance database for the PostgreSQL instance – **default Database.**
- User with superuser roles - the data is given once during the installation, it is not saved and used anywhere.

The screenshot shows the 'DBPLUS Performance Monitor - Installation Wizard' window. The title bar indicates it is 'Step 2 from 7'. The main heading is 'DBPLUS database repository' with the subtitle 'Specify database where repository user can be installed'. Below this is a progress bar with five steps: 'Catcher' (highlighted in green), 'Repository' (highlighted in yellow), 'IIS', 'App', and 'Finish'. An information box states: 'You need to specify the database server where dbplus repository would be located. Database details like name, files and any specific features you can select in the next following steps'. The form contains the following fields:

- Connection name: Repository instance
- Host name: 125.45.8.98
- TCP Port: 5432
- Default Database: postgres

Below these fields is a section titled 'Set an user account with superuser rights.' with the note 'It will be used to perform DPM monitoring objects instalation on selected instance'. This section includes:

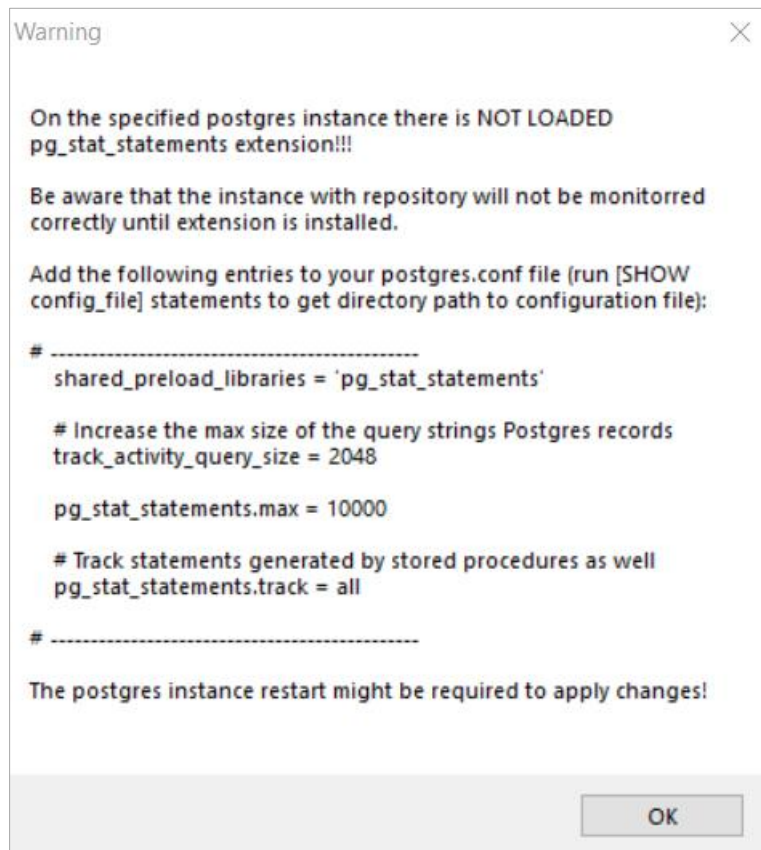
- Authentication: Postgres Authentication (dropdown menu)
- Username: postgres
- Password: masked with four dots
- A 'Test credentials' button

At the bottom right, there are 'Back' and 'Continue' buttons.

The DBPLUS Performance Monitor application to collect information about query statistics requires to turn on "pg_stat_statements" extension in the monitored PostgreSQL instance.

For this purpose, on the server where the PostgreSQL instance is installed, find the configuration file of the instance (for Windows 10 and PostgreSQL instances), the default path is: [C:\Program Files\PostgreSQL\11\data\postgresql.conf](#)).

In the next step you need to make changes according to the instructions below.



After making modifications to the configuration file, you must restart the PostgreSQL instances to make the changes ready.

We go to the next configuration step by clicking the [**Continue**] button.

2.3.2 Selection of the Repository database

The next step in the repository configuration is selecting the database where the DBPLUS technical tables will be created. All performance statistics of the monitored databases will be stored there.

DBPLUS Performance Monitor - Installation Wizard

DBPLUS database repository

Specify an database details used for repository purposes

Catcher	Repository	IIS	App	Finish

Information: You need to specify the database which will be used a system repository. We strongly advise to use separate database for Dbplus objects, but you can also use existing one. For new database, please specify its name and tablespace location

Create new database and tablespace

Database template: Tablespace name:

Database name: Location directory:

Use existing database

Use existing database

Database name:

Use existing tablespace

Use existing tablespace

Existing Tablespace:

Step 3 from 7

Two possibilities are available:

- **Create new database and tablespace**
- **Use existing database**

You can also create a new tablespace (**Use existing tablespace**).

2.3.3 *Creating a monitoring user*

The next step is to select the database user for monitoring. This user will be used as the owner of the repository database. These are two options:

- Use existing login
- Create new login/user

Clicking the [**Continue**] button accepts the selection and proceeds to the next stage of configuration.

DBPLUS Performance Monitor - Installation Wizard
✕

DBPLUS database repository
Specify login account which will be used by DBPLUSCATCHER service and user application to connect to database

Catcher	Repository	IIS	App	Finish

i You need to specify the user which will be used for connection purposes by DBPLUSPOSTGRES CATCHER service and DBPLUS Performance Monitor application

We strongly recommend to use the same user account as specified for DBPLUSPOSTGRES CATCHER service. If login doesn't exist, then please to create the new one. Specified login would be set as an owner for database repository. In addition it would get grants to executes system procedures or read system views on the postgres instance

Create new login/user

Authentication: Postgres authentication

User name: dbplus_mon

Password: ••••

Use existing login

Use existing user

User name: postgres

Password:

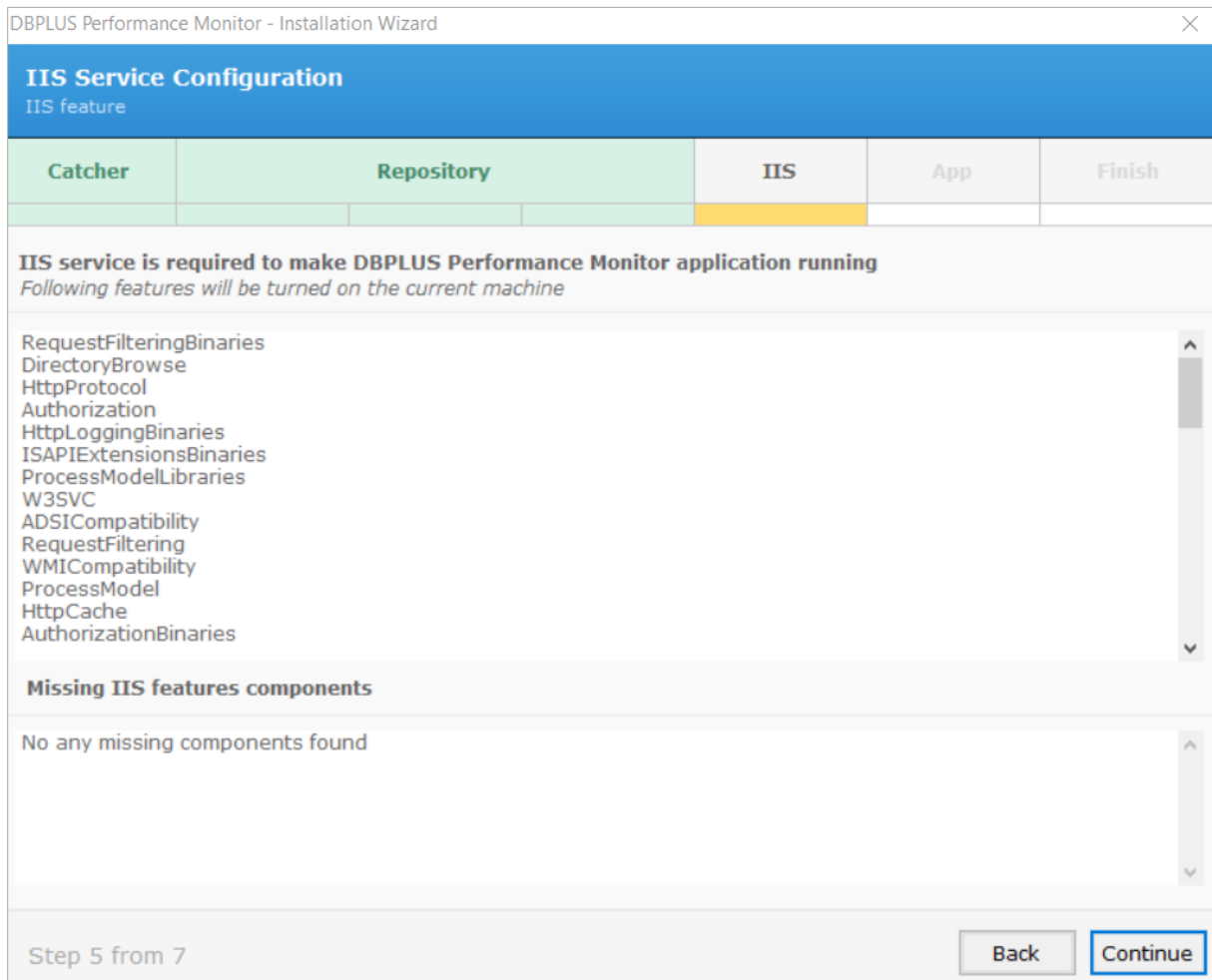
Test credentials

Step 4 from 7

Back
Continue

2.4 IIS service configuration

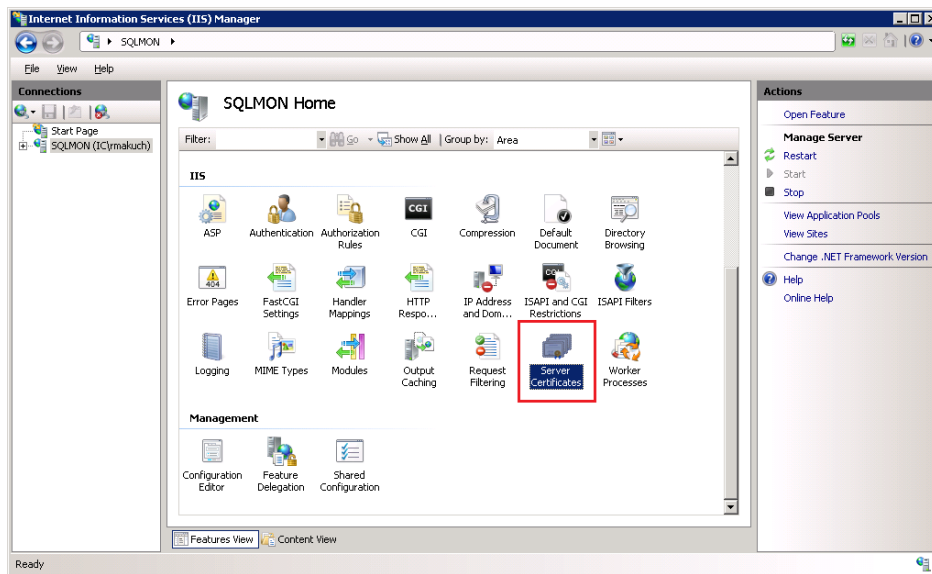
Launching of the IIS service on the server is required to run the user interface. The creator shows following features of IIS application server that will be installed. If the “**Missing IIS features components**” box is empty, no other configuration is required.



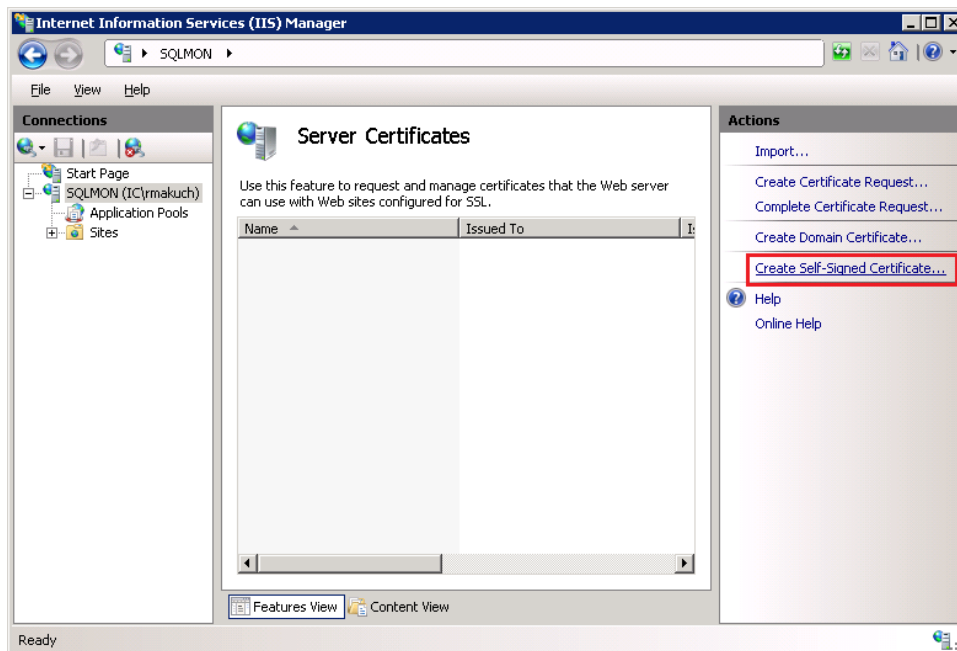
2.4.1 Configuration of SSL in the IIS (additional option)

In case you want to enable the SSL functions in the DBPLUS Performance Monitor application, you need to perform the steps on the server with the installed DBPLUS software:

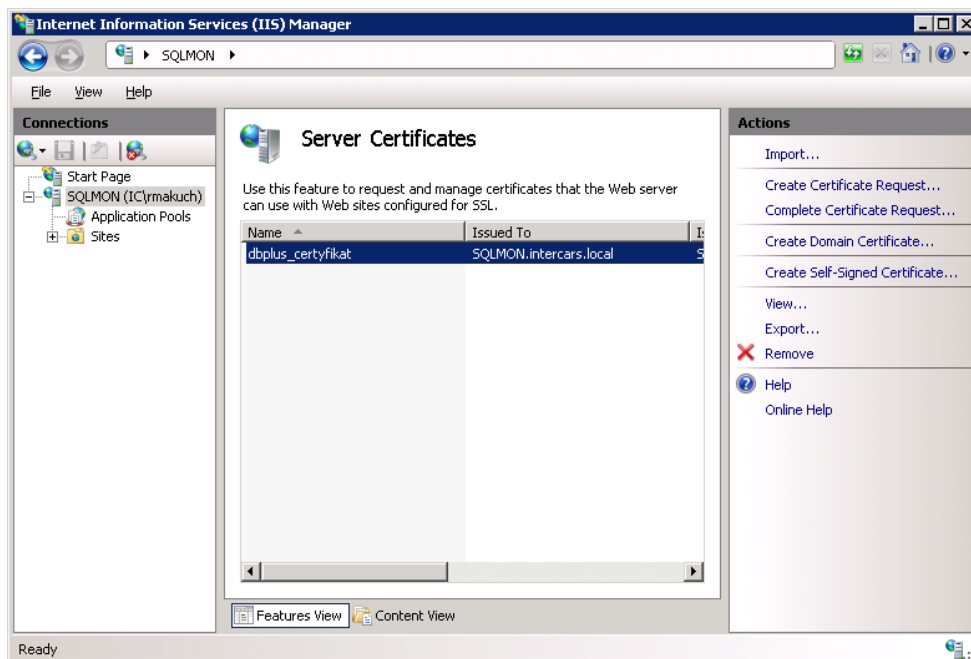
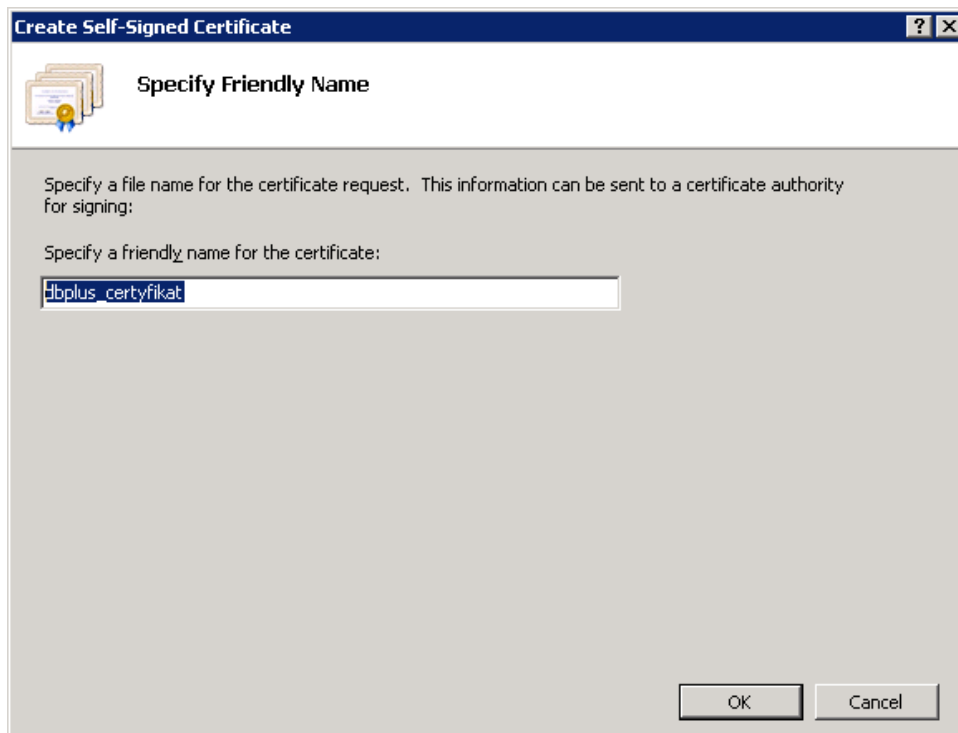
1. Run the IIS Manager (Internet Information Manager) from the command line with the **inetmgr** command
2. For the selected server, find the **Server Certificates** icon and enter to generate or import a certificate



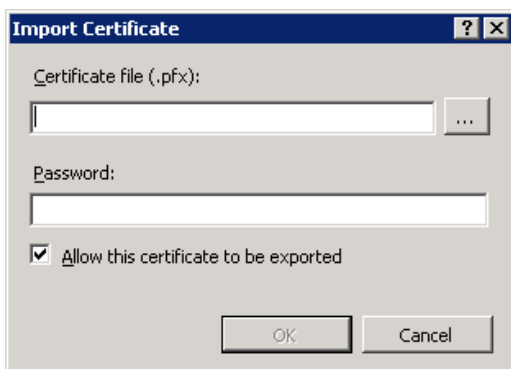
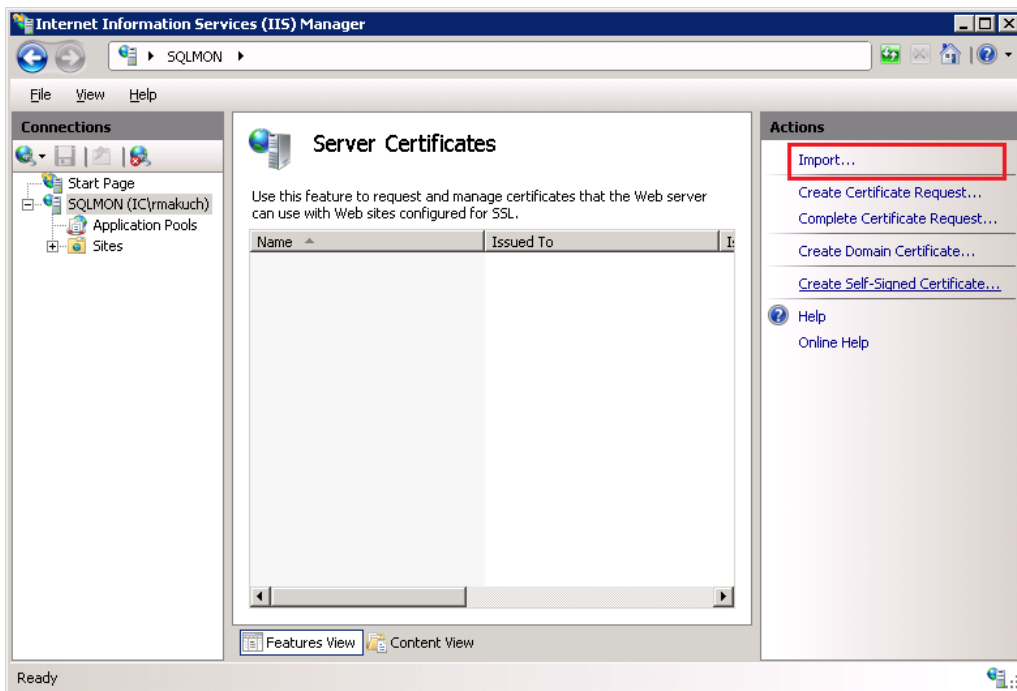
3. Generation of the certificate on the IIS server (in case we don't have it). Run options according to the below screenshots.



Create of the certificate.



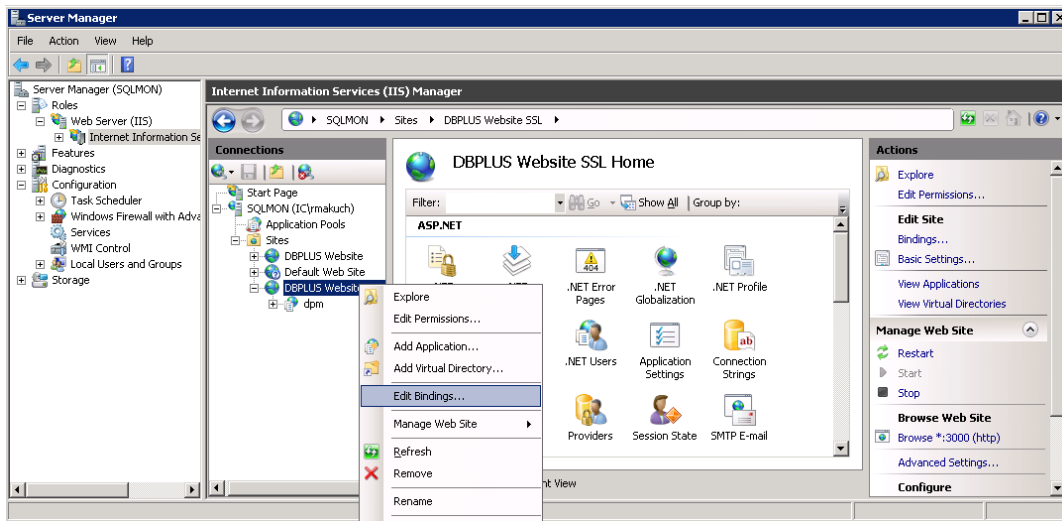
4. Certificates import (in case the certificate was not generated directly on the IIS server). We run according to the following screens:



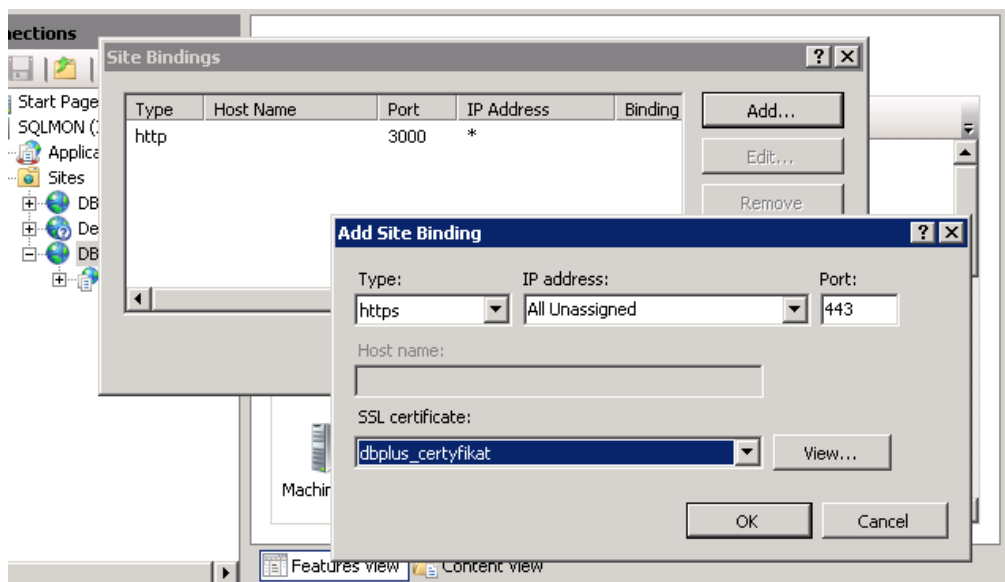
Pass the password if the certificate was exported with a password

5. Edit bindings

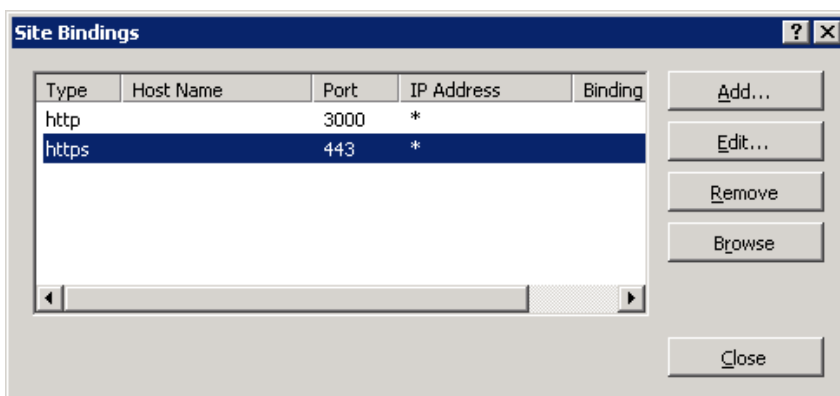
Update the link for the DBPLUS Website. Click on the site, then chose Edit Bindings option.



In the Site Bindings window, add a new link specifying the SSL protocol and select the certificate previously created or imported as on screen below:



As a result, we receive:



Remove settings with the http type. On the configured DBPLUS Website, we click the restart (Refresh button).

2.5 User application configuration

Another element is the creation of user interface objects. Belong to them:

- Authentication Type:
 - LocalSystem,
 - LocalService,
 - NetworkService,
 - Windows Domain Account,
 - ApplicationPoolIdentity

When choosing login type = LocalService there is no need to enter username and password, the service will work on the default user for Windows (LocalService)

- Parameters:
 - Port number (default 80)
 - Binding property /Host Name
 - Access to application - whether users at the login to the site will be authorized (login and password) or not.

As a result of the entire setup process completion, application will be available at the following url:

http://server_name:port_number/DPMPostgres

If the system will be running on port 80, link will be as follows:

http://server_name/DPMPostgres

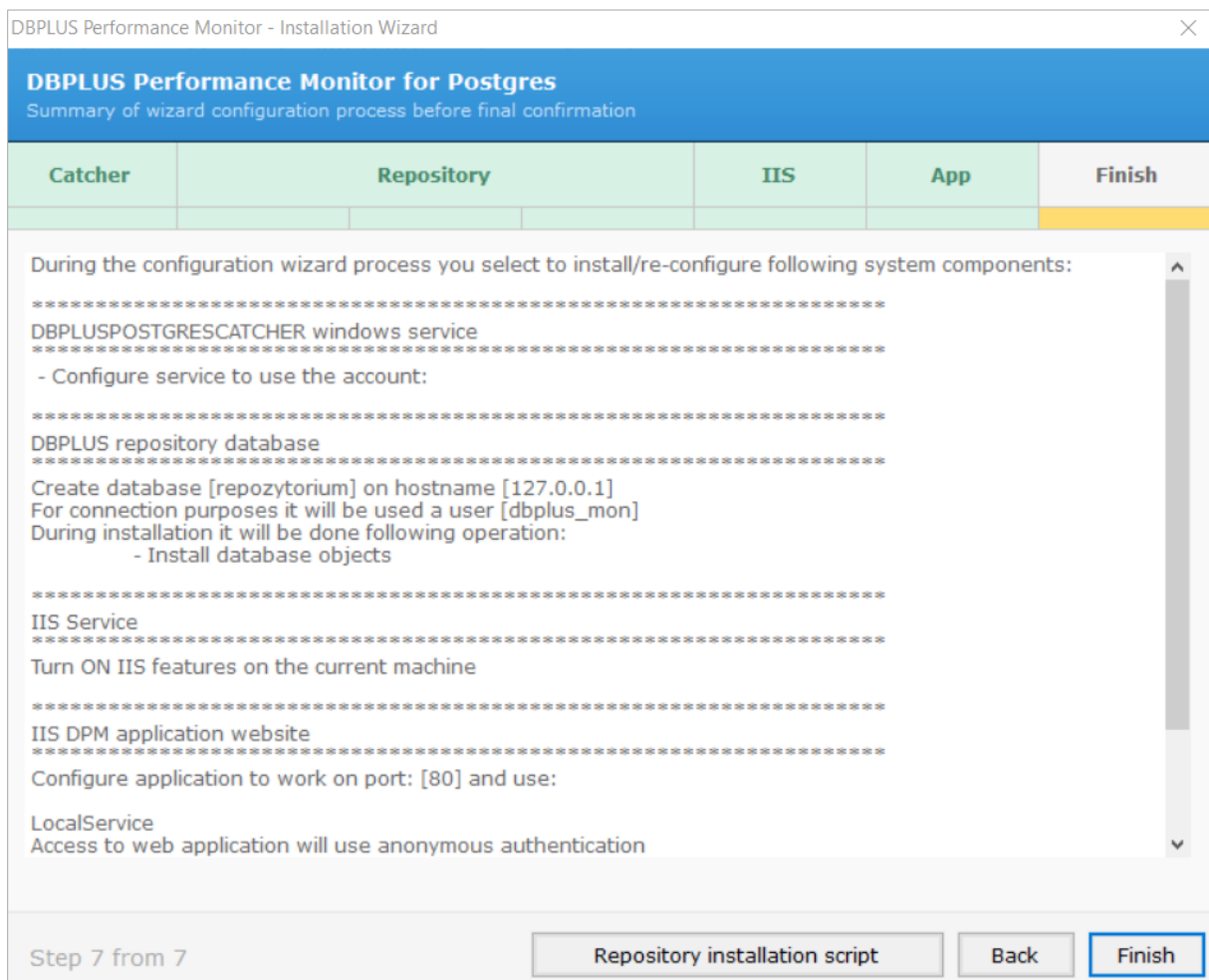
Click on the **[Continue]** button to proceed the next step

2.6 Configuration summary

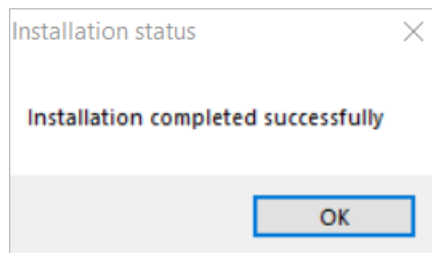
The last configuration step is to confirm all settings according to steps defined in the configurator. The final screen shows a configuration summary.

Attention! In addition, there is a Repository installation script available, by clicking on [Repository installation script] it is possible to save it on the disk.

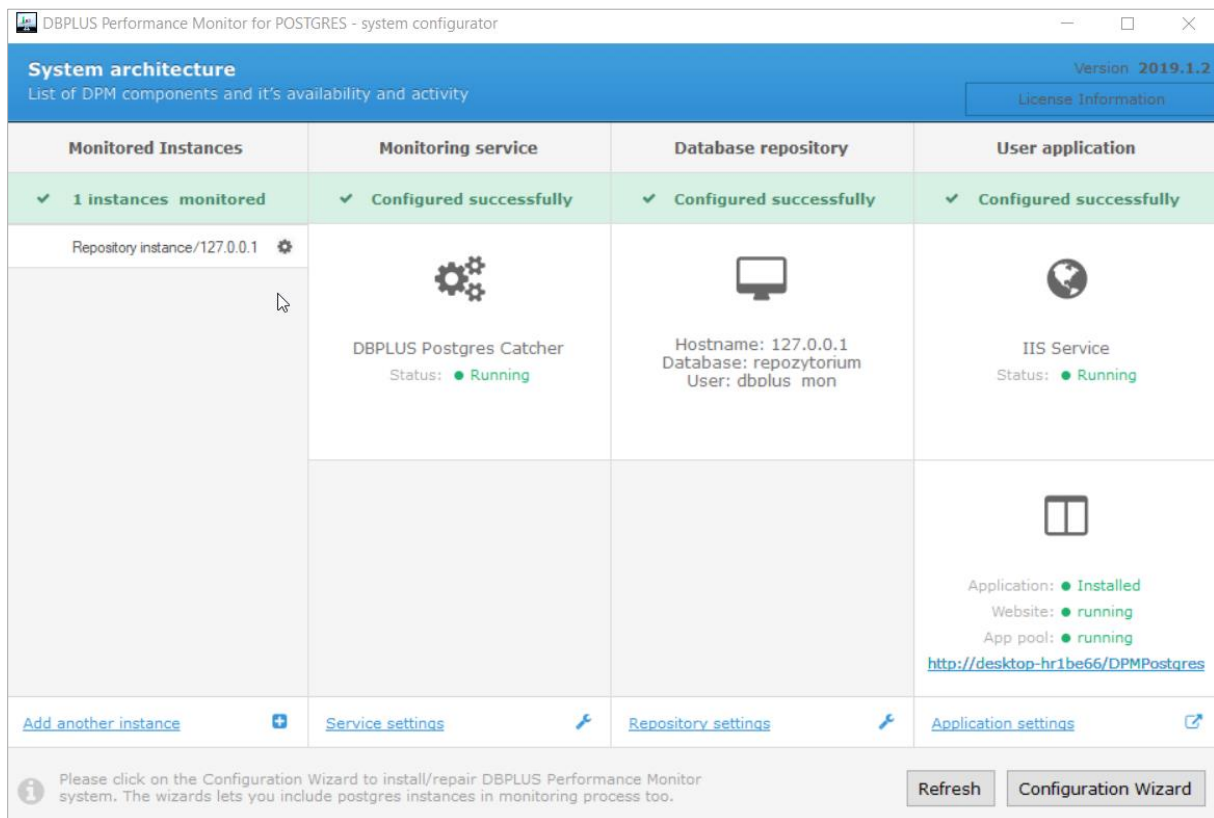
To confirm changes, click on **[Finish]** button.



At the end of the configuration status of the installation will be shown:



As a result, system configuration main window looks like below:



From the above sample screen, we can read that DBPLUS system PERFORMANCE MONITOR is:

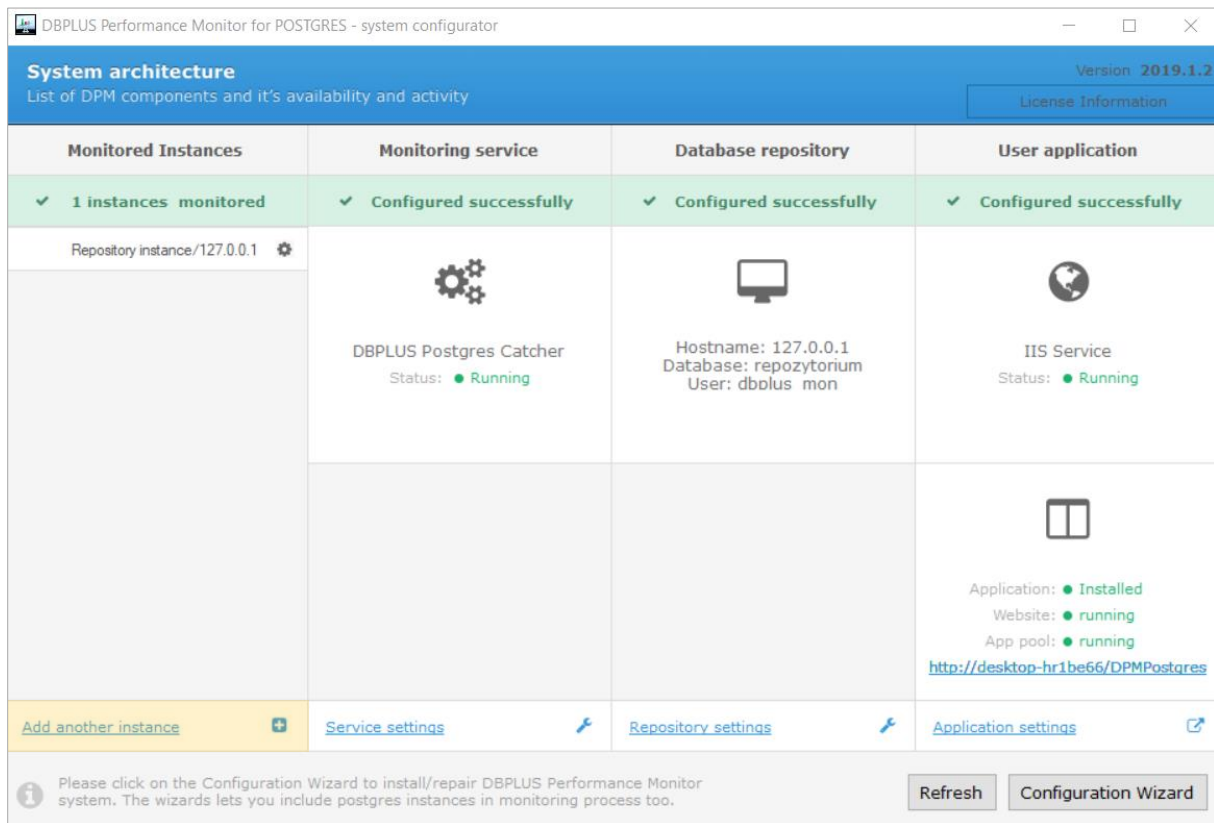
- Installed on a server DESKTOP-HR1BE66 (link to the application in the lower right corner),
- all components are properly configured (the bar with information “Configured successfully”)
- appropriate services are running:
 - DBPLUSPOSTGRESCATCHER – a service responsible for database monitoring,
 - IIS, Website, App pool – which means that the application is available to the user.
- We have 1 monitored PostgreSQL instance,
- Information from the monitoring of all instances (currently one) are stored in „repository” database in server 127.0.0.1 and connects from the user *dbplus_mon*.
- Interface / User application is available at <http://desktop-hr1be66/DPMPostgres>

Note: in the case when a port number used to configure the application is other than [80] , the link to the application will contain the port number. For example, if the port [81] is used, the link to the application will be as follows <http://127.0.0.1:81/DPMPostgres>

3 Additional functionalities

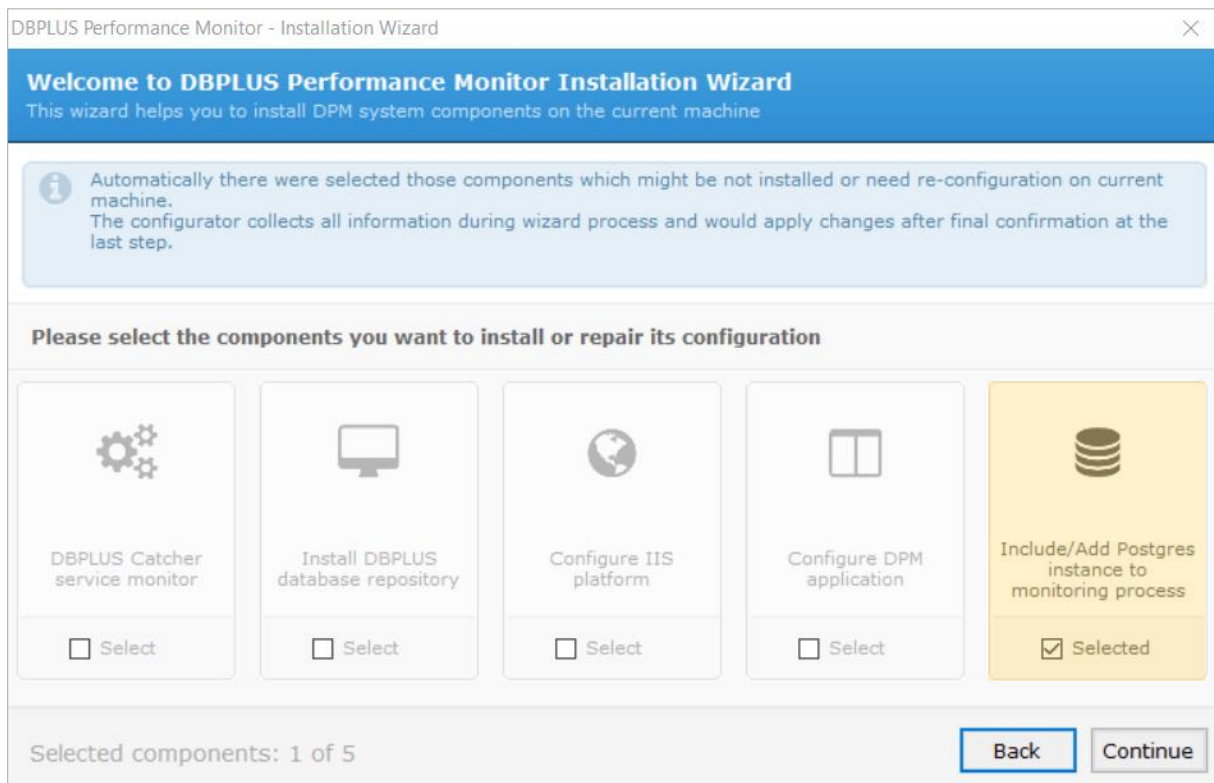
3.1 Add PostgreSQL Instance for monitoring

After the initial configuration, you can proceed to add more PostgreSQL instances for monitoring. For this purpose, in the main System Configurator window click **[Add Another instance]** button.



IMPORTANT: If the **[Add another instance]** button is not available, this is the result of the lack of a license for a certain number of instances.

The second option to add PostgreSQL instance for monitoring is click **[Configuration Wizard]** button and select the component **[Include / Add SQL instance to monitoring process]**



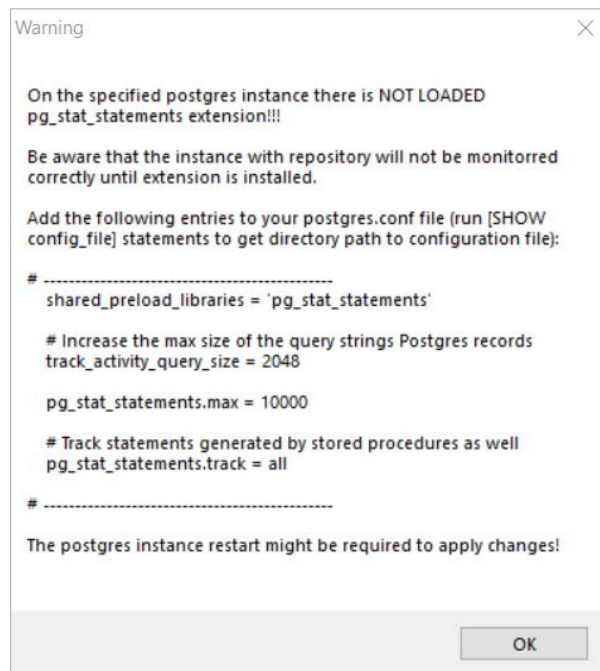
In both cases as a result, we go to the wizard to add a new database. In the first step user need to enter basic information about new instance:

- **Connection name** (any),
- Host name,
- TCP Port,
- Default maintenance database for the PostgreSQL instance – **default Database**.
- User with superuser roles - the name is given once during the installation, it is not saved and used anywhere.

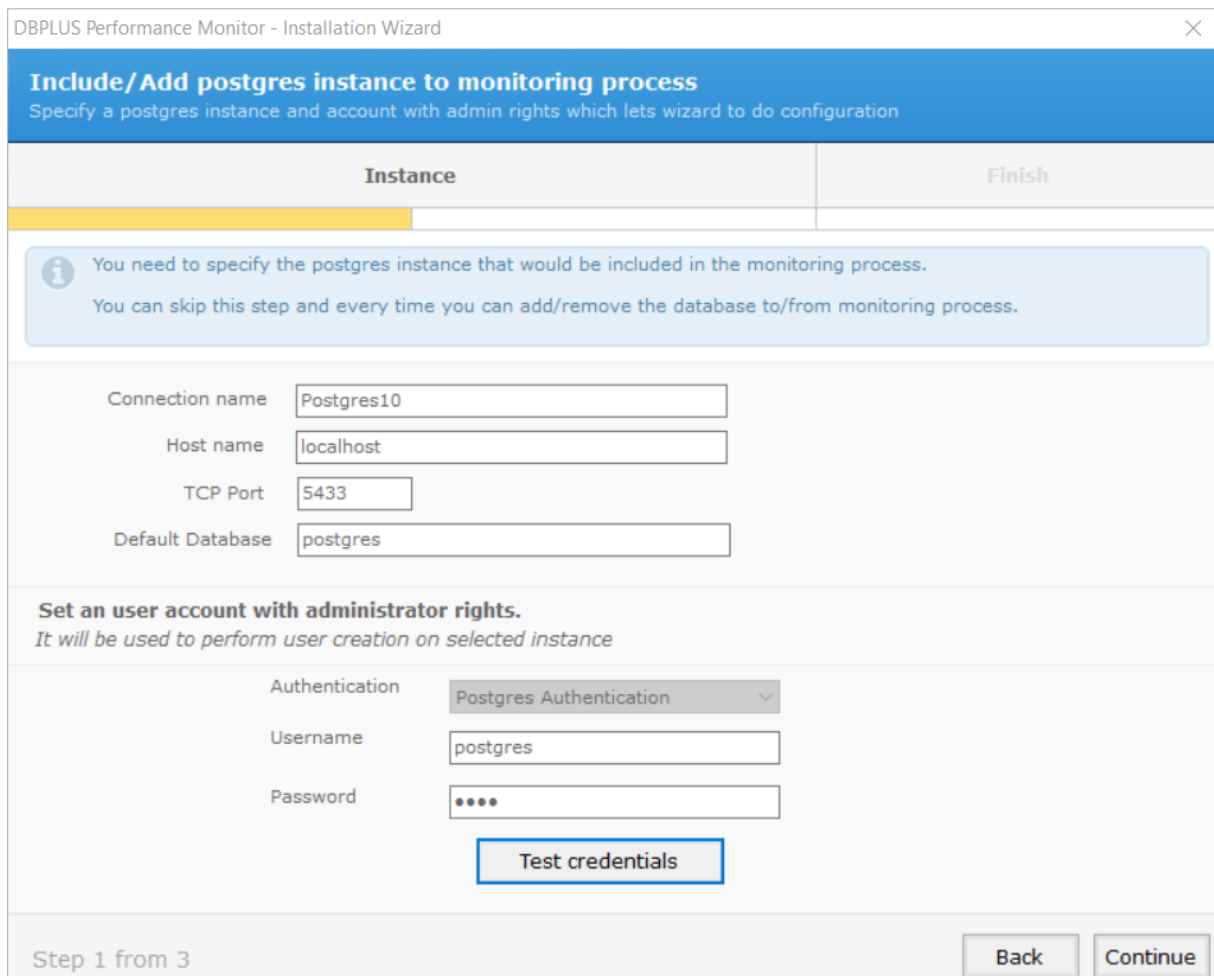
The DBPLUS Performance Monitor application to collect information about query statistics requires to turn on "pg_stat_statements" extension in the monitored PostgreSQL instance.

For this purpose, on the server where the PostgreSQL instance is installed, find the configuration file of the instance (for Windows 10 and PostgreSQL instances), the default path is: <C:\Program Files\PostgreSQL\11\data\postgresql.conf>).

In the next step you need to make changes according to the instructions below.



After modifications to the configuration file, you must restart the PostgreSQL instances to make the changes ready. Click the **[Continue]** button will take you to the next configuration step.



Connecting a new instance for monitoring, it is possible to create a new database user or use existing one. This user will be used to collect statistics from the monitored instance (DBPLUSPOSTGRESCATCHER service will log in to this user).

DBPLUS Performance Monitor - Installation Wizard

Include/Add postgres instance to monitoring process

Specify a postgres instance and account with admin rights which lets wizard to do configuration

Instance | Finish

Information: You need to specify the user which will be used for connection purposes by DBPLUSPOSTGRESCATCHER service. We strongly recomend to create new user and to not use an account with system privileges. For specified user, it would be grant rights to read system views on the monitored database

Create new login/user

Authentication: Postgres authentication

User name: db_mon

Password: [masked]

Use existing login

Use existing user

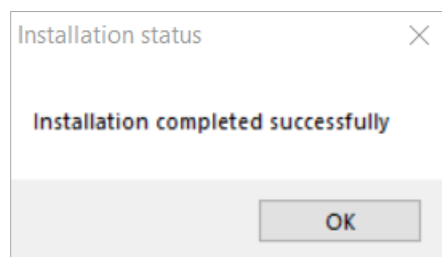
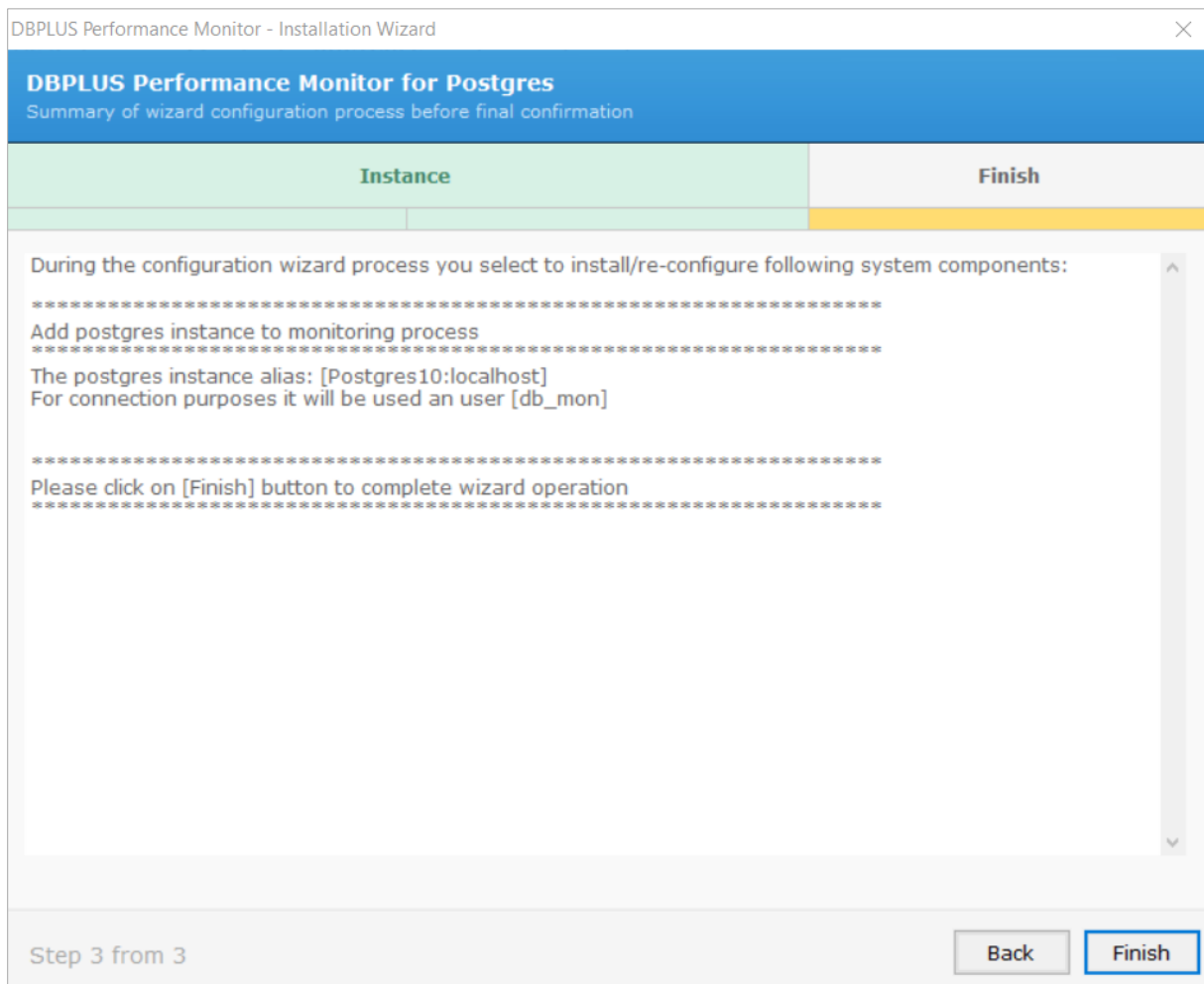
User name: postgres

Password: [empty]

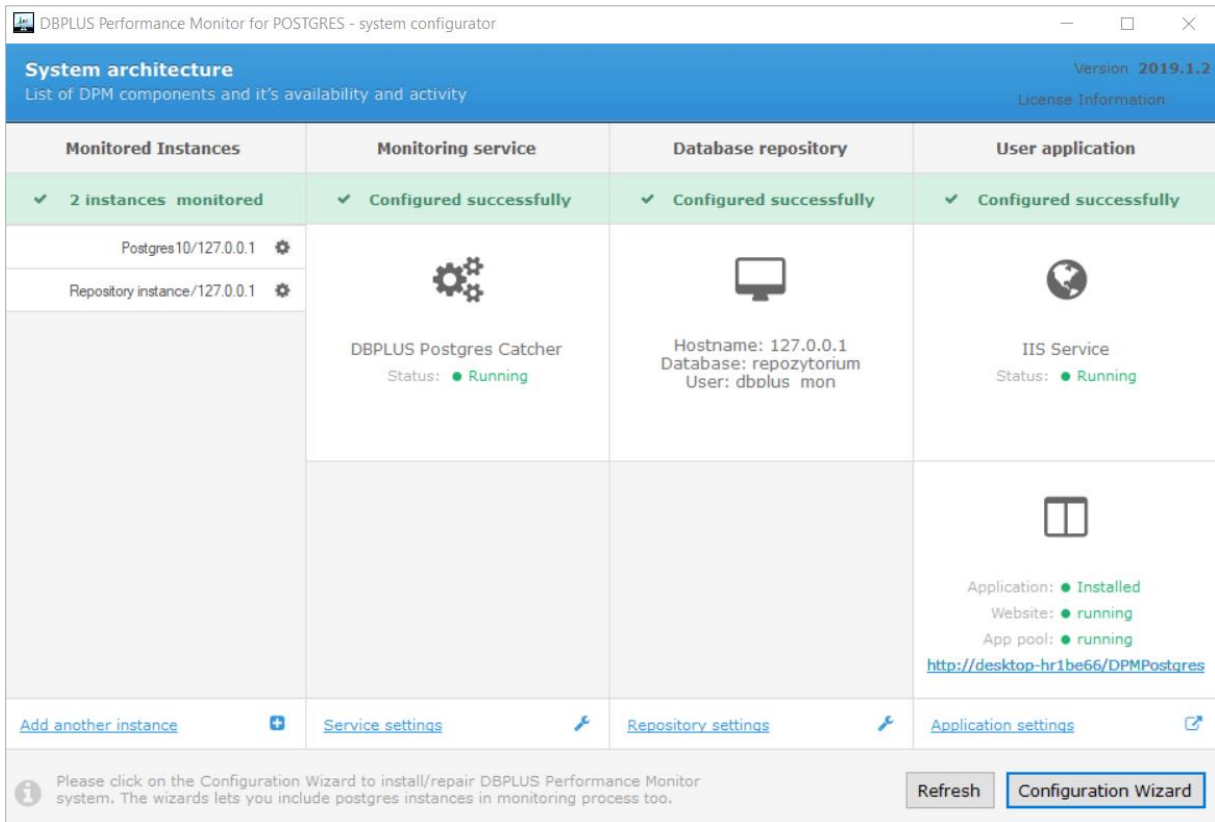
Test credentials

Step 1 from 3 | Back | Continue

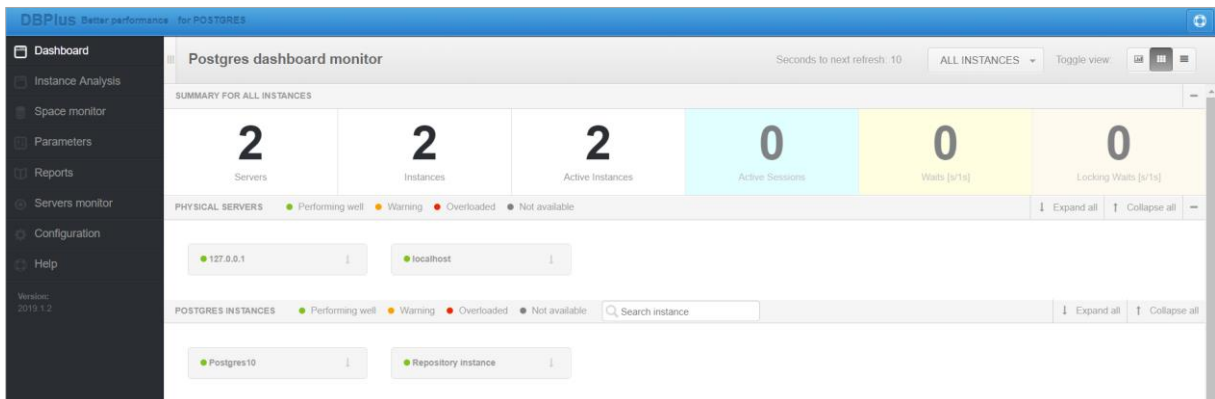
After click **[Continue]** button, it shows the final screen being the summary of configuration process.



Click **[Finish]** button to add an instance to monitoring. As a result, changes are visible in the system configuration main window - DBPLUS Performance Monitor supports two PostgreSQL instances.



Click the link to the application (in this case, <http://desktop-hr1be66/DMPPostgres>) an application with monitored instances will be displayed:



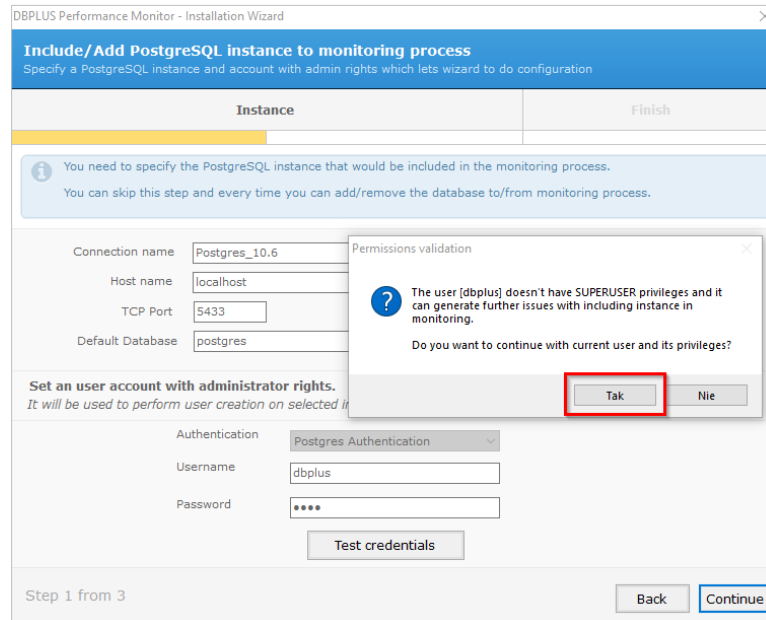
3.1.1 Add an PostgreSQL instance without a Superuser (AWS)

In the latest application User has the ability to add instances to monitoring without having to indicate a user with Superuser privileges. Necessity for the Postgres version in a "cloud" solution (eg AWS).

Warning! For postgresQL versions lower than 10, it is not possible to add an instance to the monitoring without the Superuser role.

For PostgreSQL versions higher than or equal to version 10, the administrator user must have the permissions: grant connect to all databases located in the instance that we want to monitor and the roles pg_read_all_settings.

To start, add instances as before using the [Add another instance] button, on the newly opened page we complete the details of connection details by entering the login and password of the user with administrator privileges but not Superuser.



If you move to the next page, you will see a message that Superuser has not been authorized by the user designated as the administrator. In this case, continue with the installation.

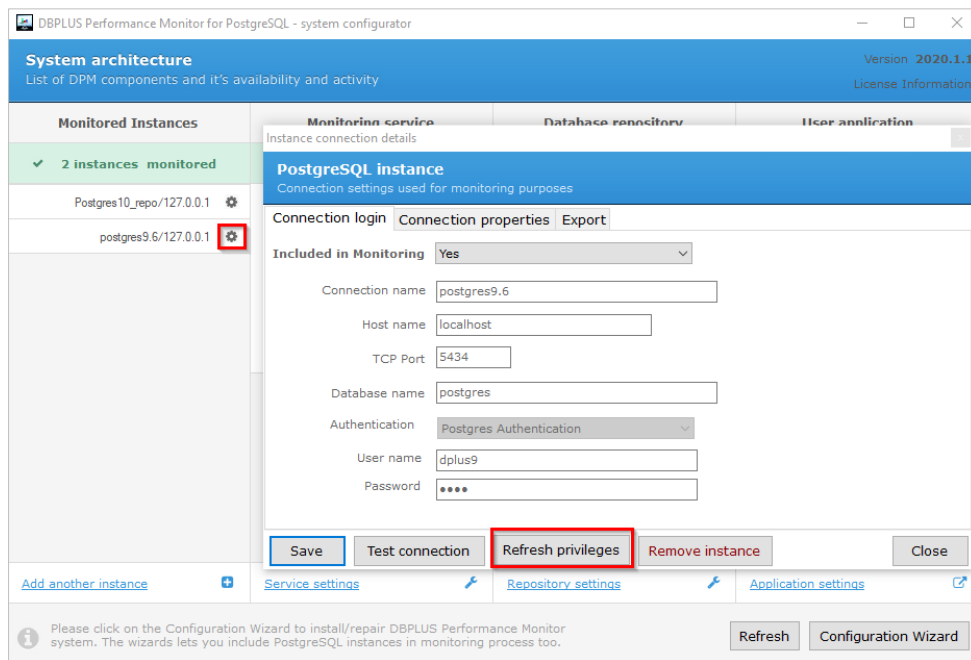
The next screen when creating a user will also display a warning about the lack of Superuser role by the administrator, in this case you should also accept and continue the installation. The next step will be a window with a summary of the installation. After clicking the [Finish] button we finish the installation process.

3.2 Modifying existing user's privileges

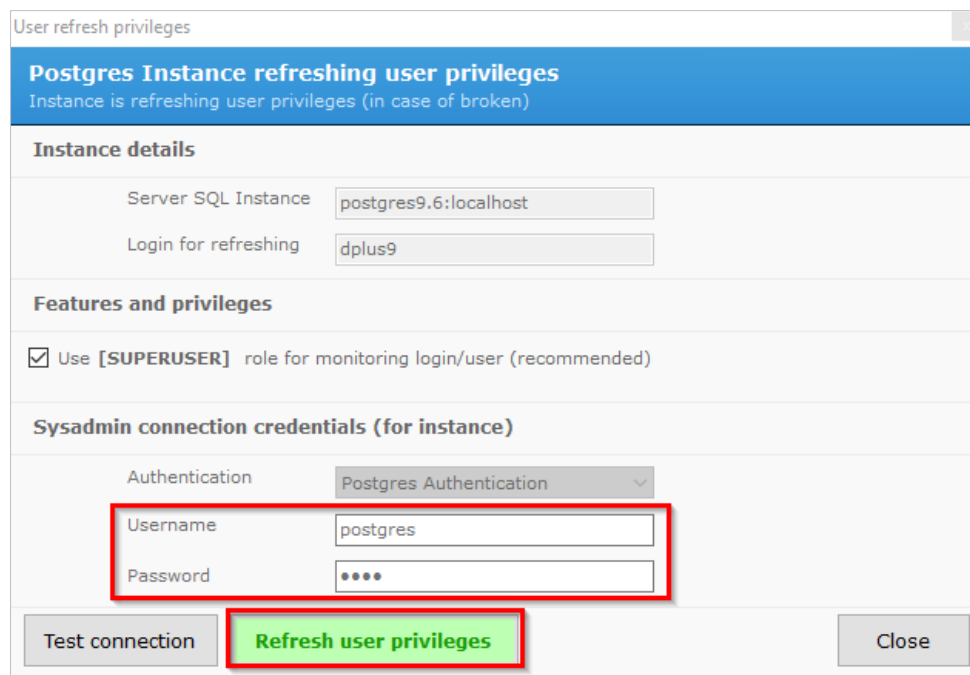
In application the User has option of changing the privileges of an existing user indicated in the monitoring. We use refreshing when the monitoring user does not have access to all databases on the instance or we want to change the currently granted permissions.

Refreshing privileges is also useful if you need to grant additional permissions that have not been previously granted (or have been revoked) and are needed to display data correctly.

To modify the privileges, open the "DBPLUS Configuration Wizard" program, then go to the settings of the given instance by clicking the button next to the PostgreSQL instance name for which you want to refresh / grant permission. Then click the **[Refresh privileges]** button.



On the next screen we provide user data with sysadmin privileges for a given instance to authorize and update changes.



The screen shows the current privileges of the given monitoring user. Checking or unchecking a given option grants or withdraws the given rights. To make changes, confirm by clicking the **[Refresh user privileges]** button.

4 System Upgrade

The DBPLUS maintenance support provides the access to new software updates that are published 4 times a year, as well as to DBPLUS engineers help PostgreSQL instance diagnostics using **DBPLUS Performance Monitor™** software.

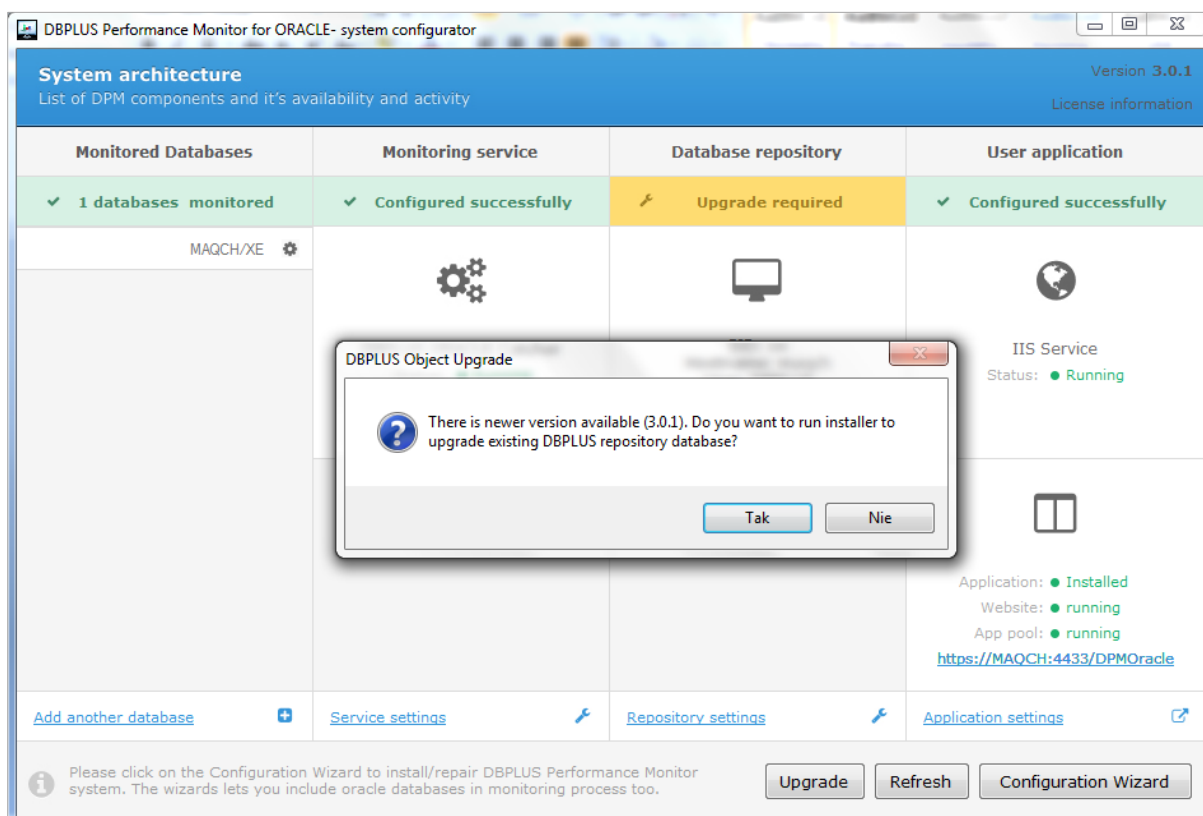
System upgrade combines with two steps:

- Run the installation file (which goes the same as the first installation process)
- Upgrade of database objects repository on DBPLUS user to the latest version.

Attention! The upgrade process involves running the dpmPostgreSQLInstaller.exe file containing the newest version of the application. Remember to select **exactly the same** folder you used during the first installation.

4.1 Upgrade to the latest version

In order to go through the upgrade process, user must run DBPLUS Configuration Wizard, which also start automatically after installation. In the result:



System automatically detects the need to update to the latest version. We accept the dialog box and we run the wizard that will guide us through the upgrade process.

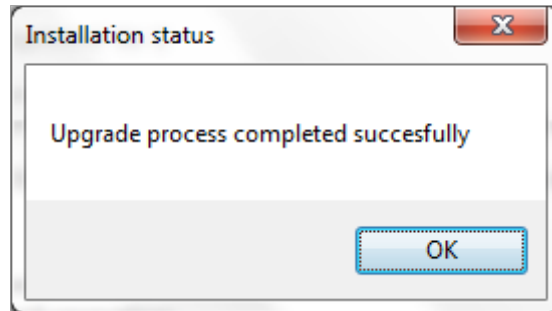
In case of interruption upgrade process, user can always return to it by click **[Upgrade]** in the main system configurator window.

As the first screen we have information about new system version, to which application will be upgrade.

The upgrade procedure applies to updating objects only in the database on which the DBPLUS repository is located.

Accept changes by click the **[Continue]** button. The system informs about operations that the DBPLUS database user will perform in the repository database. Approve changes by click the **[Finish]** button.

Depending on the version, the upgrade process can take from a few seconds to 1-3 minutes. Finally, user receive information about the status of the entire process.



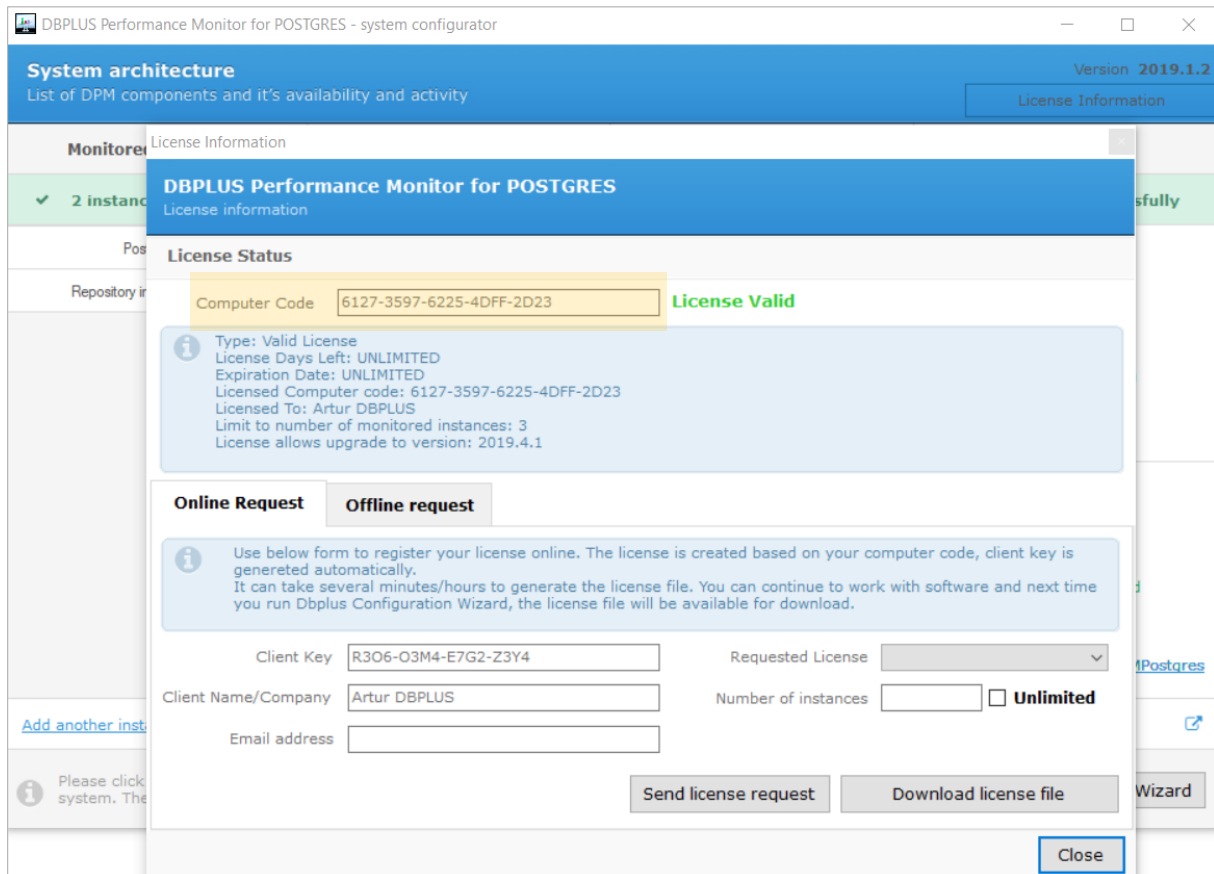
Click **[OK]** button to close the system configurator window.

5 License

The license is linked to the application server with the DBPLUS Performance Monitor software installed.

System license includes:

- Time of system availability The number of monitored PostgreSQL instances.
- It contains an automatically generated computer code.



Information about license is available from the configurator, i.e. **DBPLUS Configuration Wizard**

After installation, the system works in the trial version. This period continues 30 days and the system is available in full functionality. By the end of the period you must register the license in order to continue working with the program. This can be done in two ways:

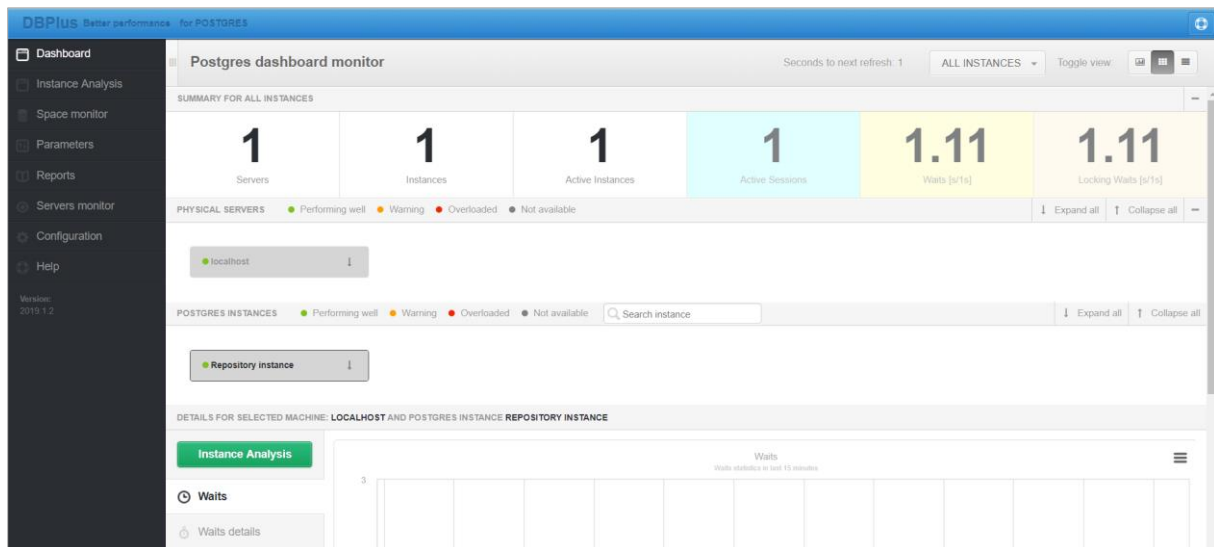
- Send a license request from the form by click the [**Send license request**] button (Internet access required on the machine) or
- Sending computer code by e-mail (Computer Code, shown in the figure above).

6 Working with program

The user interface is accessible from a web browser at the previously configured address. The default page of the system is a Dashboard shows the current performance of the monitored PostgreSQL instances.

6.1 „Dashboard”

After start **DBPLUS Performance Monitor™** web application, it opens a dashboard that shows the current performance of monitored PostgreSQL instances.



Dashboard is divided into areas:

- information bar
- summaries for all instances area
- physical servers' area,
- PostgreSQL instances area,
- details for the selected instance.

6.1.1 Information bar

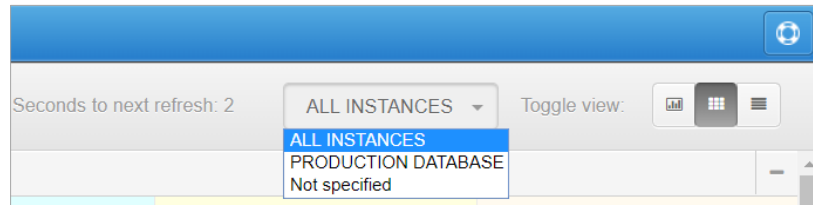


User can switch between dashboard and different view using the button on the information bar. List of available views:

- **Icon View** – displays monitored servers / databases as icons (default)
- **Grid View** – databases are displayed in a grid / table view
- **Television Mode** – shows PostgreSQL instances in a form of developed icons with automatically switching performance indicators.

Additionally, user is informed how much time is left until next Dashboard refresh with the most recent data about the current performance of all monitored instances.

Users can change display information's about database instances via the drop-down menu in the information bar. The database type can be freely defined and assigned in the Configuration> Instances menu, described later in this chapter.



In case when information bar changes colour from blue to orange – means that the program will go into alarm mode. Causes:

- insufficient space in the DBPLUS repository database,
- DBPLUSPOSTGRESQCATCHER service doesn't work.

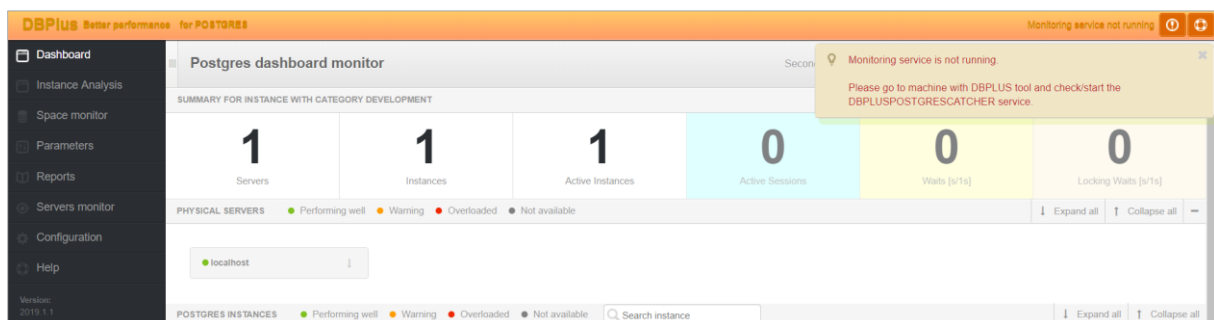
Insufficient space at repository database

When there is no place in the database schema which is the repository to collect data, a message will appear. The toolbar on Dashboard page turns orange and a lack of space message „Repository Space Warning” about lack of space will be shown.

DBPLUSPOSTGRESQCATCHER service doesn't work

Detection of a monitoring problem will result that bar change to orange on the Dashboard page and the message "Monitoring service is not running" display.

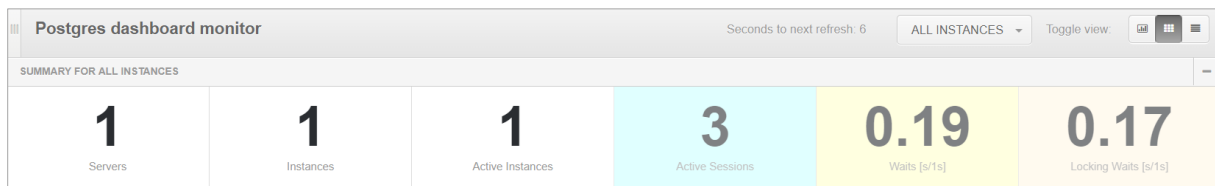
To fix the problem check if there are any errors on the application server with the installed DBPLUS Client and restart the DBPLUSPOSTGRESQCATCHER service.



6.1.2 The Summary area

The main area presents a general summary of:

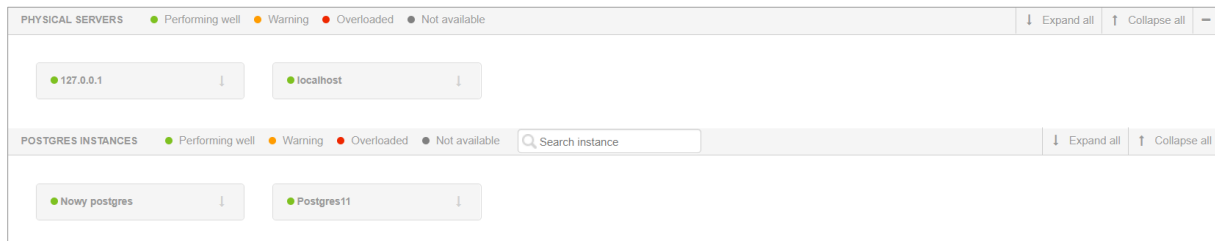
- Number of monitored servers and instances
- Number of active instances
- Number of databases on all instances
 - Active Sessions,
 - Level of waits
 - Locking Waits



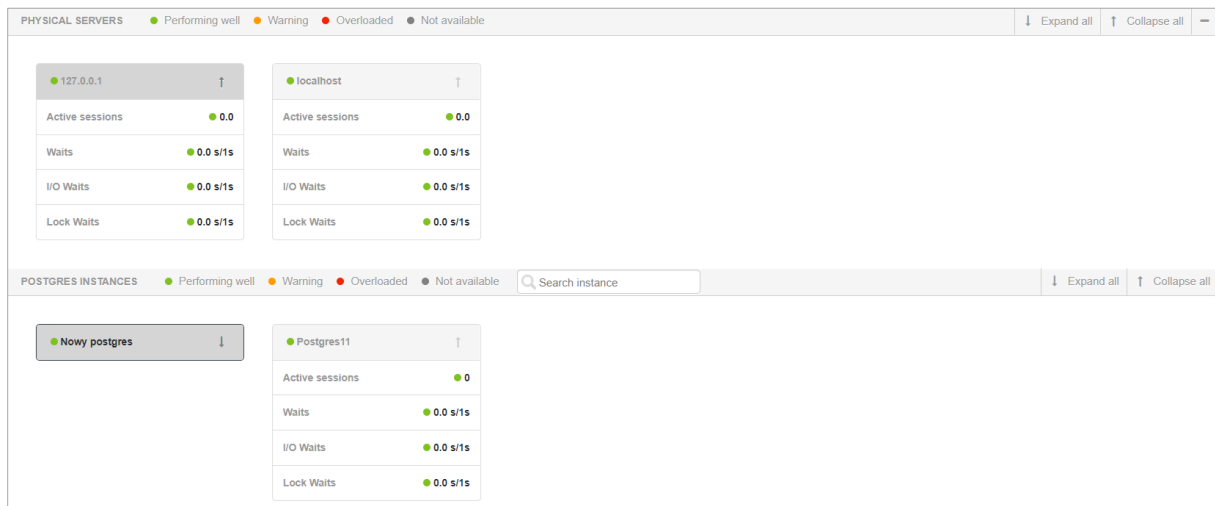
From the summary area, it's easy to tell if the wait level is high and whether user should still look for a problem.

6.1.3 Servers and instances area

In Physical servers part, user see the icon of servers run POSTGRESQL instance. By click the icon of any server in the instance area, the POSTGRESQL instances that work on a given machine will be highlighted.



The icon of each server or instance can be expanded by click the "arrow" or the **[Expand All]** button. At the instance area level, user can see exactly which PostgreSQL instance has the highest level of wait.



Additional options:

- Ability to search instance - the search option is available in every version of the Dashboard. This is useful to monitor more instances.

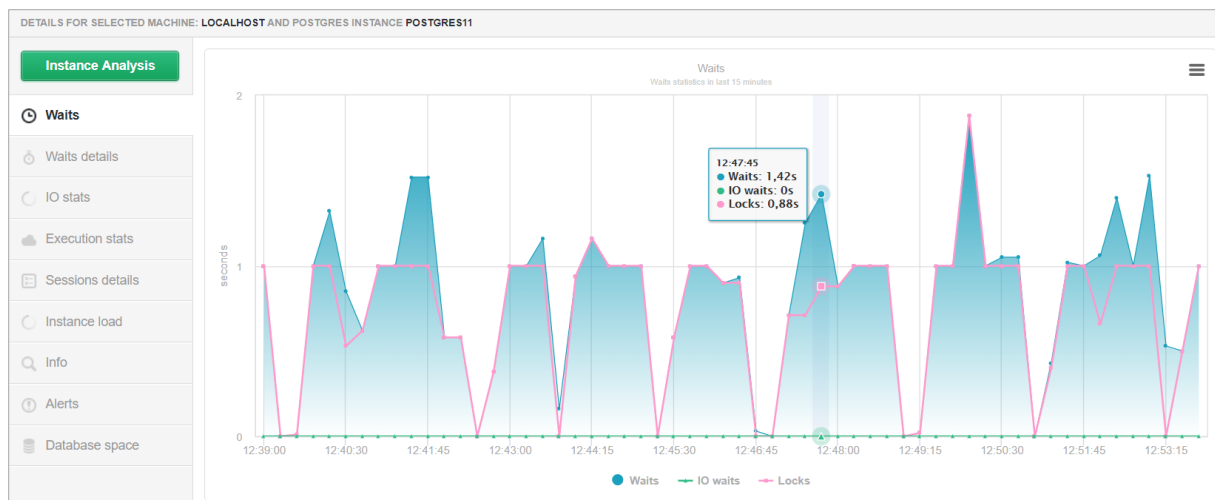


POSTGRES INSTANCES											
Search instance											
Server Type	Machine	Instance	Active	Waits [s/1s]	IO Waits [s/1s]	Locks [s/1s]	Alerts	Sessions	Transactions	Total space [GB]	
PRODUCTION DA	127.0.0.1	Nowy postgres	<input checked="" type="checkbox"/>	0.00	0.00	0.00	0	1	0	0	
DEVELOPMENT	localhost	Postgres11	<input checked="" type="checkbox"/>	0.00	0.00	0.00	0	0	0	0	

- In the Grid view, table that shows the current instance performance has options:
 - the ability to change the width of the columns
 - in the case of more records, scrolling the data does not hide the table header
 - use arrows to navigate between instances.

6.1.4 Details of PostgreSQL database performance

In order to analyze the current load, click on the icon of PostgreSQL instance. As a result, the bottom dashboard area reloads and presents details of the selected instance.



Available information:

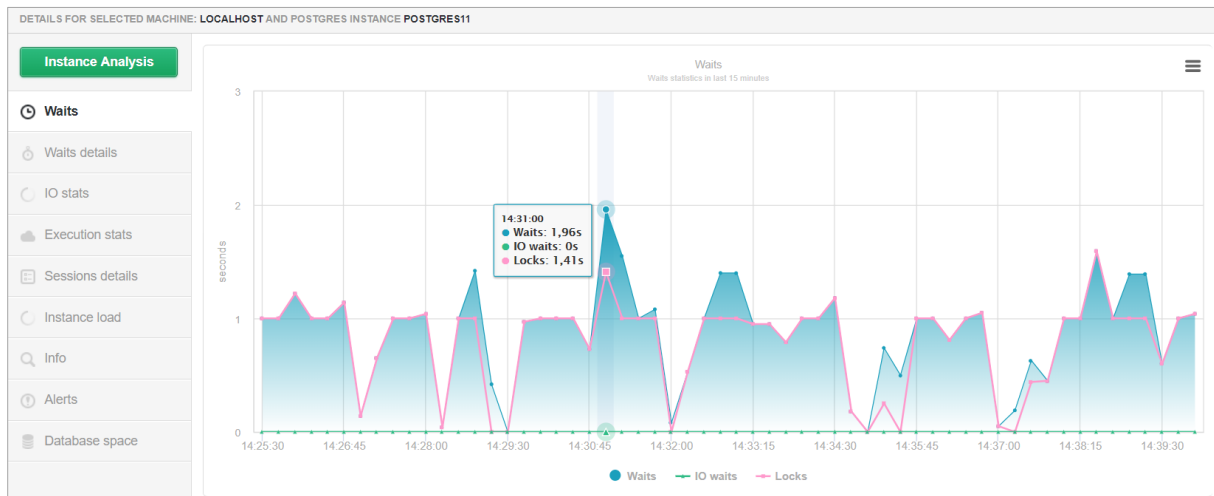
- current level of waits and lock – **Waits** tab,
- on what waits instance spend time - **Waits details** tab,
- IO statistics and Buffer Cache Hit Ratio level - **IO stats** tab,
- execution statistics – **Execution stats** tab,
- number of sessions and their status – **Sessions details** tab,
- PostgreSQL instance load – **Instance Load** tab,
- view basic information about instance – **Info** tab,
- check Alerts – **Alerts** tab,
- the size of the database in the instance – **Database space** tab.

Waits, IO, queries information is presented here in the horizon of the last 15 minutes. For example, on the **Waits** chart follow information:

- I/O waits – read of disk devices,
- Locks – wait time for queries on blockades,
- Waits – total wait time.

Chart means that at a specific moment in time (the time read from the X-axis), all users (active sessions) waited for the query result indicated number of seconds (the result read from the Y axis). IO waits and Locks categories help to state why sessions stay idle.

All series are visible by default:



After click the **Waits, IO Waits** series, only the **Lock** series is visible (next click on the legend bar, it will display the selected series again).

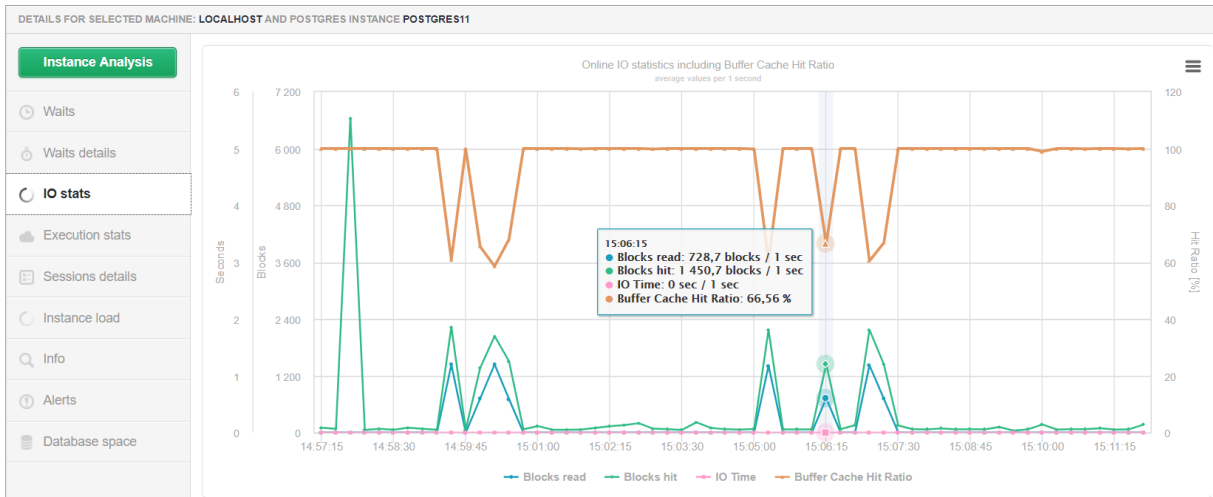


IMPORTANT: the level of expectations is calculated per one second.

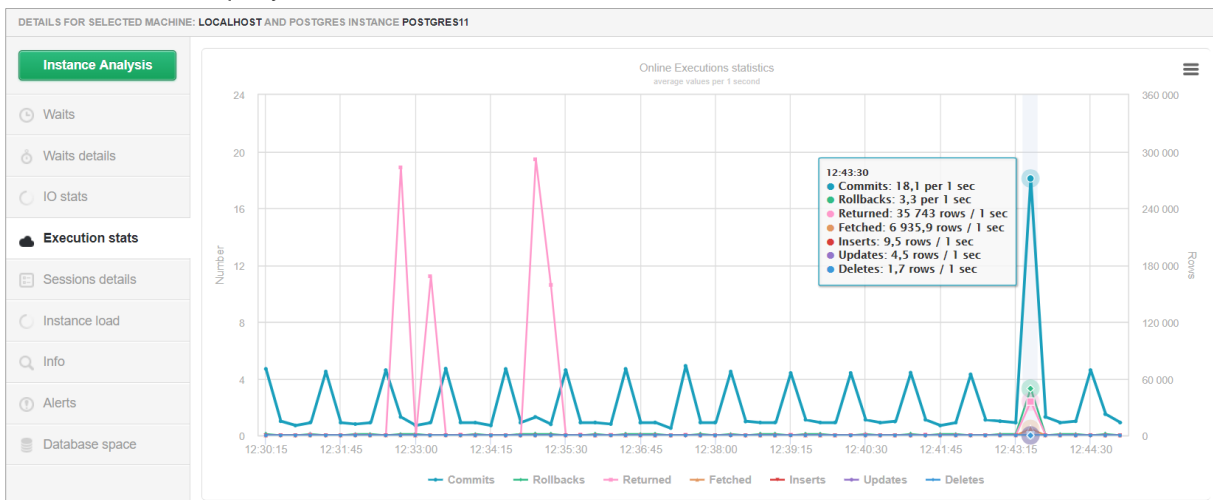
Details about waits can be found in the **Waits details** tab.



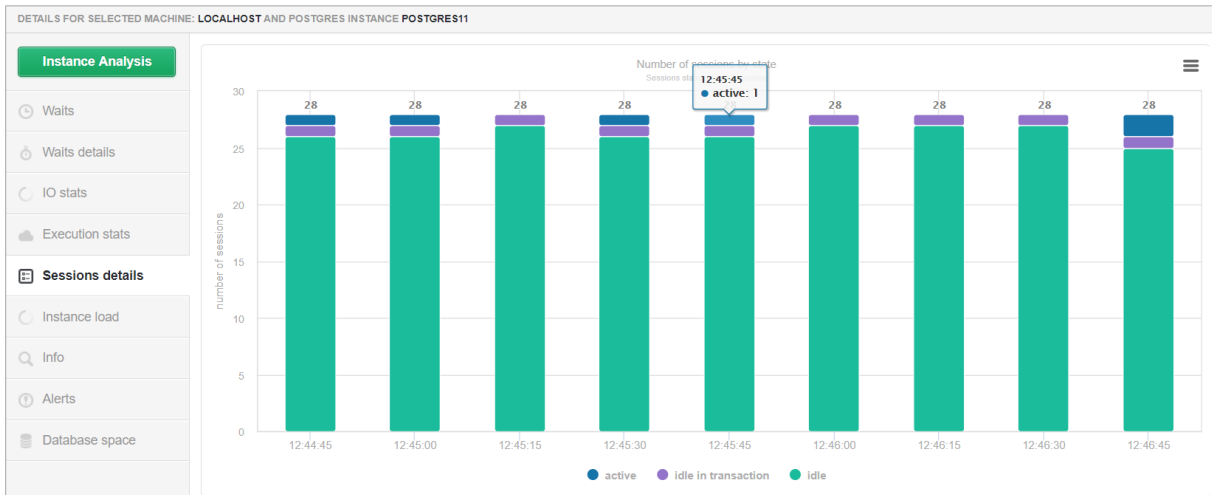
The next available tab is **IO Stats**. There is information about the number of blocks read from the disk (Blocks read), the number of read blocks from the memory buffer (Blocks hit), read times IO (IO Time) and Buffer Cache Hit Ratio.



Information about query statistics is in the **Execution stats** tab.



The next tab provides information about the status and number of sessions run on the PostgreSQL instance.



Instance Load is one of the modules used by DBPLUS engineers to analyze performance. Chart shows data for the last 24 hours.

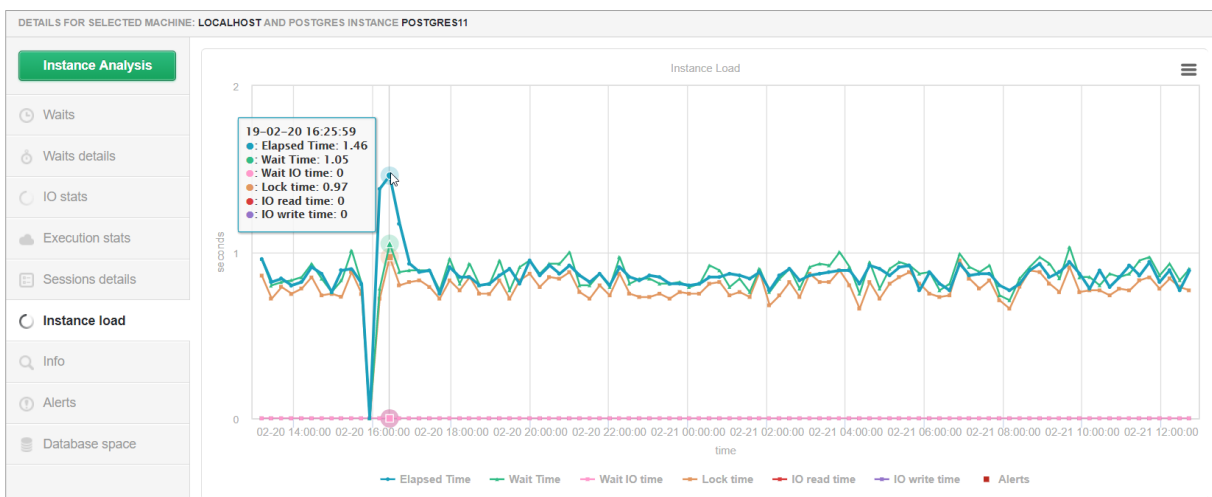


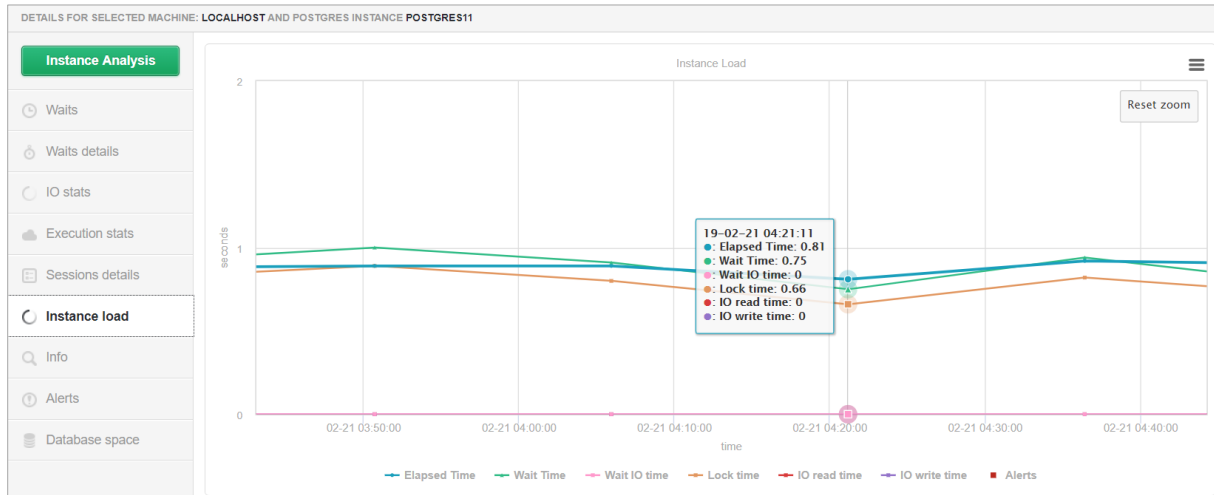
Chart consists of the following series:

- Elapsed Time - shows the summary time all users are waiting on the result of a query at a given second of time. On the graph for the displayed point Elapsed Time is 1,46 seconds, which can be interpreted as follows:
 - 3 users launched different queries – 2 users waited for 0.5 second, 3th user waited 0,46 second,
- Wait time – wait time for query execution,
- Wait IO time – time when queries waited for IO,
- Lock time – time when queries were blocked by other queries,
- IO read time
- IO write time
- Alerts – alert states.

For better readability of the chart:

- User can click to disable (or enable) given series of the chart - it can be done in the area of the chart legend
- User can zoom the chart

Here is an example of a chart in the narrower time horizon (after zooming a part of the chart)



Dashboard also allows the user to view basic information about the monitored Postgres instances, among others:

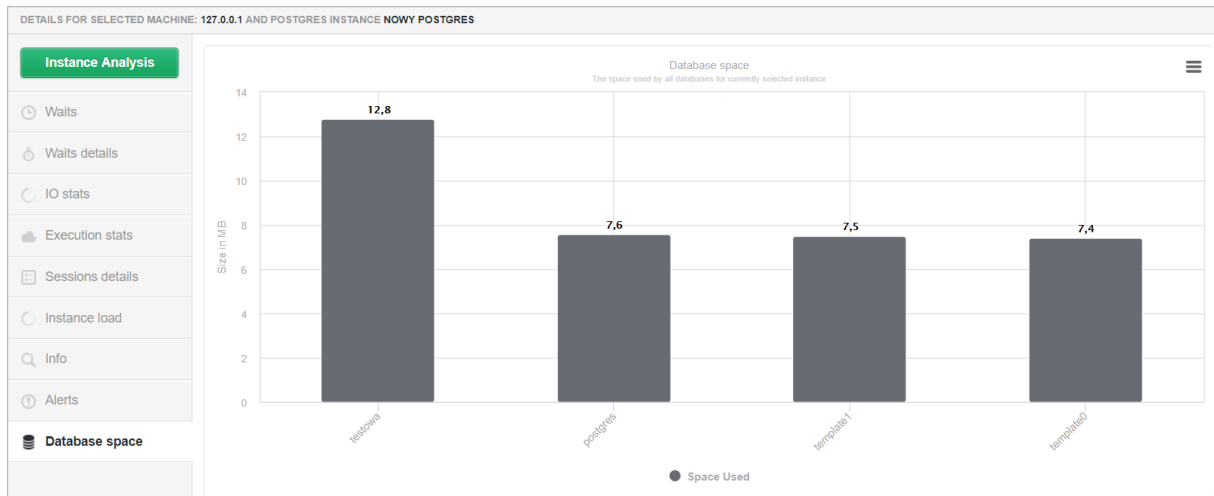
- Server type (set by the user in DPM),
- Path to the PostgreSQL configuration file,
- Path to the PostgreSQL instance data file,
- Path to the PostgreSQL instance log file
- Size of the data block,
- Maximum number of connections,
- Status of the autovacuum parameter,
- Server version number,
- Number of databases in the PostgreSQL instance,
- Number of tablespaces.

User get it after click on the **Info** tab:

DETAILS FOR SELECTED MACHINE: LOCALHOST AND POSTGRES INSTANCE POSTGRES11				
<ul style="list-style-type: none"> Instance Analysis Waits Waits details IO stats Execution stats Sessions details Instance load Info Alerts Database space 	Basic information		Last changes	
	Parameter	Value	Date change	Description
	Server type	PRODUCTION DATABASE	2019-01-28 01:17:52	Server parameter Parameter config_file changed to C:/PostgreSQL/data/pg10/postgresql.conf
	config_file	C:/PostgreSQL/data/pg10/postgresql.conf	2019-01-28 00:56:30	Last tablespace created pg_default
	data_directory	C:/PostgreSQL/data/pg10	2019-01-28 00:56:30	Last database created template1
	log_directory	C:/POSTGR-1/data/logs/pg10		
	block_size	8192		
	max_connections	100		
	autovacuum	on		
	server_version	10.2		
Databases count	10			
Tablespaces count	3			

When user click in the **Database space** can get to know the current size of database on instance (size is expressed in MB). The data on the chart show the information about the space used (Space Used),

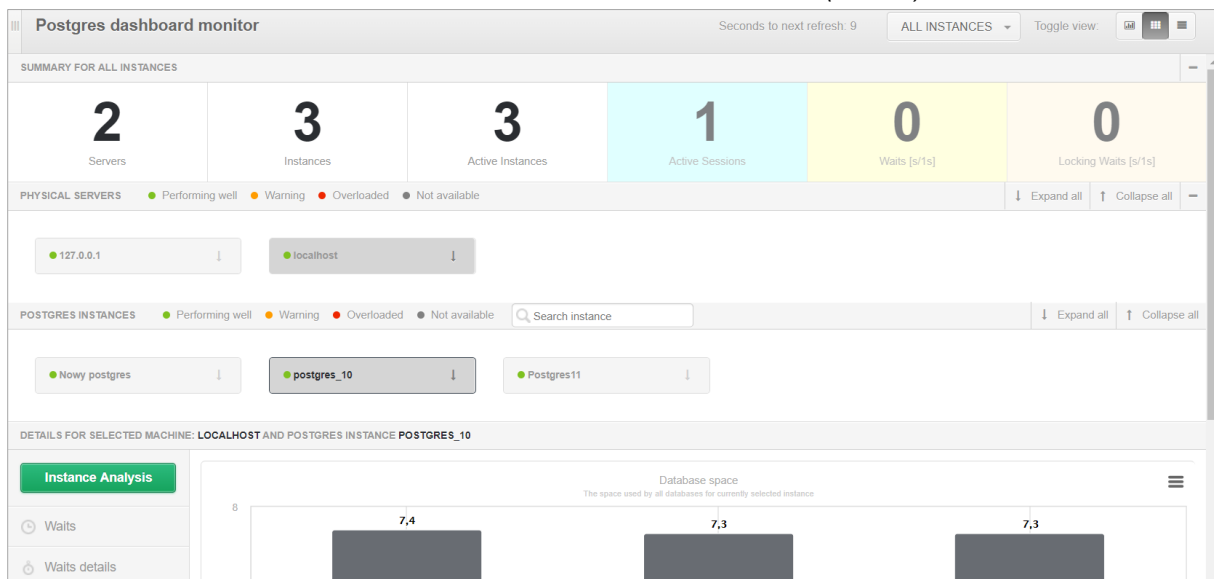
space free (Space Free) and the space the database file can be extended to the maximum (Note: it is impossible to check available space on the disk).



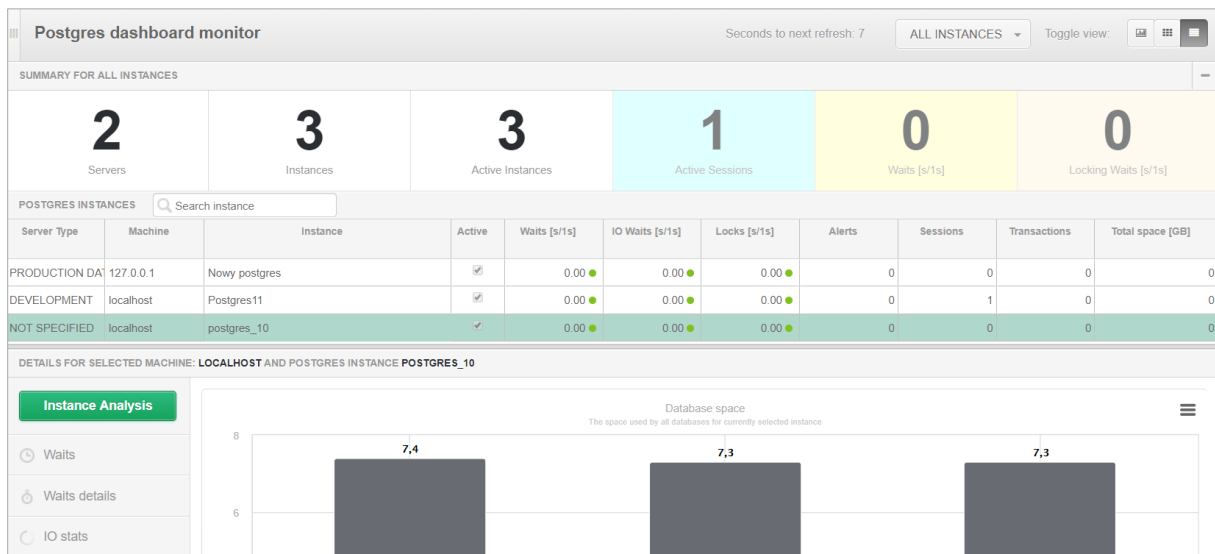
6.1.5 Dashboard – various forms of presentation

Dashboard is available in 3 modes, switched by click **[Toggle View]** icon in the top right corner. Available modes:

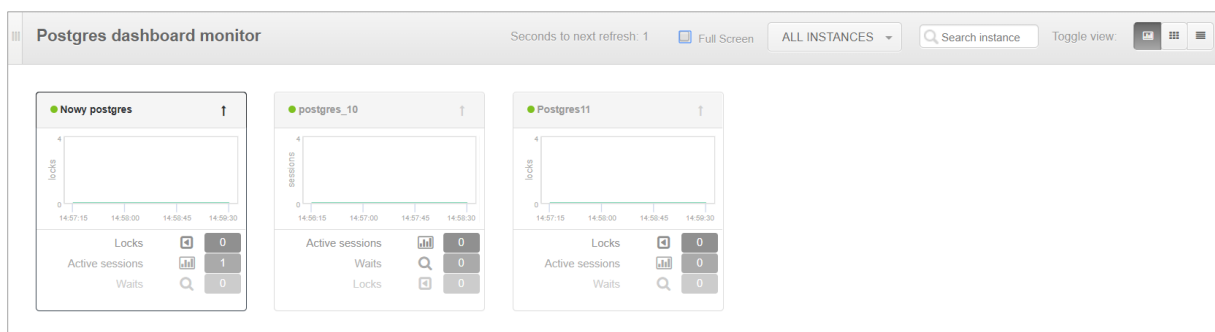
Icon View – shows monitored servers/databases in the form of icons (default)



Grid View – shows instances in the table form



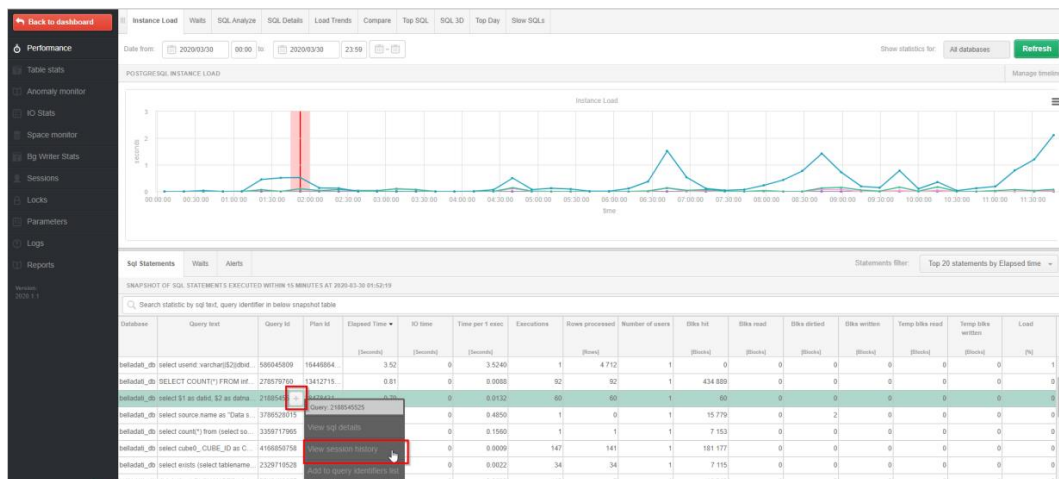
Television Mode – shows instances in the icons form with automatically switching performance indicators.



6.1.6 Additional functionalities

6.1.6.1 Quick access to session history for queries

In the application User has the ability to quickly go to session history for a given query. In the DBPLUS Performance Monitor application, the [+] button will always appear when the Query hash query identifier is presented (after press this button a window with available actions appears). In the new version, apart from the option to go to the SQL Details screen, the option to go to session history has been added.



After pressing the button, a session history window dedicated to the given query is opened. The window always opens in the context of a given day (sysdate). The user also has the option of applying several filters available so far on the session history screen.

A quick transition to the session history screen speeds up the analysis of the performance of a given query and enables, e.g. identification of the User who performs the analyzed query.

The option of going to session history by pressing the button available on the bar above the query text has been added to the SQL Details table screen.

6.1.6.2 Format SQL text queries

On each page where the query text is presented, a [SQL Format] button has been added, after which query text will be formatted.

The screenshot shows the 'SQL Statements' section of the DBPLUS Performance Monitor. It displays a table of SQL statements with columns for Database, Query text, Query id, Plan id, Elapsed Time, IO time, Time per 1 exec, Executions, Rows processed, Number of users, Elks hit, Elks read, Elks distrd, Elks writen, Temp blks read, Temp blks writen, and Load. The fourth row is highlighted in green. Below the table, the 'STATEMENT TEXT FOR QUERY ID: 34532881' is shown, along with the SQL query text. A red box highlights a button labeled 'Format SQL' located at the bottom right of the query text area.

6.1.6.3 Grid manager

The ability to change settings will be introduced in stages. First part are introduced changes on the **Load Trends** and **SQL Details** pages.

The User for these tables on the pages can change for each of the columns:

- Order of displayed columns
- Visibility of columns
- Change the format
- Change of precision
- Change of width

Additionally, it is now possible to hide the Summary row on each page, using the settings available after press the **[cog]** icon. As before, the data contained in the grid can be freely exported to a file.

The order of displayed columns

To change the order of columns, click on the header of the column, hold down the mouse button, drag the columns and drop them to the desired place on the table.

Logdate	Elapsed Time	Executions	Active sessions	Blks read	Blks written	Temp blks written	Wait time	IO time	Lock time	Rollbacks	Tuples returned	Rows	No of temp files	Data written to temp	Blks read time	Blks write time	Blks hit
	[Seconds]			[Blocks]	[Blocks]	[Blocks]	[Seconds]	[Seconds]	[Seconds]			[Rows]		[MB]	[Seconds]	[Seconds]	[Blocks]
2020-02-28	409.030	86.827	0	2.689	0	28.188	0	0	0	3.751	22.097.954	455.787	37	220 MB	0	0	2.331.374
2020-03-02	78.360	55.821	0	313	22.705	0	22.127	0	0	2.760	16.272.699	308.416	29	173 MB	0	0	1.497.210
2020-03-03	158.780	60.155	0	344	22.258	0	21.274	0	0	2.974	17.893.563	319.732	28	166 MB	0	0	1.599.230
2020-03-04	217.670	59.481	0	6.151	20.756	0	20.668	0	0	2.937	20.901.482	338.150	27	161 MB	0	0	1.796.034
2020-03-05	151.890	63.469	0	1.003	23.097	0	22.964	1	0	3.087	23.357.777	352.929	30	179 MB	0	0	1.863.128
2020-03-06	166.140	61.587	0	633	22.912	0	22.924	0	0	2.839	25.747.499	359.203	30	179 MB	0	0	1.987.850
2020-03-09	77.110	51.877	0	496	21.084	0	20.720	0	0	2.574	15.619.704	279.140	27	162 MB	0	0	1.466.907
2020-03-10	110.700	59.781	0	369	24.667	0	23.790	0	0	2.944	18.680.150	306.760	31	186 MB	0	0	1.665.009
2020-03-11	83.590	48.204	0	525	20.383	0	19.157	0	0	2.397	16.038.215	269.096	25	150 MB	0	0	1.411.514
2020-03-12	107.670	67.061	0	12.030	26.495	0	26.692	0	0	3.119	25.940.476	369.647	35	210 MB	0	0	2.059.855

Visibility of columns

To hide a column, right-click on the column header to be hidden. A popup menu will open where the Hide column button should be selected. The indicated column is hidden.

Logdate	Elapsed Time	Rows	Executions	Blks hit	Blks read	Blks dirtied	Blks written	Temp blks read	Temp blks written	Wait time
	[Seconds]	[Rows]		[Blocks]	[Blocks]	[Blocks]				[Seconds]
2020-03-09	77.11	279.140	51.877	1.466.907	496	21.084	20.747	20.720	0	0
2020-03-02	78.36	308.416	55.821	1.497.210	313	22.705	22.127	0	0	0
2020-03-11	83.50	260.606	48.204	1.411.514	525	20.383	19.157	0	0	0
2020-03-12	107.87	369.647	67.061	2.059.855	12.030	26.495	26.692	0	0	0
2020-03-10	110.70	306.760	59.781	1.665.009	369	24.667	23.790	0	0	0
2020-03-05	151.89	352.929	63.469	1.863.128	1.003	23.097	22.964	1	0	0
2020-03-03	158.78	310.732	60.155	1.599.230	344	22.258	21.274	0	0	0
2020-03-06	166.14	359.203	61.587	1.987.850	633	22.912	22.924	0	0	0

To reveal a column, click the **[cog]** button in the upper right corner of the table. After the popup menu open, select the **[Show hidden columns]** option, then indicate the column you want to rediscover in the table. The uncovered column will appear last on the right side of the table.

Logdate	Elapsed Time	Rows	Blks hit	Blks dirtied	Temp blks read	Temp blks written	IO time	Active sessions	Sessions	Connecto	Commits	Rollbacks	Tuples returned	Tuples fetched	Tuples inserted	Tuples updated	Blks read time	Blks write time	Blks hit
	[Seconds]	[Rows]	[Blocks]	[Blocks]	[Blocks]	[Blocks]	[Seconds]										[Seconds]	[Seconds]	[Blocks]
2020-03-09	77.11	279.140	1.466.907	21.084	20.747	20.720	0	0	10	221	34.014	2.574	15.619	1.696.148	23.612	0	0	0	0
2020-03-02	78.36	308.416	1.497.210	22.705	22.156	22.127	0	0	11	259	36.894	2.760	16.272	1.680.252	25.262	0	0	0	0
2020-03-11	83.50	260.606	1.411.514	20.383	19.182	19.157	0	0	10	265	31.745	2.397	16.038	1.369.643	21.825	9.237	21.416	0	25
2020-03-12	107.87	369.647	2.059.855	26.495	26.927	26.692	0	0	10	292	43.902	3.119	25.940	2.164.687	29.696	12.373	24.629	0	35
2020-03-10	110.70	306.760	1.665.009	24.667	23.821	23.790	0	0	10	257	39.351	2.844	18.680	1.594.995	26.992	11.495	13.382	0	31
2020-03-05	151.89	352.929	1.863.128	23.097	22.994	22.964	0	0	16	421	41.915	3.087	23.357	1.867.391	28.029	11.356	29.597	0	30
2020-03-03	158.78	310.732	1.599.230	22.259	21.302	21.274	0	0	12	278	39.813	2.874	17.803	1.414.794	27.081	11.200	12.110	0	28
2020-03-06	166.14	359.203	1.987.850	22.912	22.954	22.924	0	0	14	359	41.954	2.839	25.747	2.045.698	25.653	10.320	21.641	0	30
2020-03-04	217.67	338.150	1.796.034	20.756	20.695	20.668	0	0	16	377	41.716	2.937	20.901	1.669.824	23.436	9.518	19.064	0	27

Change of data format / precision

To change the data format settings, precision, right-click on the column heading where you want to change the data. After making changes, save the changes by click **[Apply]** button.

Logdate	Elapsed Time	Rows	Blks hit	Blks dirtied	Temp blks read	Temp blks written	IO time	Active sessions	Sessions	Connecti..	Commits	Rollbacks	Tuples returned	Tuples fetched	Tuples inserted	Tuples updated	Tuples deleted	Conflicts	
2020-03-02																			
2020-03-03																			
2020-03-04																			
2020-03-05																			
2020-03-06																			
2020-03-09																			
2020-03-10																			
2020-03-11	52.77	13 902	17 053	0	0	0	0	0	1	81	10 212	44	831 409	43 034	0	0	0	0	0
2020-03-12	71.77	19 490	20 008	0	0	0	0	0	1	105	13 232	61	1 085 776	57 771	0	0	0	0	0
2020-03-13	21.96	6 680	6 982	0	0	0	0	0	1	36	4 524	23	368 729	19 262	0	0	0	0	0
2020-03-16	33.76	8 902	9 152	0	0	0	0	0	1	48	6 038	28	494 780	26 137	0	0	0	0	0

Change of width

To change the column width, click the column edge, hold and move it to the right or left to change the width. The current solution used in the DBPLUS application adjusts the width of the columns to the width of the screen. Therefore, with many columns in the table, the width of the columns will always be converted in proportion to the width of the screen.

Logdate	Elapsed Time	Rows	Blks hit	Blks dirtied	Temp blks read	Temp blks written	IO time	Sessions	Tuples inserted	Tuples updated	Tuples deleted	Conflicts	No. of temp files	Data written to temp	Deadlocks	Blk read time	Blk write time	Executions	Blks written	
2020-03-02	49.83	16 120	16 510	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	8 109	0
2020-03-03	55.78	15 605	17 161	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	7 842	0
2020-03-04	89.25	10 898	48 383	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	8 305	0
2020-03-05	63.48	18 113	27 936	0	0	0	0	2	0	0	0	0	0	0	0	0	0	0	8 668	0
2020-03-06	55.19	16 672	17 567	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	8 379	0
2020-03-09	52.85	15 095	15 398	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	7 547	0
2020-03-10	72.16	17 223	17 570	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	8 663	0
2020-03-11	52.37	13 902	17 053	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	8 304	0
2020-03-12	71.37	19 490	20 008	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	9 816	0
2020-03-13	21.96	6 680	6 982	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	3 362	0

Storage of table configurations

The configuration for each of the tables is saved in two ways: at the browser cache level on the user's computer or in the repository database.

In order to permanently save the settings to the repository database, Windows authorization must be enabled in the DBPLUS Performance Monitor application (enabled at the Configuration Wizard level), and the Security module (Menu Configuration> Setings: Security "ON") must be started. The settings are saved for all monitored instances for each user separately.

Restore default settings

If User need to return to the default settings, they can do this by click the cog button and select **[Restore grid defaults]**.

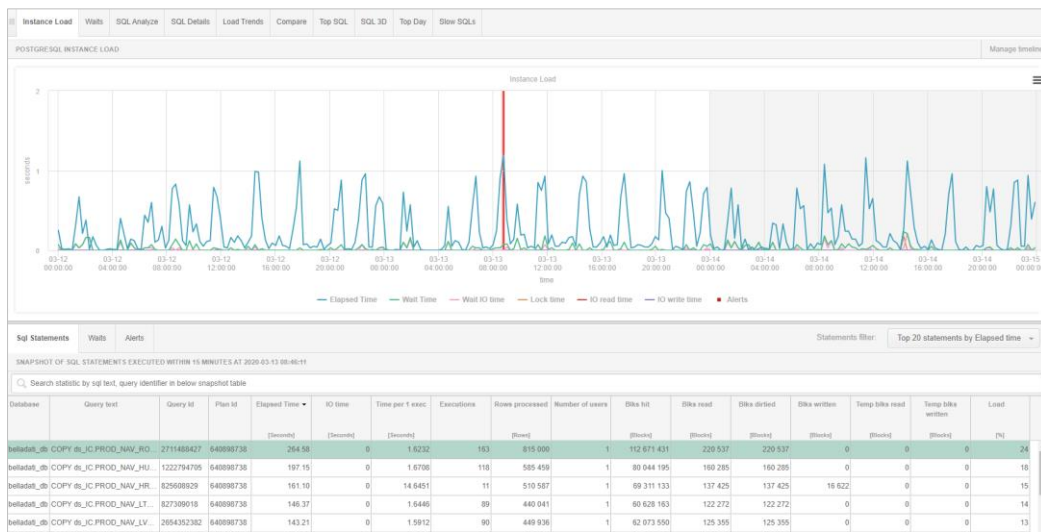
Date	Plan id	Elapsed Time	Blks read time	Blks write time	Executions	Blks hit	Blks read	Blks dirtied	Blks written	Rows per 1 Exec	Blks hit per 1 Exec
2020-03-16 10:09:01	2628428938	2.9	0	0	80	60	0	0	0	1.00	1.00
2020-03-16 10:39:21	2628428938	2.8	0	0	80	60	0	0	0	1.00	1.00
2020-03-16 10:24:11	2628428938	2.6	0	0	80	60	0	0	0	1.00	1.00
2020-03-16 13:10:11	2628428938	2.4	0	0	59	59	0	0	0	1.00	1.00

At any time, the User can restore the default setting for a given column by click on the **[Restore defaults]** button for a given column.

6.1.6.4 Remembering settings on the screen

The function works at the level of database details (Instance Analysis) and consists in remembering the last selection / indication or filter that is selected or searched by the User on a given page in the application.

If we have a "clickable" chart presented on the page, the selected snap indication on the chart is remembered.

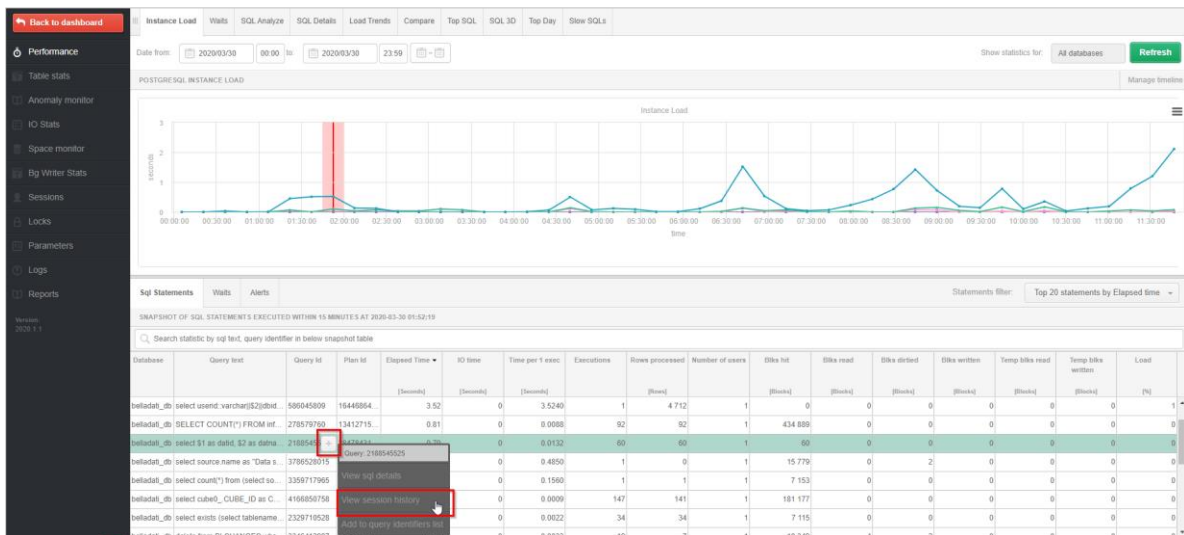


Remembering works only within a given database and after the analysis (exiting to Dashboard or changing the database to another one) the application returns to the default settings.

This feature is based on remembering and saving information at the user's session level. Clearing the browser cache returns to the default settings.

6.1.6.5 Quick access to session history for queries

In the DBPLUS Performance Monitor application, the [+/-] button will always appear when the Query hash query identifier is presented (after press this button a window with available actions appears). In the new version, apart from the option to go to the SQL Details screen, the option to go to session history has been added.



After pressing the button, a session history window dedicated to the given query is opened. The window always opens in the context of a given day (sysdate). The user also has the option of applying several filters available so far on the session history screen.

A quick transition to the session history screen speeds up the analysis of the performance of a given query and enables, e.g. identification of the User who performs the analyzed query.

The option of going to session history by pressing the button available on the bar above the query text has been added to the SQL Details screen.

6.2 „Instance Analysis” Menu

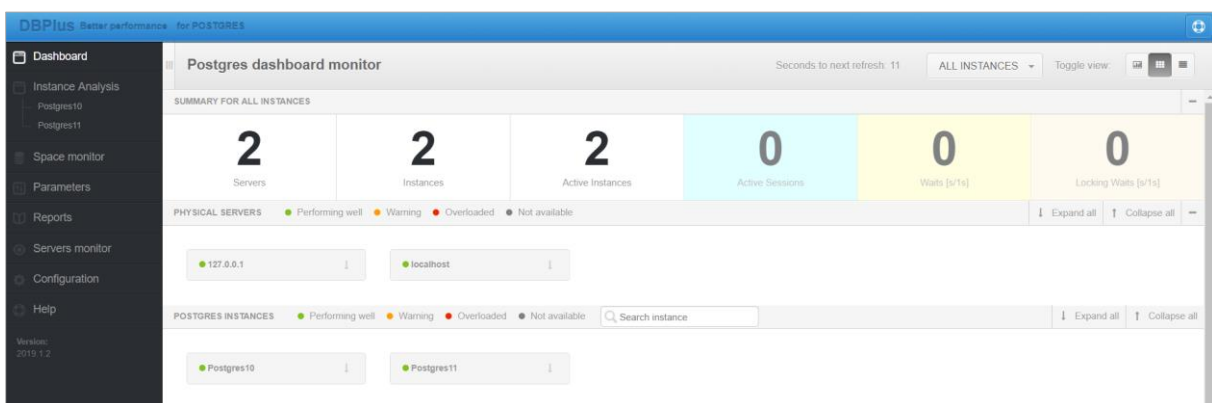
6.2.1 „Performance” Menu – Instance Analysis

Dashboard of the DBPLUS System Performance Monitor allows the user to track the performance of PostgreSQL instances, and show how it looked over the last 15 minutes or the last 24 hours. For a detailed analysis of the load at any given moment in time, and seek answers to questions like:

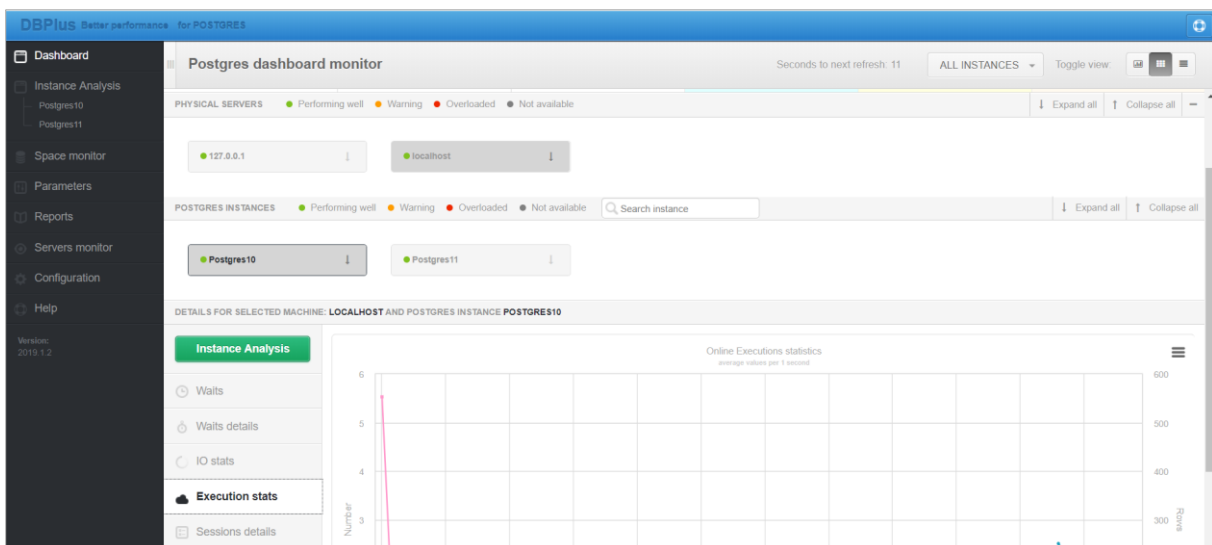
- Why database run slow?
- Why user had problems in the application 3 days ago at 15:48?
- Why my report performed 15 minutes?
- etc.

... Enter the module **Instance Analysis** and there are two options:

- On the left side of the menu, click on **[Instance Analysis]** shows a list of PostgreSQL instances.



- Display the detail of the instance after it has been selected on the Dashboard.



6.2.1.1 „Instance Load” Tab

Instance Load is a screen that shows the load of PostgreSQL instances over time. In the Performance module, there is the option to modify the date range (unlike the Dashboard). First, user can here:

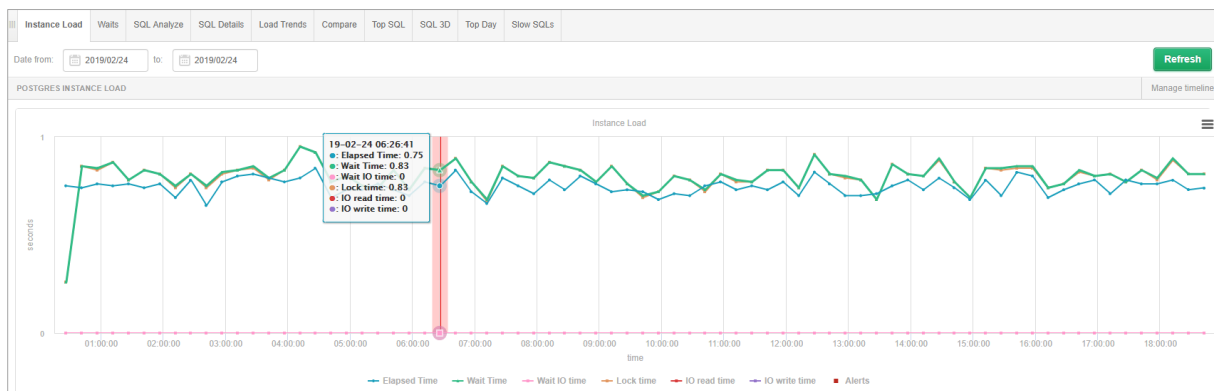
- Check the load of the instance in the wider horizon e.g. today, yesterday, a month or even 3.5 years ago,
- Look at the SQL queries / commands, which generated the load
- Asses what SQL instance did at this time i.a. if performed a lot of disk operations, whether there were locks, etc.

Instance Load screen consists of:

- filtration fields - fields of dates by which user define the period in which they want to check- the load
- chart presents data for selected statistics,
- the load information at a given moment of time:
 - list of queries with execution statistics,
 - waits - what SQL Instance was doing at the time to perform queries
 - load from the point of view of databases on instances

Chart consists of:

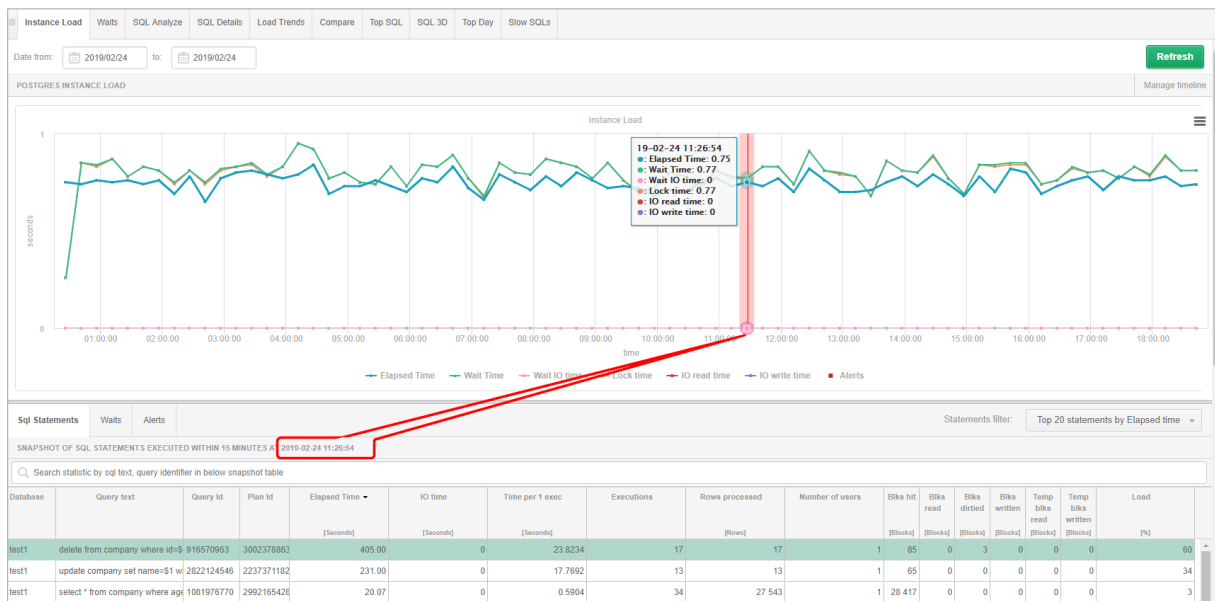
- Elapsed Time – shows the summary time all users wait on the result of a query at a given second of time.
- Wait time – total wait time of queries in a given second of time
- Wait IO time – queries wait time for I/O tasks
- Lock time – time the queries wait in a second
- IO read time – data read time (by the queries),
- IO write time – data write time (by the queries),
- Alerts – the number of alerts for a given snap.



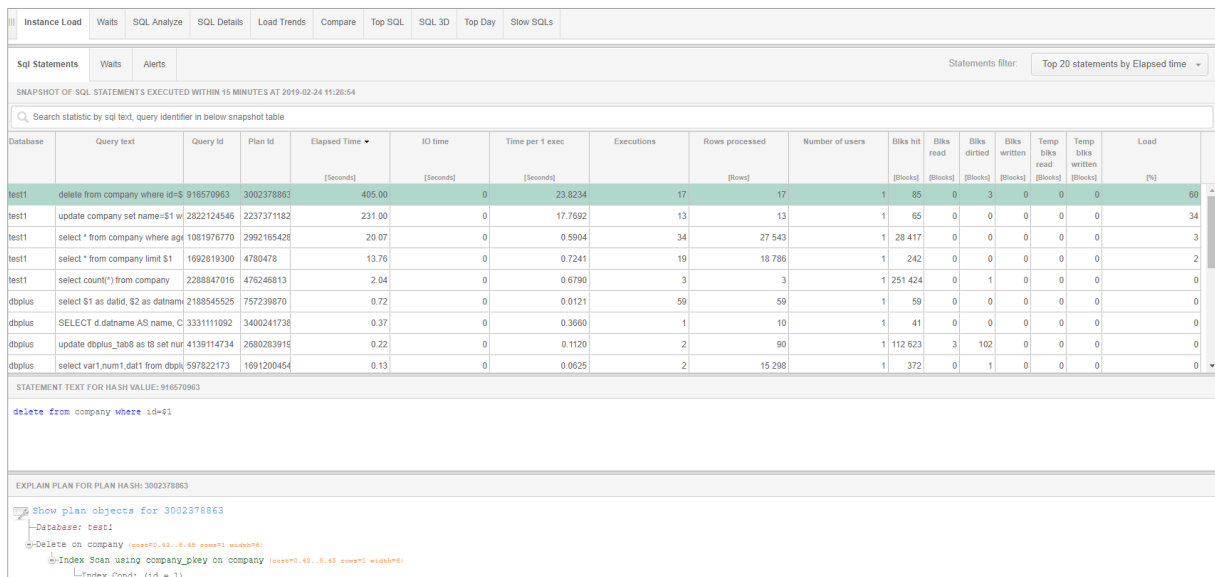
The chart is "clickable" - click on the selected part / section will refresh the bottom of the screen with information about requests and waits that generated the data load.

Data for chart of the Instance load is calculated by monitoring service DBPLUSCATCHER - a component of the DBPLUS Performance Monitor. Monitoring Service performs a few procedures examining SQL instance performance. The result of the operation of these procedures is a snapshot (snap) that is created every 15 minutes.

After click on the selected point in time, the lower part of the screen is refreshed, information about queries and waits.



If user scroll down the screen, they get information about queries performed in a given snap:



These are 3 additional sub-tabs:

- SQL Statements,
- Waits,
- Alerts.

6.2.1.1.1 SQL Statements

SQL Statements is the query statistics presented in the form of a table. System displays the most aggravating question for the duration of Elapsed Time by default. The method of display can be changed if user view a complete list of queries that participated in the load.

The table with queries:

- Sort on any column

- Search e.g. after a part of the query text

The table contains information from the **pg_stat_statements**

- **Database** – name of the database where the query was activated,
- **Query Text** – full text of SQL command,
- **Query Id** –
- **Plan id** – an identifier of execution plan generated by DBPLUS PM,
- **Elapsed Time** – the duration in seconds for all query executions within last 15 minutes
- **IO Time** – time spent on reading / saving blocks
- **Time per 1 exec**– duration of query for a single execution (seconds),
- **Executions** – number of executions of the query in last 15 minutes,
- **Rows processed** – number of rows returned by the query in last 15 minutes,
- **Number of users** – the number of users performing queries in a given period of time,
- **Blks hit** – number of read blocks from memory by the queries
- **Blks read** - number of read blocks from disks by the queries
- **Blks dirtied** – the number of “dirty” blocks
- **Blks written** – the number of written blocks by the queries
- **Temp blks read** – number of temporary blocks read by the queries,
- **Temp blks written** - number of temporary blocks written by queries,
- **Load** – percentage of the duration of a given query in relation to other queries during the last 15 minutes.

IMPORTANT - in PostgreSQL statistics are counted after the query. In the case of a long-time query (e.g. more than 1 hour), the information about the query will appear only in the snap in which the query has been completed and all statistics will be counted for the entire query.

In the column Query ID (each line presents statistics of the execution) shows **[Plus]** button

Sql Statements							
Waits		Alerts					
SNAPSHOT OF SQL STATEMENTS EXECUTED WITHIN 15 MINUTES AT 2019-02-24 11:26:54							
<input type="text" value="Search statistic by sql text, query identifier in below snapshot table"/>							
Database	Query text	Query Id	Plan Id	Elapsed Time ▾	IO time	Time per 1 exec	
				[Seconds]	[Seconds]	[Seconds]	
test1	delete from company where id=\$	916570963	3002378863	405.00	0	23.8234	
test1	update company set name=\$1 w	2822124546	2237371182	231.00	0	17.7692	
test1	select * from company where age	1081976770	2992165428	20.07	0	0.5904	
test1	select * from company limit \$1	1692819300	4780478	13.76	0	0.7241	
test1	select count(*) from company	2288847016	476346812	2.04	0	0.6790	
dbplus	select \$1 as datid, \$2 as datnam	2188545525	757239870	0.72	0	0.0121	
dbplus	SELECT d.datname AS name, C	3331111092	3400241738	0.37	0	0.3660	
dbplus	update dbplus_tab8 as t8 set nur	4139114734	2680283919	0.22	0	0.1120	

When user click on **[Plus]** it shows additional context menu, which allows for detailed analysis of a query, discussed in the **"SQL Details"** section.

Sql Statements		Waits	Alerts				
SNAPSHOT OF SQL STATEMENTS EXECUTED WITHIN 15 MINUTES AT 2019-02-24 11:26:54							
Search statistic by sql text, query identifier in below snapshot table							
Database	Query text	Query Id	Plan Id	Elapsed Time [Seconds]	IO time [Seconds]	Time per 1 exec [Seconds]	
test1	delete from company where id=\$	916570963	3002378863	405.00	0	23.8234	
test1	update company set name=\$1 w	2822124546	2237371182	231.00	0	17.7692	
test1	select * from company where age	1081976770	2992165428	20.07	0	0.5904	
test1	select * from company limit \$1	1692819300	4780478	13.76	0	0.7241	
test1	select count(*) from company	2288847016		2.04	0	0.6790	
dbplus	select \$1 as datid, \$2 as datnam	2188545525		0.72	0	0.0121	
dbplus	SELECT d.datname AS name, C	3331111092		0.37	0	0.3660	
dbplus	update dbplus_tab8 as t8 set nur	4139114734	2680283919	0.22	0	0.1120	

If user select **“Add to query hash list”**, user move a query identifier to the clipboard with a list of queries for later analysis of specific queries.

Below the slide of 4 queries added to the analysis in **SQL Details** functionality.

Instance Load
Waits
SQL Analyze
SQL Details
Load Trends
Compare
Top SQL
SQL 3D
Top Day
Slow SQL

Click on Query Id to analyze Query Performance Details

Query Identifiers list

- 18272832
- 916570963
- 2822124546
- 2188545525

Sql Statements		Waits	Alerts				
SNAPSHOT OF SQL STATEMENTS EXECUTED WITHIN 15 MINUTES AT 2019-02-24 11:26:54							
Search statistic by sql text, query identifier in below snapshot table							
Database	Query text	Query Id	Plan Id	Elapsed Time [Seconds]	IO time [Seconds]	Time per 1 exec [Seconds]	
test1	delete from company where id=\$	916570963	3002378863	405.00	0		
test1	update company set name=\$1 w	2822124546	2237371182	231.00	0		
test1	select * from company where age	1081976770	2992165428	20.07	0		
test1	select * from company limit \$1	1692819300	4780478	13.76	0		
test1	select count(*) from company	2288847016		2.04	0		
dbplus	select \$1 as datid, \$2 as datnam	2188545525		0.72	0		
dbplus	SELECT d.datname AS name, C	3331111092		0.37	0		
dbplus	update dbplus_tab8 as t8 set nur	4139114734	2680283919	0.22	0		

STATEMENT TEXT FOR HASH VALUE: 916570963

```
delete from company where id=$1
```

In the **SQL Statements** tab, after select a row in the table with queries is presented the full text of the query with the execution plan. Click on query will refresh

information about the content and the query plan.

STATEMENT TEXT FOR HASH VALUE: 1081976770

```
select * from company where age = $1 limit $2
```


EXPLAIN PLAN FOR PLAN HASH: 2992165428

Show plan objects for 2992165428

- Database: test1
 - Limit (cost=0.43..4.28 rows=1 width=68)
 - Index Scan using idx_company_age on company (cost=0.43..30795.28 rows=8001 width=68)
 - Index Cond: (age = 1)

Available information in the **(Explain plan)**:

- Name of the database in which the query is performed
- Algorithm of the explain plan
- List of parameters (example parameter values) used when compile the first explain plan.

In the area of the explain plan, there is a link  that allows to display statement script with filled parameters

STATEMENT TEXT FOR HASH VALUE: 1081976770

```
select * from company where age = $1 limit $2
```

EXPLAIN PLAN FOR PLAN HASH: 2992165428

Show plan objects for 2992165428

Explain plan options

Show statement script with filled parameters

- company (cost=0.43..30795.28 rows=8001 width=68)
 - Index Cond: (age = 1)

User click on the **Show plan objects for ...** link, user is moved to the screens that allows analysis of the objects take part in the query, i.a.:

- what tables, indexes participated in the execution of the query
- how the engine referred to the given objects
 - seek for data (seek)
 - read full data (scan index or table)
- whether the query was performed in multithreaded mode
- what mechanism was used to download and connect "data" from objects:
 - Nested Loop

- Hash / Merge Join connection

By click the [Show Plan Objects](#) link, user receive:

SQL TEXT		EXPLAIN PLAN			
<pre>update company set name='1' where id=62</pre>		<pre> Database: test1 ├─Update on company (company) │ └─Index Scan using company_pkey on company │ └─Index Cond: (id = 1) </pre>			
OBJECTS USED IN EXPLAIN PLAN		INDEXES FOR SELECTED OBJECT PUBLIC.COMPANY			
Type	Schema	Object Name	Name		
TABLE	public	company	company_pkey		
INDEX	public	company_pkey	idx_company_name		
			idx_company_age		
Object columns Details for TABLE: public.company Load object properties (slow)					
Column	Type	Position	Is nullable	Unique values	Most common values
id	integer	1	NO		-1
name	text	2	NO		James; Paul; Mark; David; Allen; Teddy; Kim
age	integer	3	NO		819
address	character(50)	4	YES		25; 32; 27; 22; 24; 344; 23; 87; 702; 93; 248; 299; 538; 41
salary	real	5	YES		6; 10000; 20000; 65000; 85000; 15000; 45000

In the window user has information about the query text and the explain plan. Below user can see areas:

Objects Used in Explain Plan – a list of all objects used by the query in the explain plan

Indexes for selected object – list of indexes for selected table - row selected in the "Objects Used in Explain Plan"

Area consists of 3 tabs (Load object properties checkbox selected):

- Object Columns** – a list of the individual columns of the selected object, with information: column name, data type, column id, density (the lower density, the higher selectivity of the column)
- Info** – DDL command that creates the object,
- Properties** – additional properties of selected object

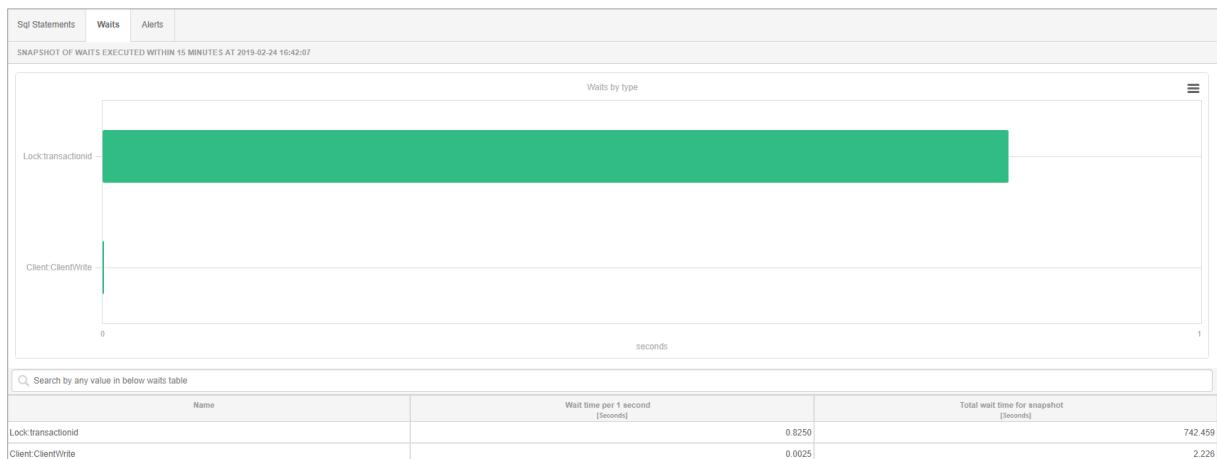
When the explain plan is analyzed, pay attention to:

- **Limit the choice of data, or of the data with the where clause and table joins**
- **Whether the request is with parameters or literals**
- **The operation the PostgreSQL engine chose to retrieve/download data**
- **Whether the table has appropriate indexes**

6.2.1.1.2 Waits

Waits tab shows the expectations PostgreSQL instance was spending time. The data is presented in graphical and tabular form.

The graph shows the duration for each second of the selected snapshot (time 15 minutes) of each type of wait / wait that occurred at that time on the instance.



A table is located below the graph with columns:

- Name - the name of wait
- Wait time - per 1 second (sec.) - Duration of wait type in seconds
- Total wait time for snapshot (sec.) - the total duration of wait type in the snapshot (15 minutes)

6.2.1.1.3 Alerts

The next tab is **Alerts** - this tab contains performance messages for a given snapshot.

6.2.1.2 „Waits” tab

Waits tab shows the duration of waits, which occurred at a time for all sessions on PostgreSQL Instance. The module consists of two elements available in the sub-tabs:

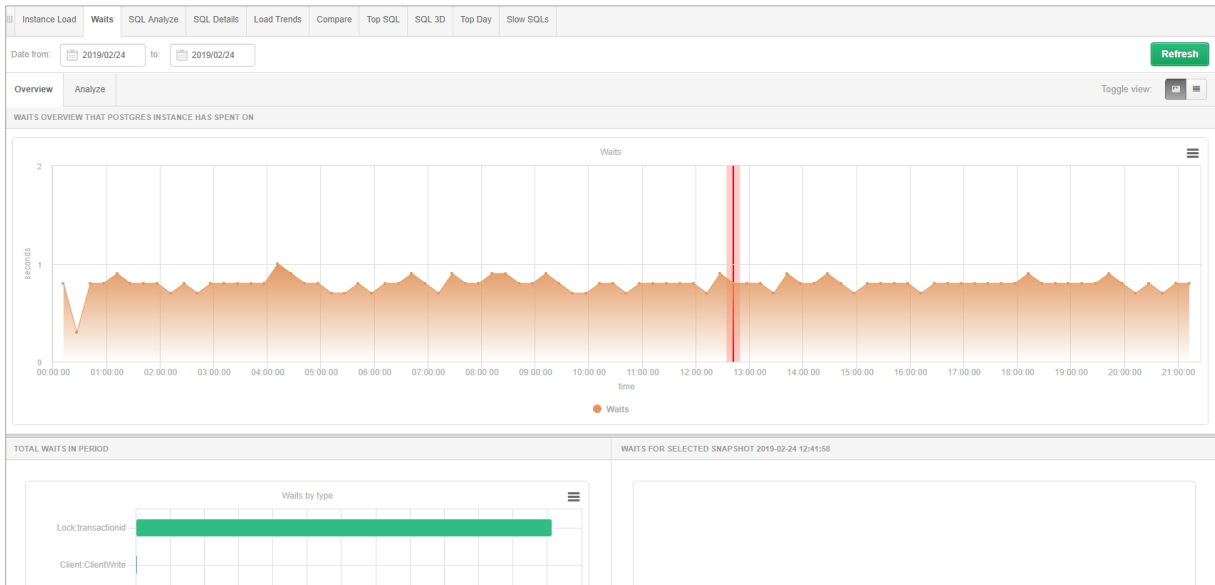
- **Overview** – allows to review wait times at a given time or for a selected snapshot,
- **Analyze** – gives the opportunity to analyze individual waits over time.

The Overview sub-tab shows the duration of wait time that occurred at a given time. Depend on the select the Toggle View option, data can be presented in graphic form or in the form of a table.

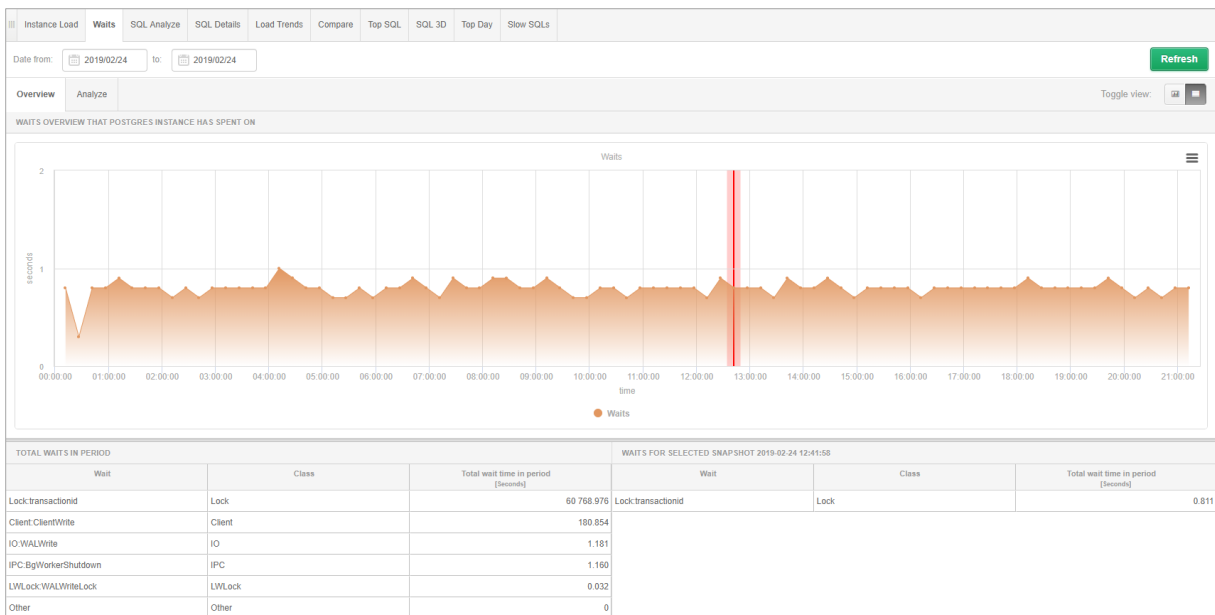
Waits screen in a similar way to **Instance Load** screen, consists of the fields:

- filtration fields - fields of dates by which user define the period in which user want to become familiar with the expectations / waits of the instance
- graph presents the level of waits
- detailed information about waits in a given moment of time

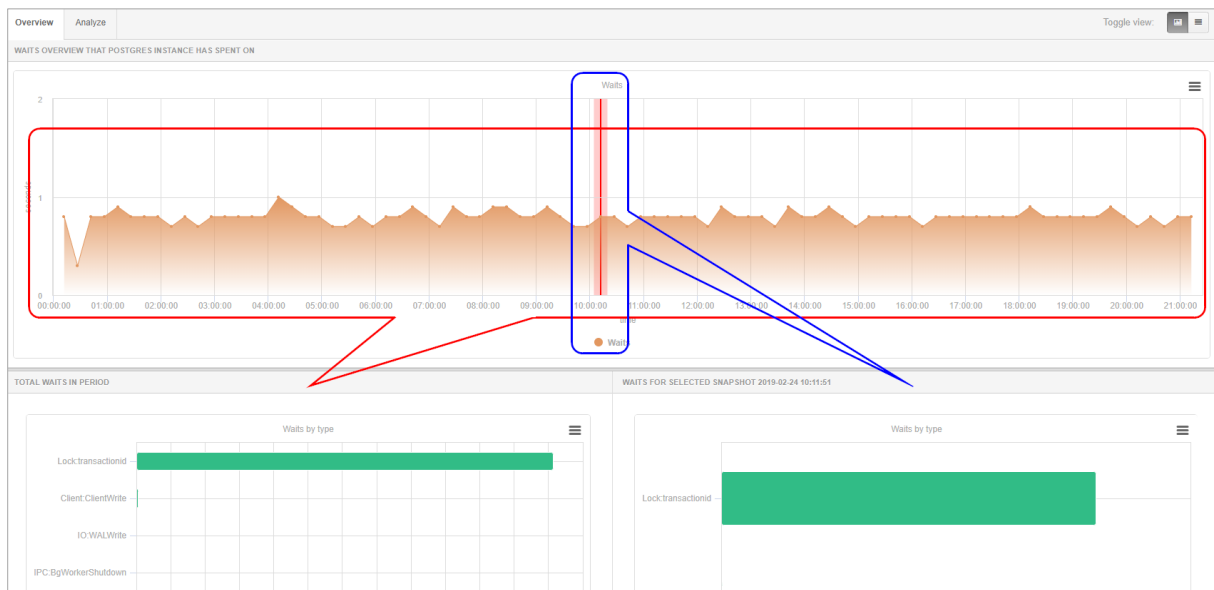
The Y axis of the graph illustrates (in seconds for a given second) time of all waits that occurred during the period shown on the X-axis. X-axis of the graph shows the period waits were occurred.



After switch the view with **[Toggle view]** user get:



Similar like in the screen **[Instance Load]**, **[Waits]** chart is "clickable". Click on the part of the graph (its point) will show user waits summary, appropriate for a snapshot in time.



Above chart gives information:

- **Total waits in period** - what the instance does during the day (the default) or a selected period limited by dates in the filter
- **Waits for selected snapshot** - what Instance does during the last snap

The system also allows to analyze individual Waits - the frequency, length and time of occurrence. To do this, click on the subtab „**Analyze**”:

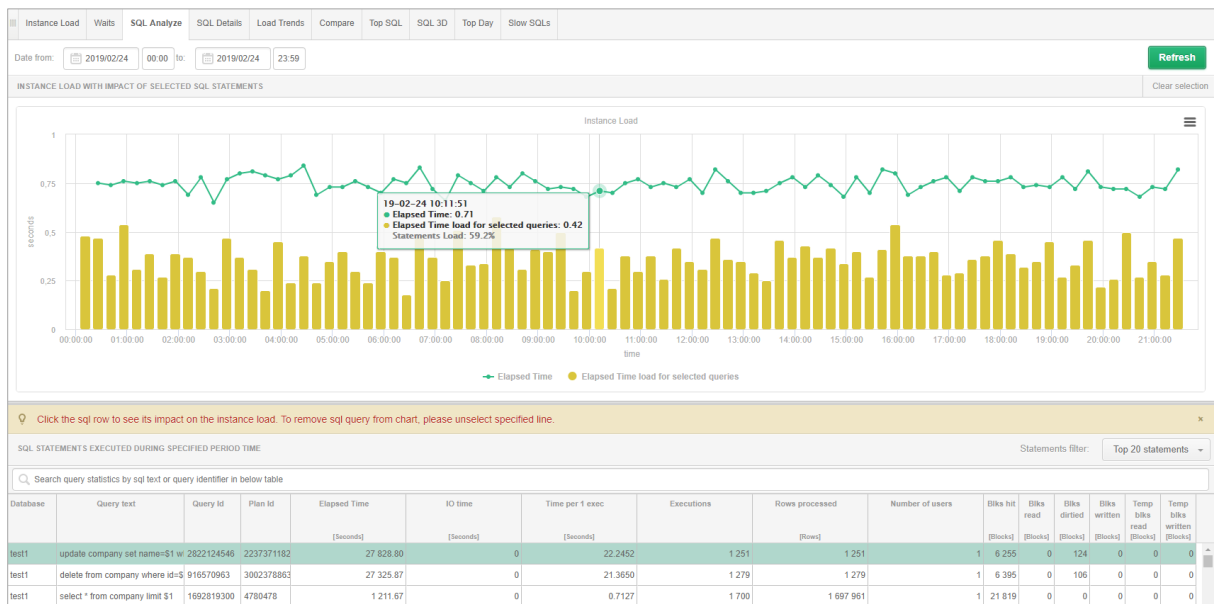
User can analyze performance waits or all waits grouped by day, hour, snapshot. “SQL Analyze” tab.

6.2.1.3 „SQL Analyze” Tab

SQL Analyze functionality presents an additional view of Instance load. As with the **Instance Load** chart, user has a graph that shows duration of queries in the PostgreSQL Instance.

The screen consists of areas:

- Filtration fields:
 - date and time fields by which the time is determine in which users can to familiarize themselves with the instance load
- the graph shows Elapsed Time statistic,
- cumulative statistic broken down into queries that generated a specific load in a given period.
- after indicate the row in the table user have information about the text and the query plan



Graphs Y-axis shows the number of seconds for each second of duration of the query in PostgreSQL instance database.

The X-axis represents the time at which the query caused the utilization of Database' server. Differences that can show up between the load shown in the Instance Load graph, and utilization statistics of the Database' server from the operating system side, arise due to including in the chart all kinds of waits, which is not shown in the operating system. The graph shows a full picture or performances, not just time.

On the example screenshot after click the first query, user see the share in the instance load for a given period presented with an accuracy of 15 minutes of samples.

Table in **StatementsSQL** tab shows statistics for each query:

- **Database** – name of the database where the query was activated,
- **Query Text** – full text of SQL command,
- **Query Id** – an identifier of a query
- **Plan id** – an identifier of execution plan generated by DBPLUS PM,
- **Elapsed Time** – the duration in seconds for all query executions within last 15 minutes
- **IO Time** – time spent on reading / saving blocks
- **Time per 1 exec**– duration of query for a single execution (seconds),
- **Executions** – number of executions of the query in last 15 minutes,
- **Rows processed** – number of rows returned by the query in last 15 minutes,
- **Number of users** – the number of users performing queries in a given period of time,
- **Blks hit** – number of read blocks from memory by the queries
- **Blks read** - number of read blocks from disks by the queries
- **Blks dirtied** – the number of “dirty” blocks
- **Blks written** – the number of written blocks by the queries
- **Temp blks read** – number of temporary blocks read by the queries,
- **Temp blks written** - number of temporary blocks written by queries,

IMPORTANT: SQL Analyze screen maintains similar functionality to the Instance Load:

- Click on a query row (not including the load graph) will display the full text of the query and its execution plan
- Next to the query identifier the [Plus] button is located, which adds a query to the clipboard with a list of queries

SQL STATEMENTS EXECUTED DURING SPECIFIED PERIOD TIME								
<input type="text" value="Search query statistics by sql text or query identifier in below table"/>								
Database	Query text	Query id	Plan id	Elapsed Time [Seconds]	IO time [Seconds]	Time per 1 exec [Seconds]	Executions	Rows pro [Re
test1	update company set name=\$1 w	28221244		27 828.80	0	22.2452	1 251	
test1	delete from company where id=\$	916570963		27 325.87	0	21.3650	1 279	
test1	select * from company limit \$1	1692819300		1 211.67	0	0.7127	1 700	
test1	select * from company where age	1081976770		941.81	0.36	0.5391	1 747	
test1	select count(*) from company	2288847016	476246813	281.04	0	0.5091	552	
dbplus	select \$1 as datid, \$2 as datnam	2188545525	757239870	72.68	0	0.0141	5 145	
dbplus	update dbplus_tab8 as t8 set nur	4139114734	2680283919	14.83	0.01	0.0862	172	
dbplus	select var1,num1,dat1 from dbpl	597822173	1691200454	7.24	0	0.0421	172	

STATEMENT TEXT FOR LAST SELECTED QUERY ID: 2822124546

```
update company set name=$1 where id=$2
```

It is worth to note that for individual components of the screen user can change the height - this applies to i.a.: charts, data tables, query text controls, execution plan.

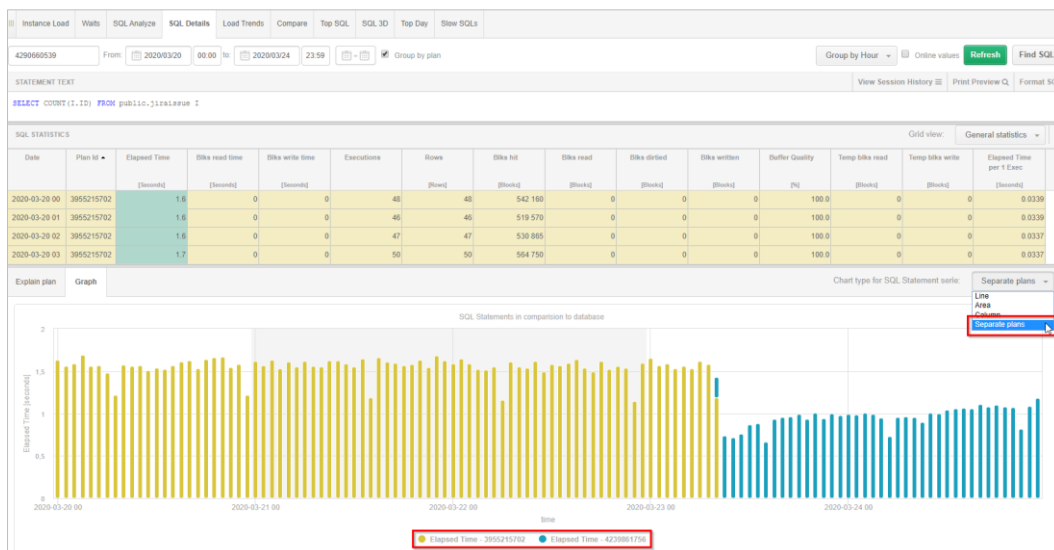
6.2.1.4 „SQL Details” Tab

SQL Details tab shows detailed information about the query: execution plan, whether the request has changed execution plan, the number of returned records, the number of executions etc.

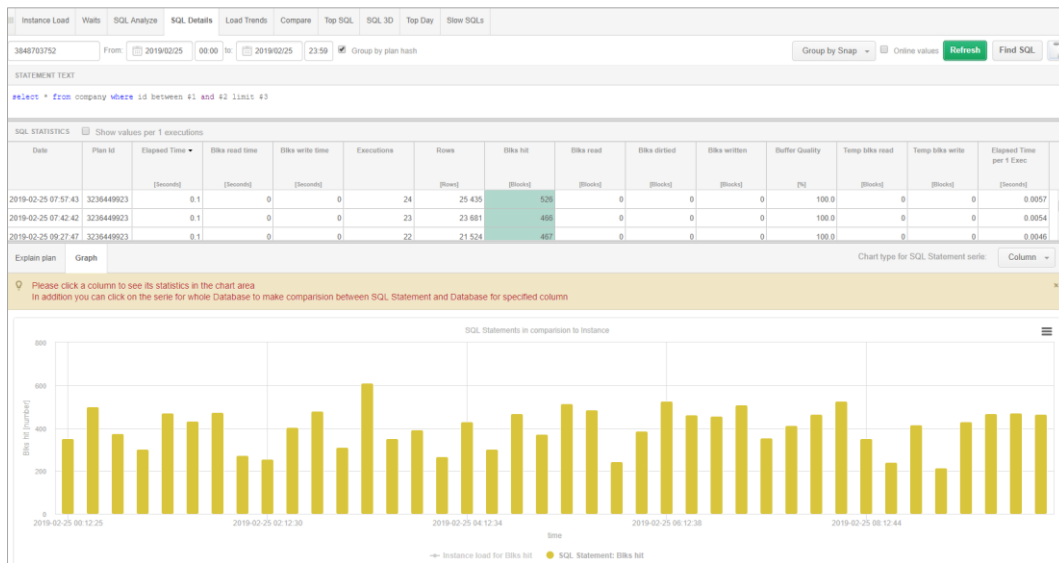
This information provides the opportunity to decide whether it makes sense to optimize query and assess participation in the instance load.

In the latest version we have changed the default settings in the application related to the default chart made available on the Graph tab on the SQL Details page.

This chart presents the statistics selected in the table, broken down into explain plans. If the query is made based on several plans, each query explain plan will be marked with a different color in the table and bar in the chart. To change the chart type and return to the previous version, select a different chart type, e.g. Column.

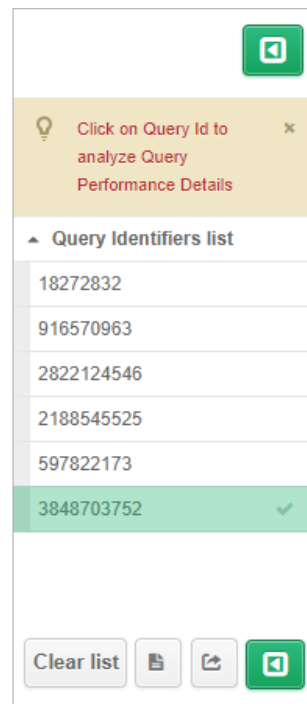


SQL Details column type graph:



„SQL Details” tab is divided into areas:

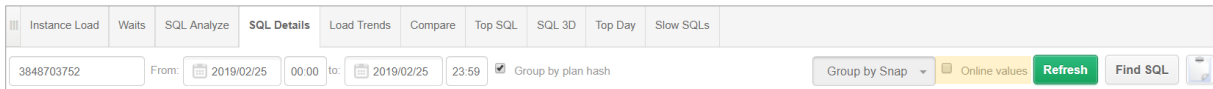
- Clipboard with a list of query IDs (hidden and developed as a result of click the green button in the left side of the screen) - the queries to the clipboard are added from screens:
 - Performance ->Instance Load
 - Performance ->SQL Analyze
 - Performance ->Top SQL
 - Performance ->SQL 3D
 - Performance ->Top Day
 - Performance ->Slow SQLs



IMPORTANT: List of queries is remembered under proper Instance for specified User. That list can be saved to file or opened again.

- Filters area and the way to display statistics for:
 - Specified Query ID of the query
 - Selected date range
 - Group statistic by snap, hour, day, etc.

- Navigation buttons which allow to:
 - Refresh the screen,
 - Search another query **[Find SQL]**,
 - Show the statistic of queries in report.



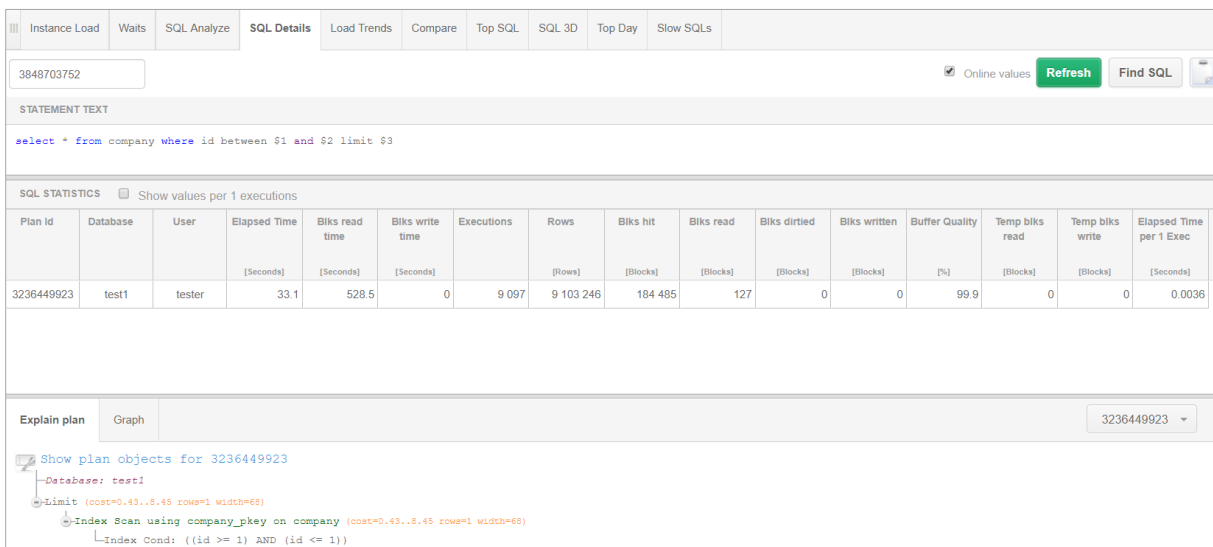
Select checkbox **Online Values** and click the **[Refresh]** button will present statistic online for a query by information which are available in the system view *pg_stat_statement*.

Online values option allows Users to display current information about queries stored in the PostgreSQL Instance buffer. Information about queries in other tabs is presented for a 15-minute period. For this screen, data is read directly from the instance.

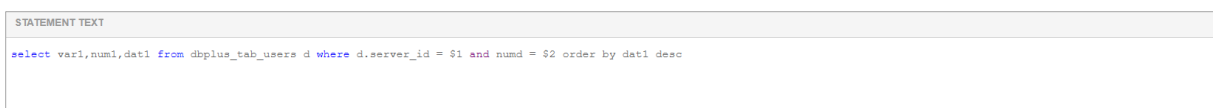
To check whether the query is performed now, after supply the Query Hash identifier, activate the **Online Values** checkbox and click the **[Refresh]** button to observe the value in the columns Execution, Elapsed Time. If the values change, it means that the query still executes. If the values are fixed, the query stop run.

Filter **Group by period** - shows statistics for a given query grouped according to the choice:

- **No group by period** - selection date ranges from 1 to 20 days of the month will show summary statistics for the selected period
- **Month** – shows statistic broken down by months
- **Day** - shows statistics broken down by day
- **Hour** - shows statistics broken down by one hour
- **Snap** - shows statistics broken down by snapshots - periods of 15 minutes



- Area with query text - where user can control the height of the window, i.a. convenient for longer query content



- Detailed statistics of performance in the form of a table (the ability to change the table height).

SQL STATISTICS <input type="checkbox"/> Show values per 1 executions														
Date	Plan Id	Elapsed Time	Blks read time	Blks write time	Executions	Rows	Blks hit	Blks read	Blks dirtied	Blks written	Buffer Quality	Temp blks read	Temp blks write	Elapsed Time per 1 Exec
		[Seconds]	[Seconds]	[Seconds]		[Rows]	[Blocks]	[Blocks]	[Blocks]	[Blocks]	[%]	[Blocks]	[Blocks]	[Seconds]
2019-02-25 00:12:25	3236449923	0.0	0	0	17	16 602	351	0	0	0	100.0	0	0	0.0020
2019-02-25 00:27:26	3236449923	0.1	0	0	23	22 928	500	0	0	0	100.0	0	0	0.0029
2019-02-25 00:42:27	3236449923	0.0	0	0	18	18 993	374	0	0	0	100.0	0	0	0.0024
2019-02-25 00:57:27	3236449923	0.0	0	0	15	14 468	302	0	0	0	100.0	0	0	0.0023
2019-02-25 01:12:27	3236449923	0.1	0	0	22	22 918	472	0	0	0	100.0	0	0	0.0028
2019-02-25 01:27:28	3236449923	0.1	0	0	20	21 023	434	0	0	0	100.0	0	0	0.0027

- Statistics show: Plan Id – DBPLUS internal identifier of the query plan,
 - Elapsed Time – total time (sec) of query duration in the selected time,
 - Blks read time – time to read the data block
 - Blks write time – time to write the data block
 - Executions – the number of query executions in the selected time
 - Rows– the number of rows processed by the query in the selected time
 - Blks hit – number of read blocks from memory by the queries
 - Blks read – number of read blocks from disks by the queries
 - Blks dirtied – number of dirtied blocks
 - Blks written – number of written blocks by the queries
 - Buffer Quality – percentage of buffer usage
 - Temp blks read – number of temporary blocks read
 - Temp blks write – number of temporary blocks write
 - Elapsed time per 1 Exec – duration of a single query execution in the selected time.
- Explain Plan (**Explain Plan** tab selected).

Explain plan 3236449923

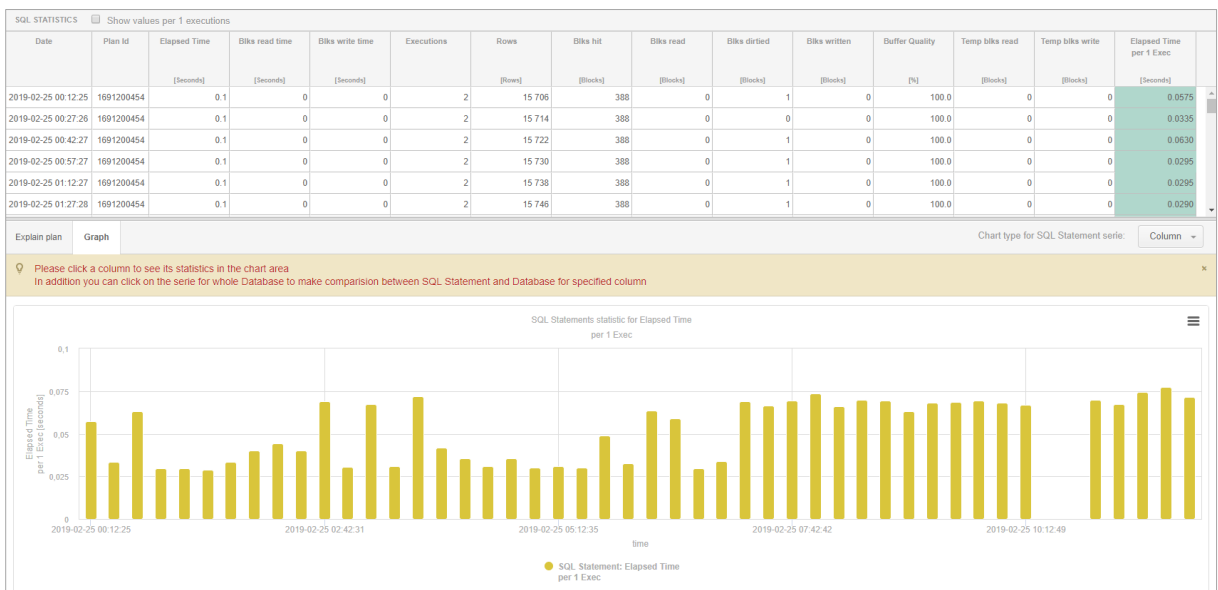
Show plan objects for 3236449923

```

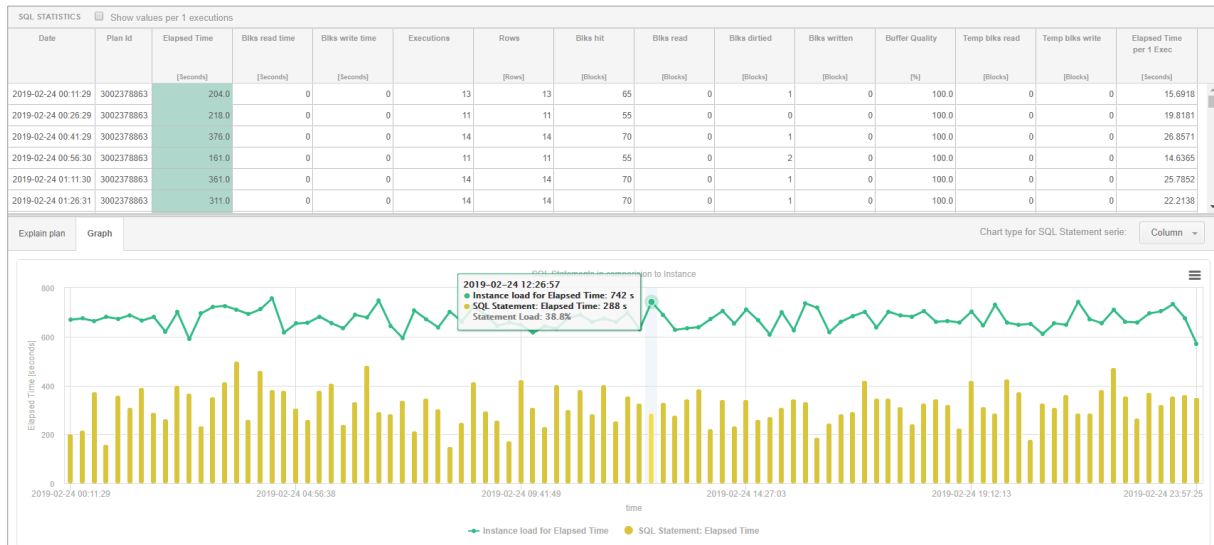
Database: test1
Limit (cost=0.45..8.45 rows=1 width=68)
├─Index Scan using company_pkey on company (cost=0.45..8.45 rows=1 width=68)
│   └─Index Cond: ((id >= 1) AND (id <= 1))

```

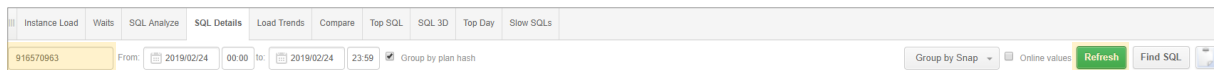
- Graphic presentation (with the SQL Statement Loads tab selected) of any indicator / column from the statistics table.



When click the **SQL Statement Load** tab, user can see the load generated by the given query (line / yellow area) against the background of the total instance load:



Enter query ID in the field: [Enter Query Id](#)



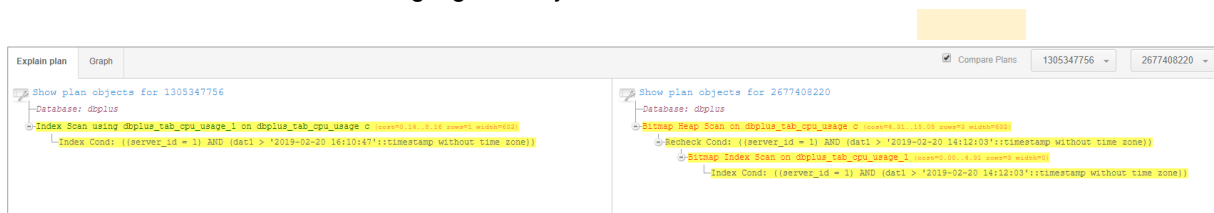
displays statistics for given query identifier with filters set.

IMPORTANT: If User does not know and do not has any query ID and clipboard with a list of queries is empty, they can:

- go to one of the screens (Instance Load, SQL Analyze, Top SQL, SQL 3D, Top Day), where user can search suboptimal/long-lasting query,
- click on the [Find SQL] to search for a specific query (search by its text)

Explain Plan Tab

Shows the query explain plan. If there is more than one for the query then user can click on the Compare plans checkbox, which will display two explain plans - it makes it easier to find differences between them, which in turn are highlighted in yellow:



- In the control of the execution plan is available: Link with additional options for text and explain plan,
- Link **Show Plan Objects**, which allows analyze the explain plan.

6.2.1.4.1 Explain plan options

In the area of the explain plan, there is a link that allows to perform operations:

- Show statement script with filled parameters,
- Generate & Estimate plan.

By click **Show statement scripts** with filled parameters, a new T-SQL block appears in the new window:

- Reference to the database
- Parameter declarations and set their value
- Statement with replaced parameter values

Another function is **Generate & Estimate plan**

This is a useful function when explain plans contain badly substituted parameters or there are no substituted parameters. In this case, an error is visible in place of the query plan, e.g. incorrect query syntax.

The screenshot shows the SQL Analyze interface. At the top, there are tabs for Instance Load, Waits, SQL Analyze, SQL Details, Load Trends, Compare, Top SQL, SQL 3D, Top Day, and Slow SQLs. The instance ID is 414511336. The time range is from 2019/02/27 00:00 to 2019/02/27 23:59. The 'Group by plan hash' checkbox is checked.

The STATEMENT TEXT is:

```
select to_char(t2.logdate,$4) as logdate ,sum(num2) num2,sum(num3) num3 from
dbplus_tab2_size_s t2, (
select max(x.logdate) as logdate from dbplus_tab2_size_s x
WHERE x.server_id = $1
AND x.logdate >= $2
```

The SQL STATISTICS table shows the following data:

Date	Plan Id	Elapsed Time	Blks read time	Blks write time	Executions	Rows	Blks hit	Blks read	Blks
		[Seconds]	[Seconds]	[Seconds]		[Rows]	[Blocks]	[Blocks]	
2019-02-27	0	0.0	0	0	1	8	28	0	

The Explain plan section shows an error:

```
Show plan objects for 520541563
Database: dbplus
42803: column "t2.logdate" must appear in the GROUP BY clause or be used in an aggregate function, error in position: 24
```

In this case, verify that the parameters used to execute the query plan is correct and that all is completed. Too correct the query plan, go to the Generate & Estimate plan. Complete the missing parameters or correct existing ones.

The PLAN GENERATOR window shows the corrected SQL statement:

```
select to_char(t2.logdate,$0) as logdate ,sum(num2) num2,sum(num3) num3 from dbplus_tab2_size_s t2, (
select max(x.logdate) as logdate from dbplus_tab2_size_s x WHERE x.server_id = $1 AND
x.logdate >= $2 AND x.logdate < $3 group by to_char(x.logdate,$4) ) x where
t2.server_id = $5 and t2.logdate = x.logdate GROUP BY to_char(t2.logdate,'YYYY-MM-DD')
```

The Database is set to dbplus. There are buttons for 'Save parameter values', 'Generate plan', and 'Cancel'.

The PARAMETERS table is as follows:

	Type	Value
\$0	Varchar	YYYY-MM-DD
\$1	Number	1
\$2	Datetime	2019-02-27 09
\$3	Datetime	2019-02-27 09
\$4	Varchar	-DD'
\$5	Number	1

The EXPLAIN PLAN section shows the same error as in the previous screenshot:

```
Database: dbplus
42803: column "t2.logdate" must appear in the GROUP BY clause or be used in an aggregate fu
```

After complete the parameter list, save the changes by click the **Save parameter values** button. Then the correct plan for the given query will be generated.

Instance Load Waits SQL Analyze **SQL Details** Load Trends Compare Top SQL SQL 3D Top Day Slow SQLs

414511336 Online values Refresh Find SQL

STATEMENT TEXT

```
select to_char(t2.logdate,$4) as logdate ,sum(num2) num2,sum(num3) num3 from
dbplus_tab2_size_s t2, (
select max(x.logdate) as logdate from dbplus_tab2_size_a x
WHERE x.server_id = $1
AND x.logdate >= $2
```

SQL STATISTICS Show values per 1 executions

Plan id	Database	User	Elapsed Time	Blks read time	Blks write time	Executions	Rows	Blks hit	Blks read	Blks dirtied	Blks written	Buffer Quality	Temp blks read	Temp blks write	Elapsed Time per 1 Exec
			[Seconds]	[Seconds]	[Seconds]		[Rows]	[Blocks]	[Blocks]	[Blocks]	[Blocks]	[%]	[Blocks]	[Blocks]	[Seconds]
520541563	dbplus	dbplus	1.5	178.1	0	24	233	461	16	11	0	96.7	0	0	0.0617

Explain plan Graph 520541563

Show plan objects for 520541563

```
Database: dbplus
GroupAggregate (cost=16.65..16.65 rows=1 width=96)
  Group Key: (to_char(t2.logdate, 'YYYY-MM-DD':text))
  Sort (cost=16.65..16.65 rows=1 width=76)
    Sort Key: (to_char(t2.logdate, 'YYYY-MM-DD':text))
    Nested Loop (cost=8.57..16.64 rows=1 width=76)
      Join Filter: (t2.logdate = (max(x.logdate)))
      Index Scan using idx_dbplus_tab2_size_s_2 on dbplus_tab2_size_s t2 (cost=0.27..5.29 rows=1 width=52)
        Index Cond: (server_id = 1)
```

6.2.1.4.2 Show Plan Objects functionality

The functionality of **Show Plan Objects** is available on screens where the query text and the execution plan are visible. After click the link with the same name, the window is opened:

SQL TEXT

```
select * from dbplus_tab_cpu_usage c where c.server_id = $1 and c.dat1 > $2
```

EXPLAIN PLAN

```
Database: dbplus
Index Scan using dbplus_tab_cpu_usage_1 on dbplus_tab_cpu_usage c (cost=0.28..1.29 rows=1 width=492)
  Index Cond: ((server_id = 1) AND (dat1 > '2019-02-28 15:04:00':timestamp without time zone))
```

OBJECTS USED IN EXPLAIN PLAN

Type	Schema	Object Name	Name
INDEX	public	dbplus_tab_cpu_usage_1	dbplus_tab_cpu_usage_2
TABLE	public	dbplus_tab_cpu_usage	dbplus_tab_cpu_usage_1

INDEXES FOR SELECTED OBJECT PUBLIC.DBPLUS_TAB_CPU_USAGE_1

Index Name	Index Type	Index Columns
dbplus_tab_cpu_usage_1	INDEX	server_id, dat1

Object columns Info Properties Details for INDEX public.dbplus_tab_cpu_usage_1 Load object properties (slower)

Column	Position	Unique values	Most common values
server_id	1		
dat1	2		

By click the **Show plan objects for ...** link, user is moved to the screens that allow analyze the objects that participate in the query:

- What tables, indexes were when the query was executed?
- How the engine referred to the given objects?
- Was the query performed in multithreaded mode?
- What kind of mechanism was used to download and connect "data" from objects:
 - Nested Loop
 - Hash/Merge Join

The window contains information about the query text and the execution plan as well as information about:

Objects Used in Explain Plan – a list of all objects used by the query in given execution plan

Indexes for selected object– list of indexes for selected table - row selected in the "Objects Used in Explain Plan"

3 tabs in the area (checkbox Load object properties checked):

- a) **Object Columns** – a list of individual columns of the selected object, along with information: column name, data type, id columns, density (the lower density - the higher selectivity of the column)
- b) **Info** – DDL command create the given object,
- c) **Properties** – additional properties of selected object

6.2.1.4.3 Searching for queries in SQL Details

In the **SQL Details** screen, the **[Find SQL]** button is available, allow the user to search for queries:

- Search a fragment of the query text,
- Find queries that change the execution plan,
- Search for queries that are active in the selected period (omitting another period)

In each case, the list of search queries contains information:

- Query Id – query ID, with the [Plus] button, which allows user to add a specific ID to the list of queries
- Last execution date – last day when query was made,
- Elapsed Time – duration of query,
- Executions – number of executions
- IO time - time spent on reading / writing blocks,
- Rows processed – number of returned rows,
- Blks hit – number of read blocks from memory
- Blks read - number of read blocks from disks
- Blks dirtied – number of „dirty” blocks
- Blks written – number of written blocks
- Temp blks read – number of temporary blocks read by queries,
- Temp blks written - number of temporary blocks written by queries,
- Query text – query text.

Each tab allows user to search for queries within a certain time period. When user search for a query after a fragment of the query text, they can enter several expressions in the search field. The result will be returned in two grids:

- FIND RESULT FOR **EXACT** QUERY TEXT MATCHING WITH –result exactly like the typed in part,

- FIND RESULT FOR **SIMILAR** QUERY TEXT MATCHING WITH –similar result contain the phrases entered

Statement by text ✕

Plan Flip-Flop Statements

New statements

select * from pg_stat

Date from: Date to: Max. returned statements:

FIND RESULTS FOR EXACT QUERY TEXT MATCHING WITH SELECT * FROM PG_STAT

Query Id	Last execution date	Elapsed Time [Seconds]	IO time [Seconds]	Executions	Rows processed [Rows]	Blks hit [Blocks]	Blks read [Blocks]	Blks dirtied [Blocks]	Blks written [Blocks]	Temp blks read [Blocks]	Temp blks written [Blocks]	Query te
2828282953	2019/02/25	2.39	0	63	630	63	0	0	0	0	0	select *
2952862856	2019/02/25	0.90	0	63	63	0	0	0	0	0	0	select *

FIND RESULTS FOR SIMILAR QUERY TEXT MATCHING WITH SELECT%*%FROM%PG_STAT

Query Id	Last execution date	Elapsed Time [Seconds]	IO time [Seconds]	Executions	Rows processed [Rows]	Blks hit [Blocks]	Blks read [Blocks]	Blks dirtied [Blocks]	Blks written [Blocks]	Temp blks read [Blocks]	Temp blks written [Blocks]	Query te
1149331173	2019/02/25	1.92	0	80	78	207	0	0	0	0	0	select q,
2317907690	2019/02/25	1.17	0	3 767	10 181	0	0	0	0	0	0	select ca

With the **Plan Flip-Flop Statements** tab selected, a search for queries that have changed the plan of execution in a given period.

For queries that change the explain plan, additional information is grouped:

- Statistics with a summary for all performance plans on which the query worked,
- Slowest plan statistics summary
- Fastest plan statistics summary
- Comparison between Slowest and Fastest
- Possible time reductions for queries statistic.

Below is an example of the search results for those questions that will change the execution plan within one week:

View of the area's *Total statistics, Slowest plan statistics*:

Statement by text

Date from: 2019/02/18 00:00 Date to: 2019/02/25 23:59

Plan Flip-Flop State...

New statements Search

FIND RESULTS

Query Id	Query text	Total statistics			Slowest plan statistics			
		Elapsed Time [Seconds]	Executions	Number of plans	Plan Id	Elapsed Time [Seconds]	Executions	Elapsed Time Per 1 exec [Seconds]
4139114734	update dbplus_tab8 as t8 set num11 = t8.num11 + t4.nu	97.29	856	2	2979494333	11.50	98	0.1
172827488	select coalesce(r_s.var1,\$2) as var1, ss.var7 as var2, l2	72.06	3	2	2176099901	32.74	1	32.7
1720256591	DELETE FROM dbplus_tab_sessions WHERE ctid = ar	44.81	67 090	2	1018538233	43.02	60 153	0.0
2533289700	DELETE FROM dbplus_tab_cpu_usage WHERE ctid =	34.45	67 090	2	2623212990	32.97	58 526	0.0
3169426278	DELETE FROM dbplus_tab_waits WHERE ctid = any (e	26.45	67 090	2	751750294	22.52	48 512	0.0
2122691603	insert into dbplus_tab8 (server_id,dat1, num1,num2,nu	8.84	855	4	4104500066	1.28	90	0.0
1749764993	insert into dbplus_tab8sd_day (server_id, dat1, qs_num	6.14	509	2	277037531	5.97	464	0.0
2868835096	select * from dbplus_tab_cpu_usage c where c.server_i	2.97	1 824	2	2677408220	2.84	1 687	0.0
1504620578	select * FROM dbplus_tab10_ext t10 where t10.server_	1.63	854	2	383050935	0.28	45	0.0
3876793342	insert into dbplus_tab17 (snap_id,loqdate, server_id,var	1.44	1 142	2	1675441822	1.04	806	0.0

View of the areas *Fastest plan statistics*, *Slowest vs. Fastest*, *Estimation statistics*.

Slowest plan statistics			Fastest plan statistics				Slowest vs Fastest		Estimation statistics
Elapsed Time [Seconds]	Executions	Elapsed Time Per 1 exec [Seconds]	Plan Id	Elapsed Time [Seconds]	Executions	Elapsed Time Per 1 exec [Seconds]	Times faster	Elapsed Time Per 1 exec difference [Seconds]	Elapsed Time to reduce [Seconds]
11.50	98	0.1174	2680283919	85.78	758	0.1132	1	0.0042	0.4145
32.74	1	32.7450	1532008776	39.31	2	19.6550	2	13.0900	13.0900
43.02	60 153	0.0007	1371767627	1.79	6 937	0.0003	3	0.0005	27.5330
32.97	58 526	0.0006	2556213323	1.48	8 564	0.0002	3	0.0004	22.8772
22.52	48 512	0.0005	1264340238	3.92	18 578	0.0002	2	0.0003	12.2738
1.28	90	0.0142	1795420420	1.32	158	0.0083	2	0.0059	1.7210
5.97	464	0.0129	681720614	0.17	45	0.0038	3	0.0090	4.1925
2.84	1 687	0.0017	1305347756	0.14	137	0.0010	2	0.0007	1.1766
0.28	45	0.0062	1738785768	1.35	809	0.0017	4	0.0045	0.2020
1.04	806	0.0013	3843479467	0.40	336	0.0012	1	0.0001	0.0815

An important area of the **Flip-Flop Statements** plan screen is the *Statistics Estimation*. The columns **Elapsed Time to reduce**, is a calculation about the possible reduction of time for the case when the query would work to be disabled on the fastest explain plan.

Below is the result of an example search query run on a specific day (not run in earlier days) - (**New statements**). This is a particularly useful functionality to search queries after changes: upload the code of a new version of a database-based application.

Statement by text

Plan Flip-Flop Statements

Statement executed in period
Date from: 2019/02/24 00:00 Date to: 2019/02/25 23:59 Min. elapsed time (sec): 100

And statement not executed in the period range
Date from: 2019/02/01 00:00 Date to: 2019/02/02 23:59

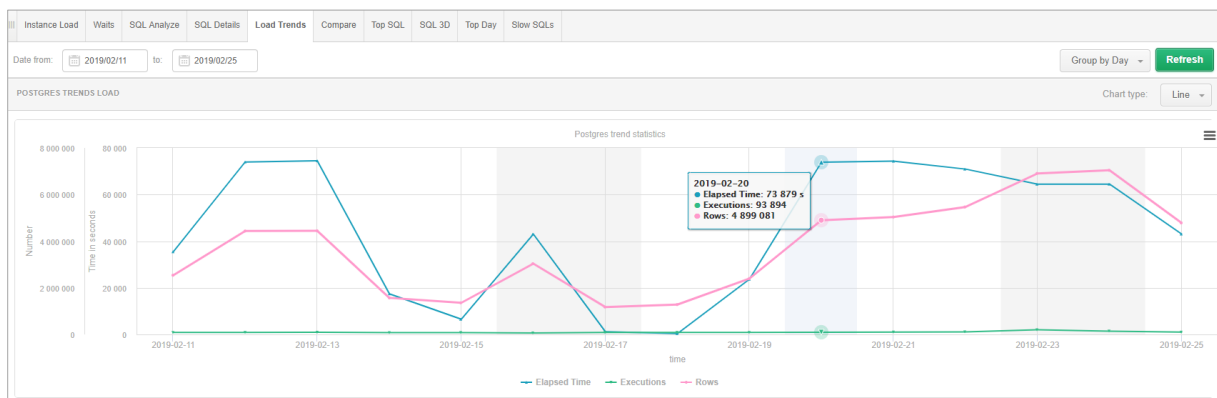
Search

FIND RESULTS

Query Id	Elapsed Time [Seconds]	IO time [Seconds]	Executions	Rows processed [Rows]	Blks hit [Blocks]	Blks read [Blocks]	Blks dirtied [Blocks]	Blks written [Blocks]	Temp blks read [Blocks]	Temp blks written [Blocks]	Query text
2188545525	138.68	0	9 570	9 570	9 570	0	0	0	0	0	select \$1 as datid, \$2 as

6.2.1.5 „Load Trends” Tab

Load Trends tab allows to get detailed information on trends in PostgreSQL Instance.



The page consists of three components:

- Filter with the date range and group option
- Graph presents certain indicators over time
- The table of statistics

Information displayed on the graph can be shown in groups of:

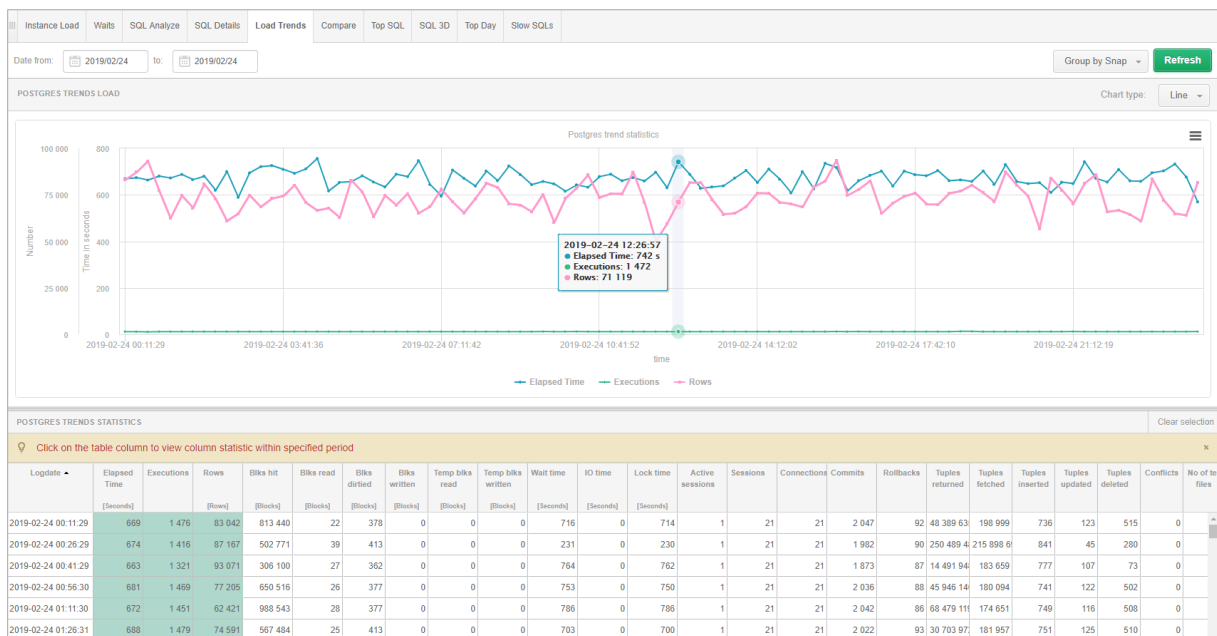
- **No group by period** –selection of date range
- **Month** – statistics broken for months
- **Day** – statistics broken by day
- **Hour** – statistics broken by one hour
- **Snap** – statistics broken by 15 minutes

Load Trends Statistics include the information:

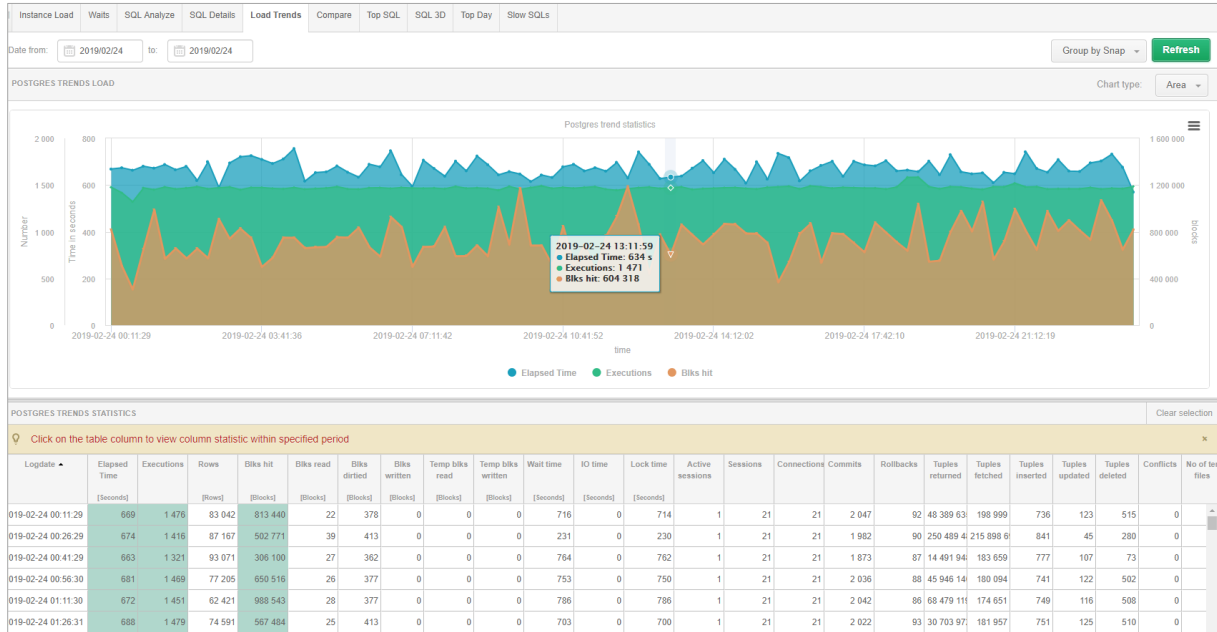
- Logdate – represents the point in time for which the statistics are presented (i.a.: day, hour, minutes, entire period)
- Elapsed time – total length of time in seconds of all queries for the selected group period
- Executions – number of performances of all searches for the selected group period
- Rows– number of rows processed by all queries for the selected group period
- Blks hit – number of read blocks by queries from the memory,
- Blks read – number of read blocks by queries from discs
- Blks dirtied – number of „dirty” blocks
- Blks written – number of blocks written by queries,
- Temp blks read – number of temporary blocks read by queries,
- Temp blks written – number of temporary blocks written by queries
- Wait time – wait time in a selected time unit,
- IO time – time of readings / writings in a selected time,

- Active sessions – average of active sessions in a selected period,
- Sessions – number of sessions for a given period,
- Connections – number of connected users in a given period,
- Commits – number of approved transactions,
- Rollbacks – number of withdrawn transactions,
- Tuples returned – number of rows returned by queries,
- Tuples fetched – number of rows downloaded by queries,
- Tuples inserted – number of rows inserted by queries
- Tuples deleted – number of rows deleted by queries,
- Conflicts – number of queries canceled due to conflicts with recovery.
- No of temp files – number of temporary files created by queries.,
- Temp files written amount – total amount of data written to temporary files by queries,
- Deadlocks – number of detected deadlocks,
- Blk read time – time spent reading data file blocks,
- Blk write time - time spent writing data file blocks.

Click selected columns present their behavior as function of time:



Change the graph type to 'Area' results in Graph changes to the example below:

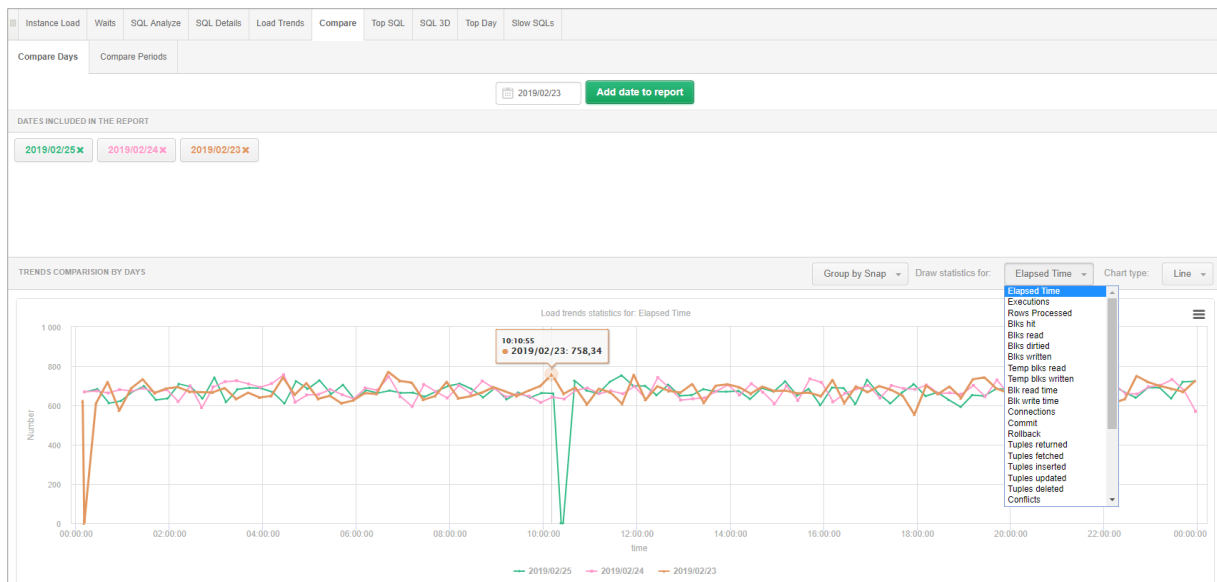


In the filter area there is a control that allows display statistics for a specific database - statistics are displayed for all databases by default.

6.2.1.6 „Compare” Tab

The **Compare** tab allows user to compare trends for performance statistics and compare days (Compare Days tab) or periods (Compare Periods) for specific performance parameters.

Example list of individual days:

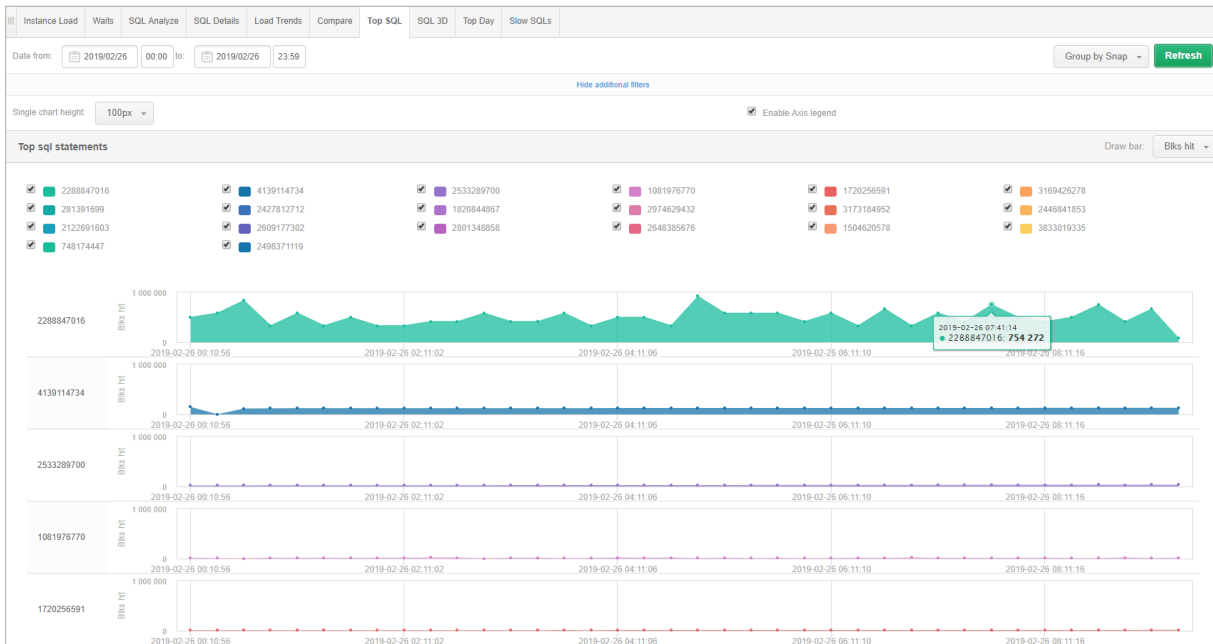


The above chart presents the Elapsed Time statistics grouped after a snap for selected 3 days and compiled for comparison. User can choose any performance statistics described on the **Load Trends** tab.

6.2.1.7 „Top SQL” Tab

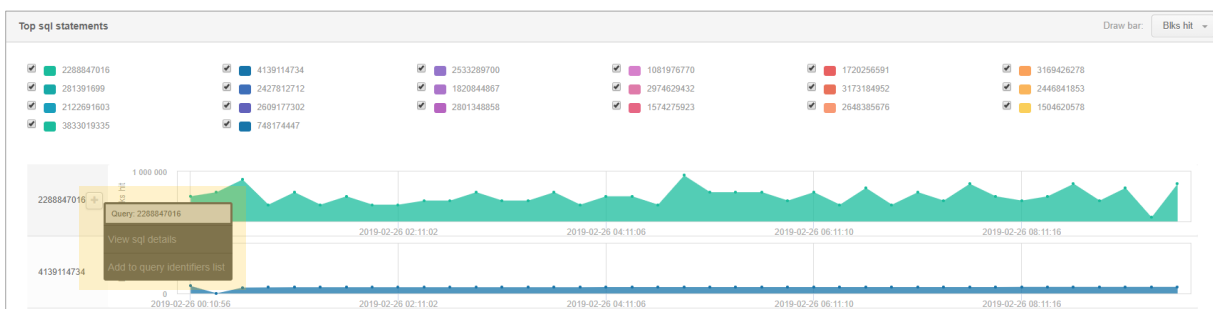
The **Top SQL** tab allows user to report the most-demanding queries depend on a specific performance indicator. The system can examine the most difficult queries in terms of Elapsed Time, the number of read data, the number of blocks processed from memory and reading of Temp files.

The queries are presented in the form of several graphs in descending order according to the Blk hit statistics (number of blocks read from the memory) of the selected period of time (or other selected indicator).

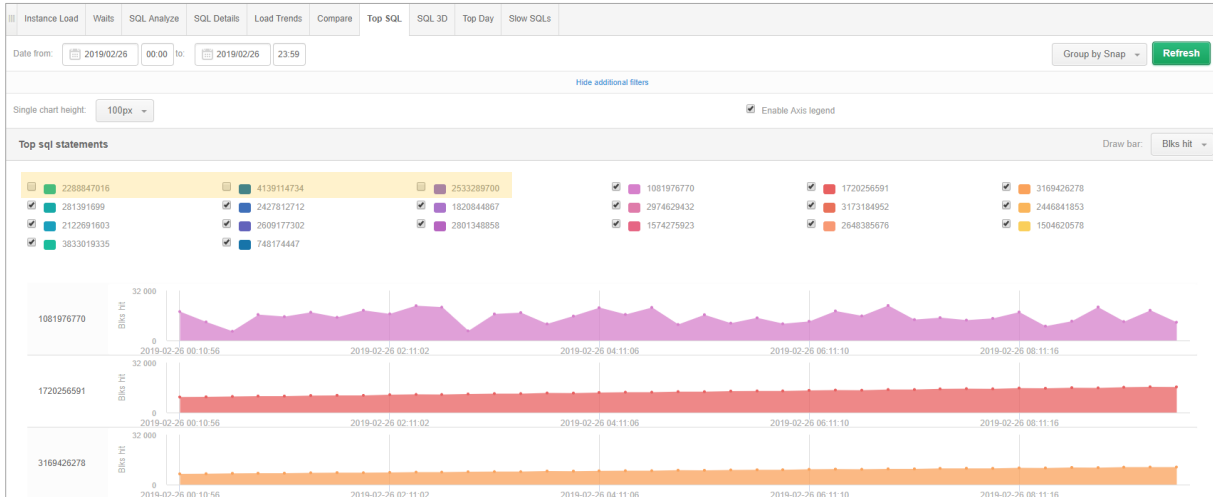


From the **Top SQL** screen, any query can be easily added to the **SQL Details** by click the **[Plus]** button next to the query identifier and click the options:

- SQL View details - to move to the SQL Details screen and analyze specific query
- Add to query hash list - to add the query to the clipboard with a list of questions for further analysis



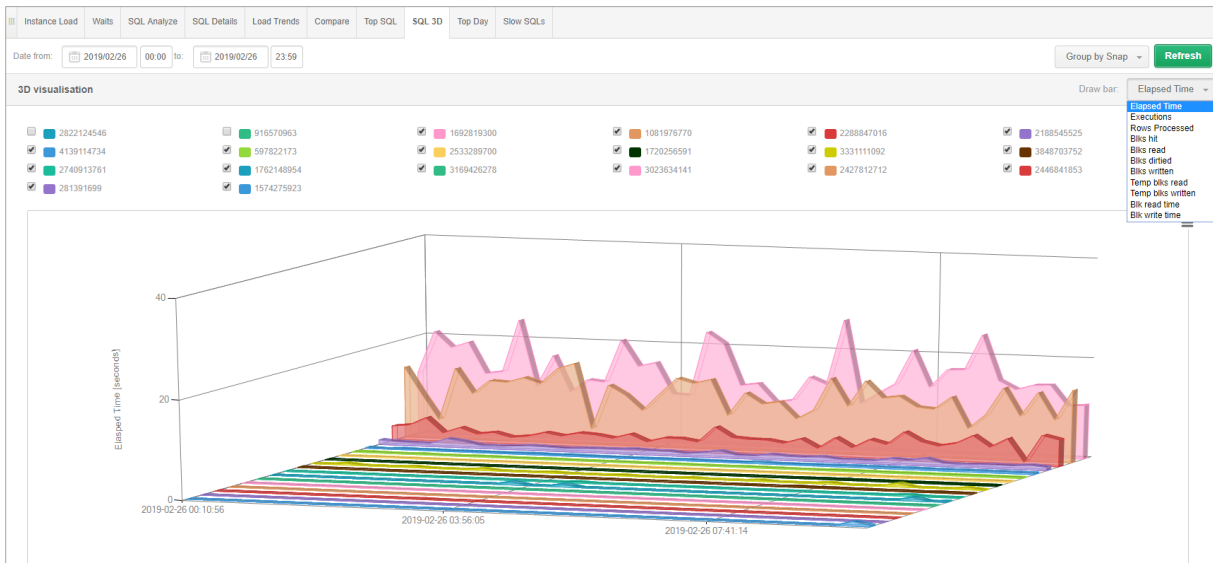
Using checkboxes in the legend, user can delete individual charts from the **Top SQL** view:



6.2.1.8 „SQL 3D” Tab

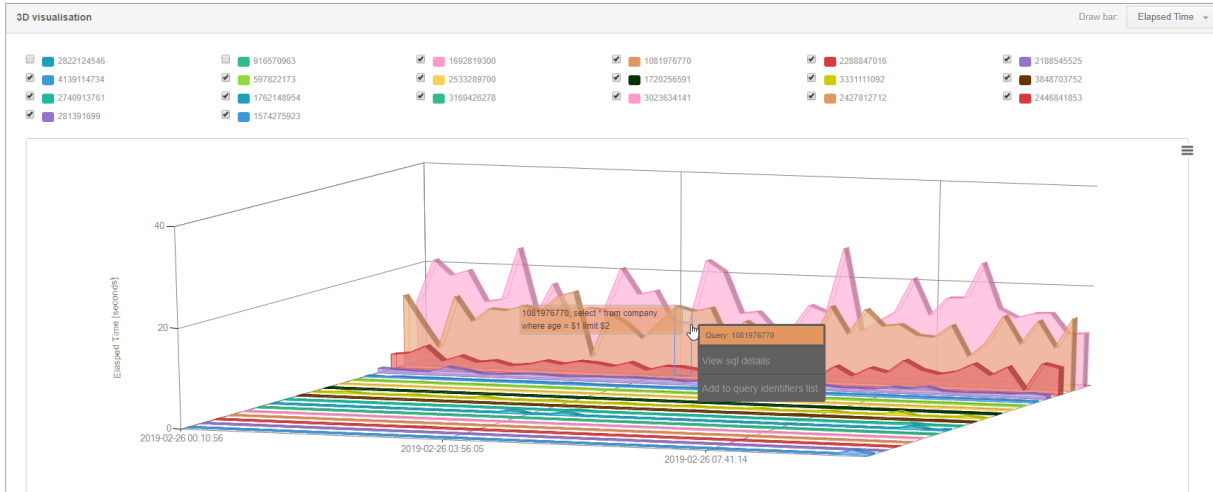
The **SQL 3D** tab contains the same functionality as in the **Top SQL** tab. The module allows user to verify the most-demanding queries depend on whether user is interested in - the execution time, the number of read data, the number of blocks processed from memory and other performance statistics.

Inquiries are presented in the form of one graph in a 3D view in decreasing order according to the duration of the query in the selected period (or another selected indicator).



The settings icon is available in the upper right corner. It allows user to control the graph. User can easily add an interesting query to the SQL Details module by click on the chart series and select options:

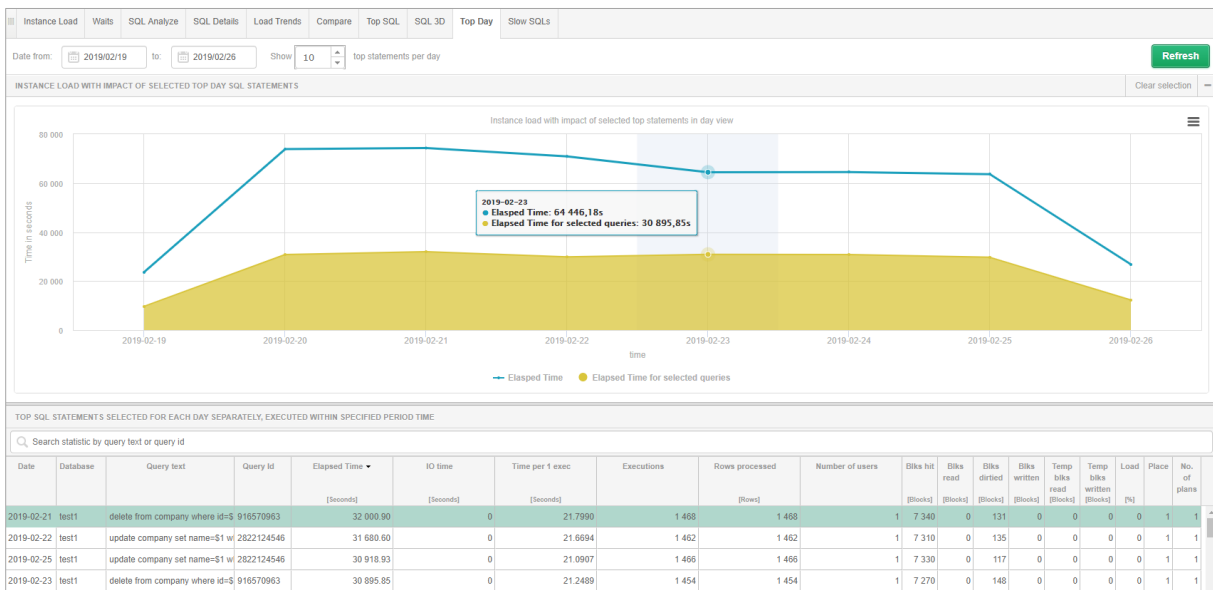
- View SQL Details – to switch to the SQL Details screen and analyze a specific query
- Add to query hash list – to add a query to the clipboard with a list of queries for further analysis



Using check boxes in the legend, user can delete individual charts from the SQL 3D view, similarly as in the **TopSQL** tab.

6.2.1.9 „Top Day” Tab

Top Day window allows to view top queries for Elapsed Time and track their behavior changes.



On the above slide, presented top queries in the last one week and the share of the first query impact against the load of entire instance.

Conclusion: optimization selected queries instance load can be reduced by 50%!!!

Table with top queries contains:

- Date – the date the query was made,
- Database – database where the query is performed,
- Query text
- Query Id – PostgreSQL query ID,
- Elapsed Time – the total execution time of the PostgreSQL query
- IO Time – total IO read time,

- Time per 1 exec – the time of a single query execution,
- Executions – number of executions at a time
- Rows processed – rows returned by the query at a particular time,
- Number of users – the number of unique users that perform the given query,
- Other....

Below the table is text of the selected query. By check the query in the table, user can add statistic to chart **Instance Load** and observe changes of its influence on the overall load of the instance.

It is important to remember about the possibility of a detailed analysis of a specific query by click on the [Plus] button next to the query identifier.

6.2.1.10 „Slow SQLs” Tab

The **Slow SQLs** window allows to display grouped statistics of queries at a time. An additional filter is **Min elapsed execution time**, which allows to filter out queries (below a certain value).

Below is an example that presents monthly statistics when the Elapsed Time is over 200 seconds.

Database	Query text	Query id	Plan id	Elapsed Time	IO time	Time per 1 exec	Executions	Rows processed	Number of users	Bkts hit	Bkts read	Bkts dirtied	Bkts written	Temp bkts read	Temp bkts written	Load
				[Seconds]	[Seconds]	[Seconds]		[Rows]		[Blocks]	[Blocks]	[Blocks]	[Blocks]	[Blocks]	[Blocks]	[%]
test1	update company set name='S1' w	2822124546	2237371182	30 918.93	0	21.0907	1 466	1 466		7 330	0	117	0	0	0	
test1	delete from company where id=5	918570963	3002378863	29 713.04	0	21.6252	1 374	1 374		6 870	0	123	0	0	0	
test1	select * from company limit \$1	1692819300	4780478	1 402.11	0	0.7179	1 953	1 954 749		25 126	0	0	0	0	0	
test1	select * from company where age	1081976770	2992165428	972.49	0	0.5165	1 883	1 355 985		1 400 52	0	0	0	0	0	
test1	select count(*) from company	2288847016	476246813	395.62	0	0.6143	644	644		53 972 2	0	45	0	0	0	

STATEMENT TEXT FOR QUERY ID: 2822124546

```
update company set name='S1' where id=2
```

EXPLAIN PLAN FOR PLAN ID: 2237371182

```

Show plan objects for 2237371182
├─Database: test1
├─Update on company (cost=0.42..0.45 rows=1 width=50)
│   └─Index Scan using company_pkey on company (cost=0.42..1.45 rows=1 width=50)
│       └─Index Cond: (id = 1)

```

Below the table there is the content of the query and the execution plan for the selected statistics.

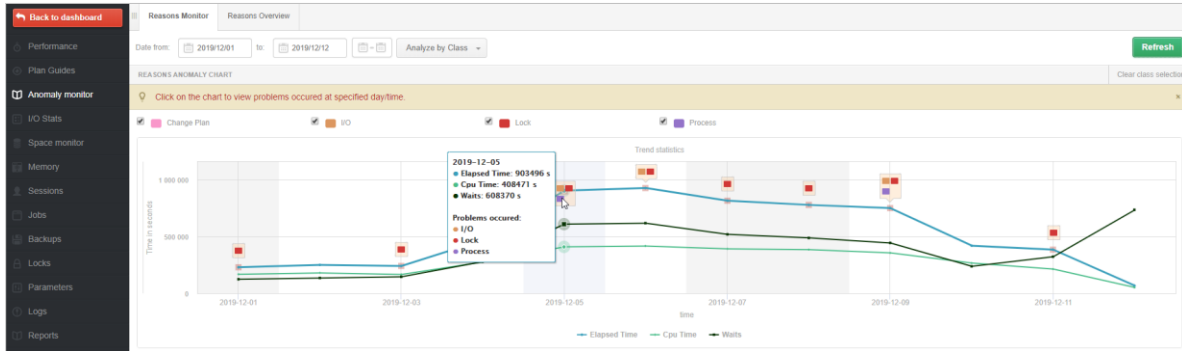
Notice: remember about the possibility of a detailed analysis of a specific query by click the [Plus] button on the query.

6.2.2 Anomaly Monitor Menu

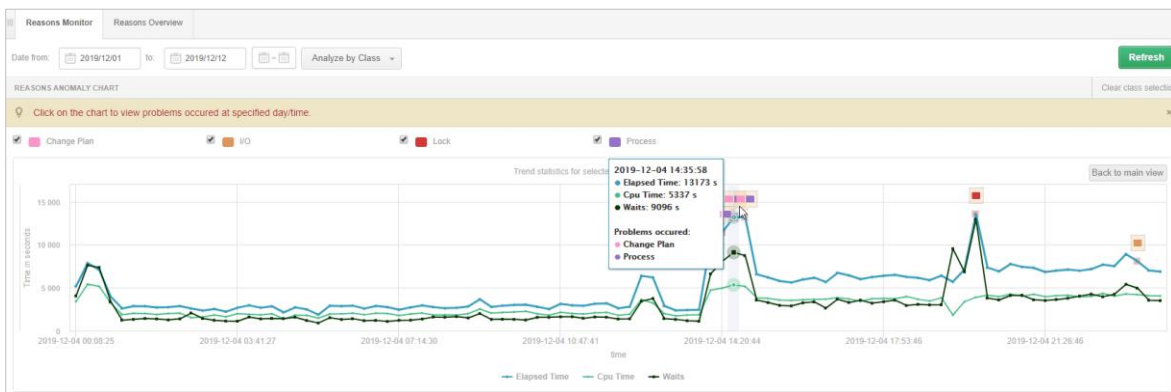
On the site, the user can view information about alerts that have occurred for a given instance. The browser is available from the level of the Instance Analysis> Anomaly Monitor instance.

6.2.2.1.1 Reasons Analysis

The Anomaly Monitor module has been improved in the new version of the application. The method of alert detection and presentation has been modified. After enter the screen, a graph from the last 14 days is presented where a performance problem occurred. The date range can be freely modified. By default, the screen presents problems grouped by class (Analyze by Class), it is also possible to change the presentation and group them by reason of problems (Analyze Reason).



Problems on the chart are marked by colored icons (a different color for each class / reason). For further analysis, select the indicated day on which the problems occurred. After select a specific day (point on the graph) a detailed graph for a given day will be presented with an indication of the point at which performance problems occurred. Each point on the graph represents a given snap (15 minutes). By select a point on the chart, the user will receive information on statistics that have been exceeded at the moment as well as information on the cause of the problem.



In the new version, the Anomaly monitor module has been extended with problem detection, which additionally analyzes database performance at a given time and presents the result of this analysis in the form of a problem. This module is embedded in the application code and is not user configurable. The current alert mechanism works all the time independently of the detection mechanism.

PROBLEMS REPORTED IN SPECIFIED TIME: 2019-12-04 19:55:28	
Increase of query processing time caused by locks	
Class	Lock
Reason details & action	Following process was the main blocker session that generated locking. Logdate: 2019-12-04 19:55:32, Sessionid: 398, Username: INTER!svrmysql, Status: running, OS User: svrmysql, Program: SQLAgent - TSQL JobStep (Job 0xB11C4134C2AF6E40ABC66BFDA1C259 : Step 1), Transaction log record count: 1308317, Last Request Runtime: 1704 s, Transaction begin: 2019-12-04 19:33:33, Transaction log size: 8554.4 MB
Additional information	Please go to Locks=>Locks history module and analyze blocking cases at specified time.
Lock Time	Alert Type: Load Trends, The measured statistic value is 134 % higher than average , Last value: 8872 s, Reference history value: 3784 s
Elapsed Time	Alert Type: Load Trends, The measured statistic value is 90 % higher than average , Last value: 13564 s, Reference history value: 7140 s

As part of defining causes of the problem in the Alerts settings menu in the "Reasons & Problems definitions" tab for a given cause of the problem, user can specify and add a detailed description of the problem with an indication of the place for detailed analysis.

REASON DEFINITION

Main description: Data writes time problem caused by slow I/O response

Reason Class: I/O

Details description: Slow data writes problem is detected. For detailed verification, go to the I/O Analyze tab in the I/O Stats menu.

Hints for further analysis: [empty]

Calculation Type: Based on Trends

Enabled:

Rules & Formulas

Notifications & Conditions

AND OR Add rule Add group Delete

Trends:Elapsed Time Delete

AND OR Add rule Add group Delete

IO: Single Block Write time Delete

IO Write time Delete

NOT IO:Disk writes Delete

Rules preview: Trends:Elapsed Time AND (IO: Single Block Write time OR IO: Write time) AND NOT IO:Disk writes AND (Trends:Wait Event Time - [buffer busy waits] OR Trends:Wait Event Time - [free buffer%])

6.2.2.1.2 Reasons Overview

As part of this tab, the application allows you to view problems in one set. We can choose the same filters as for the Reasons Analysis tab and additionally the option of marking / deselecting grouping after the Cause.

Reasons Analysis | Reasons Overview

Date from: 2018/12/03 to: 2018/12/17 Show reason type: Trends Online Using Query Hash: Enter query hash Group by reason: Refresh

Hide additional filters

Reasons list: Search by name ... Performance problem for specified SQL Statements, Increase of waits events (cause of Locks) on database, Problem - wait: PAGEIOLATCH_SH, Performance problem for specified SQL Statements, Performance problem for specified SQL Statements, Performance problem for specified SQL Statements

Reasons selected to filter: [empty]

Alerts list: Search by name ... IO Disk reads, IO Disk writes, IO MB reads, IO MB writes, IO Read time, IO Single MB Read time

Alerts selected to filter: [empty]

REASONS & ALERTS OVERVIEW

Logdate	Reason name
2018-12-14 14:26:23	I/O:Data reads time problem caused by slow I/O response
	Read time Alert Type: I/O Stat, The measured statistic value is 2.6 times higher than allowed maximum , Last value: 32871 s, Reference history value: 9204 s
	Single MB Read time Alert Type: I/O Stat, The measured statistic value is 64 % higher than allowed maximum , Last value: 0.0425 s, Reference history value: 0.0258 s
2018-12-14 14:26:23	I/O:Increase of query processing time caused by slow I/O response
	Single MB Write time Alert Type: I/O Stat, The measured statistic value is 3.5 times higher than allowed maximum , Last value: 0.1000 s, Reference history value: 0.0224 s
	Single MB Read time Alert Type: I/O Stat, The measured statistic value is 64 % higher than allowed maximum , Last value: 0.0425 s, Reference history value: 0.0258 s
	Cpu Time Alert Type: Load Trends, The measured statistic value is 11 times higher than average , Last value: 437.5 s, Reference history value: 36.3 s

Depending on the checkbox **[Group by reason]**, alert data will be displayed in various lists:

- selected

REASONS & ALERTS OVERVIEW

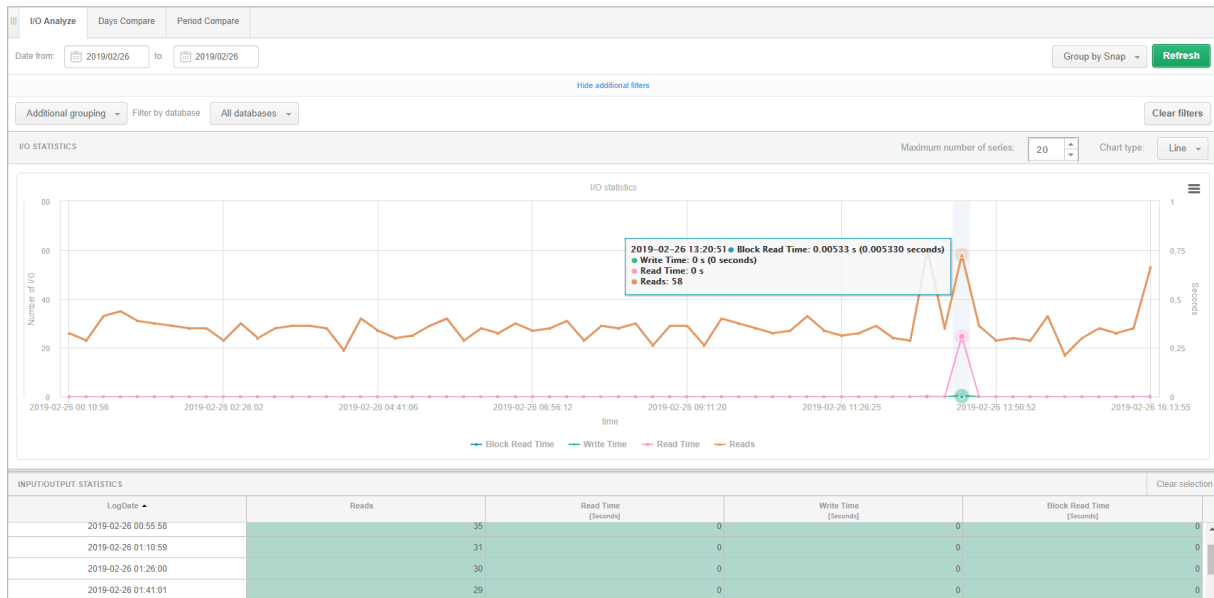
Logdate	Reason name
2018-12-02 06:32:14	I/O:Data writes time problem caused by slow I/O response
	Single Block Write time Alert Type: I/O Stat, The measured statistic value is 10.5 times higher than allowed maximum , Last value: 1.87 s, Reference history value: 0.1623 s
	Write time Alert Type: I/O Stat, The measured statistic value is 2.6 times higher than allowed maximum , Last value: 10137 s, Reference history value: 2849 s
	Wait Event Time Alert Type: Load Trends, The measured statistic value is 119 % higher than average , Wait: log file sync, Last value: 60.6 s, Reference history value: 27.6 s
	Elapsed Time Alert Type: Load Trends, The measured statistic value is 66 % higher than average , Last value: 1769 s, Reference history value: 1067 s

- unselected

REASONS & ALERTS OVERVIEW					
Logdate	Reason	Level	Alert name	Hash value	Message
2018-12-02 06:32:14	I/O/Data writes time problem caused by slow I/O response	Critical	Single Block Write time		Alert Type: I/O Stat, The measured statistic value is 10.5 times higher than allowed maximum , Last value: 1.87 s, Reference history value: 0.1623 s
2018-12-02 06:32:14	I/O/Data writes time problem caused by slow I/O response	Critical	Write time		Alert Type: I/O Stat, The measured statistic value is 2.6 times higher than allowed maximum , Last value: 10137 s, Reference history value: 2849 s
2018-12-02 06:32:14	I/O/Data writes time problem caused by slow I/O response	Critical	Wait Event Time		Alert Type: Load Trends, The measured statistic value is 119 % higher than average , Wait: log file sync, Last value: 60.6 s, Reference history value: 27.6 s
2018-12-02 06:32:14	I/O/Data writes time problem caused by slow I/O response	Warning	Elapsed Time		Alert Type: Load Trends, The measured statistic value is 66 % higher than average , Last value: 1769 s, Reference history value: 1067 s
2018-12-02 06:32:14	I/O/Increase of query processing time caused by slow I/O response	Critical	Single Block Write time		Alert Type: I/O Stat, The measured statistic value is 10.5 times higher than allowed maximum , Last value: 1.87 s, Reference history value: 0.1623 s

6.2.3 „I/O Stats” Menu

The screen is accessed from the menu and is used to analyze the performance of disk components. “I/O Analyze” functionality allows to see any performance problems on disk devices, i.a. compare the performance of records and readings for individual days, hours, 15 minutes. The data is available in a collective way for the entire instance as well as broken down into individual databases.



The area is divided into:

- Filter area with date range and additional filters
- A graph presents specific indicators
 - Table showing statistics:
 - Reads – the number of reads,
 - Read time – time to read blocks,
 - Write time – time to write blocks,
 - Block Read Time – time to read data blocks

Group by period - shows statistics for a given grouped according to the choice:

- **Day** –
- **Hour** –
- **Snap** – periods of 15 minutes,
- **No group by period**, the sum for the selected period for the database, data files or table space, depending on the filter used, will be shown.

Days Compare and **Period Compare** tabs are also available. On the tabs, user can compare data for days, or compare individual monitored periods.

6.2.4 „Space Monitor” Menu

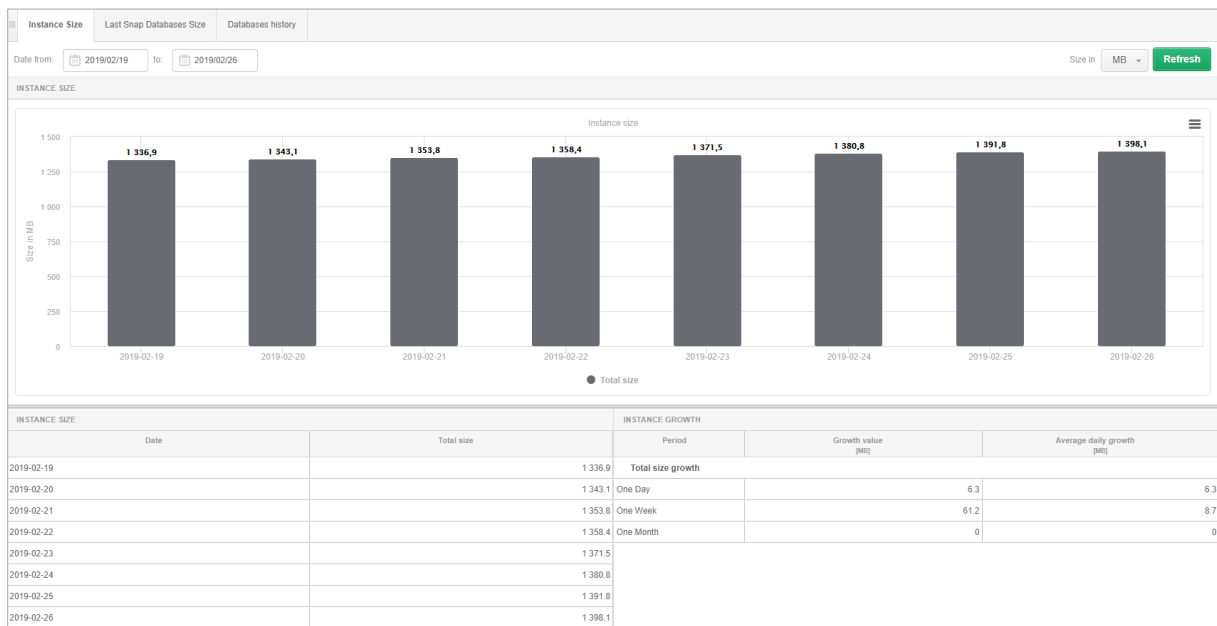
The **Space Monitor** module allows storage analysis. Three options are provided:

- Display the current size data in the instance
- Detailed information on database (broken down by database in instance),
- The history Instance size change in table and graphical form

IMPORTANT: Space Monitor module is also accessible from the main page. This allows to analyze the space used by all/ other Instances.

6.2.4.1 „Instance Size” Tab

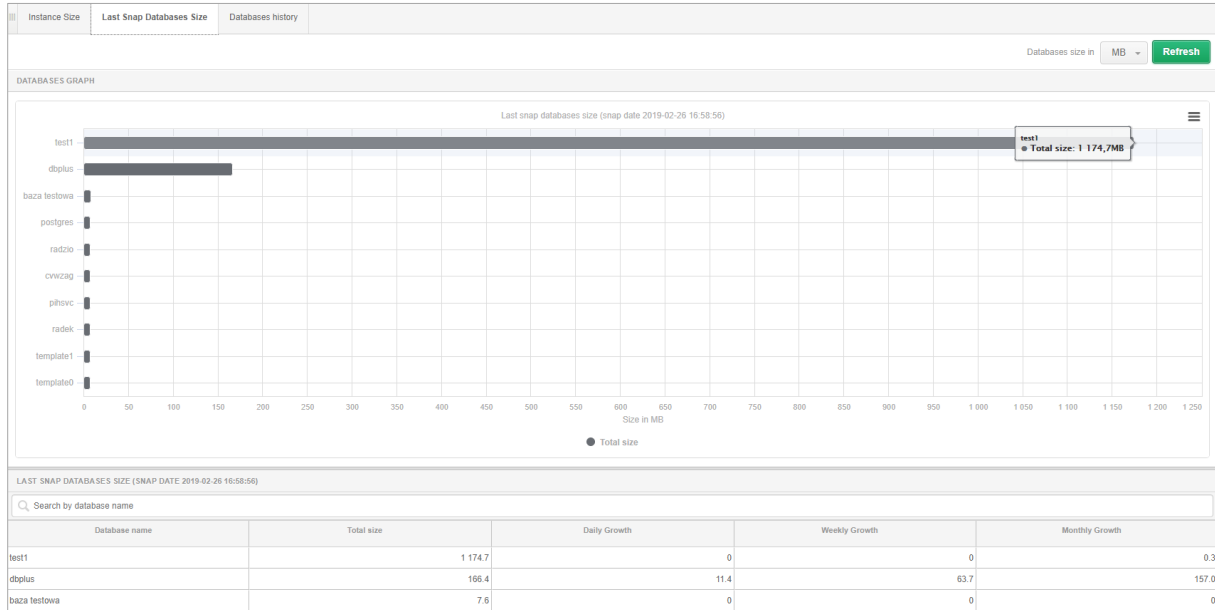
The **Instance Size** tab shows the current instance size (default in GB).



The page shows information about the data occupancy in the whole instance, broken down by day. There is also information on the data increment in the day, week and month perspective.

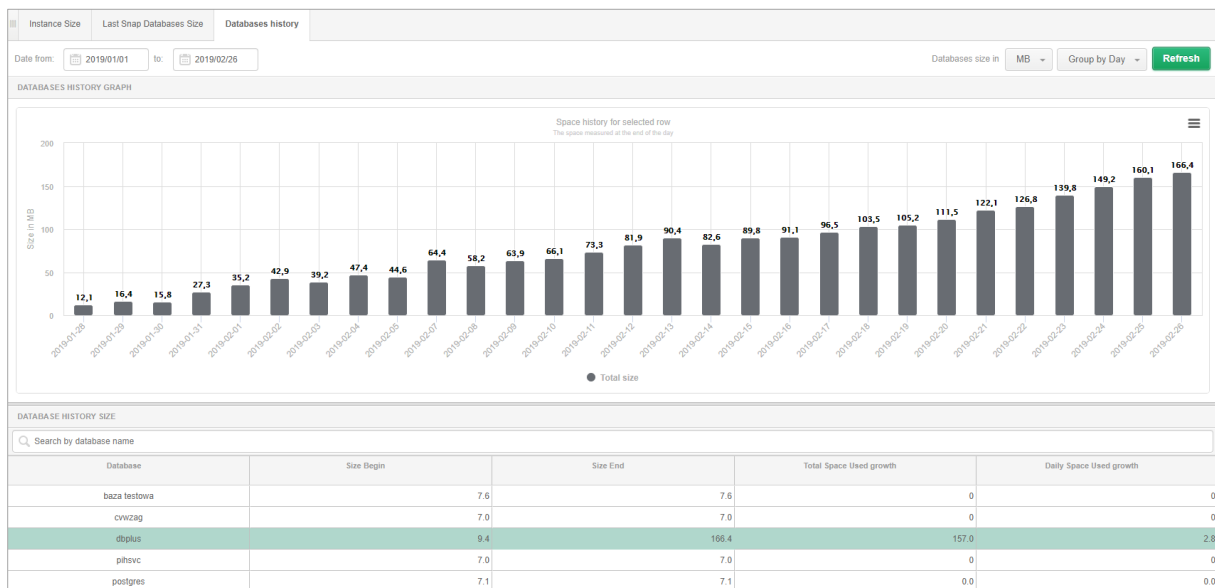
6.2.4.2 „Last Snap Databases Size” Tab

In the **Last Snap Databases Size** window, the user can check the current size of each database available in the PostgreSQL Instance.



6.2.4.3 „Databases history” Tab

On the **Databases history** screen, historical data on the size of databases in the monitored instance on days are made available.

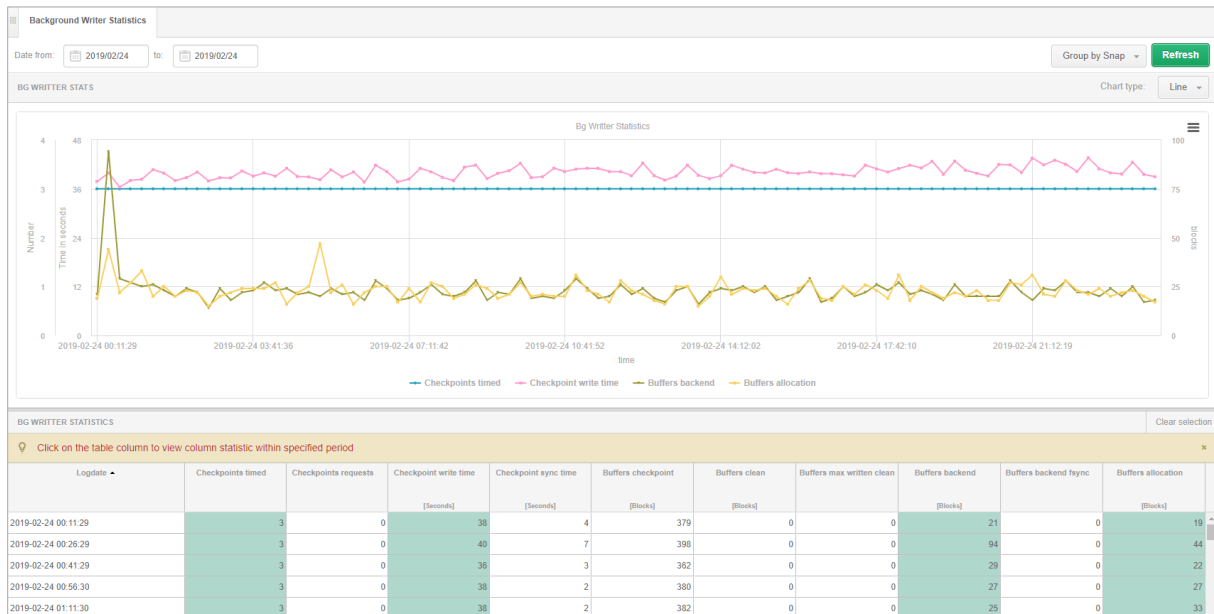


The chart area is refreshed after click the indicated line from the table below (for this case it is a dbplus database on a given PostgreSQL Instance).

Occupancy statistics can also be displayed on an hourly basis.

6.2.5 „Bg Writer Stats” Tab

The tab contains information available in the `pg_stat_bgwriter` view. This view has information about the *background writer* and *checkpointer* processes run in PostgreSQL.



Statistics in the table:

- Checkpoints_timed – Number of scheduled checkpoints that have been performed,
- Checkpoints_requests – Number of requested checkpoints,
- Checkpoint write time – Total amount of time that has been spent in the portion of checkpoint processing where files are written to disk,
- Checkpoint sync time – Total amount of time that has been spent in the portion of checkpoint process where files are synchronized to disk,
- Buffers checkpoint – Number of buffers written during checkpoints
- Buffers clean – Number of buffers written by the background writer
- Buffers max written clean – Number of times the background writer stopped a cleaning scan because it had written too many buffers
- Buffers backend - Number of buffers written directly by a backend,
- Buffers backend fsync – Number of times a backend had to execute its own fsync call (normally the background writer handles those even when the backend does its own write)
- Buffers allocation – Number of buffers allocated

Attention! The data in the rows contain incremental values for a given time unit.

6.2.6 „Sessions” Menu

Sessions functionality presents information about sessions in the PostgreSQL Instances.

Available tabs in the Session menu:

- **Sessions** – online sessions in a given database displayed by filters,
- **Session with transactions** – sessions in the transaction,
- **Session history** – information about number of sessions in the PostgreSQL Instance
- **Active Session history** – field allows user to search:
 - What queries the program / user runs
 - Which users the specified query hash is run

6.2.6.1 „Sessions” Tab

The **Session** tab shows session information on the PostgreSQL Instance.

The screenshot displays the 'Sessions' tab in the DBPLUS interface. At the top, there are navigation tabs: 'Sessions', 'Session with transactions', 'Sessions history', and 'Active Session history'. Below these, there are filters for 'Active sessions', 'Users only', and 'Min elapsed time' (set to 0). A 'Refresh' button is present. The main table shows one active session with the following details:

Logon time	Pid	Transaction start	Query start	Query Id	Database	Username	Status	Elapsed Time [Seconds]	Host	Applicat	Wait type	Wait	Statement
2019-02-27 11:10:632	10632	2019-02-27 14:57:12	2019-02-27 14:57:12		dbplus	dbplus	● active	0		DBPLUS			select * from pg_stat_activity psa where 1=1 and psa.state = \$

Below the table, there are three tabs: 'SQL', 'STATEMENT TEXT', and 'EXPLAIN PLAN'. The 'EXPLAIN PLAN' tab is selected, showing a tree diagram of the query execution plan. The plan includes a 'Nested Loop Left Join' with a 'Hash Join' and a 'Seq Scan on pg_authid u'.

The tabular part presents information:

- **Logon Time** – time of user log into database
- **Pid** – user session ID,
- **Transaction start** – transaction start date,
- **Query start** – query start date,
- **QueryID**
- **Database**
- **Username** – username in the SQL Instance,
- **Status** – status of the session,
- **Elapsed Time** – duration of the query,
- **Host** – the name of the machine from which the log to the database took place (the parameter in the configuration file must be enabled `log_hostname = on` on the PostgreSQL Instance),
- **Application** – the name of the application from which the log into the database took place,
- **Wait type** – wait type,
- **Wait** – wait name for the session,
- **Statement** – query text.

Below the table there are tabs presents detailed information for the selected session: SQL command text and explain plan.

The **SQL** tab shows the SQL query text and the execution plan. The information is displayed after click on the record for the session:

SQL

STATEMENT TEXT

```
select * from pg_stat_activity psa where l=1 and psa.state = $1 and (psa.datid is not null and psa.username is not null)
```

EXPLAIN PLAN

Show plan objects

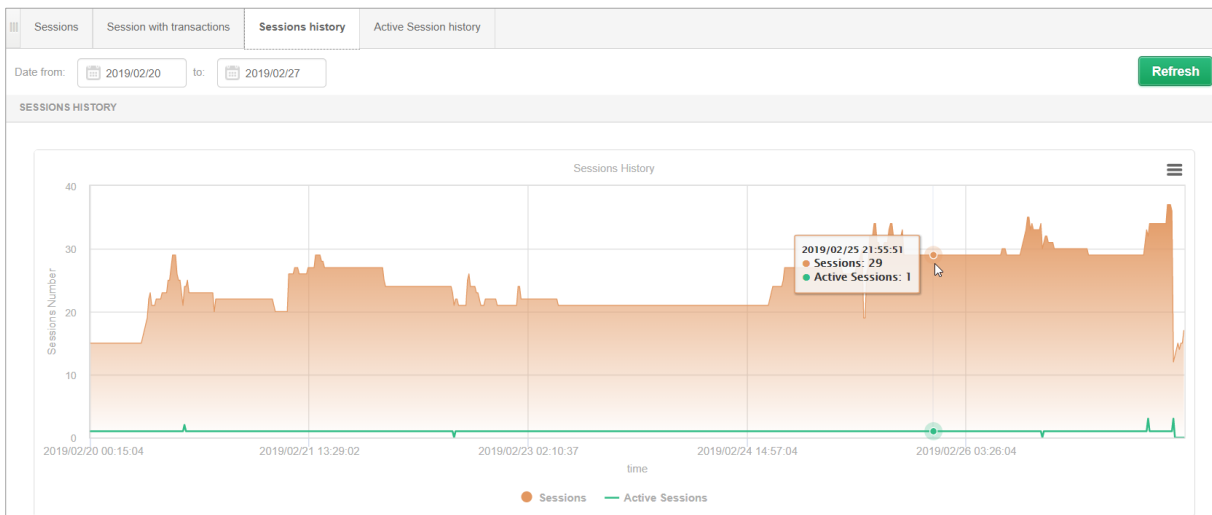
```
Database: dbplus
Nested Loop Left Join (cost=1.26..3.42 rows=1 width=440)
  Join Filter: (s.datid = d.oid)
    Hash Join (cost=1.26..2.37 rows=1 width=376)
      Hash Cond: (u.oid = s.usesyid)
        Seq Scan on pg_authid u (cost=0.00..1.07 rows=7 width=68)
          Filter: (rolname IS NOT NULL)
        Hash (cost=1.25..1.25 rows=1 width=312)
          Function Scan on pg_stat_get_activity s (cost=0.00..1.25 rows=1 width=312)
            Filter: ((datid IS NOT NULL) AND (state = '1':text))
      Seq Scan on pg_database d (cost=0.00..1.02 rows=2 width=68)
```

6.2.6.2 The [Kill Session] button allows user to kill the selected session. „Session with transactions” Tab

The screen displays sessions using the log entry. The screen contains the same information about the user's session as it is visible on the screen in the **Session** tab.

6.2.6.3 „Session history” Tab

The tab in the form of a graph presents the number of sessions and active sessions in a given time period.



6.2.6.4 „Active Session history” Tab

This tab shows detailed historical information about open sessions at a specified time. The data presents three basic data sets:

- Active sessions,
- Sessions that use the log file.

Sessions													
Session with transactions			Sessions history			Active Session history							
From:		<input type="text" value="2019/02/27 00:00"/>	to:		<input type="text" value="2019/02/27 23:59"/>	Using Query Id:		<input type="text" value="Enter query id"/>	Username:		<input type="text" value="Enter username"/>	Pid: <input type="text"/>	<input type="button" value="Refresh"/>
Show additional filters (1 filter(s) active)													
SESSIONS HISTORY												Toggle view:	
Logdate			Active Sessions				Idle sessions with transaction						
2019/02/27 13:31:44			4				2						
2019/02/27 10:19:12			4				2						
2019/02/27 01:32:22			4				2						
2019/02/27 01:10:21			4				2						
2019/02/27 13:46:45			3				1						
Sessions													
Pid	Query Id	User	Statement	Database	Application	Status	Wait type	Wait	Elapsed Time [Seconds]	Query Start	Transaction Start	Blocking Pid	
6276	916570963	tester	delete from comp	test1	dbplus	●active	Lock	transactionid	8	2019/02/27 01:32:14	2019/02/27 01:32:14	13244	
13244	916570963	tester	delete from comp	test1	dbplus	idle in transaction	Client	ClientRead	31	2019/02/27 01:31:51	2019/02/27 01:31:51		
18284	885422775	dbplus	insert into dbplus	dbplus	DBPLUS Perform	idle in transaction	Client	ClientRead	0	2019/02/27 01:32:22	2019/02/27 01:32:22		
12416	2288847016	tester	select count(*) fr	test1	dbplus	●active			0	2019/02/27 01:32:22	2019/02/27 01:32:22		
17908	2288847016	tester	select count(*) fr	test1	dbplus	●active			0	2019/02/27 01:32:22	2019/02/27 01:32:22		
3440	2288847016	tester	select count(*) fr	test1	dbplus	●active			0	2019/02/27 01:32:22	2019/02/27 01:32:22		

The main table contains snapshots performed in 60 second intervals (default setting of the module according to the parameter available in the **Configuration** option). Each snapshot contains information:

- Number of active sessions,
- Number of sessions using log files.

Click the table record presents details for the selected snapshot in the **Sessions** tab:

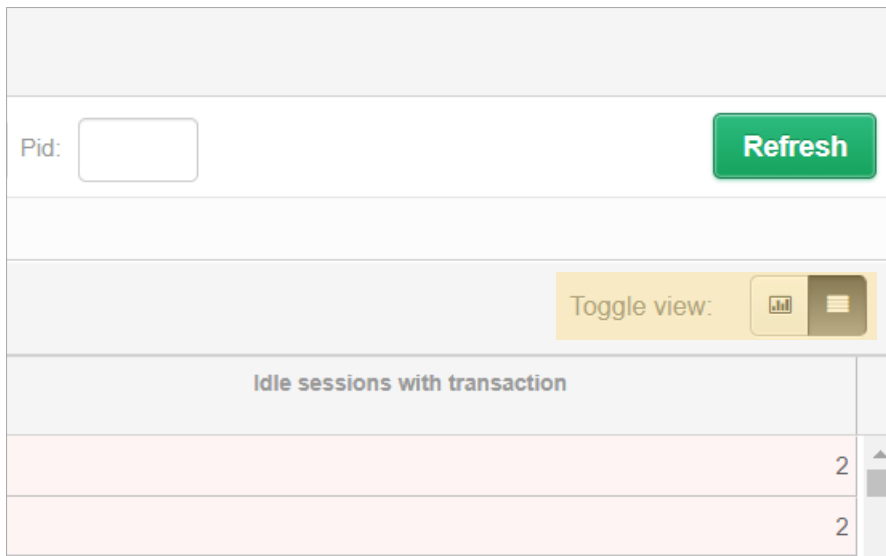
- **Pid** – user session ID,
- **QueryID** – query ID,
- **User** – username in the SQL Instance,
- **Statement** – query text,
- **Database**
- **Application** – the name of the application from which the logging into the database took place,
- **Status** – status of the session,
- **Wait type**
- **Wait** – wait name for the session,
- **Elapsed Time** – duration of the query,
- **Query start** – query start date,
- **Transaction start** – transaction start date,
- **Blocking Pid** – information about the identifier of the blocking session.

The application has ability to search information about the user's session using a given type of validity. User start the search by press the "Hide additional filters" button and then from the list of available waits they add the ones they want to view.

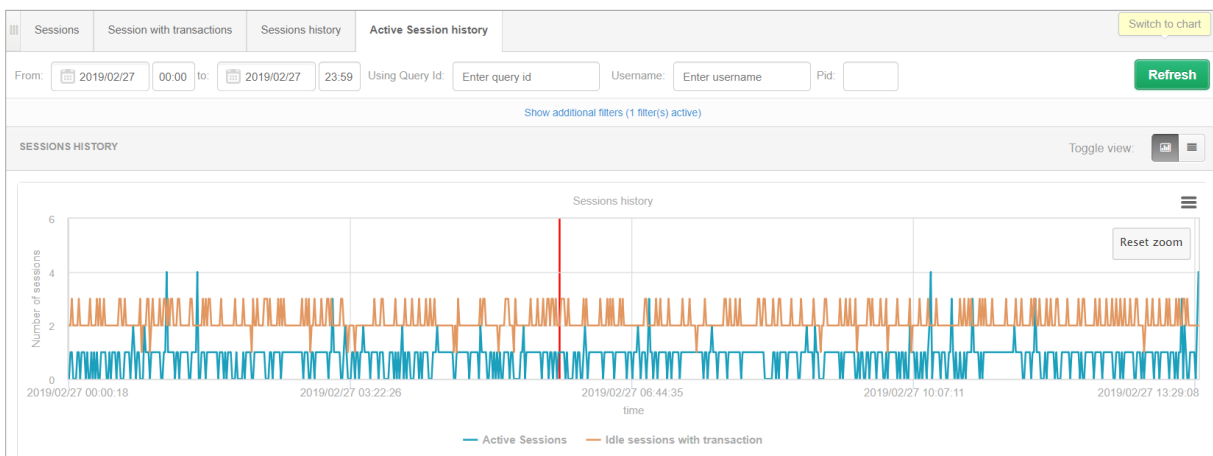
After press the Refresh button, only those sessions that wait, for a wait selected by the user from the list will be presented in the given period. At the same time, user can also select other filters, e.g. PID session ID or QueryID.

Logdate	Pid	Query Id	User	Statement	Database	Application	Status	Wait type	Wait	Elapsed Time [Seconds]	Query Start	Transaction Start	Blocking Pid
2019-02-27 00:00:18	13244		tester	update compai	test1	dbplus	idle in transa	Client	ClientRead	61	2019/02/26 23:59:18	2019/02/26 23:59:18	
2019-02-27 00:00:18	18284	885422775	dbplus	insert into dbpl	dbplus	DBPLUS Perfo	idle in transa	Client	ClientRead	0	2019/02/27 00:00:18	2019/02/27 00:00:18	
2019-02-27 00:01:18	18284	885422775	dbplus	insert into dbpl	dbplus	DBPLUS Perfo	idle in transa	Client	ClientRead	0	2019/02/27 00:01:18	2019/02/27 00:01:18	
2019-02-27 00:01:18	6276	916570963	tester	delete from co	test1	dbplus	idle in transa	Client	ClientRead	54	2019/02/27 00:00:25	2019/02/27 00:00:25	
2019-02-27 00:02:18	12416	2288847016	tester	select count(*)	test1	dbplus	idle in transa	Client	ClientRead	22	2019/02/27 00:01:56	2019/02/27 00:01:56	
2019-02-27 00:02:18	18284	885422775	dbplus	insert into dbpl	dbplus	DBPLUS Perfo	idle in transa	Client	ClientRead	0	2019/02/27 00:02:18	2019/02/27 00:02:18	
2019-02-27 00:02:18	13244	916570963	tester	delete from co	test1	dbplus	idle in transa	Client	ClientRead	22	2019/02/27 00:01:57	2019/02/27 00:01:57	

On the page that present the history of the session, user can also view information in the form of a graph. To do this, switch to the graph view on the website, as in the picture below.



The data in the graph are presented for 1-minute samples. The graph presents information on active sessions and sessions in the transaction:



6.2.7 „Locks” Menu

The lock module consists tabs:

- Locks history – allow to track blockades in time
- Online Locks – allow the current block analysis on the PostgreSQL Instance
- Locked Objects - show a list of objects on which locks are currently locked.

6.2.7.1 „Locks history” Tab

The page contains information about the history of blockades occurring in the past. The screen consists of the areas:

- The filter bar over the date range
- A graph shows the locks in time
- Tree of blocked sessions refreshed after click on the fragment / given point of the chart
 - at the top of the tree, blocking sessions are shown
 - in nodes below, waiting sessions blocked by sessions in the parent node
- Details for the selected session:
 - SQL Statement for session PID:* - the content of the query performed within a given session
 - Sessions Details – information on the session, i.a.: transaction open time, type of transaction, etc.

An example lock screen:

List of locked sessions at snapshot time: 2019-02/26 18:15:02	
▲ Session Id: 13244 Session status: idle in transaction Wait: ClientRead Wait Type: Client Last Query Runtime: 82037,9 s Last Start Time: 2019-02-26 18:13:39 Username: tester Program: dbplus	
Session Id: 6276 Session status: active Wait: transactionid Wait Type: Lock Last Query Runtime: 81989,9 s Last Start Time: 2019-02-26 18:14:27 Username: tester Program: dbplus	
SQL STATEMENT FOR SESSION PID: 13244	
delete from company where id=4644866	
SESSION DETAILS	
Pid	13244
Database	test1
UserName	tester
Status	idle in transaction
Program	dbplus
Query start	2019-02-26 18:13:39
Transaction begin time	2019-02-26 18:13:39
Wait	ClientRead
Wait Type	Client

6.2.7.2 „Online Locks” Tab

Online Locks tab consists of the area:

- Database filter bar (defaults block for all bases)
- List of locked sessions section – list of blocked sessions.

The section is built in the form of the tree, where:

- At the top of the tree, block sessions are shown
- In nodes below, wait sessions are blocked by sessions in the parent node
- Details for the selected session:
 - SQL Statement for session PID:* - query text performed in a given session
 - Sessions Details – transaction open time, transaction type, etc.

6.2.7.3 Locked objects Tab

The tab contains information about currently ongoing locks on open transactions on the PostgreSQL database server.

Database	User	Application	Pid	Lock Type	Lock Mode	Granted	Fastpath	Relation OID	Relation name	Page	Tuple	Virtualxid
dbplusrepo	dbplusrepo	DBPLUS Performanc...	28812	relation	AccessShareLock	true	true	16939	dbplus_tab4_server_9_snap			
dbplusrepo	dbplusrepo	DBPLUS Performanc...	28812	relation	RowExclusiveLock	true	true	16939	dbplus_tab4_server_9_snap			
dbplusrepo	dbplusrepo	DBPLUS Performanc...	28812	relation	AccessShareLock	true	true	16938	idx_dbplus_tab4_server_id			
dbplusrepo	dbplusrepo	DBPLUS Performanc...	28812	relation	RowExclusiveLock	true	true	16938	idx_dbplus_tab4_server_id			
dbplusrepo	dbplusrepo	DBPLUS Performanc...	28812	relation	AccessShareLock	true	true	16937	dbplus_tab4_map_id			
dbplusrepo	dbplusrepo	DBPLUS Performanc...	28812	relation	RowExclusiveLock	true	true	16937	dbplus_tab4_map_id			
dbplusrepo	dbplusrepo	DBPLUS Performanc...	28812	relation	AccessShareLock	true	true	16533	dbplus_tab4			
dbplusrepo	dbplusrepo	DBPLUS Performanc...	28812	relation	RowExclusiveLock	true	true	16533	dbplus_tab4			
dbplusrepo	dbplusrepo	DBPLUS Performanc...	28812	virtualxid	Exclusivelock	true	true					19/73232

6.2.8 „Parameters” Menu

System allows to view and report changes of instances parameters and databases. Each function presents current status of parameters and their changes over time.

Example screens below:

Status of parameters that contain the „host” word.

Param name	Setting value	Unit	Category	Short description
log_hostname	on		Reporting and Logging / What to Log	Logs the host name in the connection logs.

Date change	Setting value
2019/02/27 15:46:48	on
2019/02/22 15:55:14	off

History of parameter changes shows what PostgreSQL instance parameters were changed in during a period:

Param name	Setting value	Unit	Category	Short description	Date change
allow_system_table_mods	off		Developer Options	Allows modifications of the structure of s	2019/02/22 15:55:14
application_name	DBPLUS Performance Monitor		Reporting and Logging / What to Log	Sets the application name to be reported	2019/02/22 15:55:14
archive_command	(disabled)		Write-Ahead Log / Archiving	Sets the shell command that will be calle	2019/02/22 15:55:14
archive_mode	off		Write-Ahead Log / Archiving	Allows archiving of WAL files using archi	2019/02/22 15:55:14
archive_timeout	0	s	Write-Ahead Log / Archiving	Forces a switch to the next WAL file if a	2019/02/22 15:55:14
array_nulls	on		Version and Platform Compatibility / Pre	Enable input of NULL elements in arrays	2019/02/22 15:55:14
authentication_timeout	60	s	Connections and Authentication / Auther	Sets the maximum allowed time to comp	2019/02/22 15:55:14
autovacuum	on		Autovacuum	Starts the autovacuum subprocess.	2019/02/22 15:55:14
autovacuum_analyze_scale_factor	0.1		Autovacuum	Number of tuple inserts, updates, or dele	2019/02/22 15:55:14
autovacuum_analyze_threshold	50		Autovacuum	Minimum number of tuple inserts, update	2019/02/22 15:55:14

IMPORTANT: The parameter module is also available from the main menu level after exit the Instance Analysis performance module. Then the system allows to analyze parameters for all monitored PostgreSQL instances at the same time.

6.2.9 „Logs” Menu

The **Logs** module allows the user to:

- DBPLUS procedure statistics
- Procedure Errors
- Plan Parsing log

6.2.9.1 Process collecting data for monitoring

In the latest version of the application, the presentation of information on the times of collecting data from monitored databases by the DBPLUSPOSTGRESCATCHER Windows service has been more detailed. This information relates to the procedure for monitoring the database at 15-minute intervals.

The data, as before, is available in the Logs tab at the detail level of the given database. From this version, by clicking on a row in the Snaps table runtime procedure, the User will receive detailed information on the next steps that make up the monitoring procedure.

The screenshot shows the 'Logs' menu selected in the left sidebar. The main window displays 'Snap details at 2020-03-29 18:06:16'. It features two tables: 'Snap procedure run time' and 'INTERNAL PROCEDURES RUN TIME'. The 'Snap procedure run time' table has columns for Date, Work time, and Status. The 'INTERNAL PROCEDURES RUN TIME' table has columns for Step, Procedure, Start, End, Duration (Seconds), and Status. A red box highlights the first row of the 'INTERNAL PROCEDURES RUN TIME' table, which is '1 Check last database restart'.

Date	Work time	Status
2020-03-29 18:06:16	4	●
2020-03-29 17:51:23	4	●
2020-03-29 17:35:46	4	●
2020-03-29 17:20:32	4	●
2020-03-29 17:05:17	4	●
2020-03-29 16:50:03	5	●
2020-03-29 16:34:49	4	●
2020-03-29 16:19:35	4	●
2020-03-29 16:04:21	4	●
2020-03-29 15:49:07	5	●
2020-03-29 15:33:53	4	●
2020-03-29 15:18:39	9	●
2020-03-29 15:03:25	6	●
2020-03-29 14:48:10	5	●
2020-03-29 14:32:56	7	●
2020-03-29 14:17:42	7	●
2020-03-29 14:02:29	5	●
2020-03-29 13:47:14	5	●
2020-03-29 13:32:00	5	●
2020-03-29 13:16:47	5	●
Average time	5	-
Min time	4	-
Max time	22	-
Count steps	22	-

Step	Procedure	Start	End	Duration (Seconds)	Status
1	Check last database restart	2020-03-29 18:06:16	2020-03-29 18:06:16	0.078	●
2	Waits events statistics	2020-03-29 18:06:16	2020-03-29 18:06:16	0.125	●
3	Latches statistics	2020-03-29 18:06:16	2020-03-29 18:06:16	0.078	●
4	Database file information	2020-03-29 18:06:16	2020-03-29 18:06:16	0.094	●
5	Database size (total, used, free space)	2020-03-29 18:06:16	2020-03-29 18:06:16	0.094	●
6	I/O operation statistics	2020-03-29 18:06:16	2020-03-29 18:06:16	0.094	●
7	Memory informations (buffer, procedures caches size)	2020-03-29 18:06:16	2020-03-29 18:06:17	0.100	●
8	Query statistics (queries, procedures) including sql text and plans	2020-03-29 18:06:17	2020-03-29 18:06:17	0.078	●
9	Merge Query statistics to day view	2020-03-29 18:06:17	2020-03-29 18:06:17	0.016	●
10	Merge I/O operations to day view	2020-03-29 18:06:17	2020-03-29 18:06:17	0.016	●
11	Instance parameters informations	2020-03-29 18:06:17	2020-03-29 18:06:17	0.078	●

Then, by pointing to the step (in the Snap details table), the User receives information about the duration of the procedure and the number of rows processed (information available only for certain steps).

The screenshot shows 'Snap details at 2019-12-23 15:39:09'. The 'INTERNAL PROCEDURES RUN TIME' table is highlighted with a red box. It has columns for Step, Procedure, Start, End, Duration (Seconds), and Status. The second row, '2 Waits events statistics', is highlighted with a red box. Below the table, there is a 'Statistics' section with columns for Type, Counter value, Start, End, and Timer Duration (Seconds). The 'Rows processed' row is highlighted with a red box.

Step	Procedure	Start	End	Duration (Seconds)	Status
1	Check last database restart	2019-12-23 15:39:09	2019-12-23 15:39:09	0	●
2	Waits events statistics	2019-12-23 15:39:09	2019-12-23 15:39:09	0.452	●
3	Latches statistics	2019-12-23 15:39:09	2019-12-23 15:39:10	0.140	●
4	Operating system information	2019-12-23 15:39:10	2019-12-23 15:39:10	0.016	●
5	Query statistics (queries, procedures) including sql text and plans	2019-12-23 15:39:10	2019-12-23 15:39:14	4.727	●
6	Database size (total, used, free space)	2019-12-23 15:39:14	2019-12-23 15:39:14	0	●
7	I/O operation statistics	2019-12-23 15:39:14	2019-12-23 15:39:14	0.140	●
8	Memory informations (SGA including shared pool, db cache size)	2019-12-23 15:39:14	2019-12-23 15:39:15	0.328	●
9	Merge Query statistics to day view	2019-12-23 15:39:15	2019-12-23 15:39:16	1.279	●
10	Merge I/O operations to day view	2019-12-23 15:39:16	2019-12-23 15:39:16	0.078	●
11	Parameters informations	2019-12-23 15:39:16	2019-12-23 15:39:16	0.094	●

Type	Counter value	Start	End	Timer Duration (Seconds)
Read data		2019-12-23 15:39:09	2019-12-23 15:39:09	0.437
Write data		2019-12-23 15:39:09	2019-12-23 15:39:09	0.016
Rows processed	58			

Information about the status of a given snap is contained in the Status column. If the monitoring process run correctly, a green dot will be displayed in the column.

If one of the monitoring procedure steps has not been performed or has been interrupted and the step concerned is not critical, the User receives information about the reason for the interruption of the step and the status of the entire snap is presented in orange.

The screenshot shows the 'Procedure statistics' and 'Procedure Errors' tabs. The 'Snap procedure run time' table shows a snap at 2019-12-23 14:06:23 with a status of 'running' (orange dot). The 'Snap details at 2019-12-23 14:06:23' table shows various steps, with step 6 'Database size (total, used, free space)' highlighted in orange. The 'ERROR LOGS FOR SELECTED STEP: DATABASE SIZE (TOTAL, USED, FREE SPACE)' table shows an error at 2019-12-23 14:14:45: 'Error reported in following program: StandardSnap_CatchOOBSize: Execution for query SELECT *->ALL_ROWS *file_id, nvl(Sum(bytes),0) bytes FROM DBA_free_space GROUP BY file_id timed-out at DBPLUS.'

If there was a problem with the connection at the time of the monitoring procedure or the problem concerned a critical step for a given procedure, the status information is written in red.

The screenshot shows the 'Procedure statistics' and 'Procedure Errors' tabs. The 'Snap procedure run time' table shows a snap at 2019-12-23 16:15:00 with a status of 'failed' (red dot). The 'Snap details at 2019-12-23 16:15:00' table shows step 1 'No any steps executed for specified snapshot' with a red status. The 'ERROR LOGS FOR SELECTED SNAPSHOT' table shows three error messages: 'Error reported in following program: SessionsUnlockSort: SnapRunnetLocks Run: ORA-12541: TNS: No listener at OracleInternal.ConnectionPool.PoolManager'3.GetConnectorString cs@WIB-DRO-NewPvt...', 'Error reported in following program: Dashboard_SnapRunner_DashboardSnapQueries: ORA-12541: TNS: No listener at DBPLUS Catcher facade SQLFacadeDashboard_DashboardSnapQueries:Boolean deleteCL...', and 'Error reported in following program: SessionsUnlockSort: SnapRunnetLocks Run: ORA-12541: TNS: No listener at OracleInternal.ConnectionPool.PoolManager'3.GetConnectorString cs@WIB-DRO-NewPvt...'.

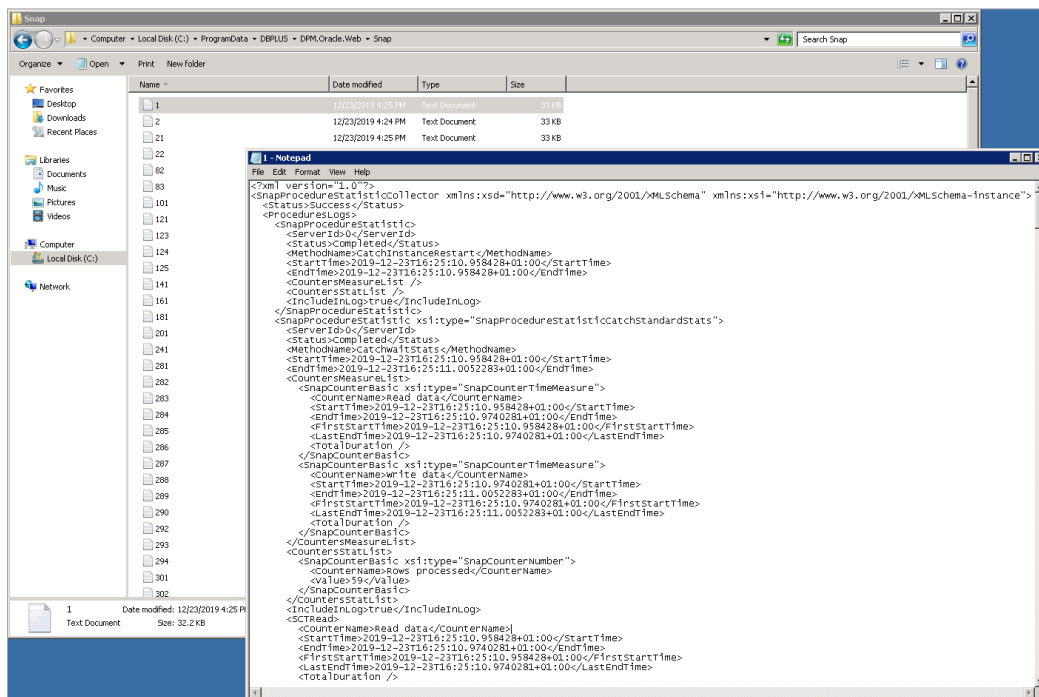
If the monitoring procedure is in progress, this information is visible in the status (running) field, as well as the Online steps refresh button is visible, after which the information on the monitoring procedure progress is refreshed.

Date	Work time	Status
2019-12-24 10:28:11	7	running
2019-12-24 10:12:39	36	●
2019-12-24 09:57:46	72	●
2019-12-24 09:42:34	35	●
2019-12-24 09:27:21	34	●
2019-12-24 09:12:09	33	●
2019-12-24 08:56:57	61	●
2019-12-24 08:41:42	36	●
2019-12-24 08:26:29	45	●
2019-12-24 08:11:16	32	●
2019-12-24 07:56:03	65	●
2019-12-24 07:40:51	30	●
2019-12-24 07:25:38	34	●
2019-12-24 07:10:25	31	●
2019-12-24 06:55:13	60	●
2019-12-24 06:40:00	29	●
2019-12-24 06:24:47	32	●
2019-12-24 06:09:34	38	●
2019-12-24 05:54:21	45	●
2019-12-24 05:39:09	84	●

In addition, all problems related to the monitoring procedure are available in the form of a list on the Procedure Errors tab.

Information on the monitoring procedure is also included in the form of a file on the application server. The file contains information about the last snap performed on a given database. The file is in the folder: C:\ProgramData\DBPLUS\DPM.Postgres.Web\Snap

Each file is marked with a digit assigned to the database when it is included in the monitoring (dbplus_central_servers table in the DBPLUS schema in the repository database).



6.2.10 „Reports” Menu

- The Reports module contains the following report Performance Report

6.2.10.1 Performance Report

The report presents the performance of the SQL Instances in the selected time period. The report contains information about:

- Top queries operating in the database for:
 - Duration: Elapsed Time,
 - High time IO,
 - High block reads,
 - High Block Hit,
 - Number of performances.
- List of top waits

6.3 Space monitor Menu

Space Monitor module allows users to analyze the storage space on servers. The module is divided into two basic groups:

- Display the current database size,
- A history of change size in tabular and graphical form.

In this menu, user can collectively verify information on disk space usage for all instances connected to monitoring.

6.4 Parameters Menu

The page allows to view and report changes instance parameters and databases over time. Additional options are available in the menu on the left:

- **Instance Parameters** – instance parameters,
- **Instance extensions** – other instance extensions.

The window presents the current status of parameters and their changes over time. From this level, it is possible to view information about given parameters simultaneously for all instances connected to monitoring.

The screenshot shows the 'Extensions overview' interface. It includes a sidebar with 'Overview' and 'History' options, a search bar, and a list of instances under 'ALL INSTANCES'. The 'ePostgres' instance is selected. A table below displays the details of extensions installed on this instance.

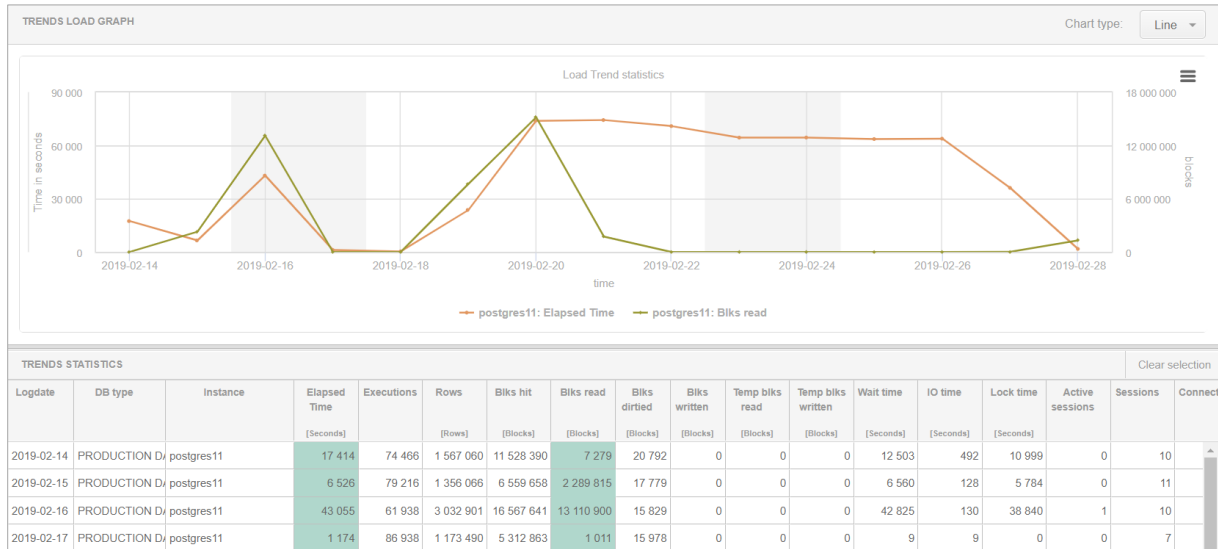
Instance type	Instance	Extension name	Default version	Installed version	Comment
Not specified	ePostgres	adminpack	2.0	2.0	administrative functions for PostgreSQL
Not specified	ePostgres	amcheck	1.1		functions for verifying relation integrity
Not specified	ePostgres	autoinc	1.0		functions for autoincrementing fields
Not specified	ePostgres	bloom	1.0		bloom access method - signature file bar
Not specified	ePostgres	btree_gin	1.3		support for indexing common datatypes
Not specified	ePostgres	btree_gist	1.5		support for indexing common datatypes

Below the table is a 'HISTORY FOR SELECTED EXTENSION' section with columns for Date change, Default value, and Installed value.

Date change	Default value	Installed value
2019/02/22 15:55:14	2.0	2.0

6.5 Reports Menu

In this menu, the application allows user to make reports based on data from monitored servers. The "Load trends" report refers to the values of the main statistics calculated for each PostgreSQL instance. As part of this report, user can compare statistics for multiple instances at the same time.



6.6 Servers Monitor Menu

6.6.1 Application architecture

The screen of the DBPLUS Performance Monitor system is available from the main menu, i.e. **Servers monitor -> Application architecture**. The module allows user to check:

- List of monitored PostgreSQL instances,
- Monitoring service activity,
- In which instance / database there is a system repository.

PostgreSQL instances are available in the area on the left. There is information about:

- when the last snapshot for the monitored PostgreSQL instance was made
- when was the last activity of the PostgreSQL instance (connection from the monitoring service with the PostgreSQL instance).

In the middle area there is information about the current state of the DBPLUSPOSTGRESCATCHER monitoring service. There is such information as:

- Is the service run
- Last activity of the service
- Memory utilization on the machine where the monitoring service works.
- Processor usage by monitoring service.

Below the statistics user can check the historical status of the service for a given time.

On the right side there is information about the SQL instance where the DBPLUS Performance Monitor repository is located.

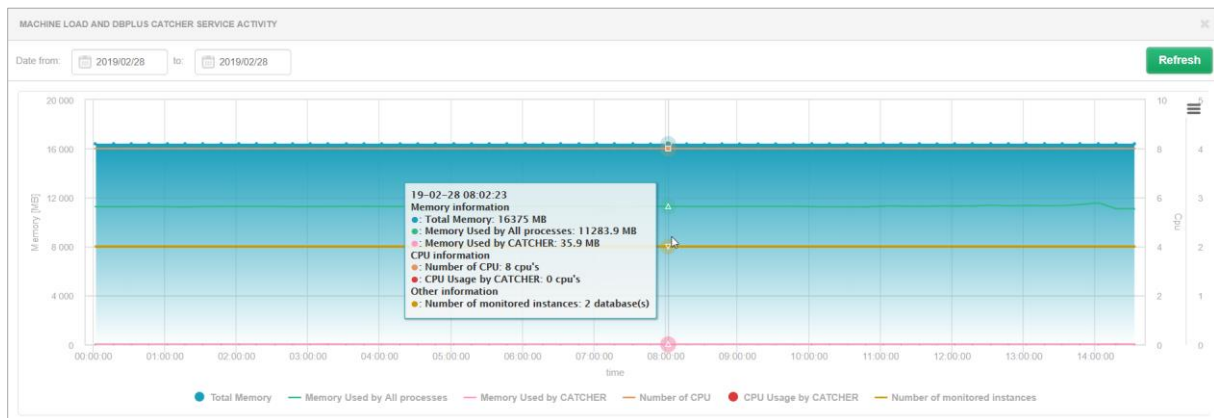
An example screens

Application architecture

List of monitored instances			Monitoring service	Dbplus Performance Monitor
Instance name	Last snapshot date	Last activity	<p>DBPLUS Postgres Catcher</p> <p>Service status: ● Running</p> <p>Last service activity: 2019-02-28 14:34:20</p> <p>Machine Total Memory: 16375 MB</p> <p>Machine Memory Usage: 11178 MB</p> <p>DBPLUSCATCHER Memory Usage: 38 MB</p> <p>DBPLUSCATCHER CPU Usage: 0,2 %</p> <p>Refresh View service activity</p>	<p>Repository Information</p> <p>Hostname: localhost Database: dbplus User: dbplus</p>
● ePostgres	2019/02/28 14:32:45	2019/02/28 14:34:30		
● postgres11	2019/02/28 14:32:54	2019/02/28 14:34:30		

● Postgres Instance is monitored ● Postgres Instance is disabled (to enable please go to Configuration module) ● Postgres Instance is not available or DBPLUS POSTGRES Catcher service is not running

In the slide below is the history of the DBPLUSCATCHER service activity, after click the **[View service activity]** button:



6.6.2 Schedules outages

After enter the tab, the user has the opportunity to view information about scheduled monitoring shutdowns. On the website, only the exclusions for the current day as well as those scheduled in the future are visible by default. User information can be viewed for all databases as well as for a specific database.

To add a new entry, click the **[Add new outage]** button.

Scheduled outages

Date from: 2018/11/26 to: Filter by database: All databases [Refresh](#)

DATABASES OUTAGES SCHEDULE [Add new outage](#)

Outages information and its schedules are refreshed within 15 minutes.

Database	Enabled	Period	Outage days	Outage hours	Reason
FK08T	<input checked="" type="checkbox"/>	From 2018-11-24 to 2018-11-28	Every Sat, Sun	between 17:00 - 17:20	Outage module testing
FK08T	<input checked="" type="checkbox"/>	From 2018-11-26 to 2018-11-26	Every Mon	between 14:40 - 15:00	testowe wyłączenie monitoringu

After click, the user selects for which base he is to be switched off, and then chooses whether the shutdown is to be:

- one-time or cyclical,
- is supposed to last one or many days,
- is supposed to appear on a specific day of the week.

After select, add information about the reason for the shutdown and accept configurations. After the correctly entered configuration, the new entry will be visible in the table. It must be remembered that the information about the shutdown will appear on the chart when the new / next snap is generated.

Information about turn off monitoring is visible on the Dashboard screen:

- in Television mode - a yellow mark next to the database and a description of "Monitoring Outage",
- in Icons view.

In this view the base is also marked in yellow, which means a break in monitoring. As well as the base where monitoring has been disabled, it is not included in the number of active databases.

- in Grid view, the row in the table is yellow

Information about turn off is also visible on the Instance Load chart. In case the instance is excluded from monitoring, yellow vertical bars are drawn in the chart. At the moment of turning off, information about the statistics is not collected.

6.6.3 Scheduled works

After enter the tab, user can view information about the upcoming scheduled work. On the page, only works for the current day and those scheduled in the future are visible by default. The information can be viewed for all PostgreSQL instances. The functionality is created to present information about scheduled work that can affect the performance of the database. To add a new entry, click the **[Add new work or tag]** button.

Scheduled works & timeline tags			
Date from: 2018/11/26 to: <input type="text"/>		Filter by database: All databases	<input type="button" value="Refresh"/>
PLANNED WORKS & TIMELINE TAGS SCHEDULE			<input type="button" value="Add new work or tag"/>
Planned works, timeline tags are visible on Database load, Load Trends charts for specified databases			
Database	Timeline	Work title	Details & Description
FK08T	2018-11-26 10:56	Wgranie poprawek	Praca testowa
FK08T	2018-11-26 13:20	wgranie poprawek 2	Praca testowa

After click, user can choose for which database the planned work should be registered, and then select whether the shutdown should be:

- Single or long period

After select the scope, user can add information about the "tag title" field (visible later in the chart), and add detailed information about the planned work, then accept the configuration. After the correctly entered configuration, the new entry will be visible in the table. It must be remembered that information about scheduled work will appear on the chart when the new / next snap is generated.

Information on scheduled work is shown in the Instance Load chart in the form of points (single events) or horizontal stripes in the case of long-term work. After indicate the point / bar, the information about the scope and the topic of the planned work will be displayed. If work is planned in the future, information about the work will be visible as a point on the right side of the chart.

Additionally, from the Instance Load level user can manage implementations by click the **[Manage timeline]** button.

6.6.4 Logs

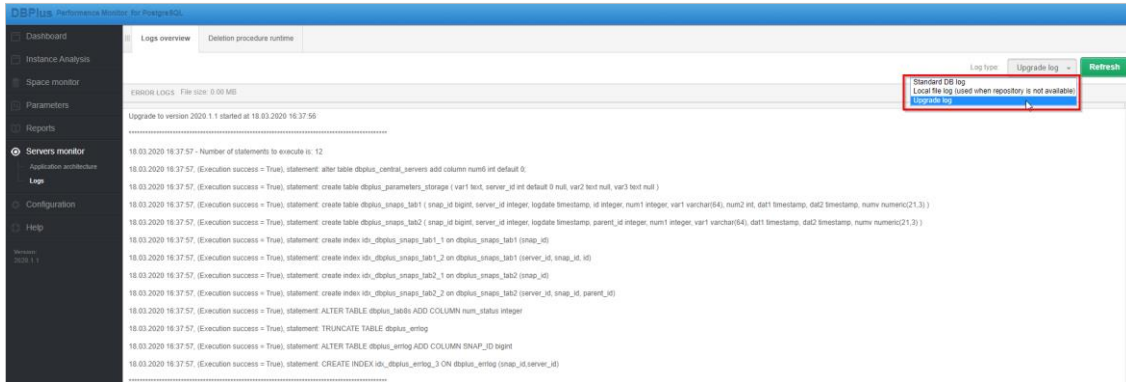
On this page it is possible to suspect logs related to the operation of the DBPLUS Performance Monitor application over the period.

After entering the Logs overview tab, the User will be presented with default logs saved in the Repository database (Standard DB Log), that contain information about problems with possible monitoring.

The User will also have the option of displaying information available in the logs available locally on the application server (Local file log). Information about problems is saved there when it is not possible to save this information in the repository database.

The next log concerns information related to the application update process. This file is created during the application upgrade process (downloading the new version). We save information about changes made to the data model as well as the update process.

In addition, information about the size of the file is displayed for each file.



6.7 Menu Configuration

6.7.1 Settings

Parameterization the DBPLUSPOSTGRESCATCHER service is available from the **Configuration->Settings**. Depends on the quality of queries and the type of problems in the system, options are enabled:

- **Number of storage days for the retail history of PostgreSQL Instances performance,**
- **The time interval between the collection of blocking samples,**
- **Query plans monitoring**

Parameter	Value	Description	
DASHBOARD_ANIMATE_PARAMETERS	ON	Setting is valid for DPM dashboard displayed in television mode. Based on it each sql server icon can toggle/animate automatically its parameters like (server cpu, waits, sessions, etc.)	Edit
KEEP_SNAPSHOT_HISTORY_DAYS	30	Number of days how long to keep detail statistics for sql statement executions, waits, latches, performance counters.	Edit
LOCKING_SNAPSHOT_FREQUENCY	60	The interval time in seconds between each snapshot of locks made by DBPLUSPOSTGRESCATCHER service. The parameter can be setup separately for each instance. In a case of frequent locks, please consider lower value for LOCKING_SNAPSHOT_FREQUENCY. In a case of rarely occurred locks, please use bigger value for it.	Edit
LOGGING_MODE	ON	Parameter used for debugging mode. By default it should be set to OFF.	Edit
MONITOR_EXPLAIN_PLANS	ON	Parameter which switch ON/OFF the module to estimate explain plans for most heavy statements run on the instance.	Edit
PLANS_TO_GENERATE_PER_SNAP	50	Number of most heavy queries for which system will estimate explain plans - Estimation is done in every snapshot.	Edit
SECURITY	OFF	Application can work in SECURITY mode set to ON or to OFF. It means that application uses (or doesnt use) user authentication. Setting the SECURITY to on, it requires at least one user created.	Edit

IMPORTANT: The selected parameters can be set at the general level or for specific SQL instances. This applies to the parameters:

Parameter	Value	Description	
LOCKING_SNAPSHOT_FREQUENCY	60	The interval time in seconds between each snapshot of locks made by DBPLUSPOSTGRESCATCHER service. The parameter can be setup separately for each instance. In a case of frequent locks, please consider lower value for LOCKING_SNAPSHOT_FREQUENCY. In a case of rarely occurred locks, please use bigger value for it.	Edit

6.7.2 Instances

On the page it is possible to configure which PostgreSQL instances should be monitored. User set the instance type. The correct type set for each instance allows the user to use this group in various functions of the DBPLUS Performance Monitor, i.a.: Space Monitor, when the size of instances assigned to a given group is presented.

On the website it is also possible to set, i.a.:

- instance type assignment
- instance visibility in monitoring
- additional information how to connect the instance to the DBPLUS Performance Monitor application

View postgres instances and its connections

INSTANCE SETTINGS

Default Instance Name Format:

POSTGRES INSTANCE LIST					DETAILS FOR SELECTED INSTANCE	
Search by name					Basic	Connection properties
Host name	Connection name	Used instance name	Type	Enabled		
localhost	ePostgres	ePostgres	Not Specified	✓	Connection name: <input type="text" value="ePostgres"/>	
localhost	Repository instance	postgres11	PRODUCTION DATABASE	☑	Hostname: <input type="text" value="localhost"/>	
					TCP Port: <input type="text" value="5444"/>	
					Type: <input type="text" value="Not specified"/>	
					Instance Name format: <input type="text" value="Connection name"/>	
					Enabled: <input type="text" value="Yes"/>	
					<input type="button" value="Save details"/>	

6.7.3 Reference lists

This tab contains the system dictionaries used in the application. Existing dictionary data can be freely modified.

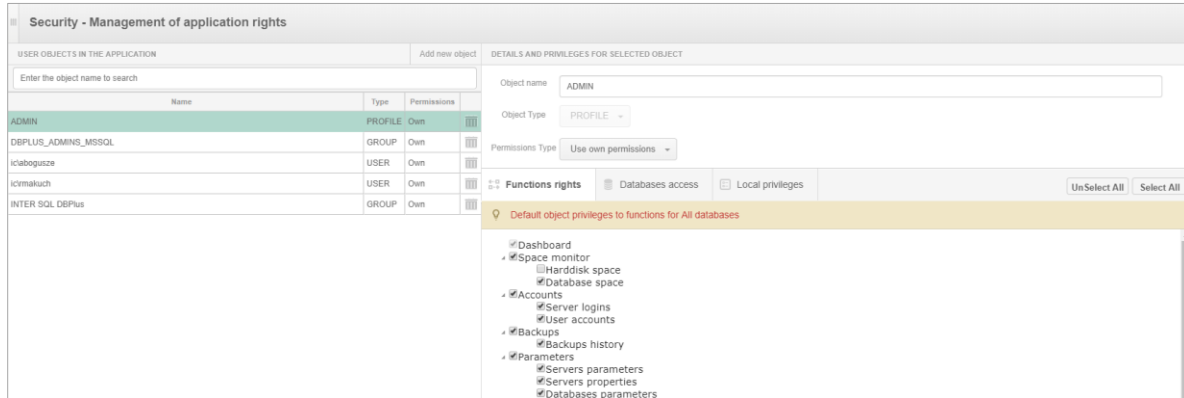
Reference types management

🔔 List of references list used to assign categories for postgres instances and its databases. Please click on the list from left site to see items belong to specified reference. ✕

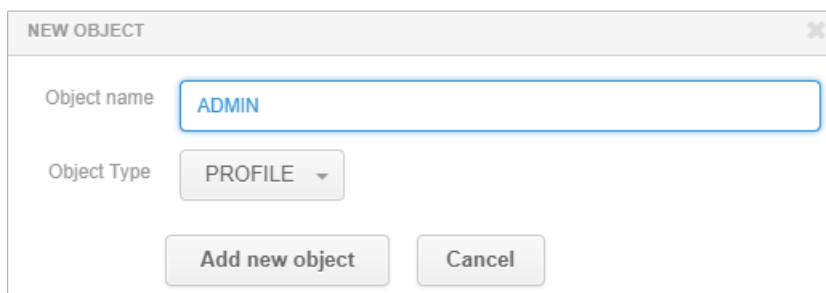
REFERENCE	REFERENCE LIST ITEMS
List Name	<input type="text" value="Enter the name for new item"/> <input type="button" value="Add item"/>
Server types	Name
Reason class	PRODUCTION DATABASE <input type="button" value="Edit"/> <input type="button" value="✕"/>
	TESTING DATABASE <input type="button" value="Edit"/> <input type="button" value="✕"/>
	DEVELOPMENT <input type="button" value="Edit"/> <input type="button" value="✕"/>
	TEMPORARY <input type="button" value="Edit"/> <input type="button" value="✕"/>
	OTHER <input type="button" value="Edit"/> <input type="button" value="✕"/>

6.7.4 Security

This tab provides the option of setting access for a user, group of users or profiles (templates - set of rights). Access is granted at the SQL Instance level and at the level of available pages in the menu.



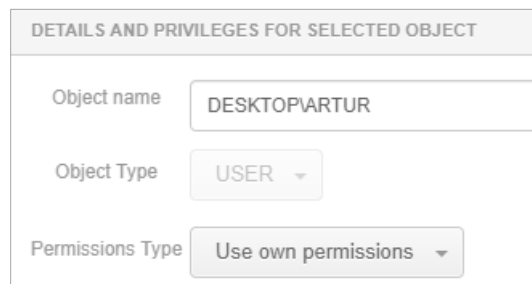
In order to create a new object, for example a profile (PROFILE), click on [Add new object], then select the object type "PROFILES" and name the object.



To assign permissions to a given object, select it from the list on the left side of the screen. After click on the object on the right side, the page with the access configuration will display.

First user needs to choose whether the permissions will be:

- own (Use own permissions).
- inherited (inherited permissions form parents).



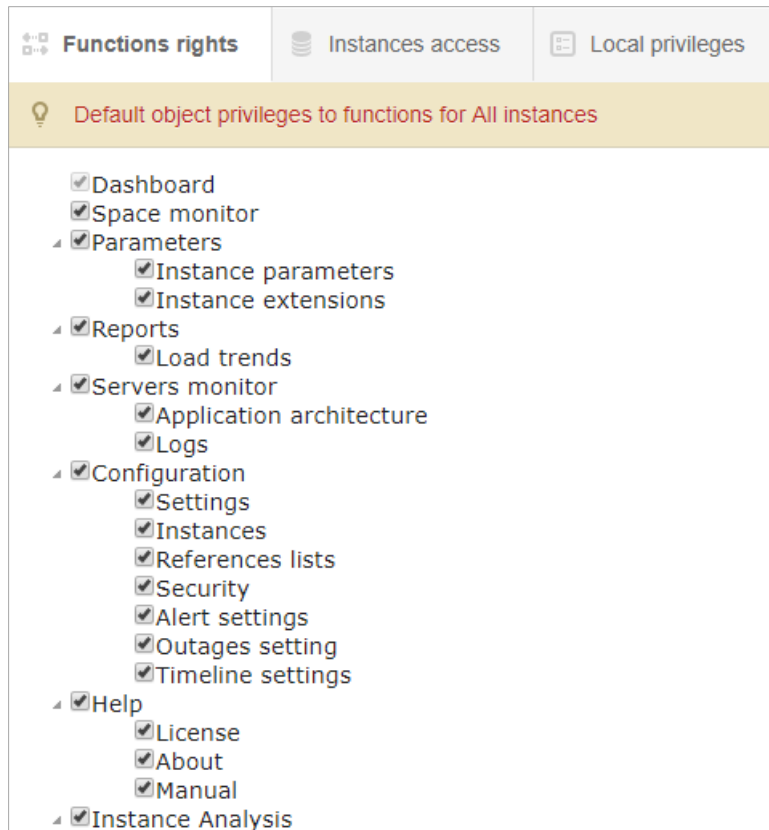
6.7.4.1 Own permissions

When user select own permissions, they have three tabs to configure permissions:

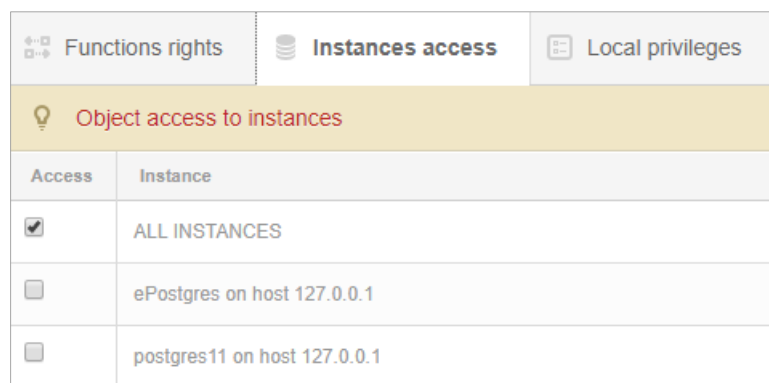
- Function rights,
- Databases access,
- Local privileges

Functional settings allow user to give rights to pages or functionality in the application at the global level for a given user / group or profile for all sql instances. User can override these rights by granting custom permissions for a specific instance. Custom permissions can only be changed for the **Instance Analysis**

module. Custom permission is superordinate to a given Instance in relation to functional rights. If you assign custom permissions, the (permissions overwritten) message will be displayed.



In addition, user can restrict access to specific databases. To do this, in the **Instances access** tab, select the appropriate check boxes for a chosen Instance or select ALL_INSTANCES. If certain bases are restricted, this will also limit the **Local privileges** tab.



6.7.4.2 Inherited permissions form parents

If user choose inherited rights, they can specify which profile or profiles to use for a given user or user group. Each profile contains a list of objects and access to which. Grant permissions to multiple profiles for the user will result in the entitlement for a given user being the sum of rights for selected profiles.

Profiles assignment	
Permissions to inherited from assigned profiles	
Access	Profile Name
<input type="checkbox"/>	ADMIN
<input type="checkbox"/>	ADMIN2
<input type="checkbox"/>	ADMIN3

Attention! In order to enable the functionality of limited access to the application, user have to change the settings at the level of the DBPLUS Configuration Wizard> Applications settings> Applications Options> Configure. As well as change the status of the SECURITY parameter to “ON”.

Parameter	Value	Description	
APPLICATION PARAMETERS			
SECURITY	ON	Application can work in SECURITY mode set to ON or to OFF. It means that application uses (or doesn't use) user authentication. Setting the SECURITY to on, it requires at least one user created.	Save
DASHBOARD_ANIMATE_PARAMETERS	ON	Setting is valid for DPM dashboard displayed in television mode. Based on it each sqg server icon can toggle/animate automatically its parameters like (server cpu, waits, sessions, etc.)	Edit
LOCKING_SNAPSHOT_FREQUENCY	300	The interval time in seconds between each snapshot of locks made by DBPLUS CATCHER service. The parameter can be setup separately for each instance. In a case of frequent locks, please consider lower value for LOCKING_SNAPSHOT_FREQUENCY. In a case of rarely occurred locks, please use bigger value for it.	Edit

The next step is to enable the mechanism to authenticate in the application by change the SECURITY parameter in the **Configuration-> Settings option**

6.7.5 Alert Settings

The alert module is available from the main menu, i.e. **Configuration-> Alert settings**. From this tab users can:

- Parameter settings related to mail - i.a. data of the mail server and account from which alert messages will be sent,
- Make general module settings,
- Define alerts,
- Specify the list of alert recipients.

6.7.5.1 „Mail settings” Tab

For the information about an alert to be sent via email, user must configure the SMTP server settings.

As part of the configuration, users have the option to set the frequency of send information about the event, depend on the configuration it is from 1 minute to 1 hour.

Mail settings	General settings	Alerts definition	Reasons & Problems definition	Events subscription
<p>💡 List of email configuration parameters.</p>				
<input checked="" type="checkbox"/> Send alerts by mail				
Mail Agent Interval	once per 5 minutes			
SMTP Mail server	pop3-dbpluskonto.ogicom.pl			
Port	587			
Sender email address	alert@dbplus.pl			
<input checked="" type="checkbox"/> smtp authentication				
Username	alert@dbplus.pl			
Password	*****			
<input type="checkbox"/> enable SSL				
Test mail address			Send test mail	
<input type="button" value="Save mail settings"/>				

IMPORTANT: Email alerts for all instances are sent from one email account.

6.7.5.2 „General settings” Tab

In this tab, users can make general settings of the alert module. User has the option to configure parameters related to the alert mechanism.

DBPlus Better performance for POSTGRES																																																								
<ul style="list-style-type: none"> Dashboard Instance Analysis Space monitor Parameters Reports Servers monitor Configuration <ul style="list-style-type: none"> Settings Instances References lists Security Alert settings Help <p>Version: 2019.1.4</p>	<table border="1"> <thead> <tr> <th>Mail settings</th> <th>General settings</th> <th>Alerts definition</th> <th>Reasons & Problems definition</th> <th>Events subscription</th> </tr> </thead> <tbody> <tr> <td colspan="5"> Elapsed Time greater than <input type="text" value="200"/> seconds. Alerts would only be ran if the elapsed time for all sql statements would take at least seconds in duration of 15 minutes (snapshot time) </td> </tr> <tr> <td colspan="5"> History Days <input checked="" type="checkbox"/> Mon <input checked="" type="checkbox"/> Tue <input checked="" type="checkbox"/> Wed <input checked="" type="checkbox"/> Thu <input checked="" type="checkbox"/> Fri <input type="checkbox"/> Sat <input type="checkbox"/> Sun <i>We recomend to select working days only</i> </td> </tr> <tr> <td colspan="5"> Number of Days Back in History <input type="text" value="30"/> <i>How long history would be included in snapshot alerts calculation</i> </td> </tr> <tr> <td colspan="5"> STATEMENTS SETTINGS </td> </tr> <tr> <td colspan="5"> Number of Top Queries to check <input type="text" value="10"/> chosen by <input type="text" value="Elapsed time"/> <i>How many top statements from each snapshot would be check by Alert Engine</i> </td> </tr> <tr> <td colspan="5"> Number of Days Back in History <input type="text" value="7"/> <i>How long statement history would be included in snapshot alerts calculation</i> </td> </tr> <tr> <td colspan="5"> WAIT EVENTS SETTINGS </td> </tr> <tr> <td colspan="5"> Number of Top Waits to check <input type="text" value="5"/> </td> </tr> <tr> <td colspan="5"> Number of Days Back in History <input type="text" value="7"/> <i>How long wait history would be considered in snapshot alerts calculation</i> </td> </tr> <tr> <td colspan="5" style="text-align: center;"> <input type="button" value="Save settings"/> </td> </tr> </tbody> </table>	Mail settings	General settings	Alerts definition	Reasons & Problems definition	Events subscription	Elapsed Time greater than <input type="text" value="200"/> seconds. Alerts would only be ran if the elapsed time for all sql statements would take at least seconds in duration of 15 minutes (snapshot time)					History Days <input checked="" type="checkbox"/> Mon <input checked="" type="checkbox"/> Tue <input checked="" type="checkbox"/> Wed <input checked="" type="checkbox"/> Thu <input checked="" type="checkbox"/> Fri <input type="checkbox"/> Sat <input type="checkbox"/> Sun <i>We recomend to select working days only</i>					Number of Days Back in History <input type="text" value="30"/> <i>How long history would be included in snapshot alerts calculation</i>					STATEMENTS SETTINGS					Number of Top Queries to check <input type="text" value="10"/> chosen by <input type="text" value="Elapsed time"/> <i>How many top statements from each snapshot would be check by Alert Engine</i>					Number of Days Back in History <input type="text" value="7"/> <i>How long statement history would be included in snapshot alerts calculation</i>					WAIT EVENTS SETTINGS					Number of Top Waits to check <input type="text" value="5"/>					Number of Days Back in History <input type="text" value="7"/> <i>How long wait history would be considered in snapshot alerts calculation</i>					<input type="button" value="Save settings"/>				
Mail settings	General settings	Alerts definition	Reasons & Problems definition	Events subscription																																																				
Elapsed Time greater than <input type="text" value="200"/> seconds. Alerts would only be ran if the elapsed time for all sql statements would take at least seconds in duration of 15 minutes (snapshot time)																																																								
History Days <input checked="" type="checkbox"/> Mon <input checked="" type="checkbox"/> Tue <input checked="" type="checkbox"/> Wed <input checked="" type="checkbox"/> Thu <input checked="" type="checkbox"/> Fri <input type="checkbox"/> Sat <input type="checkbox"/> Sun <i>We recomend to select working days only</i>																																																								
Number of Days Back in History <input type="text" value="30"/> <i>How long history would be included in snapshot alerts calculation</i>																																																								
STATEMENTS SETTINGS																																																								
Number of Top Queries to check <input type="text" value="10"/> chosen by <input type="text" value="Elapsed time"/> <i>How many top statements from each snapshot would be check by Alert Engine</i>																																																								
Number of Days Back in History <input type="text" value="7"/> <i>How long statement history would be included in snapshot alerts calculation</i>																																																								
WAIT EVENTS SETTINGS																																																								
Number of Top Waits to check <input type="text" value="5"/>																																																								
Number of Days Back in History <input type="text" value="7"/> <i>How long wait history would be considered in snapshot alerts calculation</i>																																																								
<input type="button" value="Save settings"/>																																																								

- **Elapsed Time greater than** – alerts will be calculated when in a given snap-time the duration for all queries exceeds 400 seconds

- **History Days** – define the days of the week that will be considered when performance problems are checked.
- **Number of Days Back in History** – The number of historical days based on the system will test the performance of the current day.

Statements Settings:

- **Number of Top Queries to check** – the number of top queries in individual snaps to be tested for performance problems, **Chosen by Elapsed Time** - the choice the Elapsed Time queries statistics will be selected
- **Number of Days Back in History** – The number of historical days based on system will analyze the performance of top queries on the current day.

Wait Events Settings:

- **Number of Top Waits to check** – used to handle waits calculated based on the trend. The number of top waits depends on this parameter is taken into account for the calculation.
- **Number of Days Back in History** - how many days back, are considered for the calculation of history.

6.7.5.3 „Alerts definition” Tab

Define alerts in the application has been divided into two stages:

- selection and configuration of appropriate CRITICAL / WARNING thresholds for a given type of alert,
- a rule definition based on configured alerts, and the attribution of the cause of the problem.

Website displays the information in columns:

- type of alert,
- description of the alert,
- availability,
- warning level,
- critical level.

ALERTS CONFIGURATION					Add new alert
Alert type	Alert description	Enabled	Level value WARNING	Level value CRITICAL	
Online	Alert if instance is not available	<input checked="" type="checkbox"/>			
Online	Total Waits	<input checked="" type="checkbox"/>	200 %	400 %	
Online	Lock waits	<input checked="" type="checkbox"/>	200 %	400 %	
Load Trends	Elapsed Time	<input checked="" type="checkbox"/>	50 %	100 %	
Load Trends	Wait time	<input checked="" type="checkbox"/>	50 %	100 %	
Load Trends	Lock time	<input checked="" type="checkbox"/>	20 %	50 %	
Sql Query	New Statement Elapsed Time	<input checked="" type="checkbox"/>	20 %	50 %	
Sql Query	Blocks write time	<input checked="" type="checkbox"/>	50 %	100 %	

The website presents only alerts that have been added to the configuration. If the alert has not been configured, please add it when use the **[Add new alert]** button.

Alerts can be configured for all instance or for a dedicated instance. At any time, user can delete the previously configured alert by use the [Key] button and select an option "Delete", this will delete the given alert from the configured list.

The second option is to disable the alert by unmark the “**Enabled**” checkbox. This can also be done by press the [Key] button and select the Edit option.

Depend on the type of alert, threshold values are set in various ways.

Collect data about problems in the application has been divided into 5 alert categories:

- **Online alerts** - calculated every 30 seconds,
- **Load Trends alerts** - calculated every 15 minutes based on general performance statistics,
- **Alerts type IO Stats** - calculated every 15 minutes based on read / write statistics from / to disk devices,
- **Sql Query alerts** - calculated every 15 minutes based on statistics of top queries,

Alerts can be defined at the general level (for all bases) and at the level of individual databases. Two alarm thresholds can be defined for each alert:

- **WARNING** event - warning alert level
- **CRITICAL** event - high alert level - critical alert

For example: setting for the Load Trends category for the Elapsed Time alert.

Load Trends	Cpu Time		<input checked="" type="checkbox"/>	50 %	100 %
-------------	----------	--	-------------------------------------	------	-------

If the Elapsed time utilization of the server exceeds 50%

- generate an alert at the warning level,

If the Elapsed time utilization of the server exceeds 100%

- generate a critical alert

In other cases, there is no alert.

Main list of alerts is presented below:

ALERTS CONFIGURATION

Alert type	Alert description	Enabled	Level value WARNING	Level value CRITICAL
IO Stats	Single Block Write time		20 %	50 %
Load Trends	Elapsed Time		50 %	100 %
Load Trends	Wait Time		30 %	80 %
Load Trends	Lock Time		20 %	50 %
Load Trends	Wait Event Time - [log file sync]		50 %	100 %
Load Trends	Cpu Time		50 %	100 %
Load Trends	Wait Event Time - [TCP Socket%]		50 %	100 %
Load Trends	Wait Event Time - [db file sequential read]		50 %	100 %

INSTANCE ALERTS CONFIGURATION - PLEASE SELECT A DATABASE: EBAZY (1 alert/s overwritten)

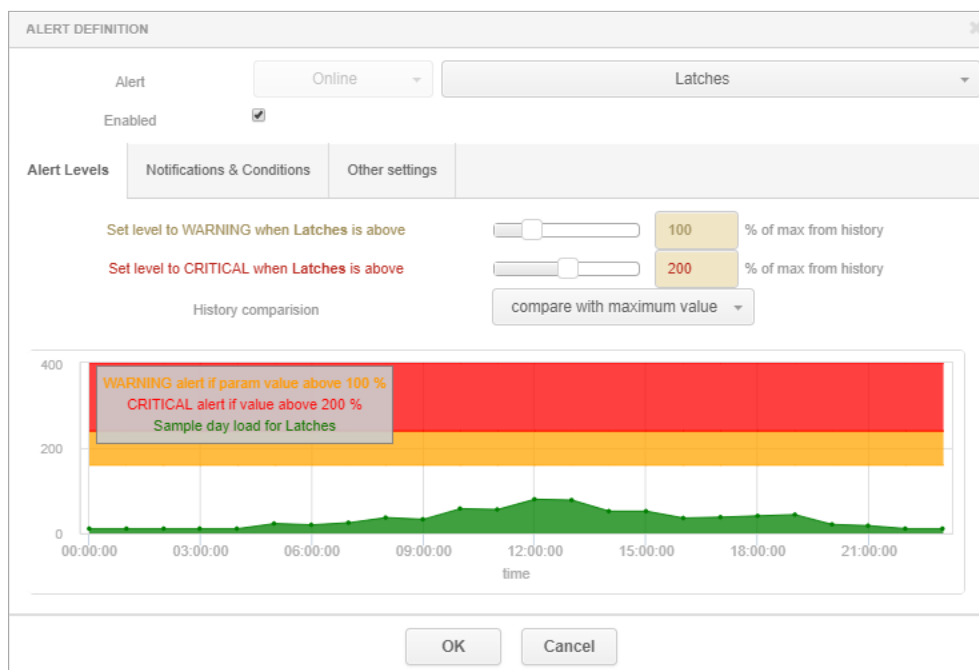
Alert type	Alert description	Enabled	Override	Level value WARNING	Level value CRITICAL
Online	Alert if database is not available		<input checked="" type="checkbox"/>		
Online	Total Waits		<input type="checkbox"/>	200 %	400 %

6.7.5.3.1 Online Alerts

The Online list includes alerts:

- **Active Sessions** – number of active sessions
- **Alert if database is not available.**
- **IO Waits** – waits related to IO readings
- **Number of Active Sessions with Elapsed Time longer than** – the number of active sessions with a duration longer than ... seconds
- **Lock waits** – lock type expectations.
- **Custom alert calculated based on sql statement** – an alert calculated based on a freely arranged query
- **Specific Waits** – the ability to indicate selected wait (any)
- **Total Waits** – all expectations together.
- **Specific Wait** – an alert for a specific expectation

The example alert tab of the alert:



Note! The field specify the type of alert (Online, Load Trends, IO Stats, Sql Query) is changeable only when new definition is created. When re-editing the alert, the field is in read-only mode. Depend on the rule chosen, the list of available and required fields to be completed is changed.

For the alert: **Specific Wait** should be completed - the name of the wait for which the alert should react.

Example will appear in the presented example:

an alert warning when the sum of expectations with a name contain reads exceeds at least 4 seconds / 1 second (a valid alert is not calculated here in percent).

critical alert when the sum of expectations with the name contain reads exceeds at least 10 seconds / 1 second (a valid alert is not calculated here in percent).

For the alert: **Custom alert calculated based on sql statement**, enter the query text.

IMPORTANT: the query must return a single-column record. The alert will occur when the value returned by the query exceeds the thresholds accord to the given definition.

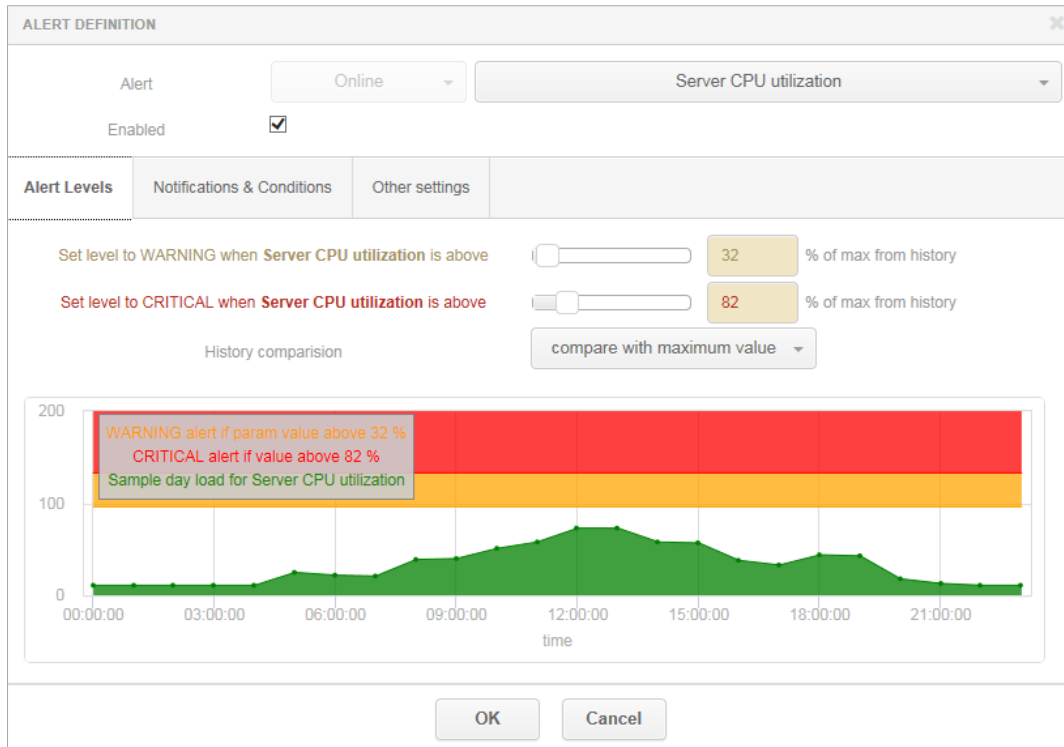
An example will appear in the presented example:

- alert warning when the number of inactive sessions with an open transaction in the database exceeds at least 10 sessions

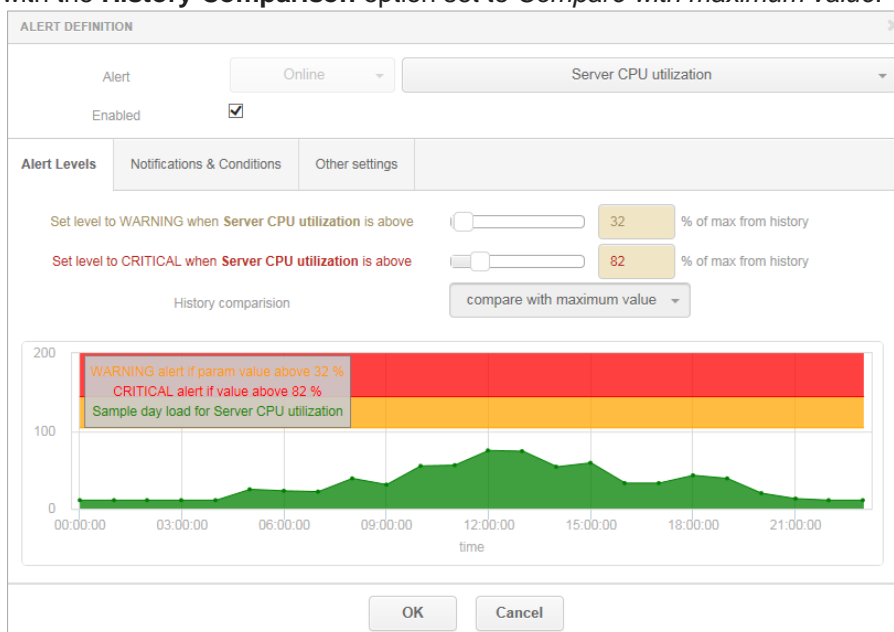
For the alert: Online:IO waits, defines standard parameters, i.e.

- Alert thresholds **WARNING**, **CRITICAL**
- The way of calculate and reaction of the History Comparison event (comparison of the performance of a given parameter with the history)
 - Compare to average value in similar time - the performance of a parameter is compared to the statistics history at similar times
 - Compare with maximum value - the performance of a parameter is compared with the maximum values that were present for a given statistic.

The screen with the option of **History Comparison** set to *Compare to average value in similar time*:



An example with the **History Comparison** option set to *Compare with maximum value*:



This slide will showcase:

Alert warning when the disposal of server processors will be 47% greater than the maximum historical value.

Critical alert when the utilization of server processors will be 90% greater than the maximum historical value.

In the alert edit tab, additional settings can be found in the **Notification & Condition** tab:

- **Mail Notification Interval** - how often to generate an email notification when an alert occurs
- **Number of snapshots to check** - the number of 30 seconds of snapshots in which there must be a "problem" for a given parameter. If a given statistic, e.g. Total Waits - stays at a high level and exceeds the alert threshold by X snapshots, then the system will generate an alert
- **Use Low Constant Value** - the minimum value that must be met first. According to the example screen below - within the dashboard snapshot (started in a 30-second cycle) the value of all wait-time must be at least 30 seconds.
- **Use High Constant Value** - the value at which the alert will always be generated, even if the WARNING, CRITICAL alert thresholds are not met.

Alert Levels	Notifications & Conditions	Other settings
	Alert Calculation Interval	once per 30 seconds
	Mailing Notification Interval	once per 5 minutes
Filter conditions		
	Use Low Constant Value	30 s. Every alert with value below entered will be skipped
	Use High Constant Value	60 s. Every alert with value above entered will be shown
Snapshot conditions		
	Number of snapshot to check	5 in which property must exceed alert level value

6.7.5.3.2 Load Trends, I/O Stats Alerts type

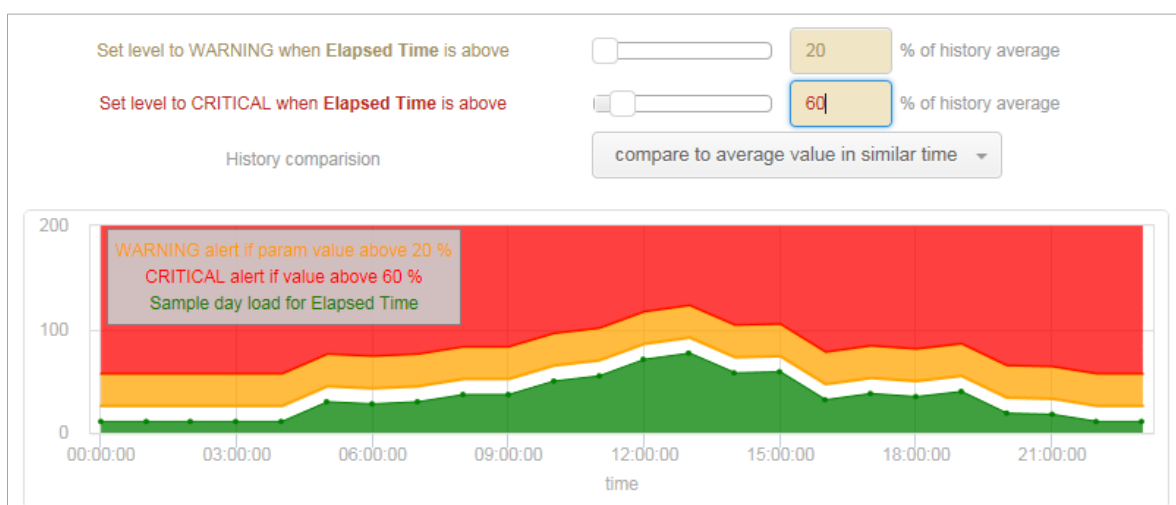
The Load Trends, I / O Stats rules refer to performance indicators available on website (functionalities) with the same names.

The edit tab for this alert:

The user specifies in the form:

- Alert type (according to the indicators given above)
- Is enabled
- Own name - **Other settings tab**
- Message format – **Other settings tab**
- E-mail settings – spam protection in case of an ongoing alert –**Notification & Conditions tab**
- **When and with what threshold an alert will occur:**
 - The rule is calculated as a percentage.
 - The alert will occur when the given alert threshold is exceeded by X% in relation to the average over the past period.
 - In the **Filter condition** section, user have additional filter settings, i.a.:
 - **Use Low Constant Value** – e.g., alert when Elapsed Time will deteriorate from X% in relation to the average, but in a situation where Elapsed Time is greater than 500 seconds.
 - **Use High Constant Value** - as above

Below are some examples of definitions for the **Elapsed Time** parameter - with the option of **History Comparison** set to Compare to average value in similar time:



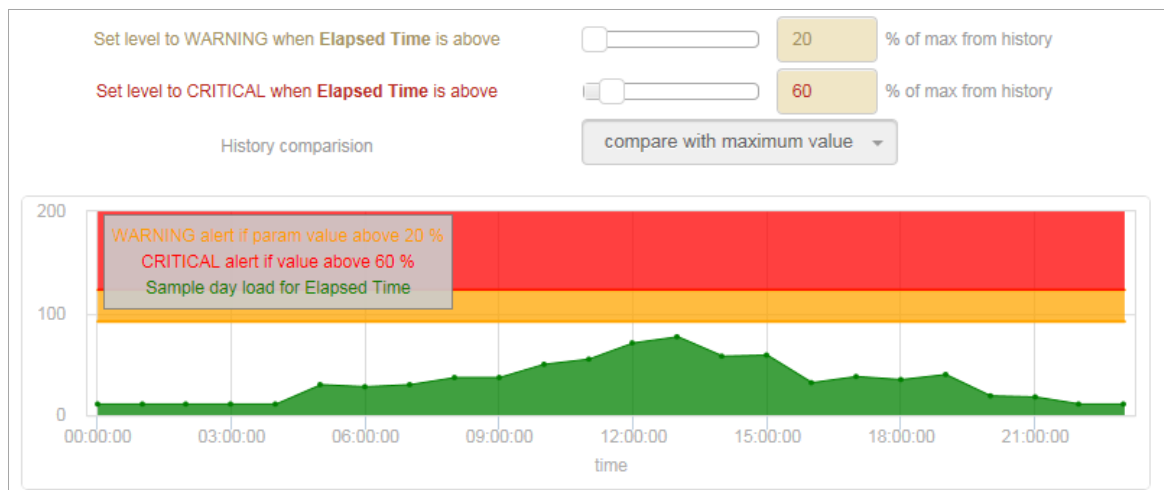
As above, the other load occurs during business hours and outside business hours. For example:

The duration of all queries, i.e. Elapsed Time at 08:00, is historically 1000 s in a 15-minute snapshots.

The duration of all queries, i.e. Elapsed Time at 12:00, is historically 5000 seconds in a 15-minute snapshots.

Alert warning type WARNING for a defined threshold $\geq 20\%$ will occur at 08:00, when the duration of all queries exceeds 1200 seconds, while around 12:00, when Elapsed time exceeds 6000 seconds.

For the second case with the History Comparison option set to Compare with maximum value:



In this example:

WARNING for the defined threshold $\geq 20\%$ will occur only if the duration of all queries exceeds 6000 seconds (reference to the maximum value of the day) regardless of the time of day.

6.7.5.3.3 Sql Query alerts type

SQL Query rules apply to performance indicators available for SQL queries and contain a similar list as for Load Trends.

For SQL queries, the system allows users to alter indicators:

- Execution
- Elapsed Time
- Elapsed Time Per 1 Exec
- New Statement Elapsed Time
- Rows
- Blocks hit
- Blocks read
- Blocks dirtied
- Blocks written
- Temp blocks read
- Temp blocks written
- Blocks read time
- Blocks write time

The SQL Query Alert Definition tab:

In the form, the user specifies similar parameters as in the alert definition for Load Trends statistics, IO Stats. In addition, user can indicate whether the alert reacts only when the execution plan is changed - the **Show Plan Changes Only** flag (if the indicator has deteriorated in relation to the history).

For example, the alert definition for e.g. Elapsed Time with the change plan check option enabled will be presented and configured separately than the Elapsed Time alert without this option selected.

Alert type	Alert description	Enabled	Level value WARNING	Level value CRITICAL
Sql Query	Execution	<input checked="" type="checkbox"/>	50 %	100 %
Sql Query	Elapsed Time (for plan changes only)	<input checked="" type="checkbox"/>	50 %	100 %
Sql Query	Elapsed Time per 1 exec (for plan changes only)	<input checked="" type="checkbox"/>	50 %	100 %
Sql Query	Disk reads (for plan changes only)	<input checked="" type="checkbox"/>	50 %	100 %
Sql Query	Execution (for plan changes only)	<input checked="" type="checkbox"/>	50 %	100 %
Load Trends	Elapsed Time	<input checked="" type="checkbox"/>	50 %	100 %
Load Trends	Wait Time	<input checked="" type="checkbox"/>	30 %	80 %

This change allows the user to more precisely define problem definitions that cause the performance of the instance to deteriorate.

For alerts with the New Statement prefix, the thresholds are determined at the level of the share in the database load.

The application allows the dependence of an alert instance on the general trend (for the entire database) for a given statistic in the snap. This option is only available for SQL Query type alerts. For the configuration shown in the picture below, this means for the SQL Query Rows processed type alarm:

- the alarm will be skipped if the value of Rows processed for a given snap for a specific Query Hash is below 10 and if the number of returned rows processed for a given query is less than 15% of all returned rows for queries (the number depends on the Number of Top Queries to

check). Additionally, the condition of exceeding the WARNING / CRITICAL alarm threshold must be met.

- the alarm will occur if the value of Rows processed for the given snap in the query is above 25%. The alarm will occur even if the alarm threshold has not been exceeded (then WARNING will occur with the Above max constant comment ...).

ALERT DEFINITION

Alert: Sql Query, Rows processed

Enabled:

Alert Levels | Notifications & Conditions | Other settings

Alert Calculation Interval: once per 15 minutes

Filter conditions

Use Low Constant Value: 10 (Every alert with value below entered will be skipped)

Use High Constant Value: 25 (Every alert with value above entered will be shown)

Query impact on load is above: 15 %

OK Cancel

6.7.5.3.4 Alert setting at the PostgreSQL Instance level

The list of alerts can be set for each base independently. Alerts are inherited from general settings by default. If any alert parameter is changed then the information appears in the Override column about overriding this rule.

As the example below:

ALERTS CONFIGURATION					Add new alert
Alert type	Alert description	Enabled	Level value WARNING	Level value CRITICAL	
Online	Alert if database is not available	<input checked="" type="checkbox"/>			
Online	Number of active sessions with Elapsed time longer than 0,03 seconds	<input checked="" type="checkbox"/>	2	5	

List of alerts on the instance level which are specific for particular database. Below settings overwrite main configuration. Those alerts which are marked in light gray color, are inherited from main configuration

INSTANCE ALERTS CONFIGURATION - PLEASE SELECT AN INSTANCE					Add new alert	Restore defaults
Alert type	Alert description	Enabled	Override	Level value WARNING	Level value CRITICAL	
Online	Alert if database is not available (test)	<input type="checkbox"/>	<input checked="" type="checkbox"/>			
Online	Total Waits	<input checked="" type="checkbox"/>	<input type="checkbox"/>	200 %	400 %	

The system will generate an alert for all instances except this one. At the indicated Instance Alert level, *Alert if database is not available*, has been disabled (Enabled = false).

6.7.5.4 „Reasons and Problems definition” Tab

The next stage of alarm configuration consists of assigned rules and defined the dedicated cause of the problem. Screen below shows an example of a list of alarms defined by DBPLUS analysts by default. Definitions can be assigned at a general level to all databases or create dedicated definitions for selected databases.

Mail settings | General settings | Alerts definition | **Reasons & Problems definition** | Events subscription

[Refresh](#)

List of performance problems which apply to all oracle databases. Please be aware that Online Issues are calculated every 30 seconds other problems every 15 minutes. Any changes in below lists are recognizes by DBPLUS.Catcher monitoring service up to 15 minutes

[Add new definition](#)

Type	Class	Reason/Problem description	Enabled	
Trends	Lock	Problems cause locking wait	<input checked="" type="checkbox"/>	Trends:Lock Time AND Trends:Wait Event Time
Trends	I/O	Problems with Disk reads increase cause query change plan	<input checked="" type="checkbox"/>	(Trends:Cpu Time AND Trends:Elapsed Time) AND ((SQLQuery:Cpu Time (for plan changes only) AND SQLQuery:Cpu Time pe
Trends	Other	Problems with Query CPU Time Increase cause query change plan	<input checked="" type="checkbox"/>	Trends:Cpu Time AND (SQLQuery:Cpu Time per 1 exec (for plan changes only) OR (SQLQuery:Cpu Time (for plan changes only)
Trends	Other	Problems cause Query CPU Time Increase	<input checked="" type="checkbox"/>	Trends:Cpu Time AND (SQLQuery:Cpu Time AND SQLQuery:Cpu Time per 1 exec)
Online	Online	Increase of waits events (cause of Locks) on database in last 3 minutes	<input checked="" type="checkbox"/>	Online:Lock waits
Trends	Other	Problems cause wait: PAGEIOLATCH_SH	<input checked="" type="checkbox"/>	Trends:Wait Time AND Trends:Wait Event Time - [PAGEIOLATCH_SH]
Trends	I/O	Problems cause increase Executions and Disk Reads.	<input checked="" type="checkbox"/>	(Trends:Cpu Time AND Trends:Elapsed Time) AND ((SQLQuery:Cpu Time AND SQLQuery:Cpu Time per 1 exec AND SQLQuer

To add a new rule, first define the reason for the problem (Reason description) for which the rule will be defined. Next, choose the type of calculation (Calculation type) - based on the trend or online and Reason class.

REASON DEFINITION

Reason description: Network problem not caused by I/O disk storage issues

Calculation Type: Based on Trends

Reason Class: I/O

Enabled:

Rules & Formulas

AND OR

Trends:Wait Event Time - [TCP Socket%] Delete

AND OR

AND OR

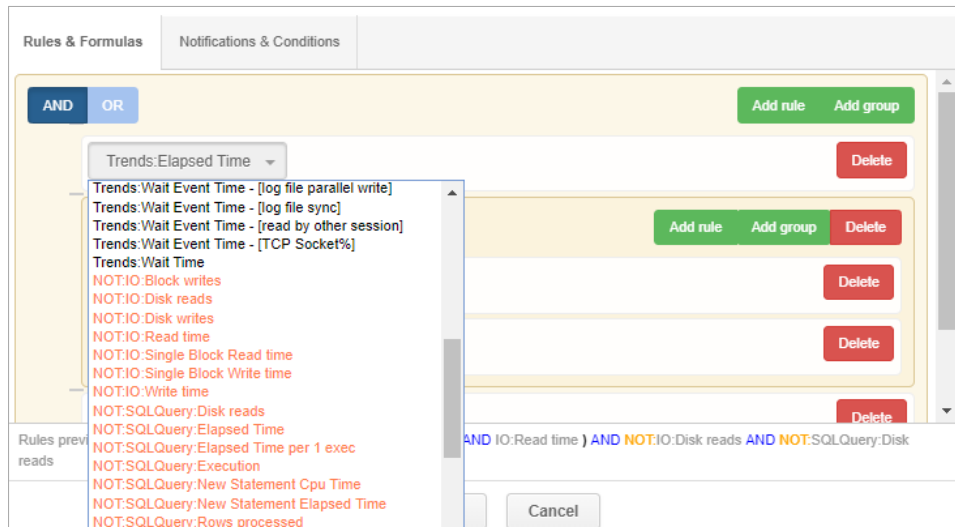
NOT:IO:Disk reads Delete

NOT:IO:Single Block Read time Delete

Rules preview: Trends:Wait Event Time - [TCP Socket%] AND ((NOT:IO:Disk reads AND NOT:IO:Single Block Read time) OR (NOT:IO:Disk writes AND NOT:IO:Single Block Write time))

OK Cancel

The most important element of the configuration is to create the cause of the problem and then define the appropriate rules based on alerts. To add a configuration, from the previously defined alerts (Alerts definition tab), create a rule using groups (Add group), AND, OR operators. In some cases, it is necessary to use negation, they are presented in the list of alerts marked in red and start with the NOT operator.

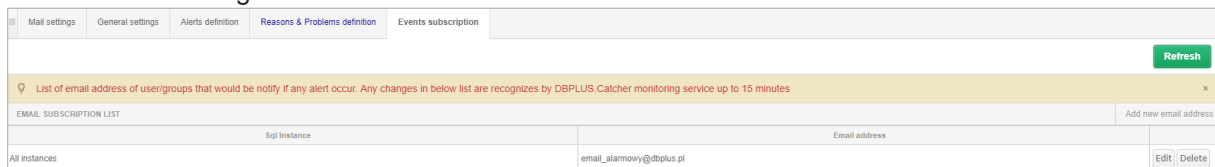


When user define the rule, correctly select the operators and complete all added alarms, the rule will be displayed below.

```
Rules preview: ( Trends:Elapsed Time AND Trends:Wait Time AND Trends:Execution ) AND ( NOT:SQLQuery:New Statement Cpu Time OR NOT:SQLQuery:New Statement Elapsed Time ) AND NOT:IO:Single Block Read time AND SQLQuery:Elapsed Time
```

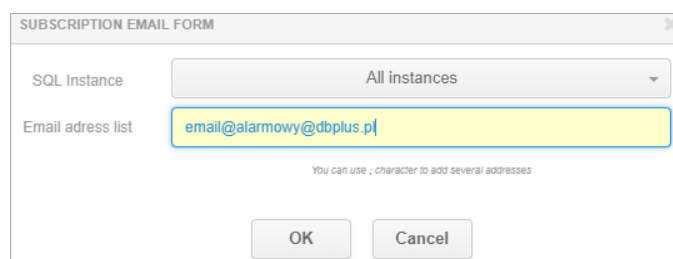
6.7.5.5 „Events subscription” Tab

In the last tab of the module user has the ability to manage the list of recipients, i.a. people who will receive alert messages.



On the subscribers list can:

- a single email address or multiple addresses separated by a separator
- assigned recipient's email address to all or selected databases.



6.7.5.6 Visibility of alerts

Alerts are visible from the Anomaly Monitor menu and from the:

- **Dashboard Level:**
 - the base icon contains information about the number of alert and critical alerts
 - after select a given SQL instance in the **Alerts** and **Instance Load** tab
- after click **[Instance Analysis]** on the **Instance Load** graph
 - if any Alert have occurred on the Elapsed Time line, relevant information is displayed about their number
- after click on a given time point (snapshot) - a list of alerts is displayed

6.8 *Help Menu*

The site contains information about licenses and information about changes in applications made in the last year.

DBPLUS ul. Bruna 9/215; 02-594 Warszawa
Tel: (+48) 22 389-73-24; e-mail: info@dbplus.pl
<http://www.dbplus.pl>

© 2018 **DBPLUS**. All rights reserved. DBPLUS, DBPLUS logo and products mentioned in this document are trademarks of DBPLUS. All other trademarks and registered trademarks are property of their respective owners.