



EDB Postgres Enterprise Manager

Version 8.0.1

1	PEM Administrator's Guide	8
1.1	PEM Overview	8
1.2	Registering a Server	9
1.3	Defining and Monitoring Postgres instances on AWS	21
1.4	Managing Certificates	23
1.5	Managing Configuration Settings	27
1.6	Managing a PEM Server	28
1.7	Managing a PEM Agent	46
2	PEM Agent User Guide	53
2.1	PEM Architecture	53
2.2	Registering an Agent	57
2.3	Managing a PEM Agent	60
2.4	PEM Agent Troubleshooting	67
3	PEM BART Management Guide	68
3.1	Managing a BART Server	68
4	PEM Enterprise Features Guide	79
4.1	What's New	79
4.2	The PEM Query Tool	79
4.3	The PEM Schema Diff Tool	91
4.4	Performance Monitoring and Management	95
4.5	Capacity Manager	146
4.6	Audit Manager	150
4.7	Log Manager	158
4.8	SQL Profiling and Analysis	169
4.9	Tuning Wizard	175
4.10	Postgres Expert - Best Practice Enforcement	179
4.11	Reports	182
4.12	Monitoring Failover Manager	186
4.13	Monitoring an xDB Replication Cluster	188
4.14	Performance Diagnostic	190
4.15	Reference	194
5	PEM Installation Guide on Linux	218
5.1	What's New	218
5.2	PEM - Hardware and Software Requirements	219
5.3	PEM Architecture	220
5.4	Installing Postgres Enterprise Manager	224
5.4.1	Prerequisites for Installing the PEM Server on Linux Platforms	224
5.4.2	Web Server Hosting Preferences	226
5.4.3	Installing the PEM Server on Linux Platforms	226
5.4.4	Creating a PEM Repository on an Isolated Network	229
5.4.5	Configuring the PEM Server on Linux Platforms	230
5.4.6	Installing a PEM Agent on Linux Platforms	232
5.4.7	Registering a PEM Agent	235
5.5	The PEM Web Interface	237
5.6	Uninstalling Postgres Enterprise Manager Components	240
5.7	Reference - Linux Service Script	241
6	PEM Installation Guide on Windows	241
6.1	What's New	242

6.2	PEM - Hardware and Software Requirements	242
6.3	PEM Architecture	243
6.4	Installing Postgres Enterprise Manager	247
6.4.1	Installing the PEM Server on Windows	247
6.4.2	Installing a PEM Agent on Windows	275
6.5	The PEM Web Interface	282
6.6	Uninstalling Postgres Enterprise Manager Components	285
7	Postgres Enterprise Manager	285
7.1	PEM Getting Started	285
7.1.1	PEM Architecture	286
7.1.2	PEM Server Logon	290
7.1.3	Managing Configuration Settings	291
7.1.4	Roles for managing PEM	292
7.1.5	The Group Dialog	294
7.1.6	Automatic Discovery of Servers	295
7.1.7	Defining a server	296
7.1.8	Defining and Monitoring Postgres instances on AWS	301
7.1.9	Connect to server	303
7.1.10	Controlling a Server	304
7.1.11	Connection error	304
7.2	Managing a PEM Agent	306
7.2.1	PEM Agent Properties	306
7.2.2	Binding an Agent to a Server	308
7.2.2.1	PEM Agent Configuration Parameters	310
7.2.2.2	High Availability Integration	314
7.2.3	Controlling the PEM Agent	315
7.2.4	High Availability Integration	316
7.2.5	PEM Agent Privileges	316
7.2.6	PEM Agent Configuration Parameters	318
7.2.7	PEM Agent Self Registration	323
7.2.8	Register/Unregister database server using PEM Agent	324
7.3	The PEM Client	326
7.3.1	PEM Main Browser Window	327
7.3.2	Browser Toolbar	335
7.3.3	The PEM Menu Bar	336
7.3.4	PEM Client Preferences	341
7.3.5	Keyboard Shortcuts	354
7.3.6	Search objects	358
7.4	Enterprise Management Features	358
7.4.1	Dashboards	359
7.4.1.1	The Alerts Dashboard	361
7.4.1.2	The Audit Log Analysis Dashboard	364
7.4.1.3	The Database Analysis Dashboard	365
7.4.1.4	The Global Overview Dashboard	368
7.4.1.5	The I/O Analysis Dashboard	370
7.4.1.6	The Memory Analysis Dashboard	373
7.4.1.7	The Objects Activity Analysis Dashboard	374
7.4.1.8	The Operating System Analysis Dashboard	377

7.4.1.9	The Probe Log Analysis Dashboard	380
7.4.1.10	The Server Analysis Dashboard	381
7.4.1.11	The Server Log Analysis Dashboard	384
7.4.1.12	The Session Activity Analysis Dashboard	385
7.4.1.13	The Session Waits Analysis Dashboard	387
7.4.1.14	The Storage Analysis Dashboard	388
7.4.1.15	The System Wait Analysis Dashboard	390
7.4.1.16	The Streaming Replication Analysis Dashboard	391
7.4.2	Server Configuration	394
7.4.2.1	Configuration Options	395
7.4.3	Log Manager	403
7.4.4	Audit Manager	411
7.4.5	Postgres Log Analysis Expert	415
7.4.6	Tuning Wizard	422
7.4.7	Postgres Expert	427
7.4.7.1	Schema Expert Recommendations	430
7.4.7.2	Security Expert Recommendations	432
7.4.7.3	Configuration Expert Recommendations	433
7.4.8	Capacity Manager	439
7.4.8.1	Tab 1 (Metrics)	444
7.4.8.2	Tab 2 (Options)	445
7.4.9	Alerting	447
7.4.9.1	Creating and Managing Alerts	450
7.4.9.2	Copy Alerts	459
7.4.9.3	Alert Templates	459
7.4.9.4	Custom Alert Templates	468
7.4.9.5	Email Groups	472
7.4.9.6	Webhooks	474
7.4.9.7	SNMP MIB Generation	477
7.4.9.8	SNMP Trap Details	478
7.4.9.9	Using PEM with Nagios	480
7.4.10	Using the Manage Charts Tab	483
7.4.10.1	Creating a New Chart	484
7.4.10.2	Importing a Capacity Manager Template	487
7.4.11	The PEM Manage Dashboards Tab	490
7.4.11.1	Creating a Custom Dashboard	490
7.4.11.2	Creating an Ops Dashboard	493
7.4.12	The Manage Probes Tab	493
7.4.12.1	Custom Probes	495
7.4.12.2	Copy Probe Configuration	499
7.4.12.3	Probe Configuration	500
7.4.12.3.1	Probes	502
7.4.13	Schedule Alert Blackout	507
7.4.14	Scheduled System Jobs	509
7.4.15	Scheduled Task Tab	510
7.4.16	Creating a PEM Scheduled Job	512
7.4.17	Sending email notifications for a job	518
7.4.18	Task Viewer	521

7.4.18.1	Log Viewer	522
7.4.19	Monitoring a Failover Manager Cluster	522
7.4.20	Performance Diagnostic	524
7.4.21	Reports	528
7.5	Management Basics	532
7.5.1	Grant Wizard	532
7.5.2	Add named restore point Dialog	534
7.5.3	Import/Export data Dialog	535
7.5.4	Maintain a database object	537
7.5.4.1	Maintenance Dialog	538
7.5.5	Storage Manager	539
7.5.6	Backup Dialog	541
7.5.7	Backup Globals Dialog	545
7.5.8	Backup Server Dialog	546
7.5.9	Restore Dialog	549
7.5.10	Managing Cluster Level Objects	553
7.5.10.1	Database Dialog	553
7.5.10.2	Move Objects Dialog	557
7.5.10.3	Resource Group Dialog	559
7.5.10.4	Login/Group Role Dialog	560
7.5.10.5	Tablespace Dialog	564
7.5.11	Managing Database Objects	568
7.5.11.1	Cast Dialog	568
7.5.11.2	Collation Dialog	570
7.5.11.3	Domain Dialog	572
7.5.11.4	Domain Constraints Dialog	576
7.5.11.5	Event Trigger Dialog	578
7.5.11.6	Extension Dialog	580
7.5.11.7	Foreign Data Wrapper Dialog	582
7.5.11.8	Foreign Server Dialog	585
7.5.11.9	Foreign Table Dialog	588
7.5.11.10	FTS Configuration dialog	593
7.5.11.11	FTS Dictionary Dialog	596
7.5.11.12	FTS Parser Dialog	599
7.5.11.13	FTS Template Dialog	601
7.5.11.14	Function Dialog	603
7.5.11.15	Language Dialog	607
7.5.11.16	Materialized View Dialog	610
7.5.11.17	Package Dialog	614
7.5.11.18	Procedure Dialog	617
7.5.11.19	Schema Dialog	621
7.5.11.20	Sequence Dialog	624
7.5.11.21	Synonym Dialog	627
7.5.11.22	Trigger function Dialog	629
7.5.11.23	Type Dialog	634
7.5.11.24	User Mapping Dialog	639
7.5.11.25	View Dialog	641
7.5.12	Creating or Modifying a Table	645

7.5.12.1	Check Dialog	645
7.5.12.2	Column Dialog	647
7.5.12.3	Compound Trigger Dialog	652
7.5.12.4	Exclusion constraint Dialog	655
7.5.12.5	Foreign key Dialog	658
7.5.12.6	Index Dialog	662
7.5.12.7	Primary key Dialog	665
7.5.12.8	RLS Policy Dialog	667
7.5.12.9	Rule Dialog	669
7.5.12.10	Table Dialog	672
7.5.12.11	Trigger Dialog	685
7.5.12.12	Unique Constraint Dialog	689
7.6	Managing a BART Server	691
7.6.1	Prerequisites for managing BART	692
7.6.2	Configuring a BART Server	692
7.6.3	Associating the BART Server with a Database Server	695
7.6.4	Viewing the BART Server Details on a PEM Dashboard	697
7.6.5	Scheduling BART Backups	698
7.6.6	Scheduling BART Obsolete Backups Deletion	700
7.6.7	BART Backup Dialog	701
7.6.8	Restoring BART Backups	703
7.7	SQL Profiler	705
7.7.1	Installing SQL Profiler	705
7.7.2	Configuring SQL Profiler	709
7.7.3	Using SQL Profiler	710
7.7.4	Using Index Advisor	715
7.7.5	The SQL Profiler Tab	716
7.8	Developer Tools	719
7.8.1	pgAdmin Debugger	719
7.8.2	Query Tool	723
7.8.3	Interpreting Graphical Query Plans	734
7.8.4	Reviewing and Editing Data	737
7.8.4.1	View/Edit Data Filter	741
7.8.5	Schema Diff	742
7.9	Configuring pgBouncer for use with PEM Agents	746
7.9.1	Connecting PEM to pgBouncer	746
7.9.2	Preparing the PEM Server for pgBouncer Connections	748
7.9.3	Configuring pgBouncer	749
7.9.4	Configuring the PEM Agent to use pgBouncer	751
7.10	pgAgent	753
7.10.1	Using pgAgent	753
7.10.2	Installing pgAgent	754
7.10.3	Creating a pgAgent Job	756
7.10.4	pgAgent Steps	761
7.10.5	pgAgent Schedules	762
7.11	Appendices	765
7.11.1	Licence	765
7.11.2	The MIT Kerberos Licence	765

7.11.3	The OpenSSL Licence	766
7.11.4	The SNMP++ Licence	766
7.11.5	The jquery table sort Licence	767
7.12	Release Notes	767
7.12.1	PEM v8.0.1	767
7.12.2	PEM v8.0	769
7.12.3	PEM v7.16	770
7.12.4	PEM v7.15	773
7.12.5	PEM v7.14	775
7.12.6	PEM v7.13	777
7.12.7	PEM v7.12	779
7.12.8	PEM v7.11	781
7.12.9	PEM v7.10	781
7.12.10	PEM v7.9	783
7.12.11	PEM v7.8	785
7.12.12	PEM v7.7.1	787
7.12.13	PEM v7.7	787
7.12.14	PEM v7.6	789
7.12.15	PEM v7.5	791
8	EDB Postgres Enterprise Manager Configuring pgBouncer for Use with PEM Agents	792
8.1	The PEM Server - PEM Agent Connection Management Mechanism	792
8.2	Preparing the PEM Database Server	793
8.3	Configuring PgBouncer	794
8.4	Configuring the PEM Agent	796
9	PEM Security Guide	798
9.1	Apache HTTPD Security Configurations	799
9.2	PEM application Security Configurations	804
10	SQL Profiler	807
10.1	Installing the SQL Profiler Plugin	808
10.2	SQL Profiling and Analysis	813
10.3	Uninstalling SQL Profiler	819
11	PEM Upgrade and Migration	820
11.1	Upgrading a PEM Installation	820
11.1.1	Upgrading a PEM Installation on Windows Host	820
11.1.2	Upgrading a PEM Native Package Installation on a Linux Host	831
11.1.3	Creating a PEM Repository on an Isolated Network	836
11.1.4	Upgrading a Graphical PEM Installation on a Linux Host	837
11.1.5	Configuring a PEM Server on a Linux Host	845
11.2	Upgrading the PEM Backend Postgres Database	846
11.3	Moving the Postgres Enterprise Manager Server	855
11.4	Troubleshooting	863

1 PEM Administrator's Guide

This document provides an introduction to Postgres Enterprise Manager (PEM). Postgres Enterprise Manager (PEM) is an enterprise management tool designed to assist database administrators, system architects, and performance analysts in administering, monitoring, and tuning PostgreSQL and EDB Advanced Server database servers. PEM is architected to manage and monitor anywhere from a handful, to hundreds of servers from a single console, allowing complete and remote control over all aspects of your databases.

For information about the platforms and versions supported by PEM, visit the EDB website at:

<https://www.enterprisedb.com/services-support/edb-supported-products-and-platforms#pem>

Please note: PEM 8.0.1 is no longer supported on CentOS/RHEL/OEL 6.x platforms. It is strongly recommended that EDB products running on these platforms be migrated to a supported platform.

This document uses *Postgres* to mean either the PostgreSQL or EDB Postgres Advanced Server database.

1.1 PEM Overview

PEM provides a number of benefits not found in any other PostgreSQL management tool:

- **Management en Masse Design.** PEM is designed for enterprise database management, and is built to tackle the management of large numbers of servers across geographical boundaries. Global dashboards keep you up to date on the up/down/performance status of all your servers in an at-a-glance fashion.
- **Distributed Architecture.** PEM is architected in a way that maximizes its ability to gather statistical information and to perform operations remotely on machines regardless of operating system platform.
- **Graphical Administration.** All aspects of database administration can be carried out in the PEM client via a graphical interface. Server startup and shutdown, configuration management, storage and security control, object creation, performance management, and more can be handled from a single console.
- **Full SQL IDE.** PEM contains a robust SQL integrated development environment (IDE) that provides ad-hoc SQL querying, stored procedure/function development, and a graphical debugger.
- **Enterprise Performance Monitoring.** PEM provides enterprise-class performance monitoring for all managed database servers. Lightweight and efficient agents monitor all aspects of each database server's operations as well as each machine's underlying operating system and provide detailed statistics back to easily navigated performance pages within the interface.
- **Proactive Alert Management.** PEM ships out-of-the-box with the ability to create performance thresholds for each key metric (e.g. memory, storage, etc.) that are monitored around-the-clock. Any threshold violation results in an alert being sent to a centralized dashboard that communicates the nature of the problem and what actions are necessary to prevent the situation from jeopardizing the overall performance of the server.
- **Simplified Capacity Planning.** All key performance-related statistics are automatically collected and retained for a specified period of time in PEM's repository. The Capacity Manager utility allows you to select various statistics and perform trend analysis over time to understand things such as peak load periods, storage consumption trends, and much more. A forecasting mechanism in the tool allows you to also forecast resource usage in the future and

plan/budget accordingly.

- Audit Manager. The Audit Manager configures audit logging on Advanced Server instances. Activities such as connections to a database, disconnections from a database, and the SQL statements run against a database can be logged. The Audit Log dashboard can then be used to filter and view the log.
 - Log Manager. The Log Manager wizard configures server logging parameters, with (optional) log collection into a central table. Use the wizard to specify your preference for logging behaviors such as log file rotation, log destination and error message severity. Use the Server Log dashboard to filter and review the collected server log entries.
 - SQL Workload Profiling. PEM contains a SQL profiling utility that allows you to trace the SQL statements that are executed against one or more servers. SQL profiling can either be done in an ad-hoc or scheduled manner. Captured SQL statements can then be filtered so you can easily identify and tune poorly running SQL statements. SQL statements can also be fed into an Index Advisor on Advanced Server that analyzes each statement and makes recommendations on new indexes that should be created to help performance.
 - Expert Database Analysis. PEM includes the Postgres Expert utility. Postgres Expert analyzes selected databases for best practice enforcement purposes. Areas such as general configuration, security setup, and much more are examined. Any deviations from recommended best practices are reported back to you, along with an explanation of each particular issue, and expert help on what to do about making things right.
 - Streaming Replication Monitoring. You can monitor the the Streaming Replication dashboard or use options on the PEM client to promote a replica node to the primary node.
 - Secure Client Connectivity. PEM supports secure client connections through an encrypted SSH tunnel. The full-featured PEM client includes an SSH Tunnel definition dialog that allows you to provide connection information for a secure connection.
 - Wide Platform Support. PEM supports most major Linux and Windows platforms.
-

1.2 Registering a Server

Before you can manage or monitor a server with PEM, you must register the server with PEM, and bind an agent. A server may be bound to a remote agent (an agent that resides on a different host), but if the agent does not reside on the same host, it will not have access to all of the statistical information about the instance.

Manually Registering a Server

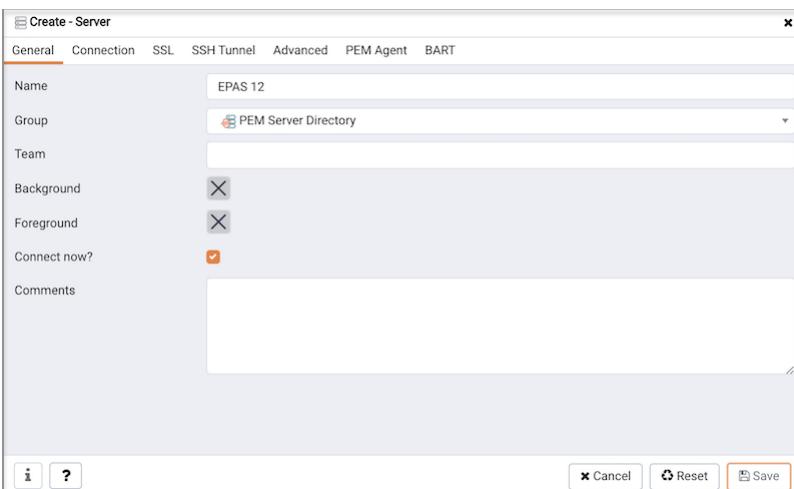
To manage or monitor a server with PEM, you must:

- Register your Advanced Server or PostgreSQL server with the PEM server.
- Bind the server to a PEM agent.

You can use the **Create – Server** dialog to provide registration information for a server, bind a PEM agent, and display the server in PEM client tree control. To open the **Create – Server** dialog, navigate through the **Create** option on the **Object** menu (or the context menu of a server group) and select **Server...**.

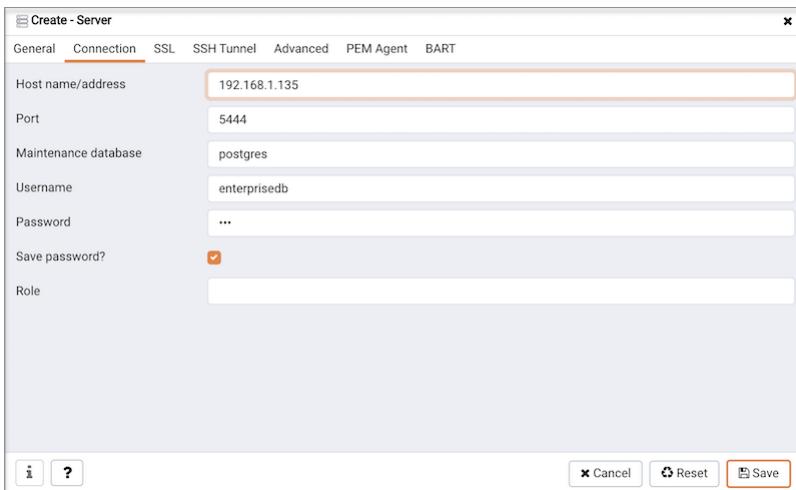


!!! Note You must ensure the `pg_hba.conf` file of the Postgres server that you are registering allows connections from the host of the PEM client before attempting to connect.



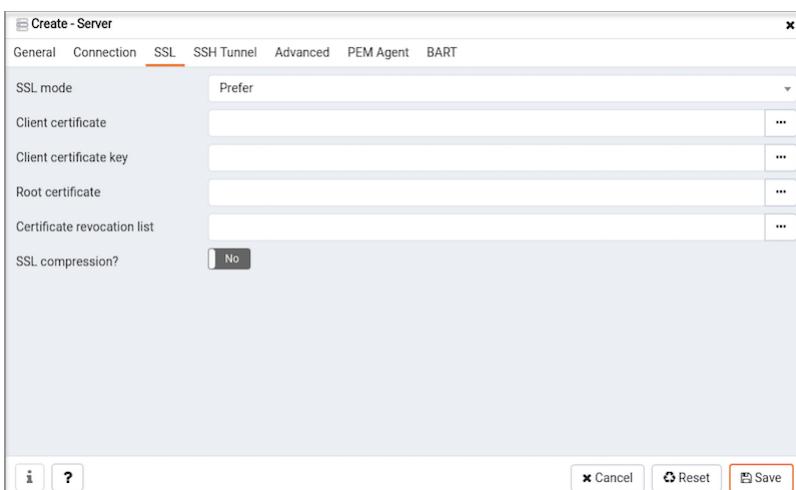
Use the fields on the **General** tab to describe the general properties of the server:

- Use the **Name** field to specify a user-friendly name for the server. The name specified will identify the server in the PEM **Browser** tree control.
- You can use groups to organize your servers and agents in the tree control. Using groups can help you manage large numbers of servers more easily. For example, you may want to have a production group, a test group, or LAN specific groups. Use the **Group** drop-down listbox to select the server group in which the new server will be displayed.
- Use the **Team** field to specify a Postgres role name. Only PEM users who are members of this role, who created the server initially, or have superuser privileges on the PEM server will see this server when they logon to PEM. If this field is left blank, all PEM users will see the server.
- Use the **Background** color selector to select the color that will be displayed in the PEM tree control behind database objects that are stored on the server.
- Use the **Foreground** color selector to select the font color of labels in the PEM tree control for objects stored on the server.
- Check the box next to **Connect now?** to instruct PEM to attempt a server connection when you click the Save button. Leave **Connect now?** unchecked if you do not want the PEM client to validate the specified connection parameters until a later connection attempt.
- Provide notes about the server in the **Comments** field.



Use fields on the **Connection tab** to specify connection details for the server:

- Specify the IP address of the server host, or the fully qualified domain name in the **Host name/address** field. On Unix based systems, the address field may be left blank to use the default PostgreSQL Unix Domain Socket on the local machine, or may be set to an alternate path containing a PostgreSQL socket. If you enter a path, the path must begin with a "/".
- Specify the port number of the host in the **Port** field.
- Use the **Maintenance database** field to specify the name of the initial database that PEM will connect to, and that will be expected to contain **pgAgent** schema and **adminpack** objects installed (both optional). On PostgreSQL 8.1 and above, the maintenance DB is normally called **postgres**; on earlier versions **template1** is often used, though it is preferable to create a **postgres** database to avoid cluttering the template database.
- Specify the name that will be used when authenticating with the server in the **Username** field.
- Provide the password associated with the specified user in the **Password** field.
- Check the box next to **Save password?** to instruct PEM to store passwords in encrypted format in PEM backend database for later reuse. Each password is stored on a per user, per server basis, and won't be shared with other team members. PEM will use the saved password to connect the database server next time. To remove a saved password, disconnect the database server first, and then use the **Clear Saved Password** menu item from the **Object/Context** menu.
- Use the **Role** field to specify the name of the role that is assigned the privileges that the client should use after connecting to the server. This allows you to connect as one role, and then assume the permissions of another role when the connection is established (the one you specified in this field). The connecting role must be a member of the role specified.



Use the fields on the **SSL** tab to configure SSL:

- Use the drop-down list box in the **SSL mode** field to select the type of SSL connection the server should use. For

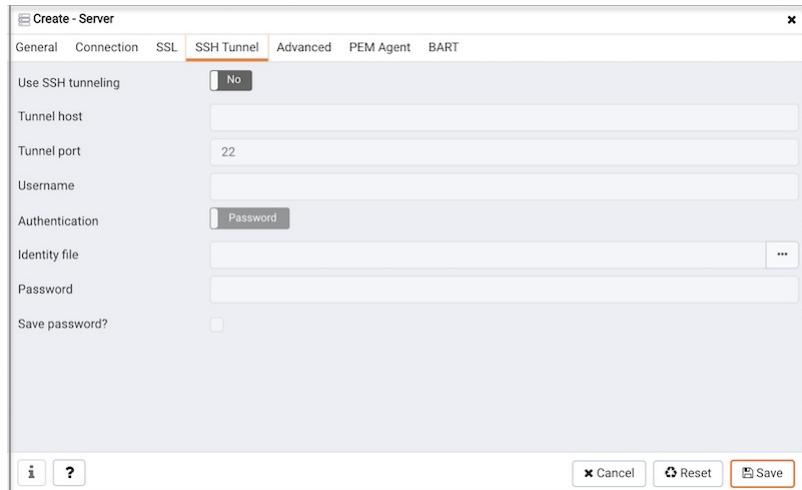
more information about using SSL encryption, see the PostgreSQL documentation at:

<https://www.postgresql.org/docs/current/static/libpq-ssl.html>

You can use the platform-specific file manager dialog to upload files that support SSL encryption to the server. To access the file manager, click the icon that is located to the right of each of the following fields:

- Use the **Client certificate** field to specify the file containing the client SSL certificate. This file will replace the default `~/.postgresql/postgresql.crt` file if PEM is installed in Desktop mode, and `<STORAGE_DIR>/<USERNAME>/postgresql/postgresql.crt` if PEM is installed in Web mode. This parameter is ignored if an SSL connection is not made.
- Use the **Client certificate key** field to specify the file containing the secret key used for the client certificate. This file will replace the default `~/.postgresql/postgresql.key` if PEM is installed in Desktop mode, and `<STORAGE_DIR>/<USERNAME>/postgresql/postgresql.key` if PEM is installed in Web mode. This parameter is ignored if an SSL connection is not made.
- Use the **Root certificate** field to specify the file containing the SSL certificate authority. This file will replace the default `~/.postgresql/root.crt` file. This parameter is ignored if an SSL connection is not made.
- Use the **Certificate revocation list** field to specify the file containing the SSL certificate revocation list. This list will replace the default list, found in `~/.postgresql/root.crl`. This parameter is ignored if an SSL connection is not made.
- When **SSL compression?** is set to **True**, data sent over SSL connections will be compressed. The default value is **False** (compression is disabled). This parameter is ignored if an SSL connection is not made.

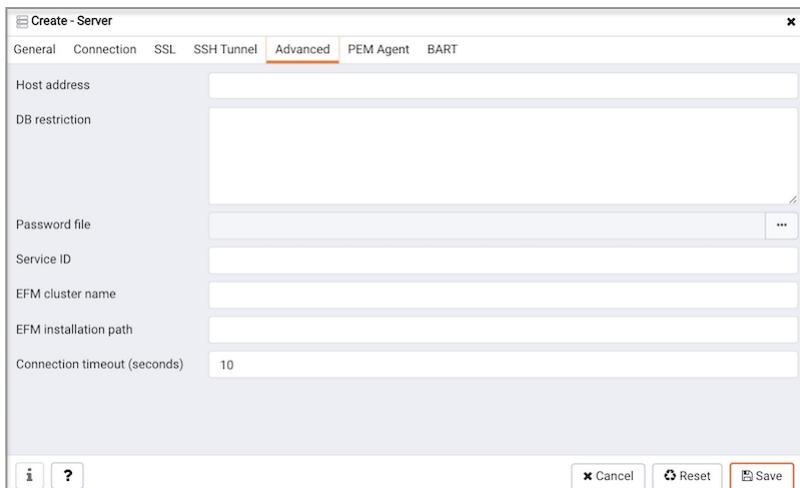
Warning: Certificates, private keys, and the revocation list are stored in the per-user file storage area on the server, which is owned by the user account under which the PEM server process is run. This means that administrators of the server may be able to access those files; appropriate caution should be taken before choosing to use this feature.



Use the fields on the **SSH Tunnel** tab to configure SSH Tunneling. You can use a tunnel to connect a database server (through an intermediary proxy host) to a server that resides on a network to which the client may not be able to connect directly.

- Set **Use SSH tunneling** to **Yes** to specify that PEM should use an SSH tunnel when connecting to the specified server.
- Specify the name or IP address of the SSH host (through which client connections will be forwarded) in the **Tunnel host** field.
- Specify the port of the SSH host (through which client connections will be forwarded) in the **Tunnel port** field.
- Specify the name of a user with login privileges for the SSH host in the **Username** field.
- Specify the type of authentication that will be used when connecting to the SSH host in the **Authentication** field.
- Select **Password** to specify that PEM will use a password for authentication to the SSH host. This is the default.
- Select **Identity file** to specify that PEM will use a private key file when connecting.

- If the SSH host is expecting a private key file for authentication, use the **Identity file** field to specify the location of the key file.
- If the SSH host is expecting a password, use the **Password** field to specify the password, or if an identity file is being used, the passphrase.



Use fields on the **Advanced** tab to specify details that are used to manage the server:

- Specify the IP address of the server host in the **Host Address** field.
- Use the **DB restriction** field to specify a SQL restriction that will be used against the `pg_database` table to limit the databases displayed in the tree control. For example, you might enter: '`'live_db'`', '`'test_db'`' to instruct the PEM browser to display only the `live_db` and `test_db` databases. Note that you can also limit the schemas shown in the database from the database properties dialog by entering a restriction against `pg_namespace`.
- Use the **Password file** field to specify the location of a password file (`.pgpass`). The `.pgpass` file allows a user to login without providing a password when they connect. It must be present on the PEM Server. For more information, see the Postgres documentation at:

<http://www.postgresql.org/docs/current/static/libpq-pgpass.html>

- Use the **Service ID** field to specify parameters to control the database service process. For servers that are stored in the Enterprise Manager directory, enter the service ID. On Windows machines, this is the identifier for the Windows service. On Linux machines, the name of the init script used to start the server is `/etc/init.d` and the name of the systemd script to start the server is `systemctl`. For example, the name of the Advanced Server 11 service is `edb-as-11`. For local servers, the setting is operating system dependent:

- If the PEM client is running on a Windows machine, it can control the postmaster service if you have sufficient access rights. Enter the name of the service. In case of a remote server, it must be prepended by the machine name (e.g. `PSE1\pgsql-8.0`). PEM will automatically discover services running on your local machine.
- If the PEM client is running on a Linux machine, it can control processes running on the local machine if you have enough access rights. Provide a full path and needed options to access the `pg_ctl` program. When executing service control functions, PEM will append status/start/stop keywords to this. For example:

`sudo /usr/pgsql-x/bin/pg_ctl -D /var/lib/pgsql/x/data` where `x` is the version of the PostgreSQL database server.

- If the server is a member of a Failover Manager cluster, you can use PEM to monitor the health of the cluster and to replace the primary node if necessary. To enable PEM to monitor Failover Manager, use the **EFM cluster name** field to specify the cluster name. The cluster name is the prefix of the name of the Failover Manager cluster properties file. For example, if the cluster properties file is named `efm.properties`, the cluster name is `efm`.

- If you are using PEM to monitor the status of a Failover Manager cluster, use the **EFM installation path** field to specify the location of the Failover Manager binary file. By default, the Failover Manager binary file is installed in `/usr/edb/efm-x.x/bin`, where `x.x` specifies the Failover Manager version.



Use fields on the **PEM Agent** tab to specify connection details for the PEM agent:

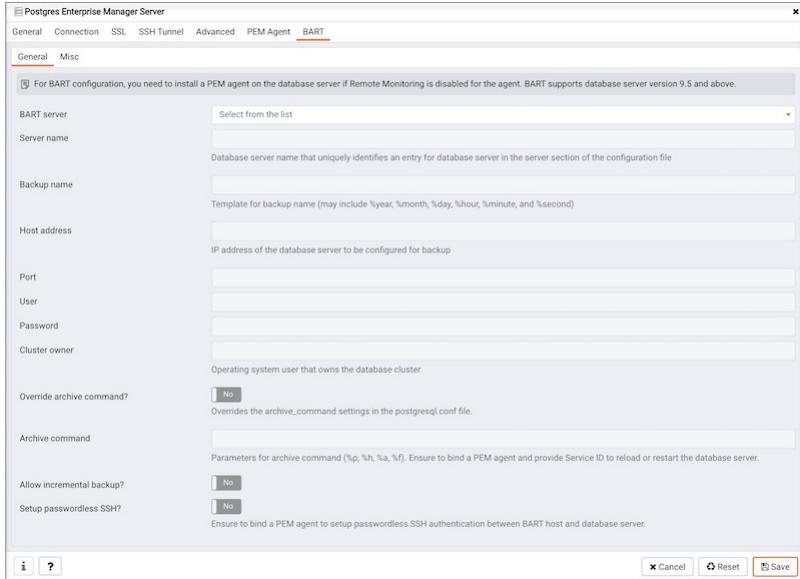
- Select an Enterprise Manager agent using the drop-down listbox to the right of the **Bound agent** label. One agent can monitor multiple Postgres servers.
- Move the **Remote monitoring?** slider to **Yes** to indicate that the PEM agent does not reside on the same host as the monitored server. When remote monitoring is enabled, agent level statistics for the monitored server will not be available for custom charts and dashboards, and the remote server will not be accessible by some PEM utilities (such as Audit Manager, Capacity Manager, Log Manager, Postgres Expert and Tuning Wizard).
- Enter the IP address or socket path that the agent should use when connecting to the database server in the **Host** field. By default, the agent will use the host address shown on the **General** tab. On a Unix server, you may wish to specify a socket path, e.g. `/tmp`.
- Enter the **Port** number that the agent will use when connecting to the server. By default, the agent will use the port defined on the **Properties** tab.
- Use the drop-down listbox in the **SSL** field to specify an SSL operational mode; specify require, prefer, allow, disable, verify-ca or verify-full. For more information about using SSL encryption, see the [PostgreSQL documentation](#).
- Use the **Database** field to specify the name of the database to which the agent will initially connect.
- Specify the name of the role that agent should use when connecting to the server in the **User name** field. Note that if the specified role is not a database superuser, then some of the features will not work as expected. For the list of features that do not work if the specified role is not a database superuser, see [Agent privileges](#).

If you are using Postgres version 10 or above, you can use the `pg_monitor` role to grant the required privileges to a non-superuser. For information about `pg_monitor` role, see [Default Roles](#).

- Specify the name of the user that agent should use when connecting to the server in the **User name** field. Note that if the specified user is not a database superuser, then some of the features will not work as expected. If you are using Postgres version 10 or above, you can use the `pg_monitor` role to grant the required privileges to a non-superuser. For information about `pg_monitor` role, see [Default Roles](#).
- Specify the password that the agent should use when connecting to the server in the **Password** field, and verify it by typing it again in the **Confirm password** field. If you do not specify a password, you will need to configure the authentication for the agent manually; for example, you can use a `.pgpass` file, and it must be present and

accessible on the system, where agent is installed.

- Set the **Allow takeover?** slider to **Yes** to specify that the server may be taken over by another agent. This feature allows an agent to take responsibility for the monitoring of the database server if, for example, the server has been moved to another host as part of a high availability failover process.



Use the fields on the **General** tab under **BART** tab to describe the general properties of the BART Server that will map to the PEM server:

- Use the **BART server** field to select the BART server name. All the BART servers configured in the PEM console will be listed in this drop down list.
- Use the **Server name** field to specify a name for the database server that you want to backup using the BART server. This name gets stored in the BART configuration file.
- Use the **Backup name** field to specify a template for user-defined names to be assigned to the backups of the database server. If you do not specify a backup name template, then the backup can only be referenced in BART sub-commands by the BART-assigned integer backup identifier.
- Use the **Host address** field to specify the IP address of the database server that you want to configure for backup.
- Use the **Port** field to specify the port to be used for the database that you want to backup.
- Use the **User** field to specify the user of the database that you want to backup using BART through PEM console. If you want to enable incremental backups for this database server, then the user must be a superuser.
- Use the **Password** field to specify the password for the user of the database that you want to backup.
- Use the **Cluster Owner** field to specify the Linux operating system user account that owns the database cluster. This is typically **enterprisedb** for Advanced Server database clusters installed in the Oracle databases compatible mode, or **postgres** for PostgreSQL database clusters and for Advanced Server database clusters installed in the PostgreSQL databases compatible mode.
- Use the **Archive command** field to specify the desired format of the archive command string to be used in the **bart.cfg** file. Inputs provided for the Archive command will overwrite the database server's **Postgresql.conf** file. Once the server gets added, the database server will be restarted or database configurations will be reloaded.
- Use the **Allow incremental backup?** switch to specify if incremental backup should be enabled for this database server.
- Use the **Setup passwordless SSH?** switch to specify if you want to create SSH certificates to allow passwordless logins between the Database Server and the BART server. Ensure to bind a PEM agent before setting up the passwordless SSH authentication. Passwordless SSH will not work for a database server being remotely monitored by a PEM agent.



Use the fields on the **Misc** tab under **BART** tab to describe the miscellaneous properties of the BART Server:

- Use the **Override default configuration?** Switch to specify if you want to override the BART server configurations with the specific database server configurations.
- Use the **Xlog** method to specify how the transaction log should be collected during the execution of `pg_basebackup`.
- Use the **Retention policy** field to specify the retention policy for the backup. This determines when an active backup should be marked as obsolete, and hence, be a candidate for deletion. You can specify the retention policy in terms of number of backup or in terms of duration (days, weeks, or months).
- Use the **WAL compression** switch to specify if you want to compress the archived Xlog/WAL files in Gzip format. To enable WAL compression, the gzip compression program must be present in the BART user account's PATH. The `wal_compression` setting must not be enabled for those database servers where you need to take incremental backups.
- Use the **Copy WALs during restore** field to specify how the archived WAL files are collected when invoking the RESTORE operation. Set to enabled to copy the archived WAL files from the BART backup catalog to the `<restore_path>/archived_wals` directory prior to the database server archive recovery. Set to disabled to retrieve the archived WAL files directly from the BART backup catalog during the database server archive recovery.
- Use the **Thread count** field to specify the number of threads to copy the blocks. You must set `thread count` to `1` if you want to take a backup with the `pg_basebackup` utility.
- Use the **Batch size** field to specify the number of blocks of memory used for copying modified blocks, applicable only for incremental backups.
- Use the **Scan interval** field to specify the number of seconds after which the WAL scanner should scan the new WAL files.
- Use the **MBM scan timeout** field to specify the number of seconds to wait for MBM files before timing out, applicable only for incremental backups.

To view the properties of a server, right-click on the server name in the PEM client tree control, and select the **Properties...** option from the context menu. To modify a server's properties, disconnect from the server before opening the **Properties** dialog.

Automatic Server Discovery

If the server you wish to monitor resides on the same host as the monitoring agent, you can use the **Auto Discovery** dialog to simplify the registration and binding process.

To enable auto discovery for a specific agent, you must enable the **Server Auto Discovery** probe. To access the

Manage Probes tab, highlight the name of a PEM agent in the PEM client tree control, and select **Manage Probes...** from the **Management** menu. When the **Manage Probes** tab opens, confirm that the slider control in the **Enabled?** column is set to **Yes**.

To open the **Auto Discovery** dialog, highlight the name of a PEM agent in the PEM client tree control, and select **Auto Discovery...** from the **Management** menu.



When the **Auto Discovery** dialog opens, the **Discovered Database Servers** box will display a list of servers that are currently not being monitored by a PEM agent. Check the box next to a server name to display information about the server in the **Server Connection Details** box, and connection properties for the agent in the **Agent Connection Details** box.

Use the **Check All** button to select the box next to all of the displayed servers, or **Uncheck All** to deselect all of the boxes to the left of the server names.

The fields in the **Server Connection Details** box provide information about the server that PEM will monitor:

- Accept or modify the name of the monitored server in the **Name** field. The specified name will be displayed in the tree control of the PEM client.
- Use the **Server group** drop-down listbox to select the server group under which the server will be displayed in the PEM client tree control.
- Use the **Host name/address** field to specify the IP address of the monitored server.
- The **Port** field displays the port that is monitored by the server; this field may not be modified.
- Provide the name of the service in the **Service ID** field. Please note that the service name must be provided to enable some PEM functionality.
- By default, the **Maintenance database** field indicates that the selected server uses a Postgres maintenance database. Customize the content of the **Maintenance database** field for your installation.

The fields in the **Agent Connection Details** box specify the properties that the PEM agent will use when connecting to the server:

- The **Host** field displays the IP address that will be used for the PEM agent binding.
- The **User name** field displays the name that will be used by the PEM agent when connecting to the selected server.
- The **Password** field displays the password associated with the specified user name.
- Use the drop-down listbox in the **SSL mode** field to specify your SSL connection preferences.

When you've finished specifying the connection properties for the servers that you are binding for monitoring, click the **OK** button to register the servers. Click **Cancel** to exit without preserving any changes.

Agent Status

Blackout	Status	Name	Alerts	Version	Processes	Threads	CPU Utilisation (%)	Memory Utilisation (%)	Swap Utilisation (%)	Disk Utilisation
<input type="checkbox"/>	UP	Postgres Enterprise Manager Host	0	7.14.0-dev	309	810	24.85	77.18	17.88	45.84
<input type="checkbox"/>	UP	PEM Agent on Remote Host	0	7.13.0	207	524	0.35	51.73	3.03	24.30

Server Status

Blackout	Status	Name	Connections	Alerts	Version	Remotely Monitored?
<input type="checkbox"/>	UP	Postgres Enterprise Manager Server	12	6	PostgreSQL 12.1 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-39), 64-bit	No
<input type="checkbox"/>	UP	EDB Postgres Advanced Server 11	3	3	PostgreSQL 11.7 (EnterpriseDB Advanced Server 11.7.14) on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-36), 64-bit	No
<input type="checkbox"/>	DOWN	PGSQL12_Centos7_1	0	0	PostgreSQL 12.2 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-39), 64-bit	Yes
<input type="checkbox"/>	UP	EPAS_12	6	5	PostgreSQL 12.2 (EnterpriseDB Advanced Server 12.2.3) on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-36), 64-bit	No

Alerts Status

Alarm Type	Object Description	Alert Name	Value	Database	Schema	Package	Object	Alerting Since
▶ ● High	EDB Postgres Advanced Server 11	Last Vacuum	Never ran					2020-04-21 21:26:54
▶ ● High	EDB Postgres Advanced Server 11	Last AutoVacuum	177.03 hrs					2020-04-22 12:04:05
▶ ● High	EDB Postgres Advanced Server 11	Database size in server	113 MB					2020-04-22 11:50:00
▶ ● High	EPAS_12	Server Down	1					2020-04-29 09:11:09
▶ ● High	EPAS_12	Table size in server	427 MB					2020-04-09 15:53:51
▶ ● High	EPAS_12	Last Vacuum	15.39 hrs					2020-04-29 08:19:11
▶ ● High	EPAS_12	Database size in server	473 MB					2020-04-09 15:52:50
▶ ● High	EPAS_12	Last AutoVacuum	15.38 hrs					2020-04-29 08:19:11
▶ ● High	N/A	Alert Errors	3					2020-01-21 14:26:04
▶ ● High	PGSQL12_Centos7_1	Server Down	1					2020-04-29 08:54:02
▶ ● High	PGSQL12_Centos7_1	Last Vacuum	Never ran					2020-04-03 14:58:57
▶ ● High	PGSQL12_Centos7_1	Last AutoVacuum	Never ran					2020-04-03 14:58:57
▶ ● High	Postgres Enterprise Manager Server	Largest index by table-size percentage	100 %					2020-04-21 22:07:52
▶ ● High	Postgres Enterprise Manager Server	Database size in server	2.748046875 GB					2020-02-05 18:26:49
▶ ● Medium	Postgres Enterprise Manager Server	Total table bloat in server	88.28 MB					2020-04-29 08:36:18
▶ ● High	Postgres Enterprise Manager Server	Table size in server	2.6591796875 GB					2020-02-20 11:29:45
▶ ● High	Postgres Enterprise Manager Server	Connections in idle state	17					2020-04-29 09:05:07
▶ ● High	Postgres Enterprise Manager Server	Last Vacuum	41.46 hrs					2020-04-28 09:38:02

After clicking the **OK** button, the newly registered server is displayed in the PEM tree control and is monitored by the PEM server.

Using the pemworker Utility to Register a Server

You can use the **pemworker** utility to register a server for monitoring by the PEM server or to unregister a database server. During registration, the **pemworker** utility will bind the new server to the agent that resides on the system from which you invoked the registration command. To register a server:

On a Linux host, use the command:

```
pemworker --register-server
```

On a Windows host, use the command:

```
pemworker.exe REGISTER-SERVICE
```

Append command line options to the command string when invoking the **pemworker** utility. Each option should be

followed by a corresponding value:

Option	Description
--pem-user	Specifies the name of the PEM administrative user. Required.
--server-addr	Specifies the IP address of the server host, or the fully qualified domain name. On Unix based systems, the address field may be left blank to use the default PostgreSQL Unix Domain Socket on the local machine, or may be set to an alternate path containing a PostgreSQL socket. If you enter a path, the path must begin with a /. Required.
--server-port	Specifies the port number of the host. Required.
--server-database	Specifies the name of the database to which the server will connect. Required.
--server-user	Specify the name of the user that will be used by the agent when monitoring the server. Required.
--server-service-name	Specifies the name of the database service that controls operations on the server that is being registered (STOP, START, RESTART, etc.). Optional.
--remote-monitoring	Include the --remote-monitoring clause and a value of no (the default) to indicate that the server is installed on the same machine as the PEM agent. When remote monitoring is enabled (yes), agent level statistics for the monitored server will not be available for custom charts and dashboards, and the remote server will not be accessible by some PEM utilities (such as Audit Manager, Capacity Manager, Log Manager, Postgres Expert and Tuning Wizard). Required.
--efm-cluster-name	Specifies the name of the Failover Manager cluster that monitors the server (if applicable). Optional.
--efm-install-path	Specifies the complete path to the installation directory of Failover Manager (if applicable). Optional.
--asb-host-name	Specifies the name of the host to which the agent is connecting.
--asb-host-port	Specifies the port number that the agent will use when connecting to the database.
--asb-host-db	Specifies the name of the database to which the agent will connect.
--asb-host-user	Specifies the database user name that the agent will supply when authenticating with the database.
--asb-ssl-mode	Specifies the type of SSL authentication that will be used for connections. Supported values include: prefer, require, disable, verify-CA, verify-full.
--group	Specifies the name of the group in which the server will be displayed.
--team	Specifies the name of the group role that will be allowed to access the server.
--owner	Specifies the name of the role that will own the monitored server.

Set the environment variable `PEM_SERVER_PASSWORD` to provide the password for the PEM server to allow the pemworker to connect as a PEM admin user.

Set the environment variable `PEM_MONITORED_SERVER_PASSWORD` to provide the password of the database server being registered and monitored by pemagent.

Failure to provide the password will result in a password authentication error. The PEM server will acknowledge that the server has been registered properly.

Using the pemworker Utility to Unregister a Server

You can use the `pemworker` utility to unregister a database server; to unregister a server, invoke the `pemworker` utility:

On a Linux host, use the command:

```
pemworker --unregister-server
```

On a Windows host, use the command:

```
pemworker.exe UNREGISTER-SERVICE
```

Append command line options to the command string when invoking the `pemworker` utility. Each option should be followed by a corresponding value:

Option	Description
<code>--pem-user</code>	Specifies the name of the PEM administrative user. Required.
<code>--server-addr</code>	Specifies the IP address of the server host, or the fully qualified domain name. On Unix based systems, the address field may be left blank to use the default PostgreSQL Unix Domain Socket on the local machine, or may be set to an alternate path containing a PostgreSQL socket. If you enter a path, the path must begin with a /. Required.
<code>--server-port</code>	Specifies the port number of the host. Required.

Use the `PEM_SERVER_PASSWORD` environment variable to provide the password for the PEM server to allow the pemworker to connect as a PEM admin user.

Failure to provide the password will result in a password authentication error. The PEM server will acknowledge that the server has been unregistered.

Verifying the Connection and Binding

Once registered, the new server will be added to the PEM `Browser` tree control, and be displayed on the `Global Overview`.



When initially connecting to a newly bound server, the **Global Overview** dashboard may display the new server with a status of “unknown” in the server list; before recognizing the server, the bound agent must execute a number of probes to examine the server, which may take a few minutes to complete depending on network availability.

Within a few minutes, bar graphs on the **Global Overview** dashboard should show that the agent has now connected successfully, and the new server is included in the **Postgres Server Status** list.

If after five minutes, the **Global Overview** dashboard still does not list the new server, you should review the logfiles for the monitoring agent, checking for errors. Right-click the agent's name in the tree control, and select the **Probe Log Analysis** option from the **Dashboards** sub-menu of the context menu.

1.3 Defining and Monitoring Postgres instances on AWS

There are two scenarios in which you can monitor a Postgres instance on an AWS host with PEM. You can monitor a:

- Postgres Instance running on AWS EC2
- Postgres Instance running on AWS RDS

Monitoring a Postgres Instance Running on AWS EC2

After creating a Postgres instance on AWS EC2, you can use the PEM server to register and monitor your instance. The following scenarios are currently supported:

- Postgres instance and PEM Agent running on the same AWS EC2 and a PEM Server running on your local machine.
- Postgres instance and PEM Agent running on the same local machine and a PEM Server running on AWS EC2.
- Postgres instance and PEM Agent running on the same AWS EC2 and a PEM Server running in different AWS EC2.

!!! Note In the first two scenarios, you must configure the VPN on AWS EC2 , so the AWS EC2 instance can access the **postgres** database. Please contact your network administrator to setup the VPN if needed.

The PEM Agent running on AWS EC2 or on your local machine should be registered to the PEM Server. Please note that when registering the PEM Agent with the PEM Server you should use the hostname of AWS EC2 instance. For more details on registering the PEM Agent see, [PEM Self Registration](#).

You can register the Postgres instance running on AWS EC2 on PEM Server using the **Create – Server** dialog. For more details on registering the server using **Create – Server** dialog see, [Registering a Server](#). Use the **PEM Agent** tab on the **Create – Server** dialog to bind the registered PEM Agent with the Postgres instance.

When the PEM Agent is registered to the PEM Server and your Postgres instance that is running on AWS EC2 is registered to the PEM Server, you can monitor your instance with PEM.

Monitoring a Postgres Instance Running on AWS RDS

While creating an AWS RDS database, choose **PostgreSQL** when prompted for **Engine options**. After creating a **Postgres(RDS)** instance on AWS, use **Create – Server** dialog to add the **Postgres(RDS)** instance to the PEM Server. Using this dialog you can describe a new server connection, bind the server to a PEM Agent, and display the server to the PEM browser tree control.

For detailed information on the **Create – Server** dialog and configuration details for each tab, see [Registering a Server](#).

The **PEM Agent** tab in the **Create – Server** dialog must have the **Remote Monitoring** field set to **Yes** to monitor the **Postgres (RDS)** instance on AWS instance using PEM Server.



As the PEM Agent will be monitoring the Postgres(RDS) AWS instance remotely, the functionality will be limited as described below:

Feature Name	Works with remote PEM Agent	Comments
Audit Manager	No	
Capacity Manager	Limited	There will be no correlation between the database server and operating system metrics.
Log Manager	No	
Manage Alerts	Limited	When you run an alert script on the database server, it will run on the machine where the bound PEM Agent is running, and not on the actual database server machine.
Manage Charts	Yes	
Manage Dashboards	Limited	Some dashboards may not be able to show complete data. For example, the operating system information of the database server will not be displayed as it is not available.
Manage Probes	Limited	Some of the PEM probes will not return information, and some of the functionalities may be affected. For details about probe functionality, see the PEM Agent Guide .
Postgres Expert	Limited	The Postgres Expert will provide partial information as operating system information is not available.
Postgres Log Analysis Expert	No	The Postgres Log Analysis Expert will not be able to perform an analysis as it is dependent on the logs imported by log manager, which will not work as required.
Scheduled Tasks	Limited	Scheduled tasks will work only for database server; scripts will run on a remote Agent.
Tuning Wizard	No	
System Reports	Yes	
Core Usage Reports	Limited	The Core Usage report will not show complete information. For example, the platform, number of cores, and total RAM will not be displayed.
Managing BART	No	BART requires password less authentication between two machines, where database server and BART are installed. An AWS RDS instance doesn't allow to use host access.

1.4 Managing Certificates

Files stored in the data directory of the PEM server backing database contain information that helps the PEM server utilize secure connections:

- `ca_certificate.crt`
- `ca_key.key`
- `server.crt`
- `server.key`
- `root.crl`
- `root.crt`

The PEM agent that is installed with the PEM server monitors the expiration date of the `ca_certificate.crt` file. When the certificate is about to expire, PEM will:

- Make a backup of the existing certificate files.
- Create new certificate files, appending the new CA certificate file to the root.crt file on the PEM server.
- Create a job that renews the certificate file of any active agents.
- Restart the PEM server.

When you uninstall an agent, the certificate associated with that agent will be added to the certificate revocation list (maintained in the `root.crl`) to ensure that the certificate cannot be used to connect to the PEM server.

The following sections contain detailed information about manually replacing certificate files.

Replacing SSL Certificates

The following steps detail replacing the SSL certificates on an existing PEM installation. If you plan to upgrade your server to a new version at the same time, invoke all of the PEM installers (first the server installer, then agent installers) before replacing the SSL certificates. Then:

1. Stop all running PEM agents, first on the server host, and then on any monitored node.

To stop a PEM agent on a Linux host, open a terminal window, assume superuser privileges, and enter the command:

On Linux with systemd, for eg: Centos 7 or 8

```
systemctl stop pemagent
```

On a Windows host, you can use the `Services` applet to stop the PEM agent. The PEM agent service is named Postgres Enterprise Manager Agent; highlight the service name in the `Services` dialog, and click `Stop the service`.

2. Take a backup of the existing SSL keys and certificates. The SSL keys and certificates are stored in the `data` directory under your PEM installation. For example, the default location on a Linux system is:

```
/var/lib/pgsql/x/data
```

where `x` is the PostgreSQL database version.

Make a copy of the following files, adding an extension to each file to make the name unique:

- `ca_certificate.crt`
- `ca_key.key`
- `root.crt`
- `root.crl`
- `server.key`
- `server.crt`

For example, the command:

```
# cp ca_certificate.crt ca_certificate_old.crt
```

Creates a backup of the `ca_certificate` file with the word `old` appended to the entry.

3. Use the `openssl_rsa_generate_key()` function to generate the `ca_key.key` file:

```
/usr/pgsql-x.x/bin/psql -U postgres -d pem --no-psqlrc -t -A -c "SELECT public.openssl_rsa_generate_key(1024)" > /var/lib/pgsql/x/data/ca_key.key
```

After creating the `ca_key.key` file, `cat` the contents to the variable `CA_KEY` for use when generating the `ca_certificate.crt` file and modify the privileges on the `ca_key.key` file:

```
CA_KEY=$(cat /var/lib/pgsql/x/data/ca_key.key)
chmod 600 /var/lib/pgsql/x/data/ca_key.key
```

4. Use the key to generate the `ca_certificate.crt` file. For simplicity, place the SQL query into a temporary file with a unique name:

```
echo "SELECT openssl_csr_to_crt(openssl_rsa_key_to_csr('${CA_KEY}', 'PEM', 'US', 'MA', 'Bedford', 'Postgres Enterprise Manager', 'support@enterprisedb.com'), NULL, '/var/lib/pgsql/x/data/ca_key.key')" > /tmp/_random.$$
```

Then use the variable to execute the query, placing the content into the `ca_certificate.crt` file.

```
/usr/pgsql-x/bin/psql -U postgres -d pem --no-psqlrc -t -A -f /tmp/_random.$$ > /var/lib/pgsql/x/data/ca_certificate.crt
```

Modify the permissions of the `ca_certificate.crt` file, and remove the temporary file that contained the SQL command:

```
chmod 600 /var/lib/pgsql/x/data/ca_certificate.crt
rm -f /tmp/_random.$$
```

5. Re-use the `ca_certificate.crt` file as the `root.crt` file:

```
cp /var/lib/pgsql/x/data/ca_certificate.crt /var/lib/pgsql/x/data/root.crt
```

Modify the permissions of the `root.crt` file:

```
chmod 600 /var/lib/pgsql/x/data/root.crt
```

6. Use the `openssl_rsa_generate_crl()` function to create the certificate revocation list (`root.crl`):

```
/usr/pgsql-x/bin/psql -U postgres -d pem --no-psqlrc -t -A -c
"SELECT openssl_rsa_generate_crl('/var/lib/pgsql/x/data/ca_certificate.crt', '/var/lib/pgsql/x/data/ca_key.key')" > /var/lib/pgsql/x/data/root.crl
```

Modify the permissions of the `root.crl` file:

```
chmod 600 /var/lib/pgsql/x/data/root.crl
```

7. Use the `openssl_rsa_generate_key()` function to generate the `server.key` file:

```
/usr/pgsql-x/bin/psql -U postgres -d pem --no-psqlrc -t -A -c "SELECT
public.openssl_rsa_generate_key(1024)" >> /var/lib/pgsql/x/data/server.key
```

After creating the `server.key` file, `cat` the contents to the variable `SSL_KEY` for use when generating the `server.crt` file and modify the privileges on the `server.key` file:

```
SSL_KEY=$(cat /var/lib/pgsql/x/data/server.key)
```

```
chmod 600 /var/lib/pgsql/x/data/server.key
```

8. Use the `SSL_KEY` to generate the server certificate. Save the certificate in the `server.crt` file. For simplicity, first place the SQL query into a temporary file with a unique name:

```
echo "SELECT openssl_csr_to_crt(openssl_rsa_key_to_csr('${SSL_KEY}', 'PEM', 'US', 'MA', 'Bedford', 'Postgres Enterprise Manager', 'support@enterprisedb.com'), '/var/lib/pgsql/x/data/ca_certificate.crt', '/var/lib/pgsql/x/data/ca_key.key')" > /tmp/_random.$$

/usr/pgsql-x/bin/psql -U postgres -d pem --no-psqlrc -t -A -f /tmp/_random.$$ >> /var/lib/pgsql/x/data/server.crt
```

9. Modify the privileges on the `server.crt` file, and delete the temporary file:

```
chmod 600 /var/lib/pgsql/x/data/server.crt

rm -f /tmp/_random.$$
```

10. Restart the Postgres server:

On Linux with `init.d`; for example, on a Centos6 host:

```
/etc/init.d/postgresql-x restart
```

On Linux with `systemd`; for example, on a Centos7 host:

```
systemctl restart postgresql-x
```

Updating Agent SSL Certificates

For each agent that interacts with the PEM server, you must:

- generate an rsa key and a certificate.
- copy the key and certificate to the agent.
- restart the agent.

Each agent has a unique identifier that is stored in the `pem.agent` table in the `pem` database. You must replace the key and certificate files with the key or certificate that corresponds to the agent's identifier. Please note that you must move the `agent.key` and `agent.crt` files (generated in Steps 2 and 3 into place on their respective PEM agent host before generating the next key file pair; subsequent commands will overwrite the previously generated file.

To generate a PEM agent key file pair:

1. Use psql to find the number of agents and their corresponding identifiers:

```
/usr/pgsql-x/bin/psql -U postgres -d pem --no-psqlrc -t -A -c "SELECT ID FROM pem.agent"
```

- On Linux, you can also find the agent identifier and location of the keys and certificates in the `PEMagent` section of the `/etc/postgres-reg.ini` file.
- On Windows, the information is stored in the registry:

- On a 64-bit Windows installation, check:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\EnterpriseDB\PEM\agent
```

- On a 32-bit Windows installation, check:

```
HKEY_LOCAL_MACHINE\SOFTWARE\EnterpriseDB\PEM\agent
```

- After identifying the agents that will need key files, generate an **agent.key** for each agent. To generate the key, execute the following command, capturing the output in a file:

```
/usr/pgsql-x/bin/psql -U postgres -d pem --no-psqlrc -t -A -c "SELECT
openssl_rsa_generate_key(1024)" > agent.key
```

Modify the privileges of the **agent.key** file:

```
chmod 600 agent.key
```

- Generate a certificate for each agent. To generate a certificate, execute the following command, capturing the output in a certificate file:

```
/usr/pgsql-x/bin/psql -U postgres -d pem --no-psqlrc -t -A -c
"SELECT openssl_csr_to_crt(openssl_rsa_key_to_csr('$(cat agent.key)',",
'agent<$ID>', 'US', 'MA', 'Bedford', 'Postgres Enterprise Manager',
'support@enterprisedb.com'),
'/var/lib/pgsql/x/data/ca_certificate.crt',
'/var/lib/pgsql/x/data/ca_key.key')" > agent.crt
```

Where \$ID is the agent number of the agent (retrieved via the psql command line).

- Modify the privileges of the **agent.crt** file:

```
chmod 600 agent.crt
```

- Replace each agent's key and certificate file with the newly generated files before restarting the PEM agent service:

On Linux with **init.d**, restart the service with the command:

```
/etc/init.d/pemagent start
```

On Linux with **systemd**, restart the service with the command:

```
systemctl start pemagent
```

On a Windows host, you can use the Services applet to start the PEM agent. The PEM agent service is named **Postgres Enterprise Manager Agent**; highlight the service name in the **Services** dialog, and click **Start the service**.

1.5 Managing Configuration Settings

Multiple configuration files are read at startup by Postgres Enterprise Manager. The files are as follows:

- **config.py**: This is the main configuration file, and should not be modified. It can be used as a reference for configuration settings, that may be overridden in one of the following files.
- **config_distro.py**: This file is read after **config.py** and is intended for packagers to change any settings that are required for their Postgres Enterprise Manager distribution. This may typically include certain paths and file locations. This file is optional, and may be created by packagers in the same directory as **config.py** if needed.
- **config_local.py**: This file is read after **config_distro.py** and is intended for end users to change any default or packaging specific settings that they may wish to adjust to meet local preferences or standards. This file is optional, and may be created by users in the same directory as **config.py** if needed.

A copy of the default **config.py** file is included in the PEM online help for reference.

1.6 Managing a PEM Server

The sections that follow provide information about tasks related to PEM server such as restarting the PEM server and agent, controlling the PEM server or PEM agent, controlling the HTTPD service on Linux and Windows, controlling the HTTPD server, managing PEM authentication and security, modifying the **pg_hba.conf** file, modifying PEM to use a proxy server etc.

Starting and Stopping the PEM Server and Agents

The PEM server starts, stops and restarts when the Postgres server instance on which it resides starts, stops or restarts; use the same commands to control the PEM server that you would use to control the Postgres server. On Linux platforms, the command that stops and starts the service script will vary by platform and OS version.

The PEM agent is controlled by a service named **pemagent**.

The Windows operating system includes a graphical service controller that displays the server status, and offers point-and-click server control. The **Services** utility can be accessed through the Windows **Control Panel**. When the utility opens, use the scroll bar to navigate through the listed services to highlight the service name.



Use the **Stop**, **Pause**, **Start**, or **Restart** buttons to control the state of the service.

Please note that any user (or client application) connected to the Postgres server will be abruptly disconnected if you stop the service. For more information about controlling a service, please consult the [EDB Postgres Advanced Server Installation Guide](#), available from the EDB website.

Remotely Starting and Stopping Monitored Servers

PEM allows you to startup and shutdown managed server instances with the PEM client. To configure a server to allow PEM to manage the service, complete the Server registration dialog, registering the database server with a PEM agent and:

- specify the **Store on PEM Server** option on the **Properties** dialog.
- specify the name of a service script in the **Service ID** field on the **Advanced** tab:
 - For Advanced Server, the service name is `edb-as-<x>` or `ppas-<x>`.
 - For PostgreSQL, the service name is `postgresql-<x>`.

Where `x` indicates the server version number.

After connecting to the server, you can start or stop the server by highlighting the server name in the tree control, and selecting **Queue Server Startup** or **Queue Server Shutdown** from the **Tools** menu.



Controlling the PEM Server or PEM Agent on Linux

On Linux platforms, the name of the service script that controls:

- a PEM server on Advanced Server is `edb-as-<x>` or `ppas-<x>`
- a PEM server on PostgreSQL is `postgresql-<x>`
- a PEM agent is `pemagent`

Where `x` indicates the server version number.

You can use the service script to control the service.

- To control a service on RHEL or CentOS version 7.x or 8.x open a command line, assume superuser privileges, and issue the command:

```
systemctl <service_name> <action>
```

Where:

`service_name` is the name of the service.

`action` specifies the action taken by the service. Specify:

- `start` to start the service.
- `stop` to stop the service.

- **restart** to stop and then start the service.
- **status** to check the status of the service.

Controlling the PEM Server or PEM Agent on Windows

The Windows operating system includes a graphical service controller that displays the server status, and offers point-and-click server control. The registered name of the service that controls:

- a PEM server host on PostgreSQL is **postgresql-<x>**
- a PEM server host on Advanced Server is **edb-as-<x>**, or **ppas-<x>**
- a PEM agent is **Postgres Enterprise Manager - pemAgent**

Where **<x>** indicates the server version number.

Navigate through the Windows **Control Panel** to open the **Services** utility. When the utility opens, use the scroll bar to browse the list of services.



Use the **Stop the service** option to stop a service. Any user (or client application) connected to the server will be abruptly disconnected if you stop the service.

Use the **Pause the service** option to instruct Postgres to reload a service's configuration parameters. The **Pause the service** option is an effective way to reset parameters without disrupting user sessions for many of the configuration parameters.

Use the **Start the service** option to start a service.

Controlling the HTTPD Server

On Linux, you can confirm the status of the **PEM-HTTPD** service by opening a command line, and entering the following command:

```
ps -ef | grep httpd
```

If Linux responds with an answer that is similar to the following example, **httpd** is not running:

```
user 13321 13267 0 07:37 pts/1 00:00:00 grep httpd
```

To start the service on a CentOS or RHEL 7.x or 8.x system, use the command:

```
systemctl start httpd
```

On Windows, you can use the **Services** applet to check the status of the **PEM HTTPD** service. After opening the Services applet, scroll through the list to locate the **PEM HTTPD** service.

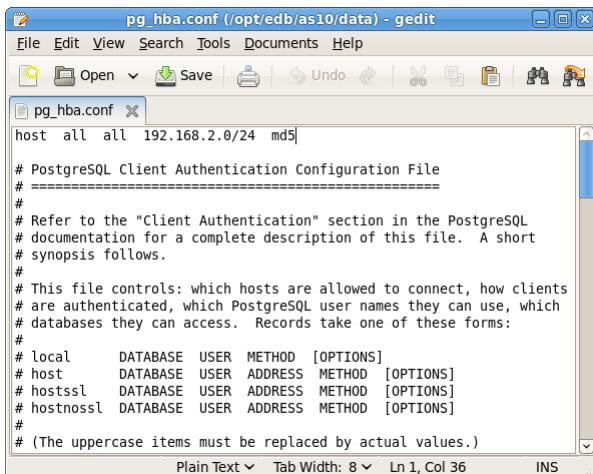


The **Status** column displays the current state of the server. Click the **Start** link to start **PEM HTTPD** if the service is not running.

Modifying the pg_hba.conf File

Entries in the **pg_hba.conf** file control network authentication and authorization. The **pg_hba.conf** file on the PEM server host must allow connections between the PEM server and PEM-HTTPD, the PEM agent, and the monitored servers.

During the PEM server installation process, you are prompted for the IP address and connection information for hosts that will be monitored by PEM; this information is added to the top of the **pg_hba.conf** file of the PEM backing database.



You may also need to manually modify the **pg_hba.conf** file to allow connections between the PEM server and other components. For example, if your PEM-HTTPD installation does not reside on the same host as the PEM server, you must modify the **pg_hba.conf** file on the PEM server host to allow PEM-HTTPD to connect to the server.

By default, the **pg_hba.conf** file resides in the data directory, under your Postgres installation; for example, on an Advanced Server 10 host, the default location of the pg_hba.conf is:

`/var/lib/edb/as10/data/pg_hba.conf`

You can modify the **pg_hba.conf** file with your editor of choice. After modifying the file, restart the server for changes to take effect.

The following example shows a `pg_hba.conf` entry that allows an md5 password authenticated connection from a user named `postgres`, to the `postgres` database on the host on which the `pg_hba.conf` file resides. The connection is coming from an IP address of `192.168.10.102`:

### TYPE	DATABASE	USER	CIDR-ADDRESS	METHOD
### IPv4 local connections:				
host	postgres	postgres	192.168.10.102/32	md5

You may specify the address of a network host, or a network address range. For example, if you wish to allow connections from servers with the addresses `192.168.10.23`, `192.168.10.76` and `192.168.10.184`, enter a CIDR-ADDRESS of `192.168.10.0/24` to allow connections from all of the hosts in that network:

### TYPE	DATABASE	USER	CIDR-ADDRESS	METHOD
### IPv4 local connections:				
host	postgres	all	192.168.10.0/24	md5

For more information about formatting a `pg_hba.conf` file entry, please see the [PostgreSQL core documentation](#).

Before you can connect to a Postgres server with PEM, you must ensure that the `pg_hba.conf` file on both servers allows the connection.

If you receive this error when connecting to the database server, modify the `pg_hba.conf` file, adding an entry that allows the connection.

Creating and Maintaining Databases and Objects

Each instance of a Postgres server manages one or more databases; each user must provide authentication information to connect to the database before accessing the information contained within it. The PEM client provides dialogs that allow you to create and manage databases, and all of the various objects that comprise a database (e.g. tables, indexes, stored procedures, etc.).

Creating a database is easy in PEM: simply right click on any managed server's `Databases` node and select `Database...` from the `Create` menu. After defining a database, you can create objects within the new database.

For example, to create a new table, right click on a `Tables` node, and select `Table...` from the `Create` menu. When the `New Table` dialog opens, specify the attributes of the new table.



PEM provides similar dialogs for the creation and management of other database objects:

- tables

- indexes
- stored procedures
- functions
- triggers
- views
- constraints, etc.

Each object type is displayed in the tree control; right click on the node that corresponds to an object type to access the **Create** menu and create a new object, or select **Properties** from the context menu of a named node to perform administrative tasks for the highlighted object.

Managing PEM Authentication

Postgres supports a number of authentication methods:

- Secure password (md5)
- GSSAPI
- SSPI
- Kerberos
- Ident
- LDAP
- RADIUS
- Certificate (SSL)
- PAM

Postgres (and PEM) authentication is controlled by the **pg_hba.conf** configuration file. Entries within the configuration file specify who may connect to a specific database, and the type of authentication required before that user is allowed to connect.

A typical entry in the **pg_hba.conf** file that allows a user named **postgres** to connect to all databases from the local host (127.0.0.1/32) using secure password (md5) authentication connections would take the form:

```
host all postgres 127.0.0.1/32 md5
```

Depending on your system's configuration, you may also need to create a password file for the user account that the PEM agent uses to connect to the server, to allow the agent to properly respond to the server's authentication request. An entry in the password file for a user named **postgres**, with a password of **1safepwd** would take the form:

```
localhost:5432:*:postgres:1safepwd
```

The password file is usually named **~root/.pgpass** on Linux systems, or **%APPDATA%\postgresql\pgpass.conf** (on Windows). For more information about configuring a password file, visit the [PostgreSQL website](#).

For more information about the authentication methods supported by Postgres, see the [PostgreSQL core documentation](#).

Editing the PEM Server Configuration

You can use the PEM client to graphically manage the configuration parameters of the PEM server to enable features or modify default settings. To open the **Server Configuration** dialog, select **Server Configuration...** from the **Management** menu.

Parameter	Value	Unit
audit_log_retention_time	30	days
auto_create_agent_alerts	True	t/f
auto_create_server_alerts	True	t/f
bart_log_retention_time	30	days
cm_data_points_per_report	50	
cm_max_end_date_in_years	5	years
dash_alerts_timeout	60	seconds
dash_db_connroll_span	168	hours
dash_db_connroll_timeout	1800	seconds
dash_db_connovervw_timeout	300	seconds
dash_db_eventlag_span	7	days
dash_db_eventlag_timeout	1800	seconds
dash_db_hottable_rows	25	rows
dash_db_hottable_timeout	300	seconds
dash_db_io_span	168	hours
dash_db_io_timeout	1800	seconds

To modify a parameter value, edit the content displayed in the **Value** field to the right of a parameter name. Click the **Save** button to preserve your changes, or click the **Close** button to exit the dialog without applying the changes. Use the **Reset** button to return the parameters to their original value.

Managing Security

PEM provides a graphical way to manage your Postgres roles and servers.

Login Roles

When you connect to the PEM server, you must provide role credentials that allow access to the database on which the PEM server stores data. By default, the `postgres` superuser account is used to initially connect to the server, but it is strongly recommended (for both security and auditing purposes) that individual roles are created for each connecting user. You can use the PEM Query Tool, the PEM web interface **Create – Login/Group Role** dialog, or a command line client (such as `psql`) to create a role.

To use the **Create – Login/Group Role** dialog to create a role, expand the node for the server on which the role will reside in the PEM tree control, and right-click on the **Login/Group Roles** node to access the context menu. Then, select **Login/Group Role...** from the **Create** menu.

Use fields on the tabs of the **Create – Login/Group Role** dialog to define the role. To display the PEM online help in a browser tab, click the help (?) button located in the lower-left corner of the dialog.

When you've finished defining the new role, click **Save** to create the role.



To modify the properties of an existing login role, right click on the name of a login role in the tree control, and select **Properties** from the context menu. To delete a login role, right click on the name of the role, and select **Delete/Drop** from the context menu.

For more complete information about creating and managing a role, see the [PostgreSQL online documentation](#).

Group Roles

Group roles can serve as containers, used to dispense system privileges (such as creating databases) and object privileges (e.g. inserting data into a particular table). The primary purpose of a group role is to make the mass management of system and object permissions much easier for a DBA. Rather than assigning or modifying privileges individually across many different login accounts, you can assign or change privileges for a single role and then grant that role to many login roles at once.

Use the **Group Roles** node (located beneath the name of each registered server in the PEM tree control) to create and manage group roles. Options on the context menu provide access to a dialog that allows you to create a new role or modify the properties of an existing role. You can find more information about creating roles [here](#).

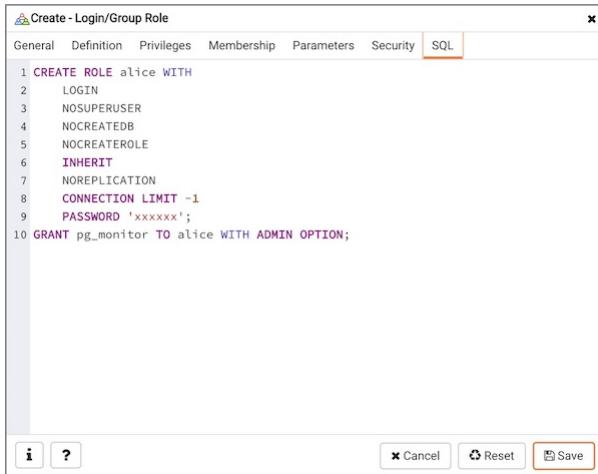
Using PEM Pre-Defined Roles to Manage Access to PEM Functionality

You can use the **Login/Group Role** dialog to allow a role with limited privileges to access PEM features such as the Audit Manager, Capacity Manager, or SQL Profiler. PEM pre-defined roles allow access to PEM functionality; roles that are assigned membership in these roles can access the associated feature.



When defining a user, use the **Membership** tab to specify the roles in which the new user is a member. The new user will share the privileges associated with each role in which it is a member. For a user to have access to PEM extended functionality, the role must be a member of the pem_user role and the pre-defined role that grants access to the feature. Use the **Roles** field to select pre-defined role names from a drop down list.

The **SQL** tab displays the SQL command that the server will execute when you click **Save**.



```

CREATE ROLE alice WITH
  LOGIN
  NOSUPERUSER
  NOCREATEDB
  NOCREATEROLE
  INHERIT
  NOREPLICATION
  CONNECTION LIMIT -1
  PASSWORD 'xxxxxx';

GRANT pg_monitor TO alice WITH ADMIN OPTION;

```

The screenshot shows a software interface titled "Create - Login/Group Role". The "SQL" tab is selected, displaying the generated SQL code for creating a user named "alice". The code includes standard PostgreSQL syntax for creating a role with specific options like LOGIN, NOSUPERUSER, and NOCREATEDB, and granting the "pg_monitor" privilege with the "ADMIN" option.

The example shown above creates a login role named **acctg_clerk** that will have access to the **Audit Manager**; the role can make unlimited connections to the server at any given time.

You can use PEM pre-defined roles to allow access to the functionality listed in the table below:

Value	Parent Role	Description
pem_super_admin		Role to manage/configure everything on Postgres Enterprise Manager.
pem_admin	pem_super_admin	Role for administration/management/configuration of all visible agents/servers, and monitored objects.
pem_config	pem_admin	Role for configuration management of Postgres Enterprise Manager.
pem_component	pem_admin	Role to run/execute all wizard/dialog based components.
pem_rest_api	pem_admin	Role to access the REST API.
pem_server_service_manager	pem_admin	Role for allowing to restart/reload the monitored database server (if server-id provided).
pem_manage_schedule_task	pem_admin	Role to configure the schedule tasks.
pem_manage_alert	pem_admin	Role for managing/configuring alerts, and its templates.
pem_config_alert	pem_config, pem_manage_alert	Role for configuring the alerts on any monitored objects.
pem_manage_probe	pem_admin	Role to create, update, delete the custom probes, and change custom probe configuration.
pem_config_probe	pem_config, pem_manage_probe	Role for probe configuration (history retention, execution frequency, enable/disable the probe) on all visible monitored objects.
pem_database_server_registration	pem_admin	Role to register a database server.
pem_comp_postgres_expert	pem_component	Role to run the Postgres Expert.
pem_comp_auto_discovery	pem_component	Role to run the Auto discovery of a database server dialog.
pem_comp_log_analysis_expert	pem_component	Role to run the Log Analysis Expert.
pem_comp_sqlprofiler	pem_component	Role to run the SQL Profiler.

Value	Parent Role	Description
pem_manage_efm	pem_admin	Role to manage Failover Manager functionality.
pem_comp_capacity_manager	pem_component	Role to run the Capacity Manager.
pem_comp_log_manager	pem_component	Role to run the Log Manager.
pem_comp_audit_manager	pem_component	Role to run the Audit Manager.
pem_comp_tuning_wizard	pem_component	Role to run the Tuning Wizard.

Using a Team Role

When you register a server for monitoring by PEM, you can specify a *Team* that will be associated with the server. A Team is a group role that can be used to allow or restrict access to one or more monitored servers to a limited group of role members. The PEM client will only display a server with a specified Team to those users who are:

- a member of the Team role
- the role that created the server
- a role with superuser privileges on the PEM server.

To create a team role, expand the node for the server on which the role will reside in the PEM tree control, and right-click on the **Login/Group Roles** node to access the context menu. Then, select **Login/Group Role...** from the **Create** menu; when the **Create – Login/Group Role** dialog opens, use the fields provided to specify the properties of the team role.

Object Permissions

A role must be granted sufficient privileges before accessing, executing, or creating any database object. PEM allows you to assign (**GRANT**) and remove (**REVOKE**) object permissions to group roles or login accounts using the graphical interface of the PEM client.

Object permissions are managed via the graphical object editor for each particular object. For example, to assign privileges to access a database table, right click on the table name in the tree control, and select the Properties option from the context menu. Use the options displayed on the Privileges tab to assign privileges for the table.

The PEM client also contains a **Grant Wizard** (accessed through the **Tools** menu) that allows you to manage many object permissions at once.

Managing Job Notifications

You can configure the settings in PEM console for sending the SMTP trap on success or failure of a system-generated job (listed under scheduled tasks) or a custom-defined agent job. These email notification settings can be configured at following three levels (in order of precedence) to send email notifications to the specified user group:

- Job level
- Agent level
- PEM server level (default level)

Configuring Job Notifications at Job Level

You can configure email notification settings at job level only for a custom-defined agent job in one of the following

ways:

- For a new agent job, you can configure the email notification settings in the **Notification** tab of **Create-Agent Job** wizard while creating the job itself.
- For an existing custom-defined job, you can edit the properties of the job and configure the notification settings.



Use the fields on the **Notifications** tab to configure the email notification settings on job level:

- Use the **Send the notifications** field to specify when you want the email notifications to be sent.
- Use the **Email group** field to specify the email group that should receive the email notification.

Configuring Job Notifications at Agent Level

Select the agent in the tree view, right click and select *Properties*. In the **Properties** dialog, select the *Job notifications* tab.



Use the fields on the Job notifications tab to configure the email notification settings on agent level:

- Use the **Override default configuration?** switch to specify if you want the agent level job notification settings to override the default job notification settings. If you select **Yes** for this switch, you can use the rest of the settings on this dialog to define when and to whom the job notifications should be sent. Please note that the rest of the settings on this dialog work only if you enable the **Override default configuration?** switch.
- Use the **Email on job completion?** switch to specify if the job notification should be sent on the successful job completion.
- Use the **Email on a job failure?** switch to specify if the job notification should be sent on the failure of a job.
- Use the **Email group** field to specify the email group to whom the job notification should be sent.

Configuring Job Notifications at Server Level

You can use the *Server Configuration* dialog to provide information about your email notification configuration at PEM server level. To open the Server Configuration dialog, select *Server Configuration...* from the PEM client's Management menu.



Four server configuration parameters specify information about your job notification preferences at PEM server level:

- Use the **job_failure_notification** switch to specify if you want to send email notification after each job failure.
- Use the **job_notification_email_group** parameter to specify the email group that should receive the email notification.
- Use the **job_retention_time** parameter to specify the number of days that non-recurring scheduled tasks should be retained in the system.
- Use the **job_status_change_notification** switch to specify if you want to send email notification after each job status change, irrespective of its status being a failure, success, or interrupted.

Managing PEM Scheduled Jobs

You can create a PEM scheduled job to perform a set of custom-defined steps in the specified sequence. These steps may contain SQL code or a batch/shell script that you may run on a server that is bound with the agent. You can schedule these jobs to suit your business requirements. For example, you can create a job for taking a backup of a particular database server and schedule it to run on a specific date and time of every month.

To create or manage a PEM scheduled job, use the PEM tree control to browse to the PEM agent for which you want to create the job. The tree control will display a **Jobs** node, under which currently defined jobs are displayed. To add a new job, right click on the **Jobs** node, and select **Create Job...** from the context menu.

When the **Create – Agent Job** dialog opens, use the tabs on the **Create – Agent Job** dialog to define the steps and schedule that make up a PEM scheduled job.



Use the fields on the **General** tab to provide general information about a job:

- Provide a name for the job in the **Name** field.
- Move the **Enabled** switch to the **Yes** position to enable a job, or **No** to disable a job.
- Use the **Comment** field to store notes about the job.



Use the **Steps** tab to define and manage the steps that the job will perform. Click the **Add** icon (+) to add a new step; then click the compose icon (located at the left side of the header) to open the step definition dialog:



Use fields on the step definition dialog to define the step:

- Provide a name for the step in the **Name** field; please note that steps will be performed in alphanumeric order by name.
- Use the **Enabled** switch to include the step when executing the job (**True**) or to disable the step (**False**).
- Use the **Kind** switch to indicate if the job step invokes SQL code (**SQL**) or a batch script (**Batch**).
 - If you select **SQL**, use the **Code** tab to provide SQL code for the step.
 - If you select **Batch**, use the **Code** tab to provide the batch script that will be executed during the step.
- Use the **On error** drop-down to specify the behavior of pgAgent if it encounters an error while executing the step. Select from:
 - Fail - Stop the job if you encounter an error while processing this step.
 - Success - Mark the step as completing successfully, and continue.
 - Ignore - Ignore the error, and continue.
- If you have selected SQL as your input for **Kind** switch, provide the following additional information:
 - Use the **Server** field to specify the server that is bound with the agent for which you are creating the PEM scheduled job.
 - Use the **Database** field to specify the database that is associated with the server that you have selected.
- Use the **Comment** field to provide a comment about the step.



- Use the context-sensitive field on the step definition dialog's **Code** tab to provide the SQL code or batch script that will be executed during the step:
 - If the step invokes SQL code, provide one or more SQL statements in the **SQL query** field.
 - If the step invokes a batch script, provide the script in the **Code** field. If you are running on a Windows server, standard batch file syntax must be used. When running on a Linux server, any shell script may be used, provided that a suitable interpreter is specified on the first line (e.g. `#!/bin/sh`). Along with the defined inline code, you can also provide the path of any batch script, shell script, or SQL file on the filesystem.
 - To invoke a script on a Linux system, you must modify the entry for **batch_script_user** parameter in the **agent.cfg** file and specify the user that should be used to run the script. You can either specify a non-root user or root for this parameter. If you do not specify a user, or the specified user does not exist, then the script will not be executed. Restart the agent after modifying the file.
 - To invoke a script on a Windows system, set the registry entry for **AllowBatchJobSteps** to **true** and restart the PEM agent. PEM registry entries are located in **HKEY_LOCAL_MACHINE\Software\Wow6432Node\EnterpriseDB\PEM\agent**.

After providing all the information required by the step, click the **Save** button to save and close the step definition dialog.

Click the add icon (+) to add each additional step, or select the **Schedules** tab to define the job schedule.

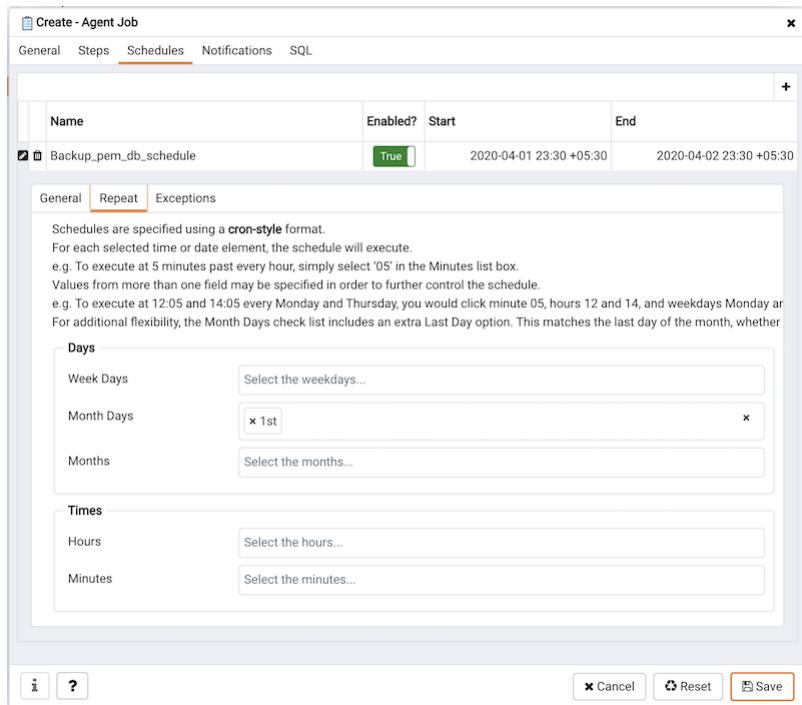
Click the add icon (+) to add a schedule for the job; then click the compose icon (located at the left side of the header) to open the schedule definition dialog:



Use the fields on the **Schedules definition** tab to specify the days and times at which the job will execute.

- Provide a name for the schedule in the **Name** field.
- Use the **Enabled** switch to indicate that pgAgent should use the schedule (**Yes**) or to disable the schedule (**No**).
- Use the calendar selector in the **Start** field to specify the starting date and time for the schedule.
- Use the calendar selector in the **End** field to specify the ending date and time for the schedule.
- Use the **Comment** field to provide a comment about the schedule.

Select the **Repeat** tab to define the days on which the schedule will execute.



Use the fields on the **Repeat** tab to specify the details about the schedule in a cron-style format. The job will execute on each date or time element selected on the **Repeat** tab.

Click within a field to open a list of valid values for that field; click on a specific value to add that value to the list of

selected values for the field. To clear the values from a field, click the X located at the right-side of the field.

- Use the fields within the **Days** box to specify the days on which the job will execute:
 - Use the **Week Days** field to select the days on which the job will execute.
 - Use the **Month Days** field to select the numeric days on which the job will execute. Specify the **Last Day** to indicate that the job should be performed on the last day of the month, regardless of the date.
 - Use the **Months** field to select the months in which the job will execute.
- Use the fields within the **Times** box to specify the times at which the job will execute:
 - Use the **Hours** field to select the hour at which the job will execute.
 - Use the **Minutes** field to select the minute at which the job will execute.

Select the **Exceptions** tab to specify any days on which the schedule will **not** execute.



Use the fields on the **Exceptions** tab to specify days on which you wish the job to not execute; for example, you may wish for jobs to not execute on national holidays.

Click the Add icon (+) to add a row to the exception table, then:

- Click within the **Date** column to open a calendar selector, and select a date on which the job will not execute. Specify **<Any>** in the **Date** column to indicate that the job should not execute on any day at the time selected.
- Click within the **Time** column to open a time selector, and specify a time on which the job will not execute. Specify **<Any>** in the **Time** column to indicate that the job should not execute at any time on the day selected.

Select the **Notifications** tab to configure the email notification settings on job level:



Use the fields on the **Notifications** tab to configure the email notification settings for a job:

- Use the **Send the notifications** field to specify when you want the email notifications to be sent.
- Use the **Email group** field to specify the email group that should receive the email notification.

When you've finished defining the schedule, you can use the **SQL** tab to review the code that will create or modify your job.

```

1 DO $$ 
2 DECLARE
3     jid integer;
4     scid integer;
5 BEGIN
6 -- Creating a new job
7 INSERT INTO pem.job(
8     agent_id, jobname, jobdesc, jobenabled, notify, email_group_id
9 ) VALUES (
10    1::integer, 'Job_backup_pem_db'::text, 'Job for taking backup of pem database'::text, true, 'ALWAYS'
11 ) RETURNING jobid INTO jid;
12
13 -- Steps
14
15 -- Inserting a step (jobid: NULL)
16 INSERT INTO pem.jobstep(
17     jstjobid, jstname, jstenabled, jstkind, jstonerror, jstcode, jstdesc,
18     server_id, database_name
19 ) VALUES (
20     jid,
21     'step1_backup'::text,
22     true,
23     's'::character(1),
24     's'::character(1),
25     ''::text, ''::text,
26     '1'::integer,
27     'pem'::text
28 );
29
30
31 -- Schedules
32 -- Inserting a schedule

```

Click the **Save** button to save the job definition, or **Cancel** to exit the job without saving. Use the **Reset** button to remove your unsaved entries from the dialog.

After saving a job, the job will be listed under the **Jobs** node of the PEM tree control of the server on which it was defined. The **Properties** tab in the PEM console will display a high-level overview of the selected job, and the Statistics tab will show the details of each run of the job. To modify an existing job or to review detailed information

about a job, right-click on a job name, and select **Properties** from the context menu.

1.7 Managing a PEM Agent

The sections that follow provide information about the behavior and management of a PEM agent.

Agent Privileges

By default, the PEM agent is installed with **root** privileges for the operating system host and superuser privileges for the database server. These privileges allow the PEM agent to invoke unrestricted probes on the monitored host and database server about system usage, retrieving and returning the information to the PEM server.

Please note that PEM functionality diminishes as the privileges of the PEM agent decrease. For complete functionality, the PEM agent should run as **root**. If the PEM agent is run under the database server's service account, PEM probes will not have complete access to the statistical information used to generate reports, and functionality will be limited to the capabilities of that account. If the PEM agent is run under another lesser-privileged account, functionality will be limited even further.

If you limit the operating system privileges of the PEM agent, some of the PEM probes will not return information, and the following functionality may be affected:

Probe or Action	Operating System	PEM Functionality Affected
Data And Logfile Analysis	Linux/ Windows	The Postgres Expert will be unable to access complete information.
Session Information	Linux	The per-process statistics will be incomplete.
PG HBA	Linux/ Windows	The Postgres Expert will be unable to access complete information.
Service restart functionality	Linux/ Windows	The Audit Log Manager, Server Log Manager Log Analysis Expert and PEM may be unable to apply requested modifications.
Package Deployment	Linux/ Windows	PEM will be unable to run downloaded installation modules.
Batch Task	Windows	PEM will be unable to run scheduled batch jobs in Windows.
Collect data from server (root access required)	Linux/ Windows	Columns such as swap usage, CPU usage, IO read, IO write will be displayed as 0 in the session activity dashboard.

!!! Note The above-mentioned list is not comprehensive, but should provide an overview of the type of functionality that will be limited.

If you restrict the database privileges of the PEM agent, the following PEM functionality may be affected:

Probe	Operating System	PEM Functionality Affected
Audit Log Collection	Linux/Windows	PEM will receive empty data from the PEM database.

Probe	Operating System	PEM Functionality Affected
Server Log Collection	Linux/Windows	PEM will be unable to collect server log information.
Database Statistics	Linux/Windows	The Database/Server Analysis dashboards will contain incomplete information.
Session Waits/System Waits	Linux/Windows	The Session/System Waits dashboards will contain incomplete information.
Locks Information	Linux/Windows	The Database/Server Analysis dashboards will contain incomplete information.
Streaming Replication	Linux/Windows	The Streaming Replication dashboard will not display information.
Slony Replication	Linux/Windows	Slony-related charts on the Database Analysis dashboard will not display information.
Tablespace Size	Linux/Windows	The Server Analysis dashboard will not display complete information.
xDB Replication	Linux/Windows	PEM will be unable to send xDB alerts and traps.

If the probe is querying the operating system with insufficient privileges, the probe may return a **permission denied** error.

If the probe is querying the database with insufficient privileges, the probe may return a **permission denied** error or display the returned data in a PEM chart or graph as an empty value.

When a probe fails, an entry will be written to the log file that contains the name of the probe, the reason the probe failed, and a hint that will help you resolve the problem.

You can view probe-related errors that occurred on the server in the **Probe Log** dashboard, or review error messages in the PEM worker log files. On Linux, the default location of the log file is:

`/var/log/pem/worker.log`

On Windows, log information is available on the **Event Viewer**.

Agent Configuration

A number of user-configurable parameters and registry entries control the behavior of the PEM agent. You may be required to modify the PEM agent's parameter settings to enable some PEM functionality. After modifying values in the PEM agent configuration file, you must restart the PEM agent to apply any changes.

With the exception of the **PEM_MAXCONN** parameter, we strongly recommend against modifying any of the configuration parameters or registry entries listed below without first consulting EDB support experts *unless* the modifications are required to enable PEM functionality.

On Linux systems, PEM configuration options are stored in the **agent.cfg** file, located in `/usr/edb/pem/agent/etc`. The **agent.cfg** file contains the following entries:

Parameter Name	Description	Default Value
pem_host	The IP address or hostname of the PEM server.	127.0.0.1.
pem_port	The database server port to which the agent connects to communicate with the PEM server.	Port 5432.

Parameter Name	Description	Default Value
pem_agent	A unique identifier assigned to the PEM agent.	The first agent is '1', the second agent is '2', and so on.
agent_ssl_key	The complete path to the PEM agent's key file.	/root/.pem/agent.key
agent_ssl_crt	The complete path to the PEM agent's certificate file.	/root/.pem/agent.crt
agent_flag_dir	Used for HA support. Specifies the directory path checked for requests to take over monitoring another server. Requests are made in the form of a file in the specified flag directory.	Not set by default.
log_level	Log level specifies the type of event that will be written to the PEM log files.	warning
log_location	Specifies the location of the PEM worker log file.	127.0.0.1.
agent_log_location	Specifies the location of the PEM agent log file.	/var/log/pem/agent.log
long_wait	The maximum length of time (in seconds) that the PEM agent will wait before attempting to connect to the PEM server if an initial connection attempt fails.	30 seconds
short_wait	The minimum length of time (in seconds) that the PEM agent will wait before checking which probes are next in the queue (waiting to run).	10 seconds
alert_threads	The number of alert threads to be spawned by the agent.	Set to 1 for the agent that resides on the host of the PEM server; 0 for all other agents.
enable_smtp	When set to true for multiple PEM Agents (7.13 or lesser) it may send more duplicate emails. Whereas for PEM Agents (7.14 or higher) it may send lesser duplicate emails.	true for PEM server host; false for all others.
enable_snmp	When set to true for multiple PEM Agents (7.13 or lesser) it may send more duplicate traps. Whereas for PEM Agents (7.14 or higher) it may send lesser duplicate traps.	true for PEM server host; false for all others.
enable_nagios	When set to true, Nagios alerting is enabled.	true for PEM server host; false for all others.
enable_webhook	When set to true, Webhook alerting is enabled.	true for PEM server host; false for all others.
max_webhook_retries	Set maximum number of times pemAgent should retry to call webhooks on failure.	Default 3.
connect_timeout	The max time in seconds (a decimal integer string) that the agent will wait for a connection.	Not set by default; set to 0 to indicate the agent should wait indefinitely.
allow_server_restart	If set to TRUE, the agent can restart the database server that it monitors. Some PEM features may be enabled/disabled, depending on the value of this parameter.	False
max_connections	The maximum number of probe connections used by the connection throttler.	0 (an unlimited number)

Parameter Name	Description	Default Value
connection_lifetime	Use ConnectionLifetime (or connection_lifetime) to specify the minimum number of seconds an open but idle connection is retained. This parameter is ignored if the value specified in MaxConnections is reached and a new connection (to a different database) is required to satisfy a waiting request.	By default, set to 0 (a connection is dropped when the connection is idle after the agent's processing loop).
allow_batch_probes	If set to TRUE, the user will be able to create batch probes using the custom probes feature.	false
heartbeat_connection	When set to TRUE, a dedicated connection is used for sending the heartbeats.	false
batch_script_dir	Provide the path where script file (for alerting) will be stored.	/tmp
connection_custom_setup	Use to provide SQL code that will be invoked when a new connection with a monitored server is made.	Not set by default.
ca_file	Provide the path where the CA certificate resides.	Not set by default.
batch_script_user	Provide the name of the user that should be used for executing the batch/shell scripts.	None
webhook_ssl_key	The complete path to the webhook's SSL client key file.	
webhook_ssl_crt	The complete path to the webhook's SSL client certificate file.	
webhook_ssl_crl	The complete path of the CRL file to validate webhook server certificate.	
webhook_ssl_ca_crt	The complete path to the webhook's SSL ca certificate file.	
allow_insecure_webhooks	When set to true, allow webhooks to call with insecure flag.	false

On 64 bit Windows systems, PEM registry entries are located in:

`HKEY_LOCAL_MACHINE\Software\Wow6432Node\EnterpriseDB\PEM\agent`

The registry contains the following entries:

Parameter Name	Description	Default Value
PEM_HOST	The IP address or hostname of the PEM server.	127.0.0.1.
PEM_PORT	The database server port to which the agent connects to communicate with the PEM server.	Port 5432.
AgentID	A unique identifier assigned to the PEM agent.	The first agent is '1', the second agent is '2', and so on.
AgentKeyPath	The complete path to the PEM agent's key file.	%APPDATA%\Roaming\pem\agent.key.
AgentCrtPath	The complete path to the PEM agent's certificate file.	%APPDATA%\Roaming\pem\agent.crt
AgentFlagDir	Used for HA support. Specifies the directory path checked for requests to take over monitoring another server. Requests are made in the form of a file in the specified flag directory.	Not set by default.
LogLevel	Log level specifies the type of event that will be written to the PEM log files.	warning
LongWait	The maximum length of time (in seconds) that the PEM agent will wait before attempting to connect to the PEM server if an initial connection attempt fails.	30 seconds

Parameter Name	Description	Default Value
shortWait	The minimum length of time (in seconds) that the PEM agent will wait before checking which probes are next in the queue (waiting to run).	10 seconds
AlertThreads	The number of alert threads to be spawned by the agent.	Set to 1 for the agent that resides on the host of the PEM server; 0 for all other agents.
EnableSMTP	When set to true, the SMTP email feature is enabled.	true for PEM server host; false for all others.
EnableSNMP	When set to true, the SNMP trap feature is enabled.	true for PEM server host; false for all others.
EnableWebhook	When set to true, Webhook alerting is enabled.	true for PEM server host; false for all others.
MaxWebhookRetries	Set maximum number of times pemAgent should retry to call webhooks on failure.	Default 3.
ConnectTimeout	The max time in seconds (a decimal integer string) that the agent will wait for a connection.	Not set by default; if set to 0, the agent will wait indefinitely.
AllowServerRestart	If set to TRUE, the agent can restart the database server that it monitors. Some PEM features may be enabled/disabled, depending on the value of this parameter.	true
MaxConnections	The maximum number of probe connections used by the connection throttler.	0 (an unlimited number)
ConnectionLifetime	Use ConnectionLifetime (or connection_lifetime) to specify the minimum number of seconds an open but idle connection is retained. This parameter is ignored if the value specified in MaxConnections is reached and a new connection (to a different database) is required to satisfy a waiting request.	By default, set to 0 (a connection is dropped when the connection is idle after the agent's processing loop).
AllowBatchProbes	If set to TRUE, the user will be able to create batch probes using the custom probes feature.	false
HeartbeatConnection	When set to TRUE, a dedicated connection is used for sending the heartbeats.	false
BatchScriptDir	Provide the path where script file (for alerting) will be stored.	/tmp
ConnectionCustomSetup	Use to provide SQL code that will be invoked when a new connection with a monitored server is made.	Not set by default.
ca_file	Provide the path where the CA certificate resides.	Not set by default.
AllowBatchJobSteps	If set to true, the batch/shell scripts will be executed using Administrator user account.	None
WebhookSSLKey	The complete path to the webhook's SSL client key file.	
WebhookSSLCrt	The complete path to the webhook's SSL client certificate file.	
WebhookSSLCrL	The complete path of the CRL file to validate webhook server certificate.	
WebhookSSLCaCrt	The complete path to the webhook's SSL ca certificate file.	
AllowInsecureWebhooks	When set to true, allow webhooks to call with insecure flag.	false

Agent Properties

The PEM Agent **Properties** dialog provides information about the PEM agent from which the dialog was opened; to open the dialog, right-click on an agent name in the PEM client tree control, and select **Properties** from the context menu.



Use fields on the PEM Agent **Properties** dialog to review or modify information about the PEM agent:

- The **Description** field displays a modifiable description of the PEM agent. This description is displayed in the tree control of the PEM client.
- You can use groups to organize your servers and agents in the PEM client tree control. Use the **Group** drop-down listbox to select the group in which the agent will be displayed.
- Use the **Team** field to specify the name of the group role that should be able to access servers monitored by the agent; the servers monitored by this agent will be displayed in the PEM client tree control to connected team members. Please note that this is a convenience feature. The Team field does not provide true isolation, and should not be used for security purposes.
- The **Heartbeat interval** fields display the length of time that will elapse between reports from the PEM agent to the PEM server. Use the selectors next to the **Minutes** or **Seconds** fields to modify the interval.



Use the fields on the **Job Notifications** tab to configure the email notification settings on agent level:

- Use the **Override default configuration?** switch to specify if you want the agent level job notification settings to override the default job notification settings. If you select **Yes** for this switch, you can use the rest of the settings on this dialog to define when and to whom the job notifications should be sent. Please note that the rest of the settings on this dialog work only if you enable the **Override default configuration?** switch.
- Use the **Email on job completion?** switch to specify if the job notification should be sent on the successful job completion.
- Use the **Email on a job failure?** switch to specify if the job notification should be sent on the failure of a job.
- Use the **Email group** field to specify the email group to whom the job notification should be sent.

Parameter	Value	Category
Agent Id	1	configuration
Running as root?	true	capability
Running as User	root	capability
Platform	"CentOS Linux 7 (Core)"	capability
Architecture	x64	capability
PEM host	127.0.0.1	configuration
PEM port	5444	configuration
Log level	debug1	configuration
Agent SSL key path	/root/.pem//agent1.key	configuration
Agent SSL crt path	/root/.pem//agent1.crt	configuration
Long wait	30	configuration
Short wait	10	configuration
Alert threads	1	configuration

At the bottom are buttons for **Cancel**, **Reset**, and **Save**.

The **Agent Configurations** tab displays all the current configurations and capabilities of a agent.

- The **Parameter** column displays a list of parameters.
- The **Value** column displays the current value of the corresponding parameter.
- The **Category** column displays the category of the corresponding parameter; it can be either **configuration** or **capability**.

2 PEM Agent User Guide

PEM is composed of three primary components: PEM server, PEM agent, and PEM web interface. The PEM agent is responsible for performing tasks on each managed machine and collecting statistics for the database server and operating system.

For information about the platforms and versions supported by PEM, visit the EDB website at:

<https://www.enterprisedb.com/services-support/edb-supported-products-and-platforms#pem>

Please note: PEM 8.0.1 is no longer supported on CentOS/RHEL/OEL 6.x platforms. It is strongly recommended that EDB products running on these platforms be migrated to a supported platform.

For information about the installation, uninstallation, or upgrading of a PEM Agent, visit the EDB website at:

<https://www.enterprisedb.com/docs/pem/latest/>

This document provides information that is required to work with PEM agents. The guide will acquaint you with the basic registering, configuration, and management of agents. The guide is broken up into the following core sections:

- Postgres Enterprise Manager - Overview - This section provides an overview of PEM architecture.
- Registering a PEM Agent - This section provides information about registration of a PEM agent.
- Managing a PEM agent - This section provides information about configuring and managing a PEM agent.
- Troubleshooting for PEM agent - This section provides information about troubleshooting for PEM agents.

This document uses *Postgres* to mean either the PostgreSQL or EDB Postgres Advanced Server database.

2.1 PEM Architecture

Postgres Enterprise Manager (PEM) is a tool designed to monitor and manage multiple Postgres servers through a single GUI interface. PEM is capable of monitoring the following areas of the infrastructure:

!!! Note The term Postgres refers to either PostgreSQL or EDB Postgres Advanced Server.

- Hosts - One or more servers (physical or virtual) and their operating systems.
- Servers - One or more instances of PostgreSQL or EDB Postgres Advanced Server running on a host.
- Databases - One or more databases and the schema objects (tables, indexes, etc.) within them.

PEM consists of a number of individual software components; the individual components are described below.

- PEM Server - The PEM Server is used as the data repository for monitoring data and as a server to which both Agents and Clients connect. The PEM server consists of an instance of PostgreSQL and an associated database for storage of monitoring data, and a server that provides web services.
- PEM Agent - The PEM Agent is responsible for executing tasks and reporting statistics from the Agent host and monitored Postgres instances to the PEM server. A single PEM Agent can monitor multiple installed instances of Postgres that reside on one or many hosts.
- PEM Web Client - The PEM web interface allows you to manage and monitor Postgres servers and utilize PEM extended functionality. The web interface software is installed with the PEM server and is accessed via any supported web browser.

- SQL Profiler - SQL Profiler is a Postgres server plugin to record the monitoring data and query plans to be analysed by the SQL Profiler tool in PEM. This is an optional component of PEM, but the plugin must be installed into each instance of Postgres with which you wish to use the SQL Profiler tool. The SQL Profiler may be used with any supported version of an EDB distribution of a PostgreSQL server or Advanced Server (not just those managed through the PEM server). See the [PEM SQL Profiler Configuration Guide](#) for details and supported versions.

PEM architecture

The following architectural diagram illustrates the relationships between the PEM server, clients, and managed as well as unmanaged Postgres servers.



PEM Architecture

The PEM Server



The PEM server consists of an instance of Postgres, an instance of the Apache web-server providing web services to the client, and a PEM Agent. PEM utilizes a server-side cryptographic plugin to generate authentication certificates.

The instance of Postgres (a database server) and an instance of the Apache web-server (HTTPD) can be on the same host or on separate hosts.

- **Postgres Instance (Database server)** - This is the backend database server. It hosts a database named pem which acts as the repository for PEM Server. The pem database contains several schemas that store metric data collected from each monitored host, server, and database.
 - pem - This schema is the core of the PEM application. It contains the definitions of configuration functions, tables, or views required by the application.
 - pemdata - This schema stores the current snapshot of the monitored data.
 - pemhistory - This schema stores the historical monitored data.
- **Apache Web Server (HTTPD)** - The PEM Web Application is deployed as a WSGI application with HTTPD to provide web services to the client. It is comprised of the following:
 - Web content presentation - The presentation layer is created by the Web Application (for example Browser, login page,..).
 - Rest API - The REST API allows integration with other apps and services.
 - Database Server Administration/Management - Database server administration and management activities like CREATE, ALTER, DROP, etc. can be performed for managed as well as unmanaged servers.
 - Dashboard/Chart generation - Internally, the web application includes functionality that generates Dashboards and Charts.
 - Management Tools - The Audit Manager, Capacity Manager, Log Manager, Postgres Expert, Postgres Log Analysis Expert, and the Tuning Wizard are made available in the Web Application.
 - Other tools provide functionality on managed or unmanaged servers:
 - SQL Profiler UI Integration - SQL Profiler generates easily analyzed traces of session content.
 - Query Editor/Data View - The Query editor allows you to query, edit, and view data.
 - Debugger - The Debugger helps you debug queries.
 - Performance Diagnostics - Performance Diagnostics help you analyze the performance of Advanced Server.

We recommend that you use a dedicated machine to host production instances of the PEM backend database. The host

may be subject to high levels of data throughput, depending on the number of database servers that are being monitored and the workloads the servers are processing.

The PEM Agent



The PEM Agent is responsible for the collection of monitoring data from the machine and operating system, as well as from each of the Postgres instances to which they are bound. Each PEM Agent can monitor one physical or virtual machine and is capable of monitoring multiple database servers locally - installed on the same system, or remotely - installed on other systems. It is also responsible for executing other tasks that may be scheduled by the user (for example, server shutdowns, SQL Profiler traces, user-defined jobs).

A PEM Agent is installed by default on the PEM Server along with the installation of the PEM Server. It is generally referred to as a PEM Agent on the PEM Host. Separately, the PEM Agent can also be installed on the other servers hosting the Postgres instances to be monitored using PEM.

Whether monitoring locally or remotely, the PEM Agent connects to the PEM Server using PostgreSQL's libpq, using SSL certificate-based authentication. The PEM Agent installer in Windows and pemworker CLI in Linux is responsible for registering each agent with the PEM Server, and generating and installing the required certificates.

Please note that there is only one-way traffic between the PEM Agent and PEM Server; the PEM Agent always connects to the PEM Server.

The PEM Agent must be able to connect to each database server that it monitors. This connection is made over a TCP/IP connection (or optionally a Unix Domain Socket on Unix hosts), and may optionally use SSL. The user must configure the connection and authentication to the monitored server.

Once configured, each agent collects statistics and other information on the host and each database server and database that it monitors. Each piece of information is known as a metric and is collected by a probe. Most probes will collect multiple metrics at once for efficiency. Examples of the metrics collected include:

- Disk I/O statistics
- Network statistics
- Database server version string
- Database server configuration option (GUC) values
- Table access statistics
- Table and index sizes

A list of PEM probes can be found [here](#).

By default, the PEM Agent bound to the database server collects the OS/Database monitoring statistics and also runs any scheduled tasks/jobs for that particular database server, storing data in the pem database on the PEM server.

The Alert processing, SNMP/SMTP spoolers, and Nagios Spooler data is stored in the **pem** database on the PEM server

and is then processed by the PEM Agent on the PEM Host by default. However, processing by other PEM Agents can be enabled by adjusting the SNMP/SMTP and Nagios parameters of the PEM Agents.

To see more information about these parameters see [Server Configuration](#).

The PEM Web Client

The PEM client is a web-based application that runs in supported browsers. The client's web interface connects to the PEM server and allows direct management of managed or unmanaged servers, and the databases and schemas that reside on them.

The client allows you to use PEM functionality that makes use of the data logged on the server through features such as the dashboards, the Postgres Log Analysis Expert, and Capacity Manager.

The SQL Profiler Plugin

You are not required to install the SQL Profiler plugin on every server, but you must install and configure the plugin on each server on which you wish to use the SQL Profiler. You may also want to install and configure SQL Profiler on un-monitored development servers. For ad-hoc use also, you may temporarily install the SQL Profiler plugin.

The plugin is installed with the EDB Postgres Advanced Server distribution but must be installed separately for use with PostgreSQL. The SQL Profiler installer is available from the [EDB website](#).

SQL Profiler may be used on servers that are not managed through PEM, but to perform scheduled traces, a server must have the plugin installed, and must be managed by an installed and configured PEM agent.

For more information about using SQL Profiler, see the [PEM SQL Profiler Guide](#)

2.2 Registering an Agent

Each PEM agent must be *registered* with the PEM server. The registration process provides the PEM server with the information it needs to communicate with the agent. The PEM agent graphical installer for Windows supports self-registration for the agent. You must use the `pemworker` utility to register the agent if the agent is on a Linux host.

The RPM installer places the PEM agent in the `/usr/edb/pem/agent/bin` directory. To register an agent, include the `--register-agent` keywords along with registration details when invoking the `pemworker` utility:

```
pemworker --register-agent
```

Append command line options to the command string when invoking the `pemworker` utility. Each option should be followed by a corresponding value:

Option	Description
<code>--pem-server</code>	Specifies the IP address of the PEM backend database server. This parameter is required.
<code>--pem-port</code>	Specifies the port of the PEM backend database server. The default value is 5432.

Option	Description
--pem-user	Specifies the name of the Database user (having superuser privileges) of the PEM backend database server. This parameter is required.
--pem-agent-user	Specifies the agent user to connect the PEM server backend database server.
--cert-path	Specifies the complete path to the directory in which certificates will be created. If you do not provide a path, certificates will be created in: On Linux, <code>~/.pem</code> On Windows, <code>%APPDATA%/pem</code>
--config-dir	Specifies the directory path where configuration file can be found. The default is the <code><pemworker path>/..etc.</code>
--display-name	Specifies a user-friendly name for the agent that will be displayed in the PEM Browser tree control. The default is the system hostname.
--force-registration	Include the force_registration clause to instruct the PEM server to register the agent with the arguments provided; this clause is useful if you are overriding an existing agent configuration. The default value is Yes.
--group	The name of the group in which the agent will be displayed.
--team	The name of the database role, on the PEM backend database server, that should have access to the monitored database server.
--owner	The name of the database user, on the PEM backend database server, who will own the agent.
--allow_server_restart	Enable the allow-server_restart parameter to allow PEM to restart the monitored server. The default value is True.
--allow-batch-probes	Enable the allow-batch-probes parameter to allow PEM to run batch probes on this agent. The default value is False.
--batch-script-user	Specifies the operating system user that should be used for executing the batch/shell scripts. The default value is none; the scripts will not be executed if you leave this parameter blank or the specified user does not exist.
--enable-heartbeat-connection	Enable the enable-heartbeat-connection parameter to create a dedicated heartbeat connection between PEM Agent and server to update the active status. The default value is False.
--enable-smtp	Enable the enable-smtp parameter to allow the PEM agent to send the email on behalf of the PEM server. The default value is False.
--enable-snmp	Enable the enable-snmp parameter to allow the PEM agent to send the SNMP traps on behalf of the PEM server. The default value is False.
-o	Specify if you want to override the configuration file options.

Before using any PEM feature for which a database server restart is required by the pemagent (such as Audit Manager, Log Manager, or Tuning Wizard), you must first set the value for `allow_server_restart` to `true` in the `agent.cfg` file.

!!! Note When configuring a shell/batch script run by a PEM agent that has PEM 7.11 or later version installed, the user for the `batch_script_user` parameter must be specified. It is strongly recommended that a non-root user is used to run the scripts. Using the root user may result in compromising the data security and operating system security. However, if you want to restore the pemagent to its original settings using root user to run the scripts, then the `batch_script_user` parameter value must be set to `root`.

You can use the `PEM_SERVER_PASSWORD` environment variable to set the password of the PEM Admin User. If the `PEM_SERVER_PASSWORD` is not set, the server will use the `PGPASSWORD` or `.pgpass` file when connecting to the PEM Database Server.

Failure to provide the password will result in a password authentication error; you will be prompted for any other

required but omitted information. When the registration is complete, the server will confirm that the agent has been successfully registered.

Setting PEM Agent Configuration Parameters

The PEM agent RPM installer creates a sample configuration file named `agent.cfg.sample` in the `/usr/edb/pem/agent/etc` directory. When you register the PEM agent, the `pemworker` program creates the actual agent configuration file (named `agent.cfg`). You must modify the `agent.cfg` file, adding the following configuration parameter:

```
heartbeat_connection = true
```

You must also add the location of the `ca-bundle.crt` file (the certificate authority). By default, the installer creates a `ca-bundle.crt` file in the location specified in your `agent.cfg.sample` file. You can copy the default parameter value from the sample file, or, if you use a `ca-bundle.crt` file that is stored in a different location, specify that value in the `ca_file` parameter:

```
ca_file=/usr/libexec/libcurl-pem7/share/certs/ca-bundle.crt
```

Then, use a platform-specific command to start the PEM agent service; the service is named `pemagent`.

On a RHEL or CentOS 7.x or 8.x host, use `systemctl` to start the service:

```
systemctl start pemagent
```

The service will confirm that it is starting the agent; when the agent is registered and started, it will be displayed on the `Global Overview` dashboard and in the Object browser tree control of the PEM web interface.

For information about using the `pemworker` utility to register a server, please see the [PEM Administrator's Guide](#)

Using a non-root User Account to Register a PEM Agent

To use a non-root user account to register a PEM agent, you must first install the PEM agent as a root user. After installation, assume the identity of a non-root user (for example, `edb`) and perform the following steps:

1. Create the `.pem` directory and `logs` directory and assign read, write, and execute permissions to the file:

```
mkdir /home/<edb>/ .pem
mkdir /home/<edb>/ .pem/logs
chmod 700 /home/<edb>/ .pem
chmod 700 /home/<edb>/ .pem/logs
```

2. Register the agent with PEM server:

```
./pemworker --register-agent --pem-server <172.19.11.230> --pem-user <postgres> --pem-port <5432> --display-name <non_root> --cert-path /home/<edb> --config-dir /home/<edb>
```

The above command creates agent certificates and an agent configuration file (`agent.cfg`) in the `~/home/edb/.pem` directory. Use the following command to assign read and write permissions to these files:

```
``chmod -R 600 /home/edb/.pem/agent*``
```

3. Change the parameters of the `agent.cfg` file:

```
agent_ssl_key=/home/edb/.pem/agent<id>.key
agent_ssl_crt=/home/edb/.pem/agent<id>.crt
log_location=/home/edb/.pem/worker.log
agent_log_location=/home/edb/.pem/agent.log
```

4. Update the values for the configuration file path and the user in the `pemagent` service file:

```
User=edb
ExecStart=/usr/edb/pem/agent/bin/pemagent -c /home/edb/.pem/agent.cfg
```

5. Stop the agent process, and then restart the agent service using the non-root user:

```
sudo systemctl start/stop/restart pemagent
```

6. Check the agent status on PEM dashboard.

2.3 Managing a PEM Agent

The sections that follow provide information about the behavior and management of a PEM agent.

Agent Privileges

By default, the PEM agent is installed with `root` privileges for the operating system host and superuser privileges for the database server. These privileges allow the PEM agent to invoke unrestricted probes on the monitored host and database server about system usage, retrieving and returning the information to the PEM server.

Please note that PEM functionality diminishes as the privileges of the PEM agent decrease. For complete functionality, the PEM agent should run as `root`. If the PEM agent is run under the database server's service account, PEM probes will not have complete access to the statistical information used to generate reports, and functionality will be limited to the capabilities of that account. If the PEM agent is run under another lesser-privileged account, functionality will be limited even further.

If you limit the operating system privileges of the PEM agent, some of the PEM probes will not return information, and the following functionality may be affected:

Probe or Action	Operating System	PEM Functionality Affected
Data And Logfile Analysis	Linux/ Windows	The Postgres Expert will be unable to access complete information.
Session Information	Linux	The per-process statistics will be incomplete.
PG HBA	Linux/ Windows	The Postgres Expert will be unable to access complete information.
Service restart functionality	Linux/ Windows	The Audit Log Manager, Server Log Manager Log Analysis Expert and PEM may be unable to apply requested modifications.

Probe or Action	Operating System	PEM Functionality Affected
Package Deployment	Linux/ Windows	PEM will be unable to run downloaded installation modules.
Batch Task	Windows	PEM will be unable to run scheduled batch jobs in Windows.
Collect data from server (root access required)	Linux/ Windows	Columns such as swap usage, CPU usage, IO read, IO write will be displayed as 0 in the session activity dashboard.

!!! Note The above-mentioned list is not comprehensive, but should provide an overview of the type of functionality that will be limited.

If you restrict the database privileges of the PEM agent, the following PEM functionality may be affected:

Probe	Operating System	PEM Functionality Affected
Audit Log Collection	Linux/Windows	PEM will receive empty data from the PEM database.
Server Log Collection	Linux/Windows	PEM will be unable to collect server log information.
Database Statistics	Linux/Windows	The Database/Server Analysis dashboards will contain incomplete information.
Session Waits/System Waits	Linux/Windows	The Session/System Waits dashboards will contain incomplete information.
Locks Information	Linux/Windows	The Database/Server Analysis dashboards will contain incomplete information.
Streaming Replication	Linux/Windows	The Streaming Replication dashboard will not display information.
Slony Replication	Linux/Windows	Slony-related charts on the Database Analysis dashboard will not display information.
Tablespace Size	Linux/Windows	The Server Analysis dashboard will not display complete information.
xDB Replication	Linux/Windows	PEM will be unable to send xDB alerts and traps.

If the probe is querying the operating system with insufficient privileges, the probe may return a **permission denied** error.

If the probe is querying the database with insufficient privileges, the probe may return a **permission denied** error or display the returned data in a PEM chart or graph as an empty value.

When a probe fails, an entry will be written to the log file that contains the name of the probe, the reason the probe failed, and a hint that will help you resolve the problem.

You can view probe-related errors that occurred on the server in the **Probe Log** dashboard, or review error messages in the PEM worker log files. On Linux, the default location of the log file is:

```
/var/log/pem/worker.log
```

On Windows, log information is available on the **Event Viewer**.

Agent Configuration

A number of user-configurable parameters and registry entries control the behavior of the PEM agent. You may be required to modify the PEM agent's parameter settings to enable some PEM functionality. After modifying values in the PEM agent configuration file, you must restart the PEM agent to apply any changes.

With the exception of the **PEM_MAXCONN** parameter, we strongly recommend against modifying any of the configuration parameters or registry entries listed below without first consulting EDB support experts *unless* the modifications are required to enable PEM functionality.

On Linux systems, PEM configuration options are stored in the **agent.cfg** file, located in **/usr/edb/pem/agent/etc**. The **agent.cfg** file contains the following entries:

Parameter Name	Description	Default Value
pem_host	The IP address or hostname of the PEM server.	127.0.0.1.
pem_port	The database server port to which the agent connects to communicate with the PEM server.	Port 5432.
pem_agent	A unique identifier assigned to the PEM agent.	The first agent is '1', the second agent is '2', and so on.
agent_ssl_key	The complete path to the PEM agent's key file.	/root/.pem/agent.key
agent_ssl_crt	The complete path to the PEM agent's certificate file.	/root/.pem/agent.crt
agent_flag_dir	Used for HA support. Specifies the directory path checked for requests to take over monitoring another server. Requests are made in the form of a file in the specified flag directory.	Not set by default.
log_level	Log level specifies the type of event that will be written to the PEM log files.	warning
log_location	Specifies the location of the PEM worker log file.	127.0.0.1.
agent_log_location	Specifies the location of the PEM agent log file.	/var/log/pem/agent.log
long_wait	The maximum length of time (in seconds) that the PEM agent will wait before attempting to connect to the PEM server if an initial connection attempt fails.	30 seconds
short_wait	The minimum length of time (in seconds) that the PEM agent will wait before checking which probes are next in the queue (waiting to run).	10 seconds
alert_threads	The number of alert threads to be spawned by the agent.	Set to 1 for the agent that resides on the host of the PEM server; 0 for all other agents.
enable_smtp	When set to true for multiple PEM Agents (7.13 or lesser) it may send more duplicate emails. Whereas for PEM Agents (7.14 or higher) it may send lesser duplicate emails.	true for PEM server host; false for all others.
enable_snmp	When set to true for multiple PEM Agents (7.13 or lesser) it may send more duplicate traps. Whereas for PEM Agents (7.14 or higher) it may send lesser duplicate traps.	true for PEM server host; false for all others.
enable_nagios	When set to true, Nagios alerting is enabled.	true for PEM server host; false for all others.
enable_webhook	When set to true, Webhook alerting is enabled.	true for PEM server host; false for all others.
max_webhook_retries	Set maximum number of times pemAgent should retry to call webhooks on failure.	Default 3.

Parameter Name	Description	Default Value
connect_timeout	The max time in seconds (a decimal integer string) that the agent will wait for a connection.	Not set by default; set to 0 to indicate the agent should wait indefinitely.
allow_server_restart	If set to TRUE, the agent can restart the database server that it monitors. Some PEM features may be enabled/disabled, depending on the value of this parameter.	False
max_connections	The maximum number of probe connections used by the connection throttler.	0 (an unlimited number)
connection_lifetime	Use ConnectionLifetime (or connection_lifetime) to specify the minimum number of seconds an open but idle connection is retained. This parameter is ignored if the value specified in MaxConnections is reached and a new connection (to a different database) is required to satisfy a waiting request.	By default, set to 0 (a connection is dropped when the connection is idle after the agent's processing loop).
allow_batch_probes	If set to TRUE, the user will be able to create batch probes using the custom probes feature.	false
heartbeat_connection	When set to TRUE, a dedicated connection is used for sending the heartbeats.	false
batch_script_dir	Provide the path where script file (for alerting) will be stored.	/tmp
connection_custom_setup	Use to provide SQL code that will be invoked when a new connection with a monitored server is made.	Not set by default.
ca_file	Provide the path where the CA certificate resides.	Not set by default.
batch_script_user	Provide the name of the user that should be used for executing the batch/shell scripts.	None
webhook_ssl_key	The complete path to the webhook's SSL client key file.	
webhook_ssl_crt	The complete path to the webhook's SSL client certificate file.	
webhook_ssl_crl	The complete path of the CRL file to validate webhook server certificate.	
webhook_ssl_ca_crt	The complete path to the webhook's SSL ca certificate file.	
allow_insecure_webhooks	When set to true, allow webhooks to call with insecure flag.	false

On 64 bit Windows systems, PEM registry entries are located in:

`HKEY_LOCAL_MACHINE\Software\Wow6432Node\EnterpriseDB\PEM\agent`

The registry contains the following entries:

Parameter Name	Description	Default Value
PEM_HOST	The IP address or hostname of the PEM server.	127.0.0.1.
PEM_PORT	The database server port to which the agent connects to communicate with the PEM server.	Port 5432.
AgentID	A unique identifier assigned to the PEM agent.	The first agent is '1', the second agent is '2', and so on.
AgentKeyPath	The complete path to the PEM agent's key file.	%APPDATA%\Roaming\pem\agent.key.
AgentCrtPath	The complete path to the PEM agent's certificate file.	%APPDATA%\Roaming\pem\agent.crt

Parameter Name	Description	Default Value
AgentFlagDir	Used for HA support. Specifies the directory path checked for requests to take over monitoring another server. Requests are made in the form of a file in the specified flag directory.	Not set by default.
LogLevel	Log level specifies the type of event that will be written to the PEM log files.	warning
LongWait	The maximum length of time (in seconds) that the PEM agent will wait before attempting to connect to the PEM server if an initial connection attempt fails.	30 seconds
shortWait	The minimum length of time (in seconds) that the PEM agent will wait before checking which probes are next in the queue (waiting to run).	10 seconds
AlertThreads	The number of alert threads to be spawned by the agent.	Set to 1 for the agent that resides on the host of the PEM server; 0 for all other agents.
EnableSMTP	When set to true, the SMTP email feature is enabled.	true for PEM server host; false for all others.
EnableSNMP	When set to true, the SNMP trap feature is enabled.	true for PEM server host; false for all others.
EnableWebhook	When set to true, Webhook alerting is enabled.	true for PEM server host; false for all others.
MaxWebhookRetries	Set maximum number of times pemAgent should retry to call webhooks on failure.	Default 3.
ConnectTimeout	The max time in seconds (a decimal integer string) that the agent will wait for a connection.	Not set by default; if set to 0, the agent will wait indefinitely.
AllowServerRestart	If set to TRUE, the agent can restart the database server that it monitors. Some PEM features may be enabled/disabled, depending on the value of this parameter.	true
MaxConnections	The maximum number of probe connections used by the connection throttler.	0 (an unlimited number)
ConnectionLifetime	Use ConnectionLifetime (or connection_lifetime) to specify the minimum number of seconds an open but idle connection is retained. This parameter is ignored if the value specified in MaxConnections is reached and a new connection (to a different database) is required to satisfy a waiting request.	By default, set to 0 (a connection is dropped when the connection is idle after the agent's processing loop).
AllowBatchProbes	If set to TRUE, the user will be able to create batch probes using the custom probes feature.	false
HeartbeatConnection	When set to TRUE, a dedicated connection is used for sending the heartbeats.	false
BatchScriptDir	Provide the path where script file (for alerting) will be stored.	/tmp
ConnectionCustomSetup	Use to provide SQL code that will be invoked when a new connection with a monitored server is made.	Not set by default.
ca_file	Provide the path where the CA certificate resides.	Not set by default.
AllowBatchJobSteps	If set to true, the batch/shell scripts will be executed using Administrator user account.	None

Parameter Name	Description	Default Value
WebhookSSLKey	The complete path to the webhook's SSL client key file.	
WebhookSSLCrt	The complete path to the webhook's SSL client certificate file.	
WebhookSSLCrL	The complete path of the CRL file to validate webhook server certificate.	
WebhookSSLCaCrt	The complete path to the webhook's SSL ca certificate file.	
AllowInsecureWebhooks	When set to true, allow webhooks to call with insecure flag.	false

Agent Properties

The PEM Agent **Properties** dialog provides information about the PEM agent from which the dialog was opened; to open the dialog, right-click on an agent name in the PEM client tree control, and select **Properties** from the context menu.



PEM Agent Properties dialog - General tab

Use fields on the PEM Agent **Properties** dialog to review or modify information about the PEM agent:

- The **Description** field displays a modifiable description of the PEM agent. This description is displayed in the tree control of the PEM client.
- You can use groups to organize your servers and agents in the PEM client tree control. Use the **Group** drop-down listbox to select the group in which the agent will be displayed.
- Use the **Team** field to specify the name of the group role that should be able to access servers monitored by the agent; the servers monitored by this agent will be displayed in the PEM client tree control to connected team members. Please note that this is a convenience feature. The Team field does not provide true isolation, and should not be used for security purposes.
- The **Heartbeat interval** fields display the length of time that will elapse between reports from the PEM agent to the PEM server. Use the selectors next to the **Minutes** or **Seconds** fields to modify the interval.



PEM Agent Properties dialog - Job Notifications tab

Use the fields on the **Job Notifications** tab to configure the email notification settings on agent level:

- Use the **Override default configuration?** switch to specify if you want the agent level job notification settings to override the default job notification settings. If you select **Yes** for this switch, you can use the rest of the settings on this dialog to define when and to whom the job notifications should be sent. Please note that the rest of the settings on this dialog work only if you enable the **Override default configuration?** switch.
- Use the **Email on job completion?** switch to specify if the job notification should be sent on the successful job completion.
- Use the **Email on a job failure?** switch to specify if the job notification should be sent on the failure of a job.
- Use the **Email group** field to specify the email group to whom the job notification should be sent.

Parameter	Value	Category
Agent Id	1	configuration
Running as root?	true	capability
Running as User	root	capability
Platform	"CentOS Linux 7 (Core)"	capability
Architecture	x64	capability
PEM host	127.0.0.1	configuration
PEM port	5444	configuration
Log level	debug1	configuration
Agent SSL key path	/root/.pem//agent1.key	configuration
Agent SSL crt path	/root/.pem//agent1.crt	configuration
Long wait	30	configuration
Short wait	10	configuration
Alert threads	1	configuration

At the bottom are buttons for **Cancel**, **Reset**, and **Save**.

PEM Agent Properties dialog - Agent Configurations tab

The **Agent Configurations** tab displays all the current configurations and capabilities of a agent.

- The **Parameter** column displays a list of parameters.

- The **Value** column displays the current value of the corresponding parameter.
- The **Category** column displays the category of the corresponding parameter; it can be either **configuration** or **capability**.

2.4 PEM Agent Troubleshooting

Restoring a Deleted PEM Agent

If an agent has been deleted from the `pem.agent` table then you cannot restore it. You will need to use the pemworker utility to re-register the agent.

If an agent has been deleted from PEM Web client but still has an entry in the `pem.agent` table with value of `active = f`, then you can use the following steps to restore the agent:

1. Use the following command to check the values of the `id` and `active` fields:

```
pem=# SELECT * FROM pem.agent;
```

2. Update the status for the agent to `true` in the `pem.agent` table:

```
pem=# UPDATE pem.agent SET active=true WHERE id=<x>;
```

Where `x` is the identifier that was displayed in the output of the query used in step 1.

3. Refresh the PEM web client.

The deleted agent will be restored again. However, the servers that were bound to that particular agent might appear to be down. To resolve this issue, you need to modify the PEM agent properties of the server to add the bound agent again; after the successful modification, the servers will be displayed as running properly.

Using the Command Line to Delete a PEM Agent with Down or Unknown Status

Using the PEM web interface to delete PEM agents with `Down` or `Unknown` status may be difficult if the number of such agents is large. In such a situation, you might want to use the command line interface to delete `Down` or `Unknown` agents.

1. Use the following query to delete the agents that are `Down` for more than `N` number of hours:

```
UPDATE pem.agent SET active=false WHERE id IN
(SELECT a.id FROM pem.agent
a JOIN pem.agent_heartbeat b ON (b.agent_id=a.id)
WHERE a.id IN
(SELECT agent_id FROM pem.agent_heartbeat WHERE (EXTRACT (HOUR FROM now())-
EXTRACT (HOUR FROM last_heartbeat)) > <N> ));
```

2. Use the following query to delete the agents with an `Unknown` status:

```
UPDATE pem.agent SET active=false WHERE id IN
```

```
(SELECT id FROM pem.agent WHERE id NOT IN
(SELECT agent_id FROM pem.agent_heartbeat));
```

3 PEM BART Management Guide

This guide will acquaint you with the dialogs that are built into the Postgres Enterprise Manager (PEM) web interface that make it easier for you to monitor and manage BART.

This document uses *Postgres* to mean either the PostgreSQL or EDB Postgres Advanced Server database.

3.1 Managing a BART Server

Postgres Enterprise Manager (PEM) is designed to assist database administrators, system architects, and performance analysts when administering, monitoring, and tuning PostgreSQL and Advanced Server database servers.

The EDB Backup and Recovery Tool (BART) is an administrative utility providing simplified backup and recovery management for multiple local or remote EDB Postgres Advanced Server and PostgreSQL database servers. For more information about BART, please visit the EDB website at:

<https://www.enterprisedb.com/enterprise-postgres/edb-postgres-backup-and-recovery-tool>

From PEM version 7.10 onwards, you can manage a BART server through PEM console. PEM provides a user-friendly interface that allows you to manage your BART server and perform all the BART operations from PEM console.

Prerequisites

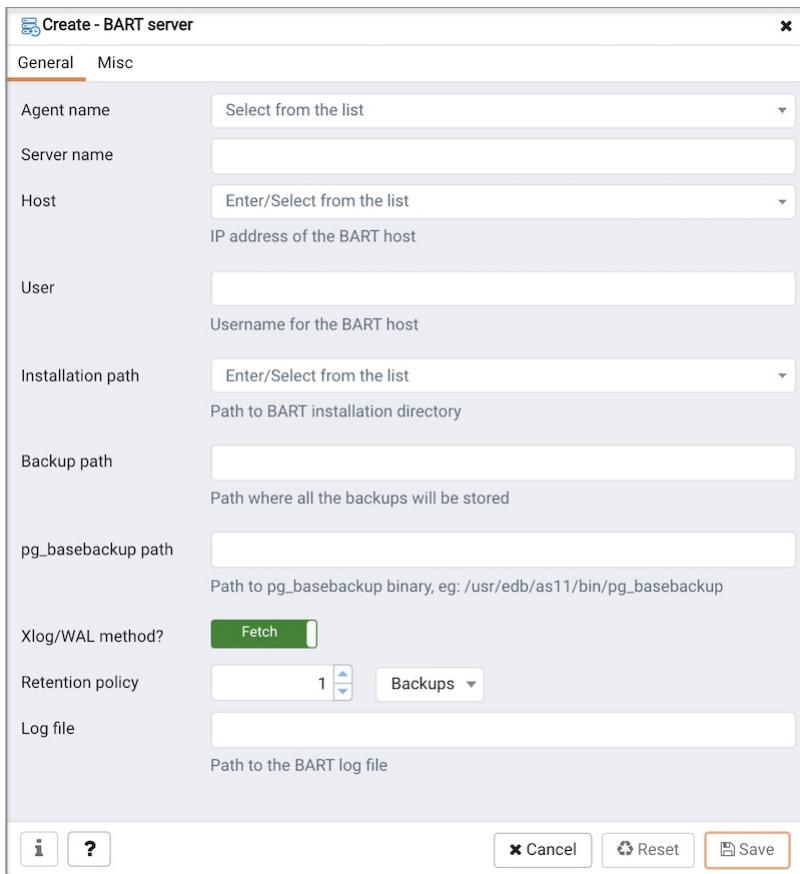
- Before adding a BART server to the PEM console, you must manually install and configure BART on the BART host. For more information about installing and configuring BART, please see the [BART Installation Guide](#).

!!! Note While integrating PEM with BART, the PEM Agent creates the backup directory required for BART, and ensures that the ownership and permissions on the directory are correct.

- Before associating a database server with a BART server, you must install SSH on the database server and the BART server.
- Before restoring a BART backup, you must install BART, a PEM agent, and SSH on the target server. SSH must also be installed on the BART server that you plan to use to restore.
- To take a backup of the replica database servers, you must ensure that the latest *pg_basebackup* utility is installed on the database server that you want to manage through BART.

Configuring a BART Server

You can use the **Create-BART server** dialog to register an existing BART server with the PEM server. To access the dialog, right-click on the **BART Servers** node and select **Create-BART Server**.



Use the fields on the **General** tab to describe the general properties of the BART Server:

- Use the **Agent Name** field to select the agent that you want to use for the BART server. Only those PEM agents that are supported for BART are listed in the drop-down list.
- Use the **Server Name** field to specify a user-friendly name for the server. The name specified will identify the server in the Browser tree.
- Use the **Host** field to specify the IP address of the host or agent where BART is installed.
- Use the **User** field to specify the user name that will be used for performing all the BART operations. You can either use the **enterprisedb** (for Advanced Server) or **postgres** (for PostgreSQL) database user account or you can create a new BART user account. This user must be an operating system user who owns the BART backup catalog directory.
- Use the **Installation path** field to specify the directory path where BART is installed on the host or BART server.
- Use the **Backup path** field to specify the file system parent directory where all BART backups and archived WAL files will be stored.
- Use the **pg_basebackup_path** field to specify the path to the **pg_basebackup** utility.
- Use the **Xlog/WAL method** field to specify how the transaction log should be collected during the execution of **pg_basebackup**. The default option is **fetch**; it specifies that the transaction log files will be collected after the backup has completed. Set the **Xlog/WAL** method to **stream** to stream the transaction log in parallel with the full base backup creation. If streaming is used, the **max_wal_senders** configuration parameter in the **postgresql.conf** file for affected database servers must account for an additional session for the streaming of

the transaction log (the setting must be a minimum of 2).

For more information about Xlog/WAL methods, see [PostgreSQL core documentation](#).

- Use the **Retention policy** field to specify the retention policy for the backup. This determines when an active backup should be marked as obsolete, and hence, be a candidate for deletion. You can specify the retention policy in terms of number of backup or in terms of duration (days, weeks, or months).
- Use the **Log file** field to specify the path to BART log file. This is an optional field.

The screenshot shows the 'Create - BART server' dialog box with the 'Misc' tab selected. The configuration includes:

- Scanner log file:** Path to the Xlog/WAL scanner log file.
- Socket dir path:** Path to the socket directory where all BART sockets will be stored. The default directory is /tmp. This parameter is added from BART version 2.5.2 onwards.
- Socket name:** Using this option overrides the default BART socket name generated using MD5 checksum. This parameter is added from BART version 2.5.6 onwards.
- WAL compression?**: A switch set to **Disabled**. Compress the archived Xlog/WAL files in gzip format (The gzip must be in the BART user account's PATH).
- Copy Xlog/WAL during restore?**: A switch set to **Disabled**. Copy the archived Xlog/WAL files from the BART backup catalog to the restore_path/archived_wals directory prior to the database server archive recovery.
- Thread count:** 1. Number of threads used to copy blocks.
- Batch size:** 49142. Number of blocks of memory used for copying the modified blocks, the default value is 49412.
- Scan interval:** 1. Number of seconds before forcing a scan of the Xlog/WAL files, default value 0 means no brute-force scanning will be started.
- MBM scan timeout:** 20. Number of seconds to wait for MBM file before timing out, applicable only for incremental backup.
- Workers:** 1. Number of parallel worker processes required to stream the modified blocks of an incremental backups to the restore host.

At the bottom are buttons for **i**, **?**, **Cancel**, **Reset**, and **Save**.

Use the fields on the **Misc** tab to describe the backup-related properties of the BART Server:

- Use the **Scanner log file** field to specify the path to the Xlog/WAL scanner log file. This is an optional field; BART does not create a WAL scanner log file if you do not specify the path.
- Use the **Socket dir path** field to specify the path to the socket directory where all BART sockets will be stored. The default directory is `/tmp`. This parameter is added from BART version 2.5.2 onwards.
- Use the **Socket name** field to specify a user-friendly BART socket file name. Using this option overrides the default BART socket name generated using MD5 checksum. This parameter is added from BART version 2.5.6 onwards.
- Use the **WAL compression?** switch to specify if you want to compress the archived Xlog/WAL files in Gzip format. To enable WAL compression, the gzip compression program must be present in the BART user account's PATH. The WAL compression setting must not be enabled for those database servers where you need to take incremental backups.
- Use the **Copy WALs during restore?** field to specify how the archived WAL files are collected when invoking the RESTORE operation. Set to enabled to copy the archived WAL files from the BART backup catalog to the `restore_path/archived_wals` directory prior to the database server archive recovery. Set to **disabled** to

retrieve the archived WAL files directly from the BART backup catalog during the database server archive recovery. Enabling this option helps you save time during the restore operation.

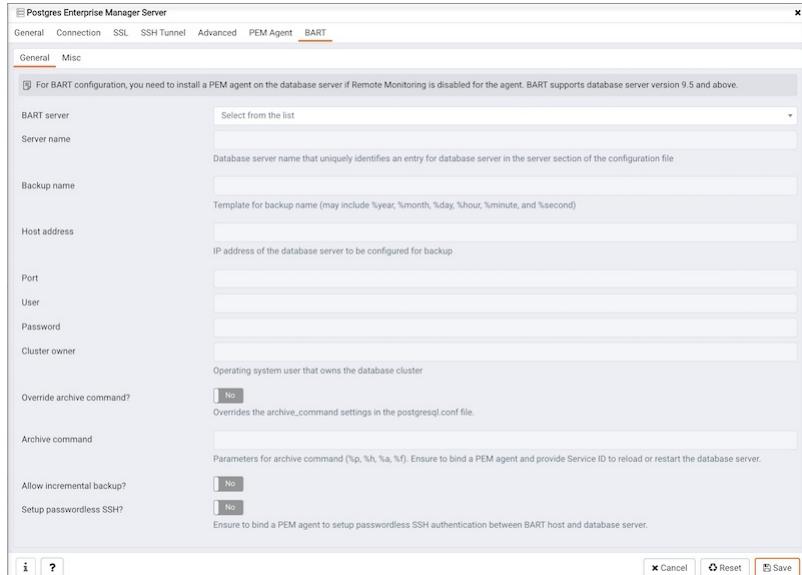
- Use the **Thread count** field to specify the number of worker threads for copying blocks or data files from the database server to the BART backup catalog. Specify a **thread count** of **1** if you want to take the backup using the **pg_basebackup** utility.
- Use the **Batch size** field to specify the number of blocks of memory used for copying modified blocks. This is applicable only for incremental backups.
- Use the **Scan interval** field to specify the number of seconds after which the WAL scanner should scan the new WAL files.
- Use the **MBM scan timeout** field to specify the number of seconds to wait for MBM files before timing out. This is applicable only for incremental backups.
- Use the **Workers** field to specify the number of parallel worker processes required to stream the modified blocks of an incremental backups to the restore host.

Associating the BART Server with a Database Server

After configuring the BART server, you must associate it with the database server whose backup you wish to manage with BART. You can do one of the following:

- Use the PEM console to modify the properties of an existing monitored database server to map it to the newly configured BART server.
- Use the PEM console to create a new monitored database server, and map it to the newly configured BART server.

To map the BART server to a new PEM database server, right-click the **PEM Server Directory** node and select **Create > Server**. Enter the details on all the generic tabs and then enter the BART-specific details on the **BART** tab.



Use the fields on the **General** tab to describe the general properties of the BART Server that will map to the PEM server:

- Use the **BART server** field to select the BART server name. All the BART servers configured in the PEM console will be listed in this drop down list.

- Use the **Server name** field to specify a name for the database server that you want to backup using the BART server. This name gets stored in the BART configuration file.
- Use the **Description** field to specify the description of the database server.
- Use the **Backup name** field to specify a template for user-defined names to be assigned to the backups of the database server. If you do not specify a backup name template, then the backup can only be referenced in BART sub-commands by the BART assigned, integer backup identifier.
- Use the **Host address** field to specify the IP address of the database server that you want to configure for backup.
- Use the **Port** field to specify the port to be used for the database that you want to backup.
- Use the **User** field to specify the user of the database that you want to backup using BART through PEM console. If you want to enable incremental backups for this database server, then the user must be a superuser.
- Use the **Password** field to specify the password for the user of the database that you want to backup.
- Use the **Cluster Owner** field to specify the Linux operating system user account that owns the database cluster. This is typically `enterprisedb` for Advanced Server database clusters installed in the Oracle databases compatible mode, or `postgres` for PostgreSQL database clusters and for Advanced Server database clusters installed in the PostgreSQL databases compatible mode.
- Use the **Archive command** field to specify the desired format of the archive command string to be used in the `bart.cfg` file. Inputs provided for the Archive command will overwrite the database server's `Postgresql.conf` file. Once the server gets added, the database server will be restarted or database configurations will be reloaded.
- Use the **Archive path** field to store the archived WAL files. The default location is the BART backup catalog. This parameter is added from BART version 2.5.2 onwards.
- Use the **Allow incremental backup?** switch to specify if incremental backup should be enabled for this database server.
- Use the **Setup passwordless SSH?** switch to specify if you want to create SSH certificates to allow passwordless logins between the Database Server and the BART server. Ensure to bind a PEM agent before setting up the passwordless SSH authentication. Passwordless SSH will not work for a database server being remotely monitored by a PEM agent.



Use the fields on the **Misc** tab to describe the miscellaneous properties of the BART Server:

- Use the **Override default configuration?** Switch to specify if you want to override the BART server configurations with the specific database server configurations.
- Use the **Xlog** method to specify how the transaction log should be collected during the execution of **pg_basebackup**.
- Use the **Retention policy** field to specify the retention policy for the backup. This determines when an active backup should be marked as obsolete, and hence, be a candidate for deletion. You can specify the retention policy in terms of number of backup or in terms of duration (days, weeks, or months).
- Use the **WAL compression** switch to specify if you want to compress the archived Xlog/WAL files in Gzip format. To enable WAL compression, the gzip compression program must be present in the BART user account's PATH. The wal_compression setting must not be enabled for those database servers where you need to take incremental backups.
- Use the **Copy WALs during restore** field to specify how the archived WAL files are collected when invoking the RESTORE operation. Set to enabled to copy the archived WAL files from the BART backup catalog to the <restore_path>/archived_wals directory prior to the database server archive recovery. Set to disabled to retrieve the archived WAL files directly from the BART backup catalog during the database server archive recovery.
- Use the **Thread count** field to specify the number of threads to copy the blocks. You must set **thread count** to **1** if you want to take a backup with the **pg_basebackup** utility.
- Use the **Batch size** field to specify the number of blocks of memory used for copying modified blocks, applicable only for incremental backups.
- Use the **Scan interval** field to specify the number of seconds after which the WAL scanner should scan the new WAL files.
- Use the **MBM scan timeout** field to specify the number of seconds to wait for MBM files before timing out, applicable only for incremental backups.
- Use the **Workers** field to specify the number of parallel worker processes required to stream the modified blocks of an incremental backups to the restore host.

Viewing the BART Server Details on a PEM Dashboard

After associating the BART server with a database server, you can review the backup and restore details for that server on the PEM Dashboard. You can also perform operations such as restoration or deletion of a backup that is listed on the dashboard.



When you select a monitored BART server, details of all the associated database servers along with their backups and restore activities are displayed as a chart on the Dashboard in the **Backup and Restore Activities** panel. You can filter the list of backups on criteria specified in the filter boxes (database server, activity, or duration).

The **Managed Database servers** panel displays a list of all the database servers managed by that particular BART server along with their high-level details.

The **Initiated Server Backups** panel displays a list of all the backups of the database servers managed by that particular BART server. You can filter the list to display the details of a particular database server. You can also filter the list on any criteria that you specify in the filter box. Typically, this filter works with any kind of string value (excluding date, time, and size) listed under the columns. For example, you can type **tar** to filter the list and display only those backups that are in tar format.

Backup details displayed include the **Backup Name**, **Backup ID**, **Status**, **Server Name**, **Start Time**, **Type**, **Parent ID**, **Format**, **Duration**, and **Size**. The **Status** column shows the status of the backups which can be one of the following: **In Progress**, **Active**, **Keep**, or **Obsolete**.

A backup is marked as **Obsolete** when the backup retention period has passed or if the number of retained backups (specified as the retention policy of the BART server) is met. If you want to make an exception so that a particular backup does not get marked as **Obsolete** even after the expiration of the retention policy, mark that particular backup as **Keep**. Similarly, if you mark a particular backup as **NoKeep**, the backup is re-evaluated to determine if its status should be changed back to **Obsolete** based on the current retention policy.

Please note that if any of the scheduled tasks for backup, restore, validate host, validate server or delete obsolete backup for any of the BART Server gets deleted, it will not display under the **BART Tool Activities** graph of BART Server's dashboard. However, it gets listed under the **Initiated Server Backups** list.

A pin in the first column under **Actions** indicates that a backup can be marked as **Keep** by clicking the pin; while an inverted pin indicates that the backup can be marked as **NoKeep**. The second column under **Actions** displays the **Restore** icon; you can perform the **Restore** operation by clicking on the icon.

You can delete all the **Obsolete** backups by clicking the **Delete Obsolete** button. You can also refresh the list of backups by clicking the **Refresh** button.

Scheduling BART Backups

To create or manage a backup, select **Schedule Backup** from the **Tools** menu. The dialog header displays general

execution information about the backup:

- Logs
- Last result
- Database server
- Last backup name
- Started on
- Type
- Parent
- Format
- Verify checksum?
- Use pg_basebackup?

Click the Add icon (+) to add information about a scheduled backup. Enter the backup details in the schedule definition dialog:

Field	Value
Database server	Select from the list
Backup name	
Type	Full
Parent	
Format	Tar
Gzip compression?	No
Compression level	6
Use pg_basebackup?	No
Thread count	1
Checksum algorithm	NONE
Verify checksum?	No

Use the fields on the **General** tab to describe the general properties of the backup:

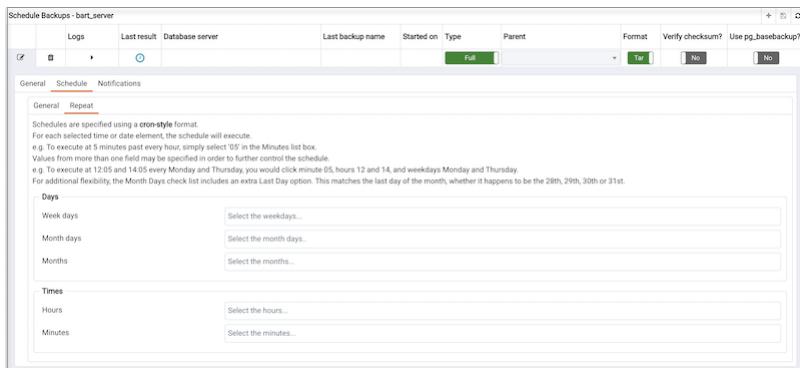
- Use the **Database Server** field to specify the target database server that you want to back up.
- Use the **Backup name** to specify a user-defined name for the backup.
- Use the **Backup type** switch to specify the backup type i.e. full backup or incremental backup.
- Use the **Parent backup** field to select the ID of the parent backup for incremental backup. This parent backup can either be a full or an incremental backup.
- Use the **Format switch** to specify the output format of the backup i.e plain text or tar. For incremental backup, you need to select plain text only.
- Use the **Gzip compression** switch to specify if gzip compression should be enabled for the backup. This option is applicable only for the tar format.
- Use the **Compression level** field to specify the gzip compression level on the tar file output.
- Use the **Thread count** field to specify the number of threads that will copy the blocks.
- Use the **MBM scan timeout** field to specify the number of seconds to wait for required MBM files before timing out.

- Use the **Verify checksum** field to specify if you want the application to verify the checksum of the backup.
- Use the **pg_basebackup** field to specify if the pg_basebackup utility should be used for the backup. Typically, pg_basebackup utility is used only for backing up the replica servers since it cannot be used for incremental backups.



Provide scheduling details for the Backup on the **Schedule** tab:

- Use the **Enabled?** switch to indicate if the schedule should be enabled (**Yes**) or disabled (**No**).
- Use the calendar selector in the **Start** field to specify the starting date and time for the schedule.
- Use the calendar selector in the **End** field to specify the ending date and time for the schedule.



Use the fields on the **Repeat** tab to specify the details about the schedule in a cron-style format. The schedule will execute on each date or time element selected on the **Repeat** tab. Click within a field to open a list of valid values for that field; click on a specific value to add that value to the list of selected values for the field. To clear the values from a field, click the **X** located at the right-side of the field.

Use the fields within the **Days** box to specify the days on which the schedule will execute:

- Use the **Week Days** field to select the days on which the schedule will execute.
- Use the **Month Days** field to select the numeric days on which the schedule will execute. Specify the Last Day to indicate that the schedule should be performed on the last day of the month, regardless of the date.
- Use the **Months** field to select the months in which the schedule will execute.

Use the fields within the **Times** box to specify the times at which the schedule will execute:

- Use the **Hours** field to select the hour at which the schedule will execute.
- Use the **Minutes** field to select the minute at which the schedule will execute.



Use the fields on the **Notifications** tab to specify the email notification settings for a scheduled backup:

- Use the **Send the notifications** field to specify when you want the email notifications to be sent.
- Use the **Email group** field to specify the email group that should receive the email notification.

Restoring BART Backups

You can restore the backups that you have earlier created using BART server on a target remote host. When you select a particular BART server, all the associated backups are listed in the Dashboard under **Initiated Server Backups**.

To restore a backup, click the **Restore** icon next to the backup that you want to restore.



In the **Restore Backup** dialog, provide information in the fields on the **General** tab:

- Use the **Target agent** field name to specify the name of the agent where you want to restore the backup.
- Use the **Remote user** field to specify the use account on the remote database server host where you want to restore the backup.
- Use the **Remote host address** field to specify the IP address of the remote host where you want to restore the backup.
- Use the **SSH port** field to specify the SSH port to be used for restoring the backup.
- Use the **Restore path** field to specify the path where you want to restore the backup.
- Use the **Number of workers** field to specify processes to run in parallel to stream the modified blocks of an

incremental backup to the restore location.

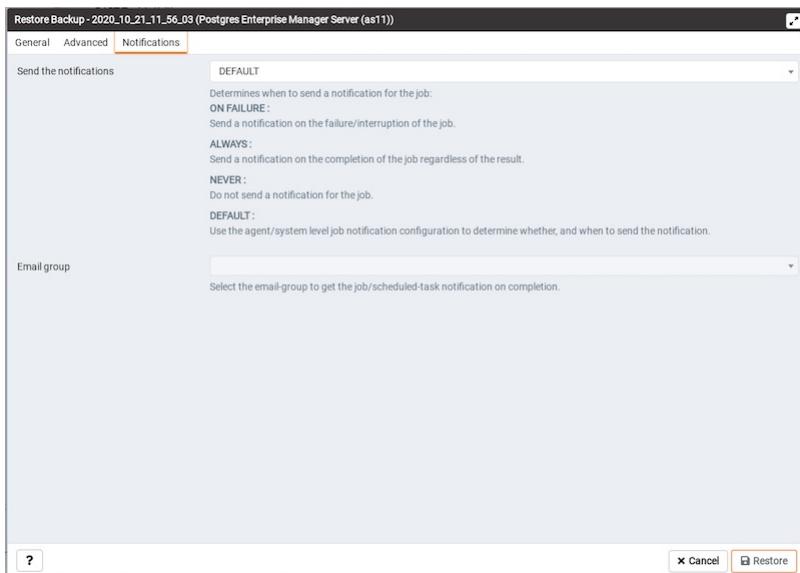
- Use the **Setup passwordless SSH?** switch to specify if you want to create SSH certificates to allow passwordless logins between the BART server and the target host for restore.



On the **Advanced** tab, specify your preferences for advanced options for restoring the backup:

- Use the **Copy WALs to restore path?** switch to specify if you want to copy WAL files to the restore path.
- Use the **Point in time recovery** switch to specify if you want point in time recovery.
- Use the **Timeline ID** field to specify the timeline ID to be used for replaying the archived WAL files for point-in-time recovery.
- Use the **Transaction ID (XID)** field to specify the transaction ID for point-in-time recovery.
- Use the **Timestamp** field to the timestamp to be used for restore.

!!! Note You can specify either **Transaction ID** or **Timestamp** for the point-in-time recovery.



Use the fields on the **Notifications** tab to specify the email notification settings for restoring the backup.

- Use the `Send the notifications` field to specify when you want the email notifications to be sent.
 - Use the `Email group` field to specify the email group that should receive the email notification.
-

4 PEM Enterprise Features Guide

This guide will acquaint you with the tools and wizards that are built into the Postgres Enterprise Manager (PEM) web interface that make it easier for you to monitor and manage your system.

This guide is not a comprehensive resource; rather, it is meant to serve as an aid to help you evaluate the tool and bring you up to speed with the basics of how to use the product. For more detailed information about using PEM's functionality, please see the online help made available by the PEM client.

This document uses `Postgres` to mean either the PostgreSQL or EDB Postgres Advanced Server database.

4.1 What's New

The following changes have been made to Postgres Enterprise Manager to create version 8.0.1:

Improved the overall performance, usability, and stability of the product by:

- Allowing the dash (-) character in the superuser name that is provided during configuration.
- Gracefully closing the operating system resources during batch probe and command execution to avoid errors.
- Improved support for unicode string handling in pemAgent.
- Fixed a regression restoring BART backup when agent is not bound with the BART server.
- In addition to the above-listed items, a few other minor bugs have also been fixed.

PEM 8.0.1 provides an option to hide or unhide the Dashboard, Statistics, Dependents, and Dependencies tabs.

Features and Functionalities removed in recent versions of PEM:

- PEM 8.0 and later is not supported on CentOS/RHEL 6.x.
 - EDB ARK UI Management is not supported by PEM 7.16 and later.
 - PEM 7.16 and later is not supported on Internet explorer version 11 and lesser.
-

4.2 The PEM Query Tool

PEM contains a feature-rich Interactive Development Environment (IDE) that allows you to issue ad-hoc SQL queries against Postgres servers.

You can access the Query Tool via the `Query Tool` menu option on the `Tools` menu, or through the context menu of

select nodes of the Browser tree control. The Query Tool allows you to:

- Issue ad-hoc SQL queries.
- Execute arbitrary SQL commands.
- Edit the result set of a SELECT query if it is [updatable](#).
- Displays current connection and transaction status as configured by the user.
- Save the data displayed in the output panel to a CSV file.
- Review the execution plan of a SQL statement in either a text, a graphical format or a table format (similar to <https://explain.depesz.com>).
- View analytical information about a SQL statement.

attnum	attname	atttypid	atttypmod	attlen	attstorage	attalign	attcollation	attinhadae
1	oid	26	-1	4	p	i	'char'(1)	true
2	proname	19	-1	64	p	c	'char'(1)	false
3	pronamespa...	26	-1	4	p	i	'char'(1)	false
4	pronower	26	-1	4	p	i	'char'(1)	false
5	prolang	26	-1	4	p	i	'char'(1)	false
6	procost	700	-1	4	p	i	'char'(1)	false
7	pronotnull	700	-1	4	p	i	'char'(1)	false
8	pronadic	26	-1	4	p	i	'char'(1)	false
9	prsupport	24	-1	4	p	i	'char'(1)	false
10	prokind	18	-1	1	p	c	'char'(1)	false
11	prosedef	16	-1	1	p	c	'char'(1)	false
12	proleakproof	16	-1	1	p	c	'char'(1)	false
13	proresolute	16	-1	1	n	n	'char'(1)	false

You can open multiple copies of the Query tool in individual tabs simultaneously. To close a copy of the Query tool, click the *X* in the upper-right hand corner of the tab bar.

The Query Tool features two panels:

- The upper panel displays the [SQL Editor](#). You can use the panel to enter, edit, or execute a query. It also shows the [History](#) tab which can be used to view the queries that have been executed in the session, and a [Scratch Pad](#) which can be used to hold text snippets during editing. If the Scratch Pad is closed, it can be re-opened (or additional ones opened) by right-clicking in the SQL Editor and other panels and adding a new panel.
- The lower panel displays the [Data Output](#) panel. The tabbed panel displays the result set returned by a query, information about a query's execution plan, server messages related to the query's execution and any asynchronous notifications received from the server.

The Query Tool Toolbar

The [Query Tool](#) toolbar uses context-sensitive icons that provide shortcuts to frequently performed tasks. If an icon is highlighted, the option is enabled; if the icon is grayed-out, the task is disabled.



Hover over an icon to display a tool-tip that describes the icon's functionality:

Icon	Behavior	Shortcut
Open File	Click the Open File icon to display a previously saved query in the SQL Editor.	Accesskey + O

Icon	Behavior	Shortcut
Save	Click the Save icon to perform a quick-save of a previously saved query, or to access the Save menu: - Select Save to save the selected content of the SQL Editor panel in a file. - Select Save As to open a new browser dialog and specify a new location to which to save the selected content of the SQL Editor panel.	Accesskey + S
Save Data Changes	Click the Save Data Changes icon to save the data changes (insert, update, or delete) in the Data Output Panel to the server.	F6
Find	Use the Find menu to search, replace, or navigate the code displayed in the SQL Editor: - Select Find to provide a search target, and search the SQL Editor contents. - Select Find next to locate the next occurrence of the search target. - Select Find previous to move to the last occurrence of the search target. - Select Persistent find to identify all occurrences of the search target within the editor. - Select Replace to locate and replace (with prompting) individual occurrences of the target. - Select Replace all to locate and replace all occurrences of the target within the editor. - Select Jump to navigate to the next occurrence of the search target.	Cmd+F Cmd+G Cmd+Shift+G Cmd+Shift+F Alt+G
Copy	Click the Copy icon to copy the content that is currently highlighted in the Data Output panel. when in View/Edit data mode.	Accesskey + C
Paste	Click the Paste icon to paste a previously row into a new row when in View/Edit data mode.	Accesskey + P
Delete	Click the Delete icon to mark the selected rows for deletion. These marked rows get deleted when you click the Save Data Changes icon.	Accesskey + D
Edit	Use options on the Edit menu to access text editing tools; the options operate on the text displayed in the SQL Editor panel when in Query Tool mode: - Select Indent Selection to indent the currently selected text. - Select Unindent Selection to remove indentation from the currently selected text. - Select Inline Comment Selection to enclose any lines that contain the selection in SQL style comment notation. - Select Inline Uncomment Selection to remove SQL style comment notation from the selected line. - Select Block Comment to enclose all lines that contain the selection in C style comment notation. This option acts as a toggle.	Tab Shift+Tab Cmd+/ Cmd+. Shift+Cmd+/ Shift+Cmd+/
Filter	Click the Filter icon to set filtering and sorting criteria for the data when in View/Edit data mode. Click the down arrow to access other filtering and sorting options: - Click Sort/Filter to open the sorting and filtering dialogue. - Click Filter by Selection to show only the rows containing the values in the selected cells. - Click Exclude by Selection to show only the rows that do not contain the values in the selected cells. - Click Remove Sort/Filter to remove any previously selected sort or filtering options.	Accesskey + F
Limit Selector	Select a value in the Limit Selector to limit the size of the dataset to a number of rows.	Accesskey + R

Icon	Behavior	Shortcut
Stop	Click the Stop icon to cancel the execution of the currently running query.	Accesskey + Q
Execute/Refresh	Click the Execute/Refresh icon to either execute or refresh the query highlighted in the SQL editor panel. Click the down arrow to access other execution options: - Add a check next to Auto-Rollback to instruct the server to automatically roll back a transaction if an error occurs during the transaction. - Add a check next to Auto-Commit to instruct the server to automatically commit each transaction. Any changes made by the transaction will be visible to others, and durable in the event of a crash.	F5
Explain	Click the Explain icon to view an explanation plan for the current query. The result of EXPLAIN is displayed graphically on the Explain tab of the output panel, and in text form on the Data Output tab.	F7
Explain analyze	Click the Explain analyze icon to invoke an EXPLAIN ANALYZE command on the current query. Navigate through the Explain Options menu to select options for the EXPLAIN command: - Select Verbose to display additional information regarding the query plan. - Select Costs to include information on the estimated startup and total cost of each plan node, as well as the estimated number of rows and the estimated width of each row. - Select Buffers to include information on buffer usage. - Select Timing to include information about the startup time and the amount of time spent in each node of the query. - Select Summary to include the summary information about the query plan.	
Commit	Click the Commit icon to commit the transaction.	Shift+CTRL+M
Rollback	Click the Rollback icon to rollback the transaction.	Shift+CTRL+R
Clear	Use options on the Clear drop-down menu to erase display contents: - Select Clear Query Window to erase the content of the SQL Editor panel. - Select Clear History to erase the content of the History tab.	Accesskey + L
Download as CSV	Click the Download as CSV icon to download the result set of the current query to a comma-separated list. You can specify the CSV settings through Preferences -> SQL Editor -> CSV output dialogue.	F8
Macros	Click the Macros icon to manage the macros. You can create, edit or clear the macros through Manage Macros option.	

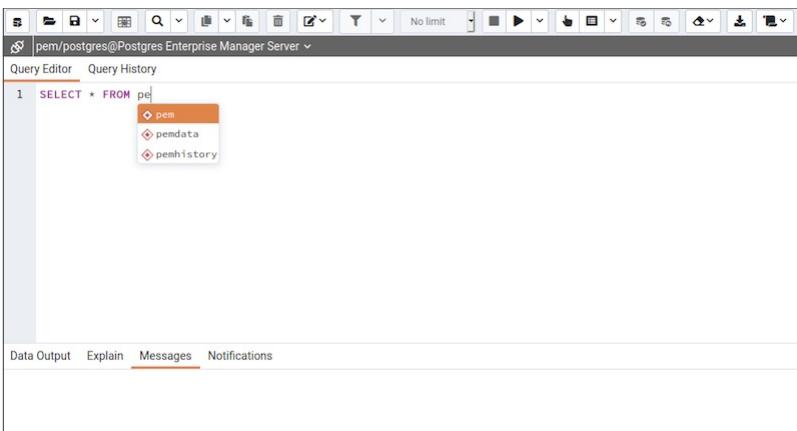
The SQL Editor Panel

The **SQL editor** panel is a workspace where you can manually provide a query, copy a query from another source, or read a query from a file. The SQL editor features syntax coloring and auto-completion.

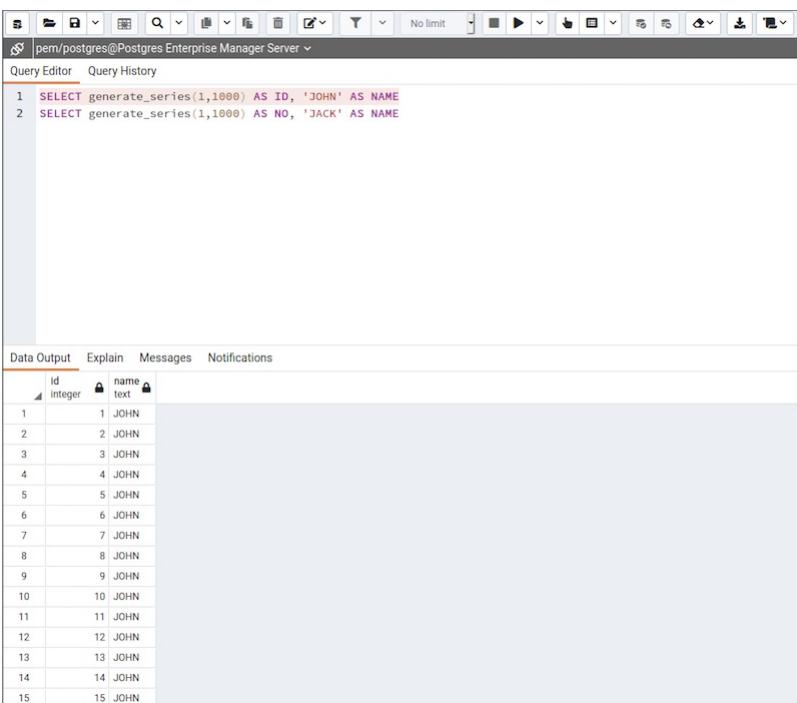


To use auto-complete, begin typing your query; when you would like the Query editor to suggest object names or commands that might be next in your query, press the Control+Space key combination. For example, type `*SELECT`

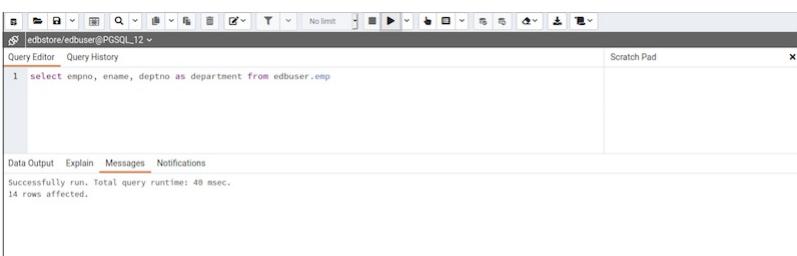
* FROM* (with a trailing space), and then press the **Control+Space** key combination to select from a popup menu of auto-complete options.



After entering a query, select the **Execute/Refresh** icon from the toolbar. The complete contents of the SQL editor panel will be sent to the database server for execution. To execute only a section of the code that is displayed in the SQL editor, highlight the text that you want the server to execute, and click the **Execute/Refresh** icon.



The message returned by the server when a command executes is displayed on the **Messages** tab. If the command is successful, the **Messages** tab displays execution details.



Options on the **Edit** menu offer functionality that helps with code formatting and commenting:

- The auto-indent feature will automatically indent text to the same depth as the previous line when you press the Return key.

- Block indent text by selecting two or more lines and pressing the Tab key.
- Implement or remove SQL style or toggle C style comment notation within your code.

You can also **drag and drop** certain objects from the tree-view to save time spent typing long object names. Text containing the object name will be fully qualified with the schema name. Double quotes will be added if required. For functions and procedures, the function name along with parameter names will be pasted in the Query Tool.

The Data Output Panel

The **Data Output** panel displays data and statistics generated by the most recently executed query.

	oid	datname	datdba	encoding	datcollate	datatype	datistemplate	datallowconn	datconnlimit	datlastsysoid
	oid	name	old	integer	name	name	boolean	boolean	integer	oid
1	13432	postgres	10	6	en_IN	en_IN	false	true	-1	13431
2	16389	edbstore	16388	6	en_IN	en_IN	false	true	-1	13431
3	1	template1	10	6	en_IN	en_IN	true	true	-1	13431
4	13431	template0	10	6	en_IN	en_IN	true	false	-1	13431

Data Output Tab

The **Data Output** tab displays the result set of the query in a table format. You can:

- Select and copy from the displayed result set.
- Use the **Execute/Refresh** options to retrieve query execution information and set query execution options.
- Use the **Download as CSV** icon to download the content of the **Data Output** tab as a comma-delimited file.
- Edit the data in the result set of a **SELECT** query if it is updatable.

A result set is updatable if:

- All columns are either selected directly from a single table, or they are not actually a table column (for example, the concatenation of two columns). Only columns that are selected directly from the table are editable, other columns are read-only.
- All the primary key columns or OIDs of the table are selected in the result set.

Any columns that are renamed or selected more than once are also read-only.

!!! Note To work with an updatable query result set, you must have **psycopg2** driver version 2.8 or above installed.

Editable and read-only columns are identified using pencil and lock icons (respectively) in the column headers.

The screenshot shows the Query Editor window with the following content:

```
edbstore/edbuser@PGSQL_12 ~
Query Editor Query History
Scratch Pad
1 select empno, ename, deptno as department from edbuser.emp
```

Below the query results, there is a table titled "Data Output" with the following data:

	empno	ename	department
	[PK] numeric (4)	character varying (10)	numeric (2)
1	7369	SMITH	20
2	7499	ALLEN	30
3	7521	WARD	30
4	7566	JONES	20
5	7654	MARTIN	30
6	7689	BLAKE	30
7	7782	CLARK	10
8	7788	SCOTT	20
9	7839	KING	10
10	7844	TURNER	30
11	7876	ADAMS	20
12	7900	JAMES	30
13	7902	FORD	20
14	7934	MILLER	10

An updatable result set is similar to the **Data Grid** in **View/Edit Data** mode, and can be modified in the same way.

If Auto-commit is **off**, data changes are made as part of the ongoing transaction; if no transaction is ongoing a new one is initiated. The data changes are not committed to the database unless the transaction is committed.

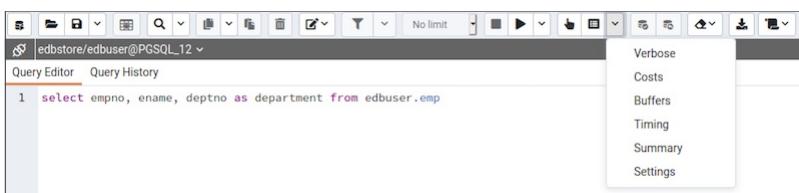
If any errors occur during saving (for example, trying to save a NULL into a column with a NOT NULL constraint) the data changes are rolled back to an automatically created SAVEPOINT to ensure any previously executed queries in the ongoing transaction are not rolled back.

All rowsets from previous queries or commands that are displayed in the **Data Output** panel will be discarded when you invoke another query; open another query tool browser tab to keep your previous results available.

Explain Tab

To generate the **Explain** or **Explain Analyze** plan of a query, click on **Explain** or **Explain Analyze** button in the toolbar.

More options related to **Explain** and **Explain Analyze** can be selected from the drop down on the right side of **Explain Analyze** button in the toolbar.



Please note that PEM generates the **Explain [Analyze]** plan in JSON format.

On successful generation of **Explain** plan, it will create three tabs/panels under the **Explain** panel.

Graphical Tab

Hover over an icon on the **Graphical** tab to review information about that item; a popup window will display information about the selected object. For information on JIT statistics, triggers and a summary, hover over the icon on top-right corner; a similar popup window will be displayed when appropriate.

Please note that **EXPLAIN VERBOSE** cannot be displayed graphically.

Use the download button on top left corner of the **Explain** canvas to download the plan as an SVG file. Please note

that [Download as SVG](#) feature is not supported on Internet Explorer.



The query plan that accompanies the [Explain analyze](#) is available on the [Data Output](#) tab.

Analysis Tab

The [Analysis](#) tab shows the plan details in table format, it generates a format similar to the format available at explain.depesz.com. Each row of the table represents the data for a [Explain Plan Node](#). The output may contain the node information, exclusive timing, inclusive timing, actual vs. planned rows, actual rows, planned rows, or loops. When you select a row, the child rows of that selected row are marked with an orange dot.

If the percentage of the exclusive/inclusive timings of the total query time is:

Greater than 90 --> Red

Greater than 50 --> Orange (between red and yellow)

Greater than 10 --> Yellow

If the planner has misestimated the number of rows (actual vs planned) by:

10 times --> Yellow color

100 times --> Orange (between Red and Yellow) color

1000 times --> Red color

The screenshot shows the 'Query Editor' tab with a complex SQL query in the 'Query History' pane. Below it, the 'Analysis' tab is selected, displaying a detailed execution plan in a table format. The table has three main sections: 'Timings', 'Rows', and 'Loops'. The 'Timings' section shows execution times for each node, with 'Exclusive' and 'Inclusive' times. The 'Rows' section shows the number of rows processed by each node. The 'Loops' section indicates the number of loops for each node. The execution plan includes various PostgreSQL operators like Unique, Sort, Nested Loop Left Join, Hash Inner Join, Hash Left Join, Hash Left Outer Join, and Merge Left Join, along with specific conditions and statistics for each node.

Statistics Tab

The **Statistics** tab displays information in two tables:

- **Statistics per Node Type** tells you how many times each node type was referenced.
- **Statistics per Table** tells you how many times each table was referenced by the query.

The screenshot shows the 'Query Editor' tab with the same query as before. The 'Statistics' tab is selected, displaying two tables. The first table, 'Statistics per Node Type', lists various node types (Hash, Hash Inner Join, Hash Left Join, etc.) with their counts and total time spent. The second table, 'Statistics per Table', lists tables (pg_attrdef, pg_attribute, pg_class, pg_constraint, pg_depend) with their scan counts and total time spent. This provides a high-level overview of the query's performance and resource usage.

Messages Tab

Use the **Messages** tab to view information about the most recently executed query:

The screenshot shows the 'Query Editor' tab with the same query as before. The 'Messages' tab is selected, displaying an error message: 'ERROR: relation "pg.roles" does not exist'. It also shows the SQL state (42P01), character set (UTF-8), and the line number (1) where the error occurred. This tab is useful for quickly identifying and addressing errors in the most recent query.

If the server returns an error, the error message will be displayed on the **Messages** tab, and the syntax that caused the error will be underlined in the SQL editor. If a query succeeds, the **Messages** tab displays how long the query took to complete and how many rows were retrieved:

The screenshot shows the Query Editor interface with the following details:

- Session:** edbstore/postgres@PGSQL_12
- Tab:** Query History
- Query:**

```

1 SELECT DISTINCT dep.deptype, dep.refclassid, cl.relkind, ad.adbin,
2 CASE WHEN cl.relkind IS NOT NULL THEN cl.relkind || COALESCE(dep.refobjsubid::character varying, '')
3 WHEN tg.oid IS NOT NULL THEN 'T'::text
4 WHEN ty.oid IS NOT NULL AND ty.typbasetype = 0 THEN 'y'::text
5 WHEN ty.oid IS NOT NULL AND ty.typbasetype != 0 THEN 'd'::text
6 WHEN ns.oid IS NOT NULL THEN 'n'::text
7 WHEN pr.oid IS NOT NULL AND prtyp.typname = 'trigger' THEN 't'::text
8 WHEN pr.oid IS NOT NULL THEN 'P'::text
9 WHEN la.oid IS NOT NULL THEN 'l'::text
10 WHEN rw.oid IS NOT NULL THEN 'R'::text
11 WHEN co.oid IS NOT NULL THEN 'C'::text || contype
12 WHEN ad.oid IS NOT NULL THEN 'A'::text
13 WHEN fs.oid IS NOT NULL THEN 'F'::text
14 WHEN fdw.oid IS NOT NULL THEN 'f'::text
15 ELSE ''

```

- Data Output:** Successfully run. Total query runtime: 61 msec.
- Notifications:** 1 rows affected.

Notifications Tab

Use the **Notifications** tab to view details of the asynchronous notifications that a client process may have sent:

The screenshot shows the Notifications tab with the following details:

- Session:** pern/postgres@Postgres Enterprise Manager Server
- Event:** listen "table_update"
- Recorded time:**

Recorded time	Event	Process ID	Payload
2019-11-07 12:53:53.645987	table_update	1668	Updated rows in table_1
2019-11-07 12:53:53.645987	table_update	5376	Updated rows in table_2

You can see details such as recorded time of the asynchronous notification event, name of the event or channel, process ID of the client process that has sent the notification, and the payload string that might have been sent along with the notification.

Query History Panel

Use the **Query History** tab to review activity for the current session:

The screenshot shows the Query History tab with the following details:

- Show queries generated internally by Postgres Enterprise Manager?** Yes
- Date:** Today - 11/17/2020
- Recent Commands:**
 - 15:42:01 SELECT DISTINCT dep.deptype, dep.refclassid, cl.relkind, ad.adbin
 - 15:42:01 CASE WHEN cl.relkind IS NOT NULL THEN cl.relkind || COALESCE(dep.refobjsubid::character varying, '')
 - 15:42:01 WHEN tg.oid IS NOT NULL THEN 'T'::text
 - 15:42:01 WHEN ty.oid IS NOT NULL AND ty.typbasetype = 0 THEN 'y'::text
 - 15:42:01 WHEN ty.oid IS NOT NULL AND ty.typbasetype != 0 THEN 'd'::text
 - 15:42:01 WHEN ns.oid IS NOT NULL THEN 'n'::text
 - 15:42:01 WHEN pr.oid IS NOT NULL AND prtyp.typname = 'trigger' THEN 't'::text
 - 15:42:01 WHEN pr.oid IS NOT NULL THEN 'P'::text
 - 15:42:01 WHEN la.oid IS NOT NULL THEN 'l'::text
 - 15:42:01 WHEN rw.oid IS NOT NULL THEN 'R'::text
 - 15:42:01 WHEN co.oid IS NOT NULL THEN 'C'::text || contype
 - 15:42:01 WHEN ad.oid IS NOT NULL THEN 'A'::text
 - 15:42:01 WHEN fs.oid IS NOT NULL THEN 'F'::text
 - 15:42:01 WHEN fdw.oid IS NOT NULL THEN 'f'::text
 - 15:42:01 ELSE ''
- Messages:** Successfully run. Total query runtime: 61 msec.
- Rows Affected:** 1 rows affected.

The **Query History** tab displays information about recent commands:

- The date and time that a query was invoked.

- The text of the query.
- The number of rows returned by the query.
- The amount of time it took the server to process the query and return a result set.
- Messages returned by the server (not noted on the **Messages** tab).
- The source of the query (indicated by icons corresponding to the toolbar).

You can show or hide the queries generated internally by pgAdmin (during **View/Edit Data** or **Save Data** operations).

To erase the content of the **Query History** tab, select **Clear history** from the **Clear** drop-down menu.

Query history is maintained across sessions for each database on a per-user basis when running in **Query Tool** mode. In **View/Edit Data** mode, history is not retained. By default, the last 20 queries are stored for each database. This can be adjusted in `config_local.py` by overriding the `MAX_QUERY_HIST_STORED` value.

Connection Status

Use the **Connection status** feature to view the current connection and transaction status by clicking on the status icon in query tool:



Macros

Query Tool Macros enable you to execute pre-defined SQL queries with a single key press. Pre-defined queries can contain the placeholder `$SELECTION$`. Upon macro execution, the placeholder will be replaced with the currently selected text in the Query Editor pane of the Query Tool.



To create a macro, select the *Manage Macros* option from the *Macros* menu on the *Query Tool*. Select the key you wish to use, enter the name of the macro, and the query, optionally including the selection placeholder, and then click the *Save* button to store the macro.

Key	Name	SQL
Ctrl + 1	View data of dept table	SELECT * FROM edbuser.dept;
Ctrl + 2	View data of cust_hist table	SELECT * FROM edbuser.cust_hist WHERE customerid=7888;
Ctrl + 3	View data of SELECTED table	SELECT * FROM \$SELECTION\$;
Ctrl + 4		
Ctrl + 5		
Ctrl + 6		
Ctrl + 7		
Ctrl + 8		
Ctrl + 9		
Ctrl + 0		

To clear a macro, select the macro on the *Manage Macros* dialogue, and then click the *Clear* button.

Key	Name	SQL
Ctrl + 1	View data of dept table	SELECT * FROM edbuser.dept;
Ctrl + 2	View data of cust_hist table	SELECT * FROM edbuser.cust_hist WHERE customerid=7888;
Ctrl + 3	View data of SELECTED table	SELECT * FROM \$SELECTION\$;
Ctrl + 4		
Ctrl + 5		
Ctrl + 6		
Ctrl + 7		
Ctrl + 8		
Ctrl + 9		
Ctrl + 0		

The server will prompt you for confirmation to clear the macro.

Clear Row		
Key	Name	SQL
Ctrl + 1	View data of dept table	SELECT * FROM edbuser.dept;
Ctrl + 2	View data of cust_hist table	SELECT * FROM edbuser.cust_hist WHERE customerid=7888;
Ctrl + 3	View data of SELECTED table	SELECT * FROM \$SELECTION\$;
Ctrl + 4		
Ctrl + 5		
Ctrl + 6		
Ctrl + 7		
Ctrl + 8		
Ctrl + 9		
Ctrl + 0		

To clear all macros, click on the *Clear* button on left side of the key. The server will prompt you for confirmation to clear all the rows.



To execute a macro, simply select the appropriate shortcut keys, or select it from the *Macros* menu.

deptno	dname	loc
1	ACCOUNTING	NEW YORK
2	RESEARCH	DALLAS
3	SALES	CHICAGO
4	OPERATIONS	BOSTON

4.3 The PEM Schema Diff Tool

Schema Diff is a feature that allows you to compare schema objects between two database schemas. Use the **Tools** menu to access Schema Diff.

The Schema Diff feature allows you to:

- Compare and synchronize the database schemas (from source to target).
- Visualize the differences between database schemas.
- List the differences in SQL statement for target schema objects.
- Generate synchronization scripts.

!!! Note The source and target databases must be of the same major version.

Click on **Schema Diff** under the **Tools** menu to open a selection panel. Choose the source and target servers, databases, and schemas that will be compared. After selecting the objects, click on the **Compare** button.

You can open multiple copies of **Schema Diff** in individual tabs simultaneously. To close a tab, click the **X** in the

upper-right hand corner of the tab bar.

The screenshot shows the Schema Diff (Beta) panel. At the top, it displays 'Select Source' and 'Select Target' both set to 'PostgreSQL 11' and database 'postgres'. Below this, the 'Schema Objects' section lists objects like 'abstract', 'asset', and various 'colN' entries. A summary at the top says 'Collations - Identical: 0 Different: 1000 Source Only: 1 Target Only: 1'. The 'Comparison Result' table shows rows for each object, with 'col1' highlighted in grey. The 'DDL Comparison' section shows the SQL scripts for creating and altering collations on both the source and target sides, with a 'Difference' column indicating the changes.

Use the **Preferences** dialog to specify if **Schema Diff** should open in a new browser tab. Set **Open in new browser tab** option to **true**.

The **Schema Diff** panel is divided into two panels; an **Object Comparison panel** and a **DDL Comparison panel**.

The Schema Diff Object Comparison Panel

In the object comparison panel, you can select the source and target servers of the same major version, databases, and schemas to be compared. You can select any server listed under the browser tree whether it is connected or disconnected. If you select a server that is not connected then it will prompt you for the password before using the server.

Next, select the databases that will be compared. The databases can be the same or different (and within the same server or from different servers).

Lastly, select the source and target schemas which will be compared.

The screenshot shows the Schema Diff (Beta) panel with the 'Compare' button highlighted. A tooltip below the button says: 'Select the server, database and schema for the source and target and click Compare to compare them.'

After you select servers, databases, and schemas, click on the **Compare** button to obtain the **Comparison Result**.

The screenshot shows the Schema Diff (Beta) interface. At the top, there are dropdowns for 'Select Source' (PostgreSQL 11, postgres, source_sc) and 'Select Target' (PostgreSQL 11, postgres, target_sc). Below these are buttons for 'Compare' and 'Generate Script'. The main area is titled 'Comparison Result' and contains a table with columns for schema objects and their status. The table shows:

Object Type	Status
Collations	Identical
Domains	Identical
Foreign Tables	Identical
FTS Configurations	Identical
FTS Dictionaries	Identical
fts1	Identical
fts10	Identical
fts100	Identical
fts1000	Identical
fts101	Identical
fts102	Identical
fts103	Identical

Use the drop-down lists of Functions, Materialized Views, Tables, Trigger Functions, Procedures, and Views to view the DDL statements of all the schema objects.

In the upper-right hand corner of the object comparison panel is a **Filter** option that you can use to filter the schema objects based on the following comparison criteria:

- **Identical** - If the object is found in both schemas with the same SQL statement, then the comparison result is identical.
- **Different** - If the object is found in both schemas but have different SQL statements, then the comparison result is different.
- **Source Only** - If the object is found in source schema only and not in target schema, then the comparison result is source only.
- **Target Only** - If the object is found in target schema only and not in source schema, then the comparison result is target only.

The screenshot shows the Schema Diff (Beta) interface with the 'Filter' dropdown open. The dropdown menu includes four options: 'Identical', 'Different', 'Source Only', and 'Target Only'. The 'Source Only' option is currently selected. The main table below shows a list of schema objects and their status, with the 'Source Only' row highlighted in yellow.

Object	Status
abstract	Source Only
asset	Target Only
col1	Different
col10	Different
col100	Different
col1000	Different
col101	Different
col102	Different
col103	Different
col104	Different
col105	Different

Click on any of the schema objects in the object comparison panel to display the DDL statements for that object in the **DDL Comparison** panel.

Schema Diff DDL Comparison Panel

The **DDL Comparison** panel displays three columns:

- The first column displays the DDL statement of the object from the source schema.
- The second column displays the DDL statement of the object from the target schema.
- The third column displays the difference in the SQL statement of the target schema object.

The screenshot shows the Schema Diff (Beta) tool interface. At the top, it displays 'Select Source' (PostgreSQL 11) and 'Select Target' (PostgreSQL 11). Below this is a 'Schema Objects' tree view with nodes for abstract, asset, col1, col10, col100, col101, col102, col103, col104, col105, and col106. The 'Comparison Result' panel indicates 1000 differences, with col1 being 'Different'. The 'DDL Comparison' panel shows the SQL code for creating and altering collations in both the source and target databases.

```

Source
1 -- Collation: coll;
2
3 -- DROP COLLATION source_sc.coll;
4
5 CREATE COLLATION source_sc.coll
6   (LC_COLLATE = 'POSIX', LC_CTYPE = 'POSIX');
7
8 ALTER COLLATION source_sc.coll
9   OWNER TO postgres;

Target
1 -- Collation: coll;
2
3 -- DROP COLLATION target_sc.coli;
4
5 CREATE COLLATION target_sc.coll
6   (LC_COLLATE = 'C', LC_CTYPE = 'C');
7
8 ALTER COLLATION target_sc.coll
9   OWNER TO postgres;

```

You can review the DDL statements of all the schema objects to check for the differences in the SQL statements.

You can also use the Schema Diff tool to generate a SQL script of the differences found in the target schema object based on the SQL statement of the source schema object. To generate the script, select the checkboxes of the schema objects in the object comparison panel and then click on the **Generate Script** button in the upper-right hand corner of the object comparison panel.

The screenshot shows the Schema Diff (Beta) tool interface. At the top, it displays 'Select Source' (PostgreSQL 11) and 'Select Target' (PostgreSQL 11). Below this is a 'Schema Objects' tree view with nodes for abstract, asset, col1, col10, col100, col101, col102, col103, col104, col105, and col106. The 'Comparison Result' panel indicates 1000 differences, with col1 being 'Different'. The 'DDL Comparison' panel shows the SQL code for creating and altering collations in both the source and target databases. A 'Copy' button is visible next to the difference text.

```

Source
1 -- Collation: coll;
2
3 -- DROP COLLATION source_sc.coll;
4
5 CREATE COLLATION source_sc.coll
6   (LC_COLLATE = 'POSIX', LC_CTYPE = 'POSIX');
7
8 ALTER COLLATION source_sc.coll
9   OWNER TO postgres;

Target
1 -- Collation: coll;
2
3 -- DROP COLLATION target_sc.coli;
4
5 CREATE COLLATION target_sc.coll
6   (LC_COLLATE = 'C', LC_CTYPE = 'C');
7
8 ALTER COLLATION target_sc.coll
9   OWNER TO postgres;

```

Select the schema objects and click on the **Generate Script** button to open the **Query Tool** in a new tab, with the difference in the SQL statement displayed in the **Query Editor**.

If you have clicked on the schema object to check the difference generated in the **DDL Comparison** Panel, and you have not selected the checkbox of the schema object, PEM will open the **Query Tool** in a new tab, with the differences in the SQL statements displayed in the **Query Editor**.

```

1 -- This script was generated by a beta version of the Schema Diff utility in Postgres Enterprise Manager.
2 -- This version does not include dependency resolution, and may require manual changes
3 -- to the script to ensure changes are applied in the correct order.
4 -- Please report an issue for any failure with the reproduction steps.
5 \t
6 BEGIN;
7 -- WARNING:
8 -- We have found the difference in either of LC_COLLATE or LC_CTYPE or LOCALE,
9 -- so we need to drop the existing collation first and re-create it.
10
11 DROP COLLATION target_sc.col100;
12
13 CREATE COLLATION target_sc.col100
14   (LC_COLLATE = 'POSIX', LC_CTYPE = 'POSIX');
15
16 END;

```

You can also use the [Copy](#) button to copy the difference generated in the [DDL Comparison](#) panel.

Apply the SQL Statement in the target schema to synchronize the schemas.

4.4 Performance Monitoring and Management

PEM contains built-in functionality that implements enterprise-wide performance monitoring of all managed servers. While you can customize many aspects of the various performance monitoring aspects of PEM, you can also elect to accept the recommended defaults that come out-of-the-box with the product.



The top-level dashboard is the [Global Overview](#). The Global Overview presents a status summary of all the servers and agents that are being monitored by the PEM server, a list of the monitored servers, and the state of any currently triggered alerts.

Using Dashboards to View Performance Information

PEM displays performance statistics through a number of dashboards; each dashboard contains a series of summary views that contain charts, graphs and tables that display the statistics related to the selected object.

The PEM client displays the [Global Overview](#) dashboard when it connects to the PEM server. Additional dashboards provide statistical information about monitored objects. These include the:

Alerts Dashboard

The Alerts dashboard displays the currently triggered alerts. If opened from the Global Overview, the dashboard displays the current alerts for all monitored nodes on the system; if opened from a node within a server, the report will reflect alerts related to that node, and all monitored objects that reside below that object in the tree control.

Audit Log Analysis dashboard

For Advanced Server users, the Audit Log Analysis dashboard allows you to browse the audit logs that have been collected from instances that have audit logging and collection enabled.

Database Analysis dashboard

The Database Analysis dashboard displays performance statistics for the selected database.

I/O Analysis dashboard

The I/O Analysis dashboard displays I/O activity across various areas such as object DML activity, log operations and more.

Memory Analysis dashboard

The Memory Analysis dashboard supplies statistics concerning various memory-related metrics for the Postgres server.

Object Activity Analysis dashboard

The Object Activity Analysis dashboard provides performance details on tables/indexes of a selected database.

Operating System Analysis dashboard

The Operating System Analysis dashboard supplies information regarding the performance of the underlying machine's operating system.

Probe Log Analysis Dashboard

The Probe Log Analysis dashboard displays any error messages returned by a PEM agent.

Server Analysis dashboard

The Server Analysis dashboard provides general performance information about the overall operations of a selected Postgres server.

Server Log Analysis dashboard

The Server Log Analysis dashboard allows you to filter and review the contents of server logs that are stored on the PEM server.

Session Activity Analysis dashboard

The Session Activity Analysis dashboard provides information about the session workload and lock activity for the selected server

Session Waits Analysis dashboard

The Session Waits Analysis dashboard provides an overview of the current DRITA wait events for an Advanced Server session.

Storage Analysis dashboard

The Storage Analysis dashboard displays space-related metrics for tablespaces and objects.

System Waits Analysis dashboard

The System Waits Analysis dashboard displays a graphical analysis of system wait information for an Advanced Server

session.

Streaming Replication Analysis dashboard

The Streaming Replication Analysis dashboard displays statistical information about WAL activity for a monitored server and allows you to monitor the status of Failover Manager clusters.

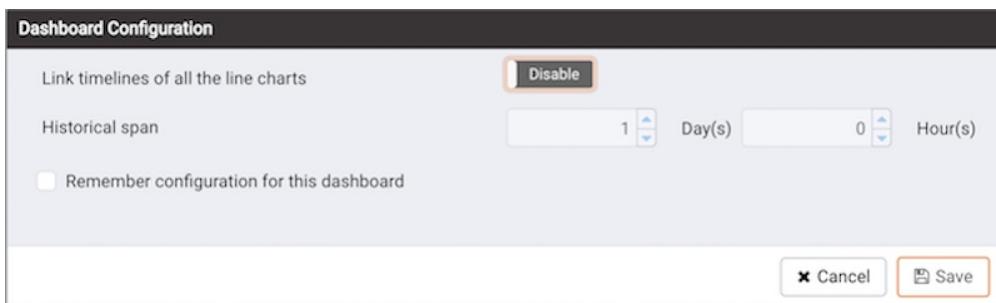
There are two ways to open a dashboard; you can:

- Select an active dashboard name from the **Dashboards** menu (accessed via the Management menu).
- Right click on the name of a monitored object in the tree control and select the name of the dashboard you would like to review from the Dashboards menu.

Each dashboard is displayed on the **Monitoring** tab in the main panel of the client window. After opening a dashboard, you can navigate to other dashboards within the same tab.

Each dashboard header includes navigation menus that allow you to navigate to other dashboards; use your browser's forward and back icons to scroll through previously-viewed dashboards. Use the Refresh icon to update the current dashboard.

Options on the **Dashboard Configuration** dialog allow you to link the time lines of all of the line graphs on the dashboard. To open the **Dashboard Configuration** dialog, click the Settings icon displayed in the dashboard header.



Use fields on the **Dashboard Configuration** dialog to control attributes of the charts displayed on the dashboard:

- Set the Link timelines of all the line charts slider to Enable to indicate that the specified timeline should be applied to line graphs displayed on the dashboard; if set to Disable, your preferences will be preserved for later use, but will not modify the amount of data displayed.
- Use the **Days** selector to specify the number of days of gathered data that should be displayed on line graphs.
- Use the **Hour(s)** selector to specify the number of hours of gathered data that should be displayed on line graphs.
- Check the box next to Remember configuration for this dashboard to indicate that the customized time span should be applied to the current dashboard only; if left unchecked, the time span will be applied globally to line graphs on all dashboards.

Please note that settings specified on the **Dashboard Configuration** dialog are applied only to the current user's session.

Managing Custom Dashboards

PEM displays performance statistics through a number of system-defined dashboards; each dashboard contains a series of summary views that contain charts, graphs and tables that display statistics related to the selected object. You can use the **Manage Dashboards** tab to create and manage custom dashboards that display the information that is most relevant to your system.

The screenshot shows the 'Manage Dashboards' tab in the EDB Postgres Enterprise Manager interface. At the top, there are tabs for Properties, SQL, Statistics, Dependencies, Dependents, Monitoring, and Manage Dashboards. Below the tabs, there's a 'Description' section with a note about creating or modifying user-defined dashboards. Under 'Quick Links', there are 'Create New Dashboard' and 'Help' buttons. The 'Custom Dashboards' section contains a table titled 'Manage Dashboards'. The table has columns for Name, Level, and Description. It lists two dashboards: 'test1' (Level: Database, Description: test1) and 'Test1' (Level: Agent). A search bar at the top right of the table area says 'Search by Name or Level'.

To create a custom dashboard, click the **Create New Dashboard** link (located in the Quick Links section of the Manage Dashboards tab).

To modify an existing dashboard, click the edit icon to the left of a dashboard name. The dashboard editor will open, displaying the definition of the dashboard. When you've finished modifying the dashboard's definition, click the **Save** button to preserve your changes; click **Cancel** to exit without saving your changes.

To delete a dashboard, click the delete icon to the left of a dashboard name. A popup will ask you to confirm that you wish to delete the dashboard; click **OK** to delete the selected dashboard.

Creating a Custom Dashboard

You can use the PEM dashboard editor to create or modify a user-defined dashboard. The custom dashboard may include pre-defined charts, user-defined charts or a mix of pre-defined and user-defined charts.

The screenshot shows the 'Configure' dialog box for creating a new dashboard. The dialog is divided into several sections: 'Name' (input field with 'test'), 'Level' (dropdown menu set to 'Global'), 'Description' (text area), 'Ops dashboard options' (checkboxes for 'Ops dashboard?' and 'Show title?' both set to 'No'), 'Font' (dropdown menu), 'Font size' (dropdown menu), 'Permissions' (checkboxes for 'Share with all users?' set to 'Yes' and 'Access permissions' with a note to specify user groups), and 'Dashboard Layout Design' (button). At the bottom are 'Cancel' and 'Save' buttons.

Use the fields in the **Configure** section to specify general information about the dashboard:

- Specify a name for the dashboard in the **Name** field. The name specified will also be the title of the dashboard if the title is displayed.
- Use the **Level** drop-down listbox to specify the level of the PEM hierarchy within the PEM client on which the dashboard will be displayed. A dashboard may be accessed via the Dashboards menu on a Global level, an Agent level, the Server level or the Database level. Each selected level within the list will expose a different set of metrics on which the custom dashboard's charts may be based.
- Provide a description of the dashboard in the **Description** field.

Provide information in the fields in the **Ops dashboard options** box if the dashboard will be used as an Ops dashboard:

- Set the **Ops Dashboard?** field to Yes to instruct the server to create a dashboard that is formatted for display on an Ops monitor.
- Set the **Show Title?** field to Yes to display the dashboard name at the top of the Ops dashboard.
- Use the **Font** drop-down list box to select a custom font style for the title. The selected font style will be displayed in the Preview box.
- Use the **Font size** drop-down list box to select a custom font size for the title. The selected font style will be displayed in the Preview box.

Use the **Permissions** box to specify the users that will be able to view the new dashboard:

- Set the **Share with all slider** to Yes to instruct the server to allow all Teams to access the dashboard, or set Share with all to No to enable the Access permissions field.
- Use the **Access permissions** field to specify which roles can view the new dashboard. Click in the field, and select from the list of users to add a role to the list of users with dashboard access.

When you've completed the **Configure Dashboard** section, click the arrow in the upper-right corner to close the section, and access the **Dashboard Layout Design** section.



Click the edit icon in a section header to specify a section name; then, click the add icon (+) to add a chart to the section.



Use the arrows to the right of each chart category to display the charts available and select a chart.



Use the chart detail selectors to specify placement details for the chart:

- Use the **Chart width** selector to indicate the width of the chart; select 50% to display the chart in half of the dashboard, or 100% to use the whole dashboard width.
- Use the **Chart alignment** selector to indicate the position of the chart within the section:

Select **Left** to indicate that the chart should be left-justified.

Select **Center** to indicate that the chart should be centered.

Select **Right** to indicate that the chart should be right-justified.

Please note that tables are always displayed centered.

When creating or editing a custom dashboard, you can use drag and drop to re-arrange the charts within a section or to move a chart to a different section.

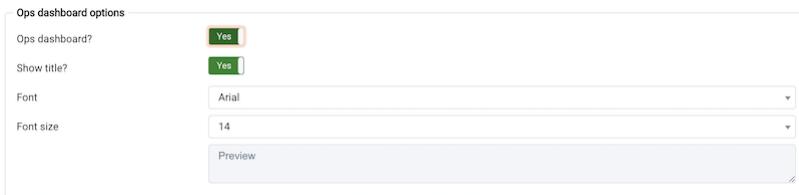
To add another chart to your dashboard, click the add icon (+) in the section header. When you've finished editing the dashboard, click the **Save** button to save your edits and exit.

To exit without saving your changes, click the **Cancel** button.

Creating an Ops Dashboard

You can use the PEM dashboard editor to create a custom dashboard formatted for display on an Ops monitor. An Ops

dashboard displays the specified charts and graphs, while omitting header information and minimizing extra banners, titles, and borders.



To create an **Ops dashboard**, provide detailed information about the Ops display in the **Ops dashboard options** section of the **Create Dashboard** dialog.

- Set the **Ops Dashboard?** field to **Yes** to instruct the server to create a dashboard that is formatted for display on an Ops monitor.
- Set the **Show Title?** field to **Yes** to display the dashboard name at the top of the Ops dashboard.
- Use the **Font** drop-down list box to select a custom font style for the title. The selected font style will be displayed in the **Preview** box.
- Use the **Font size** drop-down list box to select a custom font size for the title. The selected font style will be displayed in the **Preview** box.

After adding charts and tables to the Ops dashboard, click the **Save** button to save your work. You can then access the dashboard by navigating through the Dashboards menu of the hierarchy level specified in the **Level** field on the **New Dashboard** dialog.

Using the Manage Charts tab

You can use the **Manage Charts** tab to access dialogs that allow you to create or modify a custom line chart or table, or import a Capacity Manager template for use in a custom chart. After defining a chart, you can display the chart on a custom dashboard. To open the **Manage Charts** tab, select **Manage Charts...** from the PEM client **Management** menu.

Name	Type	Level	Metrics Category
test1	Table	Database	Database Object Activity
Test1	Line Chart	Agent	Alerts

The **Manage Charts** tab provides a **Quick Links** menu that allows you to access dialogs to:

- **Create a New Chart** for use on a custom dashboard.
- **Import a Capacity Manager** template to use as a template for creating a custom chart.
- Access online **Help**.

The **Custom Charts** table displays a list of user-defined charts; when a chart is newly added, the font displays in green. When you add an additional chart or refresh the screen, the name of the chart is displayed in black.

Custom Charts				
	Name	Type	Level	Metrics Category
<input checked="" type="checkbox"/>	test1	Table	Database	Database Object Activity
<input checked="" type="checkbox"/>	Test1	Line Chart	Agent	Alerts

Use the search box in the upper-right hand corner of the **Custom Charts** table to search through your custom charts. Specify a:

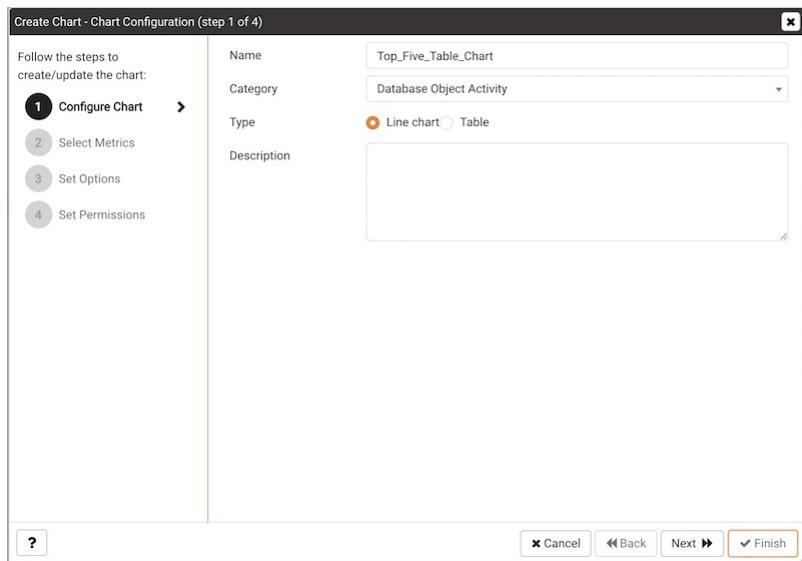
- Chart name
- Type
- Level
- Metrics Category

Use icons to the left of a charts name in the **Custom Charts** table to manage a chart:

- Click the edit icon to open the **Chart Configuration** wizard and modify aspects of the chart or table.
- Click the delete icon to delete the selected chart.

Creating a Custom Chart

Click the **Create New Chart** icon in the **Quick Links** section of the **Manage Charts** tab to open the **Create Chart** wizard. The wizard will walk you through the steps required to define a new chart.



Use the fields on the **Configure Chart** dialog to specify general information about the chart:

- Specify the name of the chart in the **Name** field.
- Use the drop-down listbox in the **Category** field to specify the category in which this chart will be displayed; when adding a custom chart to a custom dashboard, the chart will be displayed for selection in the category specified.
- Use the radio buttons in the **Type** field to specify if the chart will be a **Line chart** or a **Table**.
- Provide a description of the chart in the **Description** field. The description will be displayed to the user viewing the chart (on a custom dashboard) when they click the information icon.

When you've completed the fields on the **Configure Chart** dialog, click **Next** to continue.



Use the fields on the **Select Metrics** dialog to select the metrics that will be displayed on the chart.

- Use the **Metric level** drop-down listbox to specify the level of the PEM hierarchy from which you wish to select metrics. You can specify Agent, Database, or Server. Each level offers access to a unique set of probes and metrics.
- Use the tree control in the Available metrics box to select the metrics that will be displayed on the chart.

If you are creating a table, you may only select metrics from one probe; each node of the tree control lists the metrics returned by a single probe. Expand a node of the tree control, and check the boxes to the left of a metric name to include that metric data in the table.

If you are creating a line chart, expand the nodes of the tree control and double-click each metric that you would like to include in the chart.

- Use the fields in the Selected metrics panel to specify how the metric data will be displayed in your chart. The selection panel displays the name of the metric in the (non-modifiable) Metric [Probe] column. You can:
 - Click the garbage can icon to delete a metric from the list of selected metrics.
 - Use the drop-down listboxes in the **Selection Criteria** column to specify the order of the data displayed.
 - Use the **Limit** field to specify the number of rows in a table or lines in a chart:

The maximum number of lines allowed in a chart is 32.

The maximum number of rows allowed in a table is 100.

- If you are creating a line chart, PEM supports comparisons of cross-hierarchy metrics.
 - Click the **compare icon** to open a selection box that allows you to select one or more probe-specific attributes (i.e. CPUs, interfaces, databases, etc.) to compare in the chart.
 - Click the **copy** icon to apply your selections to all of the metrics for the same probe. When the popup opens, click **Yes** to confirm that other selections for the same probe will be overwritten, or **No** to exit the popup without copying the attributes.

When you've completed the fields on the **Select Metrics** dialog, click **Next** to continue.



Use the fields on the **Set Options** dialog to specify display options for your chart:

- Use the **Auto Refresh** field to specify the number of minutes between chart updates - choose a value from 1 to 120. The default auto refresh rate is 2 minutes.

Use fields under the **Line chart options** heading to specify display preferences for a line chart:

- Use the **Points to plot** field to specify the maximum number of points that will be plotted on the chart.
- Use the fields to the right of the **Historical span** label to specify how much historical data should be displayed on the chart:
 - Use the **Day(s)** field to specify the number of days of historical data that should be included on the chart.
 - Use the **Hour(s)** field to specify the number of hours of historical data that should be included on the chart.
 - Use the **Minute(s)** field to specify the number of minutes of historical data that should be included on the chart.

Use the fields in the **Data extrapolation** box to specify if PEM should generate extrapolated data based on historical data:

- Click the **No Extrapolation** label to omit extrapolated data from the chart.
- Click the **Span** label to use the Days and Hours selectors to specify the period of time spanned by the metrics on the chart.
- Click the **Threshold** label to use threshold selectors to specify a maximum or minimum value for the chart.

When you've completed the fields on the **Set Options** dialog, click **Next** to continue.



Use the fields on the **Set Permissions** dialog to specify display options for your chart.

- Set the **Share with all** slider to **Yes** to indicate that the chart will be available to all authorized users, or **No** to restrict access to the users or groups specified in the Access permissions field.
- Use the **Access permissions** field to select the group or groups that will have access to the chart.

	Name	Type	Level	Metrics Category
<input checked="" type="checkbox"/>	test1	Table	Database	Database Object Activity
<input checked="" type="checkbox"/>	Test1	Line Chart	Agent	Alerts
<input checked="" type="checkbox"/>	Top_Five_Table.Chart	Line Chart	Database	Database Object Activity

When you've finished defining the chart, click **Finish** to save your edits and add your chart to the list on the **Manage Charts** tab.

Importing a Capacity Manager Template

Click the **Import Capacity Manager Template** icon in the Quick Links section of the **Manage Charts** tab to open the **Create Chart** dialog, and use a Capacity Manager template as a starting point for a chart or table.

When the **Create Chart** dialog opens, provide information about the custom chart:

- Use the drop-down listbox in the **Import capacity template** field to select the name of the template on which the chart will be based.
- Specify the name of the chart in the **Name** field.
- Use the drop-down listbox in the **Category** field to specify the category in which this chart will be displayed. When adding a custom chart to a custom dashboard, the chart will be displayed for selection in the Category specified.
- Use the radio buttons in the **Type** field to specify if the chart will be a **Line chart** or a **Table**.
- Provide a description of the chart in the **Description** field. The description will be displayed to the user viewing the chart (on a custom dashboard) when they click the information icon.

Click **Next** to continue to the Select Metrics dialog.

Metrics	Metric details
Blocks Read+	Display Name: Blocks Read+ (Postgres Enterprise Manager Host) Probe: IO Analysis Metric: Blocks Read+ Host: Postgres Enterprise Manager Host
Blocks Read+	Display Name: Blocks Read+ (Postgres Enterprise Manager Host) Probe: IO Analysis Metric: Blocks Read+ Host: Postgres Enterprise Manager Host
Blocks Read	Display Name: Blocks Read (Postgres Enterprise Manager Host) Probe: IO Analysis Metric: Blocks Read Host: Postgres Enterprise Manager Host
Blocks Read	Display Name: Blocks Read (Postgres Enterprise Manager Host) Probe: IO Analysis Metric: Blocks Read Host: Postgres Enterprise Manager Host
Blocks Written+	Display Name: Blocks Written+ (Postgres Enterprise Manager Host) Probe: IO Analysis Metric: Blocks Written+ Host: Postgres Enterprise Manager Host

Buttons at the bottom: ? (Help), Cancel, Back, Next, Finish.

The **Select Metrics** window allows you to review the metrics specified by the selected template. The bottom panel of the chart editor displays the metrics that will be included in the chart. The metrics included in the chart are not modifiable via the chart editor; to modify the metrics, you must use the Capacity Manager utility to update the template.

When you've reviewed the metrics, click **Next** to continue to the Set Options dialog.

Auto refresh: 2 Minute(s)
Please specify the number of minutes between chart updates.

Data extrapolation:

- Historical days and extrapolated days Historical days and threshold
- Historical: 5 Day(s)
- Extrapolated: 0 Day(s)

Use the fields on the **Set Options** window to specify display options for your chart:

- Use the **Auto Refresh** field to specify the number of minutes between chart updates - choose a value from 1 to 120. The default auto refresh rate is 2 minutes.

Use the fields in the **Data extrapolation** box to specify the time period covered by the chart. You can either:

- click the **Historical days and extrapolated days** label and provide:
 - the number of days of historical data that should be charted in the **Historical** field.
 - the number of projected days that should be charted in the **Extrapolated** field.
- or, click the **Historical days and threshold** label and provide:
 - the number of days of historical data that should be charted in the **Historical** field

- the **threshold** value at which the chart will end.

When you've completed the Set Options window, click **Next** to continue.



Use the fields on the **Set Permissions** window to specify display options for your chart:

- Set the **Share with all slider** to Yes to indicate that the chart will be available to all authorized users, or No to restrict access to the users or groups specified in the Access permissions field.
- Use the **Access permissions** field to select the group or groups that will have access to the chart.

When you've finished defining the chart, click **Finish** to save your edits and add your chart to the list on the **Manage Charts** tab.

Probes

A **probe** is a scheduled task that retrieves information about the database objects that are being monitored by the PEM agent. PEM uses the collected information to build the graphs displayed on each homepage. The **Manage Probes** tab (accessed via the **Management** menu) allows you to modify the data collection schedule and the length of time that PEM will retain information returned by a specific probe.

System Probes

Unless otherwise noted, Postgres Enterprise Manager enables the probes listed in the table below:

Probe Name	Information Monitored by Probe	Level
Background Writer Statistics	This probe monitors information about the background writer. The information includes: The number of timed checkpoints The number of requested checkpoints The number of buffers written (by checkpoint) The number of buffers written (by background writer) The number of background writer cycles The number of background buffers written The number of buffers allocated	Server

Probe Name	Information Monitored by Probe	Level
Blocked Session Information	This probe provides information about blocked sessions.	Server
CPU Usage	This probe monitors CPU Usage information.	Agent
Data and Log File Analysis	This probe monitors information about log files. The information includes: The name of the log file The directory in which the log file resides	Server
Database Statistics	This probe monitors database statistics. The information includes: The number of backends The number of transactions committed The number of transactions rolled back The number of blocks read The number of blocks hit The number of rows returned The number of rows fetched The number of rows inserted The number of rows updated The number of rows deleted	Server
Disk Busy Info	This probe monitors information about disk activity. Note: This probe is not supported on Mac OS X, Solaris or HP-UX	Agent
Disk Space	This probe monitors information about disk space usage. The information includes: The amount of disk space used The amount of disk space available	Agent
EDB Audit Configuration	This probe monitors the audit logging configuration of EDB Postgres Advanced Server.	Server
Failover Manager Cluster Info	This probe monitors a Failover Manager cluster, returning information about the cluster. This probe is disabled unless a cluster name and path of the Failover Manager binary is provided on the Server Properties dialog.	Server
Failover Manager Node Status	This probe monitors a Failover Manager cluster, returning detailed about each node within the cluster. This probe is disabled unless a cluster name and path of the Failover Manager binary is provided on the Server Properties dialog.	Server
Function Statistics	This probe monitors a database, retrieving information about functions. The information includes: Function names Argument types Return values	Database
Index Size	This probe monitors a database, retrieving information about indexes. The information includes: The name of the index The time the data was gathered The size of the index (in MB's)	Database
Index Statistics	This probe monitors index statistics. The information includes: The number of index scans The number of rows read The number of rows fetched The number of blocks read The number of blocks hit	Database

Probe Name	Information Monitored by Probe	Level
Installed Packages	<p>This probe monitors the packages that are currently installed. The information gathered includes:</p> <ul style="list-style-type: none"> The name of the installed package The version of the installed package The date and time that the probe executed 	Agent
IO Analysis	<p>This probe monitors disk I/O information in. The information includes:</p> <ul style="list-style-type: none"> The number of blocks read The number of blocks written The date and time that the probe executed <p>Note: This probe is not supported on Mac OS X</p>	Agent
Load Average	<p>This probe monitors CPU load averages. The information includes:</p> <ul style="list-style-type: none"> The 1-minute load average The 5-minute load average The 15-minute load average <p>Note: This probe is not supported on Windows</p>	Agent
Lock Information	<p>This probe monitors lock information. The information includes:</p> <ul style="list-style-type: none"> The database name The lock type The lock mode The process holding the lock 	Server
Memory Usage	<p>This probe monitors information about system memory usage.</p>	Agent
Network Statistics	<p>This probe monitors network statistics. The information includes:</p> <ul style="list-style-type: none"> The interface IP address The number of packets sent The number of packets received The number of bytes sent The number of bytes received The link speed (in MB/second) 	Agent
Number of Prepared Transactions	<p>This probe stores the number of prepared transactions.</p>	Server
Number of WAL Files	<p>This probe monitors the number of WAL files.</p>	Server
Object Catalog: Database	<p>This probe monitors a list of databases and their properties. The information includes:</p> <ul style="list-style-type: none"> The database name The database encoding type If the database allows user connections or system connections 	Server
Object Catalog: Foreign Key	<p>This probe monitors a list of foreign keys and their properties. The information includes:</p> <ul style="list-style-type: none"> The name of the table that contains the foreign key The name of the table that the foreign key references The name of the database in which the table resides The name of the schema in which the table resides 	Schema
Object Catalog: Function	<p>This probe monitors a list of functions and their properties. The information includes:</p> <ul style="list-style-type: none"> The name of the function The name of the schema in which the function resides The name of the database in which the function resides 	Schema
Object Catalog: Index	<p>This probe monitors a list of indexes and their properties. The information includes:</p> <ul style="list-style-type: none"> The name of the index The name of the table that the index is associated with The name of the database in which the indexed table resides 	Schema

Probe Name	Information Monitored by Probe	Level
Object Catalog: Schema	This probe monitors a list of schemas and their associated databases and servers.	Database
Object Catalog: Sequence	This probe monitors a list of sequences and their properties.	Schema
Object Catalog: Table	<p>This probe monitors a list of table information. The information includes:</p> <p>The table name The name of the schema in which the table resides The name of the database in which the schema resides A Boolean indicator that indicates if the table has a primary key</p>	Schema
Object Catalog: Tablespace	This probe monitors a list of tablespaces.	Server
Operating System Information	This probe monitors the operating system details and boot time.	Agent
Package Catalog	<p>This probe monitors the packages that are currently available for installation. The information gathered includes:</p> <p>The package name The package version</p>	Agent
PG HBA Conf	This probe monitors authentication configuration information from the pg_hba.conf file.	Server
Server Information	This probe monitors server information.	Server
Session Information	<p>This probe monitors session information. The information includes:</p> <p>The name of the session user The date and time that the session connected to the server The status of the session at the time that the information was gathered (idle, waiting, etc) The client address and port number</p>	Server
Settings	This probe monitors the values currently assigned to GUC variables.	Server
SQL Protect	This probe monitors a server, retrieving information about SQL injection attacks.	Server
Slony Replication	This probe monitors lag data for clusters replicated using Slony.	Database
Streaming Replication	<p>This probe monitors a cluster that is using streaming replication, retrieving information about:</p> <p>The sent Xlog location (in bytes) The write Xlog location (in bytes) The flush Xlog location (in bytes) The replay Xlog location (in bytes) The Xlog lag (in segments) The Xlog lag (in pages)</p>	Server
Streaming Replication Lag Time	<p>This probe monitors a cluster that is using streaming replication, retrieving lag information about:</p> <p>Replication lag time (in seconds) Current status of replication (running/paused)</p>	Server
Streaming Replication Database Conflicts	<p>This probe monitors a database that is using streaming replication, retrieving information about any conflicts that arise. This includes information about queries that have been canceled due to:</p> <p>The # of drop tablespace conflicts The # of lock timeout conflicts The # of old snapshot conflicts The # of pinned buffer conflicts The # of deadlock conflicts</p>	Server

Probe Name	Information Monitored by Probe	Level
Table Bloat	This probe monitors information about the current table bloat. The information includes: The name of the table The name of the schema in which the table resides The estimated number of pages The estimated number of wasted pages The estimated number of bytes per row	Database
Table Frozen XID	This probe monitors the frozen XID of each table.	Schema
Table Size	This probe monitors table statistics. The information includes: The number of sequential scans The number of sequential scan rows The number of index scans The number of index scan rows The number of rows inserted The number of rows updated The number of rows deleted The number of live rows The number of dead rows The last VACUUM The last auto-vacuum The last ANALYZE The last auto-analyze The number of pages estimated by ANALYZE The number of rows estimated by ANALYZE	Database
Table Statistics	This probe monitors a list of tablespaces and their sizes.	Server
Tablespace Size	This probe monitors a list of tablespaces and their sizes.	Server
User Information	This probe monitors a list of the current users. The stored information includes: The user name The user type (superuser vs. non-superuser) The server to which the user is connected	Server
WAL Archive Status	This probe monitors the status of the WAL archive. The stored information includes: The # of WAL archives done The # of WAL archives pending The last archive time The # of WAL archives failed The time of the last failure	Server
xDB Replication	This probe monitors lag data for clusters replicated using xDB replication.	Database

Customizing Probes

A probe is a scheduled task that returns a set of performance metrics about a specific monitored object. A probe retrieves statistics from a monitored server, database, operating system or agent. You can use the [Manage Probes](#) tab to override the default configuration and customize the behavior of each probe.

To open the [Manage Probes](#) tab, select [Manage Probes...](#) from the [Management](#) menu. The [Manage Probes](#) tab opens in the PEM client.

Probe name	Execution Frequency		Enabled?	Data Retention	
	Default?	Minutes		Seconds	Default?
Background Writer Statistics	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	180
Blocked Session Information	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	180
Data and Log File Analysis	<input checked="" type="checkbox"/>	0	10	<input checked="" type="checkbox"/>	180
Database Frozen XID	<input checked="" type="checkbox"/>	720	0	<input checked="" type="checkbox"/>	180
Database Size	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	180
Database Statistics	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	90
Failover Manager Cluster Info	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	7
Failover Manager Node Status	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	7
Lock Information	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	180
Number of Prepared Transactions	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	180
Number of WAL Files	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	180
Object Catalog: Database	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	180
Object Catalog: Tablespace	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	180
PG HBA Conf	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	180
Server Information	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	180
Server log Configuration	<input checked="" type="checkbox"/>	0	10	<input checked="" type="checkbox"/>	180
Session Information	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	180
Settings	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	180
Streaming Replication	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	180
Streaming Replication Database Conflicts	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	180
Streaming Replication Lag Time	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	180
Tablespace Size	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	180
User Information	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	180
WAL Archive Status	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	180

The **Manage Probes** tab provides a set of Quick Links that you can use to create and manage probes:

- Click the **Manage Custom Probes** icon to open the **Custom Probes** tab and create or modify a custom probe.
- Click the **Copy Probes** icon to open the Copy Probe dialog, and copy the probe configurations from the currently selected object to one or more monitored objects.

A probe monitors a unique set of metrics for each specific object type (server, database, database object, or agent); select the name of an object in the tree control to review the probes for that object.

To modify the properties associated with a probe, highlight the name of a probe, and customize the settings that are displayed in the Probes table:

- Move the **Default** switch in the **Execution Frequency** columns to **N** to enable the Minutes and Seconds selectors, and specify a non-default value for the length of time between executions of the probe.
- Move the **Default** switch in the **Enabled?** column to **No** to change the state of the probe, and indicate if the probe is active or not active.

!!! Note If data from a disabled probe is used in a chart, the chart will display an information icon in the upper-left corner that allows you to enable the probe by clicking the provided link.

- Move the **Default** switch in the **Data Retention** column to **No** to enable the Day(s) field and specify the number of days that information gathered by the probe is stored on the PEM server.

The **Manage Probes** tab may display information about probes that cannot be modified from the current node. If a probe cannot be modified from the current dialog, the switches are disabled. Generally, a disabled probe can be

modified from a node that is higher in the hierarchy of the PEM client tree control; select another object in the tree control to modify which probes are displayed or enabled in the **Manage Probes** tab.

Creating a Custom Probe

You can use the **PEM Custom Probes** tab to create a new probe or modify an existing user-defined probe. To open the **Custom Probes** tab, select the **Manage Custom Probes...** icon from the **Manage Probes** tab.

The screenshot shows the 'Probes' tab of the PEM Custom Probes interface. At the top, there's a 'Description' section with information about system and custom probes. Below it is a table with one row, showing details for a probe named 'Top_Five_Large_Tables'. The table columns include 'Probe name', 'Collection method', 'Target type', 'Execution frequency' (Minutes: 1, Seconds: 0), 'Probe enabled?' (Yes), and 'Data retention' (1). There are buttons for adding (+) and editing (Edit).

Probe name	Collection method	Target type	Execution frequency		Probe enabled?	Data retention
Top_Five_Large_Tables	SQL	Table	1	0	Yes	1

Use the **Show System Probes?** switch to display or conceal the system probes on the **Custom Probes** tab.

You can use the **Custom Probes** tab to create a new probe or modify an existing probe. To create a new probe, click the **Add** icon in the upper-right corner of the tab; provide a name for the new probe in the **Probe Name** column. Then, select the **Edit** icon (located to the left of the probe name) to review or add the probe definition.

The screenshot shows the 'Probes' tab with the 'Edit' icon for the 'Top_Five_Large_Tables' probe selected. The 'General' tab is active, showing the probe's current configuration:

- Probe name:** Top_Five_Large_Tables (highlighted with a red error border)
- Collection method:** SQL
- Target type:** Server
- Execution frequency:** Minutes: 5, Seconds: 0
- Probe enabled?**: Yes
- Data retention:** 1 day

Below the table, there are tabs for 'General', 'Columns', 'Code', and 'Alternate Code'. The 'General' tab is selected, showing detailed configuration fields:

- Probe name:** Input field with an error message: 'Please specify Probe name'.
- Collection method:** SQL dropdown with a note: 'Use the Collection method field to specify the probe type. Use the drop-down to select: SQL (the probe will gather information via a SQL statement), WMI (the probe will gather information via a Windows Management Instrumentation extension), Batch/Shell Script (the probe will use a command script or shell script to gather information). Please note that batch probes are platform specific. If you specify a collection method of Batch, you must specify a platform type in the Platform field.'
- Target type:** Server dropdown with a note: 'Use the Target type drop-down to select the object type that the probe will monitor.'
- Execution frequency:** Minutes: 5, Seconds: 0
- Probe enabled?**: Yes (selected)
- Data retention:** 1 day
- Discard from history?**: No (selected)
- Platform:** *nix dropdown with a note: 'Use the Platform drop-down to specify the type of platform that the probe will monitor. This field is enabled only when the Collection method is Batch/Shell Script.'

Use the fields on the **General** tab to modify the definition of an existing probe or to specify the properties of a new probe:

- Use the **Probe Name** field to provide a name for a new probe.
- Use the **Collection method** field to specify the probe type. Use the drop-down listbox to select:
 - SQL - the probe will gather information via a SQL statement.
 - WMI - the probe will gather information via a Windows Management Instrumentation extension.
 - Batch - the probe will use a command-script or shell-script to gather information.

Before creating a batch probe on a Linux system, you must modify the **agent.cfg** file, setting the **allow_batch_probes** parameter equal to **true**, and restart the PEM agent. The **agent.cfg** file is located in one of the following directories:

- If you have installed PEM using graphical installer: **/opt/edb/pem/agent/etc/agent.cfg**
- If you have installed PEM using RPM: **/usr/edb/pem/agent/etc/agent.cfg**

On 64-bit Windows systems, agent settings are stored in the registry. Before creating a batch probe, modify the registry entry for the **AllowBatchProbes** registry entry and restart the PEM agent. PEM registry entries are located in **HKEY_LOCAL_MACHINE\Software\Wow6432Node\EnterpriseDB\PEM\agent**.

Please note that batch probes are platform-specific. If you specify a collection method of Batch, you must specify a platform type in the Platform field.

To invoke a script on a Linux system, you must modify the entry for **batch_script_user** parameter of **agent.cfg** file and specify the user that should be used to run the script. You can either specify a non-root user or root for this parameter. If you do not specify a user, or the specified user does not exist, then the script will not be executed. Restart the agent after modifying the file.

To invoke a script on a Windows system, set the registry entry for **AllowBatchJobSteps** to true and restart the PEM agent.

- Use the **Target Type** drop-down listbox to select the object type that the probe will monitor. Target type is disabled if Collection method is WMI.
- Use the **Minutes** and **Seconds** selectors to specify how often the probe will collect data.
- Use the **Probe Enable?** switch to specify if the probe is enabled by default. Specify Yes to enable the probe by default, or No to specify that the probe is disabled by default.

!!! Note If data from a disabled probe is used in a chart, the chart will display an information icon in the upper-left corner that allows you to enable the probe by clicking the provided link.

- Use the **Data Retention** field to specify the number of days that gathered information will be retained in the probe's history table.
- Use the switch next to **Discard from history** to specify if the server should create a history table for the probe. Select **Yes** to discard probe history, or **No** to retain the probe history in a table.
- Use the **Platform** drop-down listbox to specify the type of platform that the probe will monitor. This field is enabled only when the Collection method is Batch.

Use the **Columns** tab to define the columns in which the probe data will be stored. Navigate to the **Columns** tab, and click the **Add** button (in the upper-right corner) to define a new column. After providing a column name in the **Name** field, click the **Edit** button (to the left of the new column name) to provide information about the column:

- Provide a descriptive name for the column in the **Name** field.
- The **Internal Name** field is not enabled for user-defined probes.
- Use the **Column Type** drop-down listbox to specify if the column is a Key column (a primary key) or a Non key column. Non-key columns are generally metric items (values that can be graphed).
- Use the **Data Type** drop-down listbox to specify the type of data that will be stored in the column.
- Use the **Unit** field to specify the unit of measure that applies to the metric stored in the column. This unit is displayed on the Y-Axis of a custom chart or a Capacity Manager chart. This is an optional field.
- Use the **Graphable** switch to specify if the defined metric may be graphed, and that the probe should be accessible from the Capacity Manager or Manage Charts dialogs.
- Use the **Is PIT** switch to specify if the metric should be stored by point-in-time.

'Point-in-time' metrics are those metrics that change (increase or decrease) at any given point of time. For example, database size is a point-in-time metric; at any given point-in-time, the size of the database is fluctuating. Metrics that are not point-in-time (also referred to as cumulative metrics) are metrics whose size always increases over time. For

example, Blocks Read and Tuples Read are cumulative metrics; the value stays the same or increases.

- Use the **Calculate PIT** switch to specify that the server should calculate a point-in-time value for the metric data. **Calculate PIT** is disabled if **Is PIT** is **Yes**.

PEM allows you to store point-in time-values of cumulative metrics as well. PEM subtracts the last collected value of a cumulative metric from the current value, and stores the difference as a point-in-time value.

Use the **Code** tab to specify the default code that will be executed by the probe:

- If the probe is a SQL probe, you must specify the **SQL SELECT** statement invoked by the probe on the **Code** tab. The column names returned by the query must match the Internal Name specified on the **Columns** tab. The number of columns returned by the query, as well as the column name, data type, etc. must match the information specified on the **Columns** tab.
- If the probe is a batch probe, you must specify the shell or .bat script that will be invoked when the probe runs. The output of the script should be as follows:

The first line must contain the names of the columns provided on the **Columns** tab. Each column name should be separated by a tab (t) character. From the second line onwards, each line should contain the data for each column, separated by a tab character.

If a specified column is defined as key column, you should ensure that the script does not produce duplicate data for that column across lines of output. The number of columns specified in the **Columns** tab and their names, data type, etc. should match with the output of the script output.

- If the probe is a WMI probe, you must specify the WMI query as a **SELECT WMI** query. The column name referenced in the **SELECT** statement should be same as the name of the corresponding column specified on the **Column** tab. The column names returned by the query must match the **Internal Name** specified on the **Column** tab. The number of columns returned by the query, as well as the column name, data type, etc. must match the information specified on the **Columns** tab.

The screenshot shows the 'Probes' tab in the EDB Postgres Enterprise Manager. The 'Alternate Code' tab is active. A note at the top states: 'Move the Applies to all database server versions switch to Yes to specify that the code on the Code tab will execute for every server version. If Applies to all database server versions? is set to No, you may specify code for a specific server version below. Applies to all database server versions? is disabled when the Collection method is WMI and Batch. Do not specify the alternate probe code for a database server version to use the default code as specified in the Code Tab.' Below this, there's a section for 'Database version(s)' and 'Probe code'. A note says 'No alternate code found for custom probe'. At the bottom, a red warning box says '⚠ Please specify Probe name'.

Use the **Alternate Code** tab to provide code that will be invoked if the probe fires on a specific version of the server. To provide version-specific code, move the **Applies to any server version?** switch to **No**, and click the **Add** button. Then, use the **Database Version(s)** drop-down listbox to select a version, and click the **Edit** button (to the left of the version name) to provide the code that will execute when the probe fires.

If you select a database version, and leave the **Probe Code** column blank, PEM will invoke the code specified on the **Code** tab when the probe executes on a server that matches that version.

When you've finished defining the probe, click the **Save** icon (in the corner of the **Custom Probes** tab) to save the definition, and make the probe data available for use on custom charts and graphs.

Deleting a Probe

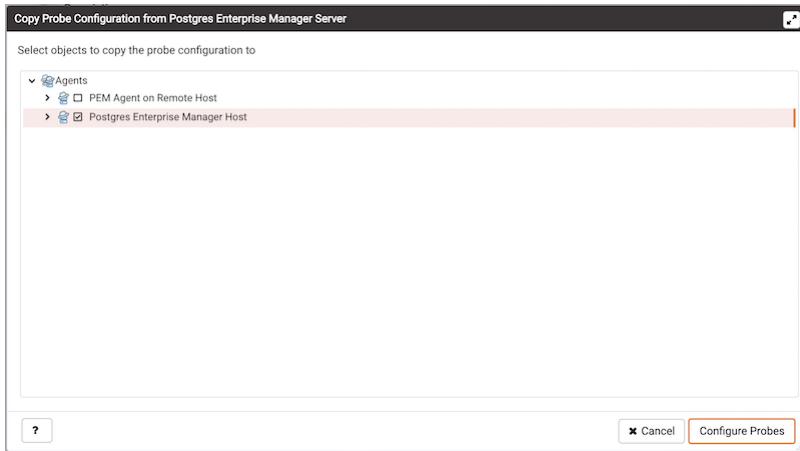
Use the **Delete** icon (located to the left of a probe name) to delete a user-defined probe. When you delete a probe, the probe is marked for deletion and will be deleted later (when custom probes are purged). During the deletion, the probe definition is deleted and any corresponding tables are dropped from the **pemdata** and **pemhistory** schemas.

System probes are the built-in probes provided by PEM, and are part of the PEM schema. If you attempt to delete a system probe, the PEM client will display a notice, informing you that the probe cannot be deleted.



Copying a Probe

You can use the **Copy Probe Configuration...** dialog to copy probe definitions from one monitored object to one or more monitored objects of the same type. To open the **Copy Probe Configuration...** dialog, highlight the object from which you are copying probes in the PEM client tree control, and select **Manage Probes** from the **Management** menu. When the **Manage Probes** tab opens, click on **Copy Probe** to open the **Copy Probe Configuration** dialog:



The dialog will copy the probe definitions from the object through which the Copy Probe Configuration dialog was opened, to the location(s) selected on the tree control.

Note that if you specify a parent node in the [Copy Probe Configuration](#) tree control, PEM will copy the probe configurations to each object (of the same type) that resides under that node in the tree control. For example, to copy the probe definitions from one schema to all schemas that reside within a database, select only the parent database of the target schemas. Please note that a red warning symbol is displayed to the left of the name of a listed target object if that object is the source of the probe that is being copied.

When you have selected the target object or objects, click the [Configure Probes](#) button to copy the probe definitions to the location selected on the dialog.

Alerting

PEM continually monitors registered servers and compares performance metrics against pre-defined and user-specified thresholds that constitute good or acceptable performance for each statistic. Any deviation from an acceptable threshold value triggers an alert. An alert is a system-defined or user-defined set of conditions that PEM compares to the system statistics. Alerts call your attention to conditions on registered servers that require your attention.

Viewing the Alerts via Global Dashboard

When your system statistics deviate from the boundaries specified for that statistic, the alert triggers, displaying a high (red), low (yellow), or medium (orange) severity warning in the left-most column of the [Alert Status](#) table on the [Global Overview](#) dashboard.

Alerts Status							
	Alarm Type	Object Description	Alert Name	Value	Database	Schema	Package
▶	● High	EDB Postgres Advanced Server 11	Database size in server	113 MB			
▶	● High	EDB Postgres Advanced Server 11	Last Vacuum	Never ran			
▶	● High	EDB Postgres Advanced Server 11	Last AutoVacuum	140.21 hrs			
▶	● High	EPAS_12	Table size in server	410 MB			
▶	● Medium	EPAS_12	Last Vacuum	5.18 hrs			
▶	● High	EPAS_12	Database size in server	455 MB			
▶	● Medium	EPAS_12	Last AutoVacuum	5.16 hrs			
▶	● High	N/A	Alert Errors	3			
▶	● High	PGSQL12_Centos7_1	Server Down	1			
▶	● High	PGSQL12_Centos7_1	Last Vacuum	Never ran			
▶	● High	PGSQL12_Centos7_1	Last AutoVacuum	Never ran			
▶	● High	Postgres Enterprise Manager Server	Largest index by table-size percentage	100 %			
▶	● High	Postgres Enterprise Manager Server	Database size in server	2.072265625 GB			
▶	● High	Postgres Enterprise Manager Server	Table size in server	1.9814453125 GB			
▶	● Medium	Postgres Enterprise Manager Server	Connections in idle state	12			
▶	● Medium	Postgres Enterprise Manager Server	Last Vacuum	4.99 hrs			

The PEM server includes a number of pre-defined alerts that are actively monitoring your servers. If the alert definition makes details available about the cause of the alert, you can click the down arrow to the right of the severity warning to access a dialog with detailed information about the condition that triggered the alert.

Alert Details (Auto-refresh paused whilst rows are expanded. ⏪)

Ack'd	Alert Type	Name	Value	Agent	Server	Database	Schema	Package	Object	Alerting Since
▶	☐ ● High	Table size in server	1.9814453125 GB	Postgres Enterprise Manager Server						2020-02-20 11:29:45

General Parameters

Table name	Schema name	Database name	Total table size(MB)
table_statistics	pemhistory	pem	1087
server_logs	pemdata	pem	263
index_statistics	pemhistory	pem	237
session_info	pemhistory	pem	137
lock_info	pemhistory	pem	88

PEM also provides an interface that allows you to create customized alerts. Each alert uses metrics defined on an alert template. An alert template defines how the server will evaluate the statistics for a resource or metric. The PEM server includes a number of pre-defined alert templates, or you can create custom alert templates.

Viewing the Alerts via Alerts Dashboard

Use the **Dashboards** menu (on the **Monitoring** tab) to access the **Alerts** dashboard. The **Alerts** dashboard displays a summary of the active alerts and the status of each alert:

The screenshot shows the 'Alerts Overview' section of the Postgres Enterprise Manager interface. At the top, there's a navigation bar with 'Postgres Enterprise Manager Host', 'Postgres Enterprise Manager Server', and 'Alerts'. Below that, a status bar shows 'Object Type: Server Status: UP (Since: 27/04/2020, 15:47:09)', 'Generated On: 27/04/2020, 21:00:47', 'No. of alerts: 5 (Acknowledged: 0)', and a settings gear icon.

Alert Status:

Severity	Count
High	2
Medium	1
Low	0
None	2

Alert Details:

Acked	Alert Type	Name	Value	Agent	Server	Database	Schema	Package	Object	Alerting Since
▶	☐ High	Table size in server	1.9814453125 GB	Postgres Enterprise Manager Server						2020-02-20 11:29:45
▶	☐ High	Database size in server	2.072265625 GB	Postgres Enterprise Manager Server						2020-02-05 18:26:49
▶	☐ High	Largest index by table-size percentage	100 %	Postgres Enterprise Manager Server						2020-04-21 22:07:52
▶	☐ Medium	Connections in idle state	15	Postgres Enterprise Manager Server						2020-04-27 16:20:32
▶	☐ Medium	Last Vacuum	4.99 hrs	Postgres Enterprise Manager Server						2020-04-27 20:47:50

Alert Errors:

Alert Type	Name	Value	Agent	Server	Database	Schema	Package	Object	Error Message	Error Timestamp
☒ Error	Number of WAL archives pending		Postgres Enterprise Manager Server						Required probe(s) wal_archive_status are disabled.	2020-01-21 14:25:04

The **Alerts Dashboard** header displays the date and time that the dashboard was last updated, and the number of current alerts.

The **Alerts Overview** section displays a graphic representation of the active alerts, as well as a count of the current high, low and medium alerts. The vertical bar on the left of the graph provides the count of the alerts displayed in each column. Hover over a bar to display the alert count for the selected alert severity in the upper-right hand corner of the graph.

The **Alert Details** table provides a list of the alerts that are currently triggered. The entries are prioritized from high-severity to lower-severity; each entry includes information that will allow you to identify the alert and recognize the condition that triggered the alert. Click the name of an alert to review detailed information about the alert definition.

The **Alert Errors** table displays configuration-related errors (eg. accidentally disabling a required probe, or improperly configuring an alert parameter). You can use the information provided in the Error Message column to identify and resolve the conflict that is causing the error.

Customizing the Alerts Dashboard

You can customize tables and charts that appear on the Alerts dashboard. To customize a table or chart, click the Settings icon located in the upper-right corner.



Use fields on the Personalize chart configuration dialog to provide your display preferences:

- Use the **Auto Refresh** field to specify the number of seconds between updates of the data displayed in the table or chart.
- If applicable, use the **Download as** field to indicate if you would like a chart to be downloaded as a JPEG image or a PNG image.
- If applicable, use the **Colours selectors** to specify the display colors that will be used on a chart.

- If applicable, set the **Show Acknowledged Alerts** switch to Yes indicate that you would like the table to display alerts that you have acknowledged with a checkbox in the Ack'ed column. Set the field to No to indicate that the table should hide any acknowledged alerts. The switch acts as a toggle; acknowledged alerts are not purged from the table content until the time specified in the alert definition passes.

To save your customizations, click the **Save** icon (a check mark) in the upper-right corner; to delete any previous changes and revert to the default values, click the **Delete** icon. The **Save** and **Delete** drop-down menus allow you to specify if your preferences should be applied to **All Dashboards**, or to a selected server or database.

Managing Alerts

Use the PEM Client's **Manage Alerts** tab to define, copy, or manage alerts. To open the **Manage Alerts** tab, select **Manage Alerts** from the **Management** menu.

Name	Auto created?	Template	Enable?	Interval		History retention	
				Default?	Minutes	Default?	Days
Average table bloat in server	Yes	Average table bloat in server	Yes	Yes	1	Yes	30
Connections in idle-in-transaction state	Yes	Connections in idle-in-transaction state	Yes	Yes	1	Yes	30
Connections in idle-in-transaction state, a...	Yes	Connections in idle-in-transaction state, a...	Yes	Yes	1	Yes	30
Connections in idle state	Yes	Connections in idle state	Yes	Yes	1	Yes	30
Database size in server	Yes	Database size in server	Yes	Yes	1	Yes	30
Highest table bloat in server	Yes	Highest table bloat in server	Yes	Yes	1	Yes	30
Largest index by table-size percentage	Yes	Largest index by table-size percentage	Yes	Yes	1	Yes	30
Last AutoVacuum	Yes	Last AutoVacuum	Yes	Yes	1	Yes	30
Last Vacuum	Yes	Last Vacuum	Yes	Yes	1	Yes	30

Use the **Quick Links** toolbar to open dialogs and tabs that will assist you when managing alerts:

- Click **Copy Alerts** to open the **Copy Alert Configuration** dialog and copy an alert definition.
- Click **Alert Templates** to open the **Alert Template** tab, and modify or create an alert template.
- Click **Email Groups** to open the **Email Groups** tab, and modify or create an email group.
- Click **Webhooks** to open the **Webhooks** tab, and create or manage the webhooks endpoints.
- Click **Server Configurations** to open the **Server Configuration** dialog and review or modify server configuration settings.
- Click **Help** to open the PEM online help in a new tab of the PEM web interface.

Use the table in the **Alerts** section of the **Manage Alerts** tab to create new alerts or manage existing alerts.

Alert Templates

An alert template is a prototype that defines the properties of an alert. An alert instructs the server to compare the current state of the monitored object to a threshold (specified in the alert template) to determine if a situation exists that requires administrative attention.

You can use the **Alert Templates** tab to define a custom alert template or view the definitions of existing alert templates. To open the **Alert Templates** tab, select the **Manage Alerts...** menu option from the **Management** menu. When the **Manage Alerts** tab opens, select **Alert Templates** from the **Quick Links** toolbar.

The screenshot shows the 'Alert Templates' tab in the EDB Postgres Enterprise Manager interface. At the top, there are tabs for Properties, SQL, Statistics, Dependencies, Dependents, Monitoring, Manage Alerts, and Alert Templates. The Alert Templates tab is selected. Below the tabs is a section titled 'Description' with a detailed explanation of what an alert template is. The main area contains a table with the following columns: 'Template name', 'Description', 'Target type', 'Applies to server', and 'Check frequency (minutes)'. A note at the bottom of the table area states, 'No data is currently available for custom alert template.'

Use the **Show System Template** drop-down listbox to filter the alert templates that are displayed in the **Alert Templates** table. Use the listbox to select a level of the PEM hierarchy to view all of the templates for the selected level.

Defining a New Alert Template

To define a new alert template, use the **Show System Template** drop-down listbox to select None, and click the **Add** icon (+) located in the upper-right corner of the alert template table. The alert template editor opens.

The screenshot shows the 'General' tab of the Alert Template Editor. It includes fields for 'Template name' (with a warning icon), 'Description' (with a warning icon), 'Target type' (set to 'Server'), 'Applies to server' (set to 'ALL'), 'History retention' (set to 30 days), 'Threshold unit' (empty), 'Auto create?' (set to 'No'), 'Operator' (set to '>'), 'Low' (grayed out), 'Med' (selected), 'High' (grayed out), 'Check frequency (minutes)' (set to 1), and a note about specifying the number of minutes between alert executions.

Use fields on the **General** tab to specify general information about the template:

- Use the **Template name** field to specify a name for the new alert template.
- Use the **Description** field to provide a description of the alert template.
- Use the **Target type** drop-down listbox to select the type of object that will be the focus of the alert.
- Use the **Applies to server** drop-down listbox to specify the server type (EDB Postgres Advanced Server or PostgreSQL) to which the alert will be applied; you can specify a single server type, or ALL.
- Use the **History retention** field to specify the number of days that the result of the alert execution will be stored on the PEM server.
- Use the **Threshold unit** field to specify the unit type of the threshold value.
- Use fields in the **Auto create** box to indicate if PEM should use the template to generate an automatic alert. If enabled, PEM will automatically create an alert when a new server or agent (as specified by the Target type drop-down listbox) is added, and delete that alert when the target object is dropped.

- Move the **Auto create?** slider to **Yes** to indicate that PEM should automatically create alerts based on the template. If you modify an existing alert template, changing the Auto create? slider from No to Yes, PEM will create alerts on the existing agents and servers. Please note that if you change the slider from Yes to No, the default threshold values in existing alerts will be erased, and cannot be recovered.
- Use the **Operator** drop-down listbox to select the operator that PEM will use when evaluating the current system values.

Select a greater-than sign (>) to indicate that the alert should be triggered when the system values are greater than the values entered in the Threshold values fields.

Select a less-than sign (<) to indicate that the alert should be triggered when the system values are less than the values entered in the Threshold values fields.

- Use the threshold fields to specify the values that PEM will compare to the system values to determine if an alert should be raised. Please note that you must specify values for all three thresholds (Low, Medium, and High):

Enter a value that will trigger a low-severity alert in the **Low** field.

Enter a value that will trigger a medium-severity alert in the **Medium** field.

Enter a value that will trigger a high-severity alert in the **High** field.

- Use the **Check frequency** field to specify the default number of minutes between alert executions. This value specifies how often the server will invoke the SQL code specified in the definition and compare the result to the threshold value specified in the template.

The screenshot shows the 'Probe Dependency' tab of a configuration interface. At the top, there are tabs for 'General', 'Probe Dependency' (which is selected), 'Parameters', and 'SQL'. Below the tabs, there is a section titled 'Probes' with a dropdown menu labeled 'Select probe from list'. To the right of the dropdown is a '+' button. Below the dropdown are two input fields: 'Display name' and 'Internal name'.

Use the fields on the **Probe Dependency** tab to specify the names of probes referred to in the SQL query specified on the SQL tab:

- Use the **Probes** drop-down listbox to select from a list of the available probes; highlight a probe name, and click the **Add** button to add the probe to the list of probes used by the alert template. To remove a probe from the selected probes list, highlight the probe name, and click the **Delete** icon.

The screenshot shows the 'Parameters' tab of a configuration interface. At the top, there are tabs for 'General', 'Probe Dependency', 'Parameters' (which is selected), and 'SQL'. Below the tabs, there is a table with three columns: 'Name', 'Data type', and 'Unit'. A '+' button is located at the top right of the table. A note at the bottom left states: 'Add (+) button is disabled in case of system template or value of "Auto create" is Yes in General tab.'

- Use fields on the **Parameters** tab to define the parameters that will be used in the SQL code specified on the **SQL** tab. Click the **Add** icon (+) and:

Use the **Name** field to specify the parameter name.

Use the **Data type** drop-down listbox to specify the type of parameter.

Use the **Unit** field to specify the type of unit specified by the parameter.

- Use the **Code** field on the **SQL** tab to provide the text of the SQL query that the server will invoke when executing the alert. The SQL query will provide the result against which the threshold value is compared; if the alert result deviates from the specified threshold value, an alert will be raised.



Within the query, parameters defined on the **Parameters** tab should be referenced sequentially by the variable `param_x`, where `x` indicates the position of the parameter definition within the parameter list. For example, `param_1` refers to the first parameter in the parameter list, `param_2` refers to the second parameter in the parameter list, and so on.

The query can also include the following pre-defined variables:

Variable Description	Variable Name
agent identifier	'\${agent_id}'
server identifier	'\${server_id}'
database name	'\${database_name}'
schema name	'\${schema_name}'
Table	'\${object_name}'
index	'\${object_name}'
sequence	'\${object_name}'
function name	'\${object_name}'

- Use the **Detailed Information SQL** field to provide a SQL query that will be invoked if the alert is triggered. The result set of the query may be displayed as part of the detailed alert information on the **Alerts** dashboard or **Global Overview** dashboard.

!!! Note If the specified query is dependent on one or more probes from different levels within the PEM hierarchy (server, database, schema, etc.), and a probe becomes disabled, any resulting alerts will be displayed as follows:

- If the alert definition and the probe referenced by the query are from the same level within the PEM hierarchy, the server will display any alerts that reference the alert template on the **Alert Error** table of the **Global Alert** dashboard.
- If the alert definition and the probe referenced by the query are from different levels of the PEM hierarchy, the server will display any triggered alerts that reference the alert template on the **Alert Details** table of the hierarchy on which the alert was defined.

Click the **Save** icon to save the alert template definition and add the template name to the Alert Templates list. After saving a custom alert template, you can use the Alerting dialog to define an alert based on the template.

Modifying or Deleting an Alert Template

To view the definition of an existing template (including PEM pre-defined alert templates), use the **Show System Template** drop-down listbox to select the type of object monitored. When you select the object type, the **Alert Templates** table will display the currently defined alert templates that correspond with that object type.

Highlight a Template Name in the list, and click the Edit icon (at the left end of the row) to review the template definition.

Use the tabs on the **Alert Templates** dialog to view detailed information about the alert template:

- General information is displayed on the **General** tab.
- The names of probes that provide data for the template are listed on the **Probe Dependency** tab.
- The names of any parameters referred to in the SQL code are listed on the **Parameters** tab.
- The SQL code that defines the behavior of the alert is displayed on the **SQL** tab.

To delete an alert template, highlight the template name in the alert templates table, and click the Delete icon. The alert history will persist for the length of time specified in the **History Retention** field in the template definition.

Audit Log Alerting

PEM provides alert templates that allow you to use the **Alerting** dialog to create an alert that will trigger when an **ERROR** or **WARNING** statement is written to a log file for a specific server or agent. To open the **Alerting** dialog, highlight the name of the server or agent in the PEM client Object browser tree control, and select **Alerting...** from the **Management** menu.

To create an alert that will notify you of ERROR or WARNING messages in the log file for a specific server, create an alert that uses one of the following alert templates:

Number of ERRORS in the logfile on server M in last X hours

Number of WARNINGS in the logfile on server M in last X hours

Number of ERRORS or WARNINGS in the logfile on server M in last X hours

To create an alert that will notify you of ERROR or WARNING messages for a specific agent, create an alert that uses one of the following alert templates:

Number of ERRORS in the logfile on agent M in last X hours

Number of WARNINGS in the logfile on agent M in last X hours

Number of ERRORS or WARNINGS in the logfile on agent M in last X hours

Please note that this functionality is supported only on Advanced Server.

Alerts

Use the PEM Client's **Manage Alerts** tab to define, copy, or manage alerts. To open the **Manage Alerts** tab, select **Manage Alerts** from the **Management** menu.

Defining a New Alert

The **Manage Alerts** tab displays a table of alerts that are defined on the object currently selected in the PEM client tree control. You can use the **Alerts** table to modify an existing alert, or to create a new alert.

Manage Alerts							
Name	Auto created?	Template	Enable?	Interval	History retention		
				Default?	Minutes	Default?	Days
Average table bloat in server	Yes	Average table bloat in server	Yes	Yes	1	Yes	30
Connections in idle-in-transaction state	Yes	Connections in idle-in-transaction state	Yes	Yes	1	Yes	30
Connections in idle-in-transaction state, a...	Yes	Connections in idle-in-transaction state, a...	Yes	Yes	1	Yes	30
Connections in idle state	Yes	Connections in idle state	Yes	Yes	1	Yes	30
Database size in server	Yes	Database size in server	Yes	Yes	1	Yes	30
Highest table bloat in server	Yes	Highest table bloat in server	Yes	Yes	1	Yes	30
Largest index by table-size percentage	Yes	Largest index by table-size percentage	Yes	Yes	1	Yes	30
Last AutoVacuum	Yes	Last AutoVacuum	Yes	Yes	1	Yes	30
Last Vacuum	Yes	Last Vacuum	Yes	Yes	1	Yes	30

To open the alert editor and create a new alert, click the Add icon (+) in the upper-right corner of the table. The editor opens as shown below.

The screenshot shows the 'General' tab of the alert editor. It includes fields for Name, Description, Template, Enable?, Interval, History retention, Threshold values, Auto created?, and Parameter Options.

- Name:** A red warning icon is displayed next to the input field.
- Description:** A large text area for entering alert details.
- Template:** A dropdown menu labeled "Select from the list". A tooltip explains that a template uses metrics to generate a value compared to user-specified boundaries.
- Enable?:** A switch set to "Yes". A tooltip indicates it enables the alert.
- Interval:** A section with "Default?" switch (set to "Yes") and a "Minutes" selector set to "1". A tooltip says it specifies how often the alert should confirm conditions.
- History retention:** A section with "Default?" switch (set to "Yes") and a "Days" selector set to "30". A tooltip says it specifies the number of days PEM stores data.
- Threshold values:** A section for defining triggering criteria with Operator (">> Low), Unit, Medium, and High fields.
- Auto created?:** A switch set to "No".
- Parameter Options:** A table with columns "Name" and "Value". An entry for "server" has "Postgres Enterprise Manager Server" as its value.

Use the fields on the **General** tab to provide information about the alert:

- Enter the name of the alert in the **Name** field.
- Use the drop-down listbox in the **Template** field to select a template for the alert. An alert template is a function that uses one (or more) metrics or parameters to generate a value to which PEM compares user-specified alert boundaries. If the value returned by the template function evaluates to a value that is within the boundary of a user-defined alert (as specified by the Operator and Threshold values fields), PEM raises an alert, adds a notice to the Alerts overview display, and performs any actions specified on the template.
- Use the **Enable?** switch to specify if the alert is enabled (Yes) or disabled (No).
- Use the controls in the **Interval** box to specify how often the alert should confirm if the alert conditions are satisfied. Use the Minutes selector to specify an interval value. Use the Default switch to set or reset the Minutes value to the default (recommended) value for the selected template.
- Use controls in the **History retention** box to specify the number of days that PEM will store data collected by the alert. Use the Days selector to specify the number of days that the data will be stored. Use the Default switch to set or reset the Days value to the default value (30 days).

- Use controls in the **Threshold values** box to define the triggering criteria for the alert. When the value specified in the Threshold Values fields evaluates to greater-than or less-than the system value (as specified with the Operator), PEM will raise a Low, Medium or High level alert:
- Use the **Operator** drop-down listbox to select the operator that PEM will use when evaluating the current system values:
 - Select a greater-than sign (>) to indicate that the alert should be triggered when the system values are greater than the values entered in the Threshold values fields.
 - Select a less-than sign (<) to indicate that the alert should be triggered when the system values are less than the values entered in the Threshold values fields.
- Use the **threshold** fields to specify the values that PEM will compare to the system values to determine if an alert should be raised. Please note that you must specify values for all three thresholds (Low, Medium, and High):
 - Enter a value that will trigger a low-severity alert in the **Low** field.
 - Enter a value that will trigger a medium-severity alert in the **Medium** field.
 - Enter a value that will trigger a high-severity alert in the **High** field.

The **Parameter Options** table contains a list of parameters that are required by the selected template; the table displays both pre-defined parameters, and parameters for which you must specify a value. Please note that you must specify a value for any parameter that displays a prompt in the Value column.

PEM can send a notification or execute a script if an alert is triggered, or if an alert is cleared. Use the **Notification** tab to specify how PEM will behave if an alert is raised.

Alert types	
All alerts?	<input type="checkbox"/> No <Default>
Low alerts?	<input type="checkbox"/> No <Default>
Medium alerts?	<input type="checkbox"/> No <Default>
High alerts?	<input type="checkbox"/> No <Default>

Use the fields in the **Email notification** box to specify the Email group that will receive an email notification if the alert is triggered at the specified level. Use the **Email Groups** tab to create an email group that contains the address of the user or users that will be notified when an alert is triggered. To access the **Email Groups** tab, click the **Email Groups** icon located in the **Quick Links** menu of the **Manage Alerts** tab.

- To instruct PEM to send an email when a specific alert level is reached, set the slider next to an alert level to Yes, and use the drop-down listbox to select the pre-defined user or group that will be notified.

Please note that you must configure the PEM Server to use an SMTP server to deliver email before PEM can send email notifications.

To configure notification for an alert use the below fields to specify one or multiple endpoints if the alert is triggered at the specified level. Use the dropdown to select a pre-defined endpoints that will be sent a notification.

Enable? Yes No

Override default configuration? Yes No

Alert types

- Low alerts? Yes No
- Medium alerts? Yes No
- High alerts? Yes No
- Cleared alerts? Yes No

Use the **Webhook notification** box to specify one or multiple endpoints if the alert is triggered at the specified level. Use the **webhooks tab** to create a webhook endpoint that will receive the notifications when an alert is triggered. To access the **Webhooks** tab, click the **Webhooks** icon located in the **Quick Links** menu of the **Manage Alerts** tab.

- Set the **Enable?** to **Yes** to send the alert notifications to the webhook endpoint.
- Set the **Override default configuration?** to **Yes** to set the customized alert levels as per the requirement. Once it is set to **Yes** all the alert levels are enabled to configure.
- Use the dropdown to select a pre-defined endpoint that will be sent a notification for **Low alerts?**, **Medium alerts?**, **High alerts?** and **Cleared alerts?**.

Use the trap notification options to configure trap notifications for this alert Note that you must configure PEM server to send notifications to an SNMP trap/notification receiver before notifications can be sent.

Trap notification

Send trap?	<input type="checkbox"/> No <input checked="" type="checkbox"/> Yes	SNMP version	v3 <input style="width: 20px; height: 15px; border: 1px solid black; border-radius: 5px; padding: 2px 5px;" type="button" value="v1"/>	Low alert?	<input type="checkbox"/> No <input checked="" type="checkbox"/> Yes
Medium alert?	<input type="checkbox"/> No <input checked="" type="checkbox"/> Yes	High alert?	<input type="checkbox"/> No <input checked="" type="checkbox"/> Yes		

Use the **Trap notification** options to configure trap notifications for this alert:

- Set the **Send trap** slider to **Yes** to send SNMP trap notifications when the state of this alert changes.
- Set the **SNMP Ver** to **v1**, **v2**, or **v3** to identify the SNMP version.
- Use the **Low alert**, **Med alert** and **High alert** sliders to select the level(s) of alert that will trigger the trap. For example, if you set the slider next to High alert to Yes, PEM will send a notification when an alert with a high severity level is triggered.

Please note that you must configure the PEM Server to send notifications to an SNMP trap/notification receiver before notifications can be sent. For sending SNMP v3 traps, pemAgent will use 'User Security Model(USM)' which is in charge of authenticating, encrypting, and decrypting SNMP packets.

Also note while sending SNMP v3 traps, agent will create snmp_boot_counter file. This file will get created in location mentioned by batch_script_dir parameter in agent.cfg, if this parameter is not configured or if directory is not accessible due to authentication restrictions then in operating systems temporary directory, if that is also not possible then in user's home directory.

General Notification Script execution

Email Webhook SNMP Nagios

Set "Submit passive service check result to Nagios" to "Yes" to instruct the PEM server to notify Nagios when the alert is triggered or cleared.

Nagios notification

Submit passive service check result to Nagios?

Use the field in the **Nagios notification** box to instruct the PEM server to notify Nagios network-alerting software when the alert is triggered or cleared. For more details see [Using PEM with Nagios](#)

- Set the **Submit passive service check result to Nagios** switch to **Yes** to instruct the PEM server to notify Nagios when the alert is triggered or cleared.

General Notification Script execution

Script execution

Execute script? Execute on alert cleared?

Execute script on PEM Server Monitored Server

Code

Use the fields in the Script execution box to (optionally) define a script that will be executed if an alert is triggered, and to specify details about the script execution.

- Set the Execute script slider to Yes to instruct PEM to execute the provided script if an alert is triggered.
- Set the Execute on alert cleared slider to Yes to instruct PEM to execute the provided script when the situation that triggered the alert has been resolved.
- Use the selector to indicate if the script should execute on the PEM Server or the Monitored Server.
- Provide the script that PEM should execute in the Code field. You can provide a batch/shell script, or SQL code. Within the script you can use the placeholders to replace the following:
 - %AlertID% - the id of the triggered alert.
 - %AlertName% - the name of the triggered alert.
 - %ObjectName% - the name of the server or agent on which the alert was triggered.
 - %ObjectType% - the type on which alert was generated.
 - %ThresholdValue% - the threshold value reached by the metric when the alert triggered.
 - %CurrentValue% - the current value of the metric that triggered the alert.
 - %CurrentState% - the current state of the alert.
 - %OldState% - the previous state of the alert.
 - %AlertRaisedTime% - the time that the alert was raised, or the most recent time that the alert state was changed.
 - %AgentID% - the id of the agent by which alert was generated.
 - %AgentName% - the name of the agent by which alert was generated.
 - %ServerID% - the id of the server on which alert was generated.
 - %ServerName% - the name of the server on which alert was generated.
 - %ServerIP% - the ip or address of the server on which alert was generated.
 - %ServerPort% - the port of the server on which alert was generated.
 - %DatabaseName% - the name of the database on which alert was generated.
 - %SchemaName% - the name of the schema on which alert was generated.
 - %PackageName% - the name of the package on which alert was generated.
 - %DatabaseObjectName% - the name of the database object name like table name, function name etc on which alert was generated.
 - %Parameters% - the list of custom parameters used to generate the alert.

Use the fields in the **Script execution** box to (optionally) define a script that will be executed if an alert is triggered, and to specify details about the script execution.

- Set the **Execute script** slider to **Yes** to instruct PEM to execute the provided script if an alert is triggered.
- Set the **Execute on alert cleared** slider to **Yes** to instruct PEM to execute the provided script when the situation that triggered the alert has been resolved.
- Use the radio buttons next to **Execute script on** to indicate that the script should execute on the PEM Server or the Monitored Server.
- Provide the script that PEM should execute in the **Code** field. You can provide a batch/shell script, or SQL code. Within the script, you can use placeholders for the following:

`%AlertName%` - this placeholder will be replaced with the name of the triggered alert.

`%ObjectName%` - this placeholder will be replaced with the name of the server or agent on which the alert was triggered.

`%ThresholdValue%` - this placeholder will be replaced with the threshold value reached by the metric when the alert triggered.

`%CurrentValue%` - this placeholder will be replaced with the current value of the metric that triggered the alert.

`%CurrentState%` - this placeholder will be replaced with the current state of the alert.

`%OldState%` - this placeholder will be replaced with the previous state of the alert.

`%AlertRaisedTime%` - this placeholder will be replaced with the time that the alert was raised, or the most recent time that the alert state was changed.

To invoke a script on a Linux system, you must modify the entry for the `batch_script_user` parameter of the `agent.cfg` file and specify the user that should be used to run the script. You can either specify a non-root user or root for this parameter. If you do not specify a user, or the specified user does not exist, then the script will not be executed. Restart the agent after modifying the file.

To invoke a script on a Windows system, set the registry entry for `AllowBatchJobSteps` to true and restart the PEM agent. PEM registry entries are located in
`HKEY_LOCAL_MACHINE\Software\Wow6432Node\EnterpriseDB\PEM\agent`.

When you have defined the alert attributes, click the edit icon to close the alert definition editor, and then the save icon (in the upper-right corner of the `Alerts` table). To discard your changes, click the refresh icon; a popup will ask you to confirm that you wish to discard the changes.

!!! Note Suppose you need to use the alert configuration placeholder values in an external script. You can do it either by passing them as the command-line arguments or exporting them as environment variables. Please note that the external script must have proper execution permissions.

- You can run the script with any of the placeholders as command-line argument.

For eg:

```
#!/bin/bash

bash <path_to_script>/script.sh "%AlertName% %AlertLevel% %AlertDetails%"
```

- You can define the environment variables for any of the placeholders and then use those environment variables in the script.

For eg:

```
#!/bin/bash

export AlertName=%AlertName%
export AlertState=%AlertState%

bash <path_to_script>/script.sh
```

Modifying an Alert

Use the `Alerts` table to manage an existing alert or create a new alert. Highlight an object in the PEM client tree

control to view the alerts that monitor that object.

	Name	Auto created?	Template	Enable?	Interval		History retention	
					Default?	Minutes	Default?	Days
<input checked="" type="checkbox"/>	Audit config mismatch	<input checked="" type="checkbox"/>	Audit config mismatch	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Average table bloat in server	<input checked="" type="checkbox"/>	Average table bloat in server	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Connections in idle-in-transaction state	<input checked="" type="checkbox"/>	Connections in idle-in-transaction state	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Connections in idle-in-transaction state, as a percentage	<input checked="" type="checkbox"/>	Connections in idle-in-transaction state, as a percentage	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Connections in idle state	<input checked="" type="checkbox"/>	Connections in idle state	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Database size in server	<input checked="" type="checkbox"/>	Database size in server	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Highest table bloat in server	<input checked="" type="checkbox"/>	Highest table bloat in server	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Largest index by table-size percentage	<input checked="" type="checkbox"/>	Largest index by table-size percentage	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Last AutoVacuum	<input checked="" type="checkbox"/>	Last AutoVacuum	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Last Vacuum	<input checked="" type="checkbox"/>	Last Vacuum	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Log config mismatch	<input checked="" type="checkbox"/>	Log config mismatch	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Number of prepared transactions	<input checked="" type="checkbox"/>	Number of prepared transactions	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Number of WAL archives pending	<input checked="" type="checkbox"/>	Number of WAL archives pending	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Number of WAL files	<input checked="" type="checkbox"/>	Number of WAL files	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Server Down	<input checked="" type="checkbox"/>	Server Down	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Table_Size_Alerts	<input type="checkbox"/>	Table size in server	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Table size in server	<input checked="" type="checkbox"/>	Table size in server	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Total connections	<input checked="" type="checkbox"/>	Total connections	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Total connections as percentage of max_connections	<input checked="" type="checkbox"/>	Total connections as percentage of max_connections	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Total table bloat in server	<input checked="" type="checkbox"/>	Total table bloat in server	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30

You can modify some properties of an alert in the **Alerts** table:

- The **Alert name** column displays the name of the alert; to change the alert name, simply replace the name in the table, and click the save icon.
- The **Alert template** column displays the name of the alert template that specifies properties used by the alert. You can use the drop-down listbox to change the alert template associated with an alert.
- Use the **Alert enable?** switch to specify if an alert is enabled (Yes) or disabled (No).
- Use the **Interval** column to specify how often PEM should check to see if the alert conditions are satisfied. Set the **Default** switch to **No** and specify an alternate value (in Minutes), or return the Default switch to **Yes** to reset the value to its default setting. By default, PEM will check the status of each alert once every minute.
- Use the **History retention** field to specify the number of days that PEM will store data collected by the alert. Set the **Default** switch to **No** and specify an alternate value (in Days), or return the Default switch to **Yes** to reset the value to its default setting. By default, PEM will recommend storing historical data for 30 days.

After modifying an alert, click the save icon (located in the upper-right corner of the table) to make your changes persistent.

Click the edit icon to the left of an alert name to open an editor that provides access to the complete alert definition to modify other alert attributes.

Alert Details (Auto-refresh paused whilst rows are expanded: 																																		
Ack'd	Alert Type	Name	Value	Agent	Server	Database	Schema	Package	Object	Alerting Since																								
<input checked="" type="checkbox"/>	<input type="checkbox"/>	Table size in server	1.9814453125 GB	Postgres Enterprise Manager Server						2020-02-20 11:29:45																								
General		Parameters																																
<table border="1"> <thead> <tr> <th>Table name</th> <th>Schema name</th> <th>Database name</th> <th>Total table size(MB)</th> </tr> </thead> <tbody> <tr><td>table_statistics</td><td>pemhistory</td><td>pem</td><td>1087</td></tr> <tr><td>server_logs</td><td>pemdata</td><td>pem</td><td>263</td></tr> <tr><td>index_statistics</td><td>pemhistory</td><td>pem</td><td>237</td></tr> <tr><td>session_info</td><td>pemhistory</td><td>pem</td><td>137</td></tr> <tr><td>lock_info</td><td>pemhistory</td><td>pem</td><td>88</td></tr> </tbody> </table>											Table name	Schema name	Database name	Total table size(MB)	table_statistics	pemhistory	pem	1087	server_logs	pemdata	pem	263	index_statistics	pemhistory	pem	237	session_info	pemhistory	pem	137	lock_info	pemhistory	pem	88
Table name	Schema name	Database name	Total table size(MB)																															
table_statistics	pemhistory	pem	1087																															
server_logs	pemdata	pem	263																															
index_statistics	pemhistory	pem	237																															
session_info	pemhistory	pem	137																															
lock_info	pemhistory	pem	88																															

Use fields on the **Alert details** dialog to modify the definition of the selected alert. When you've finished modifying the alert definition, click **Save** to preserve your changes, or **Cancel** to exit the dialog without saving any changes.

Deleting an Alert

To mark an alert for deletion, highlight the alert name in the Alerts table and click the delete icon to the left of the name; the alert will remain in the list, but in red strike-through font.

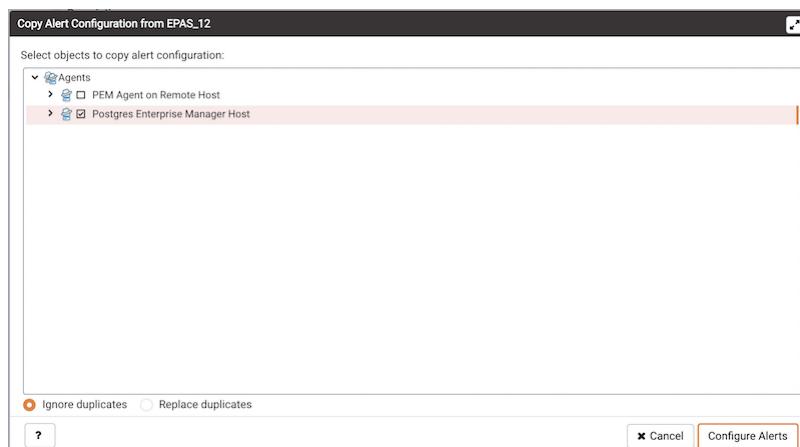
	Name	Auto created?	Template	Enable?	Interval		History retention	
					Default?	Minutes	Default?	Days
<input checked="" type="checkbox"/>	Audit config mismatch	<input checked="" type="checkbox"/> Yes	Audit config mismatch	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30
<input checked="" type="checkbox"/>	Average table bloat in server	<input checked="" type="checkbox"/> Yes	Average table bloat in server	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30
<input checked="" type="checkbox"/>	Connections in idle-in-transaction state	<input checked="" type="checkbox"/> Yes	Connections in idle-in-transaction state	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30
<input checked="" type="checkbox"/>	Connections in idle-in-transaction state, as a percentage	<input checked="" type="checkbox"/> Yes	Connections in idle-in-transaction state, as a percentage	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30
<input checked="" type="checkbox"/>	Connections in idle state	<input checked="" type="checkbox"/> Yes	Connections in idle state	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30
<input checked="" type="checkbox"/>	Database size in server	<input checked="" type="checkbox"/> Yes	Database size in server	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30
<input checked="" type="checkbox"/>	Highest table bloat in server	<input checked="" type="checkbox"/> Yes	Highest table bloat in server	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30
<input checked="" type="checkbox"/>	Largest index by table-size percentage	<input checked="" type="checkbox"/> Yes	Largest index by table-size percentage	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30
<input checked="" type="checkbox"/>	Last AutoVacuum	<input checked="" type="checkbox"/> Yes	Last AutoVacuum	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30
<input checked="" type="checkbox"/>	Last Vacuum	<input checked="" type="checkbox"/> Yes	Last Vacuum	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30
<input checked="" type="checkbox"/>	Log config mismatch	<input checked="" type="checkbox"/> Yes	Log config mismatch	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30
<input checked="" type="checkbox"/>	Number of prepared transactions	<input checked="" type="checkbox"/> Yes	Number of prepared transactions	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30
<input checked="" type="checkbox"/>	Number of WAL archives pending	<input checked="" type="checkbox"/> Yes	Number of WAL archives pending	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30
<input checked="" type="checkbox"/>	Number of WAL files	<input checked="" type="checkbox"/> Yes	Number of WAL files	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30
<input checked="" type="checkbox"/>	Server Down	<input checked="" type="checkbox"/> Yes	Server Down	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30
<input checked="" type="checkbox"/>	Table-Size-Alerts	<input checked="" type="checkbox"/> No	Table-size-in-server	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30
<input checked="" type="checkbox"/>	Table size in server	<input checked="" type="checkbox"/> Yes	Table size in server	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30
<input checked="" type="checkbox"/>	Total connections	<input checked="" type="checkbox"/> Yes	Total connections	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30
<input checked="" type="checkbox"/>	Total connections as percentage of max_connections	<input checked="" type="checkbox"/> Yes	Total connections as percentage of max_connections	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30
<input checked="" type="checkbox"/>	Total table bloat in server	<input checked="" type="checkbox"/> Yes	Total table bloat in server	<input checked="" type="checkbox"/> Yes	<input checked="" type="checkbox"/> Yes	1	<input checked="" type="checkbox"/> Yes	30

The delete icon acts as a toggle; you can undo the deletion by clicking the delete icon a second time; when you click the **Save** icon, the alert definition will be permanently deleted.

Copying an Alert

To speed up the deployment of alerts in the PEM system, you can copy alert definitions from one object to one or more target objects.

To copy alerts from an object, highlight the object in the PEM client tree control on the main PEM window, and select the **Copy Alerts...** option from the **Management** menu. When the **Manage Alerts** tab opens, click the **Copy Alerts** icon (located in the **Quick Links** toolbar) to open the **Copy Alert Configuration** dialog.



The **Copy Alert Configuration** dialog copies all alerts from the object highlighted in the PEM client tree control to the object or objects selected on the dialog. Expand the tree control to select a node or nodes to specify the target object(s). The tree control displays a red warning indicator next to the source object.

To copy alerts to multiple objects at once, select a parent node of the target(s). For example, to copy the alerts from one table to all tables in a schema, you can simply select the checkbox next to the schema. PEM will only copy alerts to targets that are of the same type as the source object.

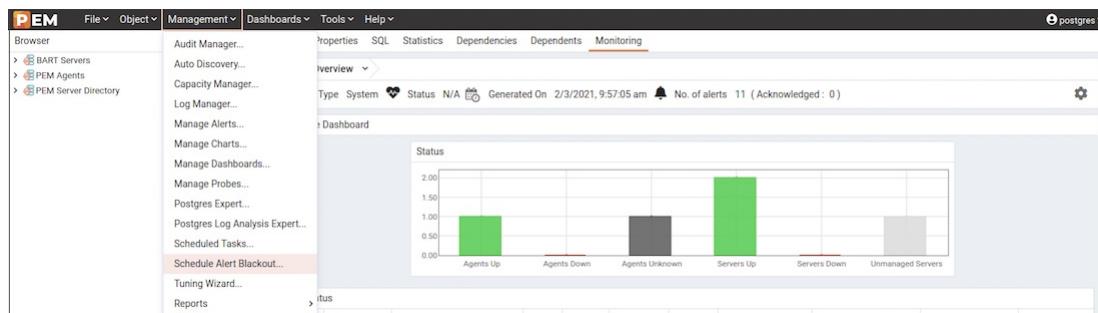
Check the **Ignore duplicates** radio button to prevent PEM from updating any existing alerts on the target objects with the same name as those being copied. Use the **Replace duplicates** option to replace existing alerts with alerts of the same name from the source object.

Click the **Configure Alerts** button to proceed to copy the alerts from the source object to all objects of the same type in, or under those objects selected on the **Copy Alert Configuration** dialog.

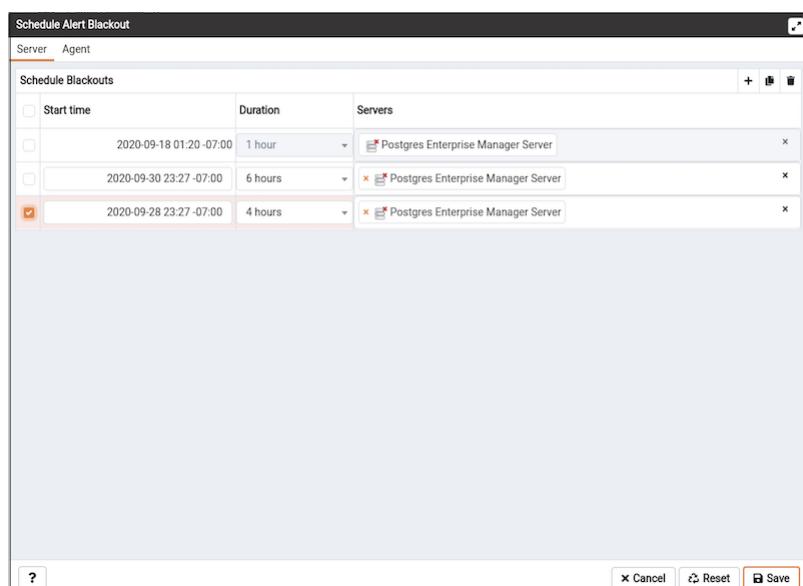
Schedule Alert Blackout

You can use the **Schedule Alert Blackout** option on the **Management** menu to schedule an alert blackout for your Postgres servers and PEM Agents during maintenance. Alerts will not be raised during a defined blackout period.

To schedule an alert blackout, click on the **Management** menu and select **Schedule Alert Blackout**.



When the **Schedule Alert Blackout** dialog opens, use the tabs on the dialog to define the blackout period for servers and agents. Open the **Server** tab and click the Add icon (+) at the top right corner to add new row.



Use the fields on the **Server** tab to provide information about an alert blackout period:

- Use the **Start time** field to provide the date and time to start the alert blackout.

- Use the **Duration** field to provide the interval for which you want to blackout the alerts.
- Use the **Servers** field to provide the server name for which you want to blackout the alerts. You can also select multiple servers to blackout the alerts for those servers simultaneously.

After providing details, you can save the details by clicking on **Save** button on the right bottom corner of the dialog. Once saved, it cannot be edited. The alerts will not be displayed on the **Alerts** dashboard for the scheduled interval of that particular server.

You can also schedule a blackout period for PEM Agents via the **Agent** tab on the dialog. Open the **Agent** tab and click the Add icon (+) at the top right corner to add new row.

Start time	Duration	Agents
2020-09-18 03:33 -07:00	1 hour	Postgres Enterprise Manager Host
2020-09-25 03:33 -07:00	4 hours	Agent2
2020-09-30 03:33 -07:00	23 hours	Postgres Enterprise Manager Host
2020-09-21 23:23 -07:00	6 hours	Agent2

Buttons at the bottom: ? Cancel Reset Save

Use the fields on the **Agent** tab to provide the information about about an alert blackout period:

- Use the **Start time** field to provide the date and time to start the alert blackout.
- Use the **Duration** field to provide the interval for which you want to blackout the alerts.
- Use the **Agents** field to provide the Agent name for which you want to blackout the alerts. All server level alerts, for the servers bound to that particular agent will blackout.

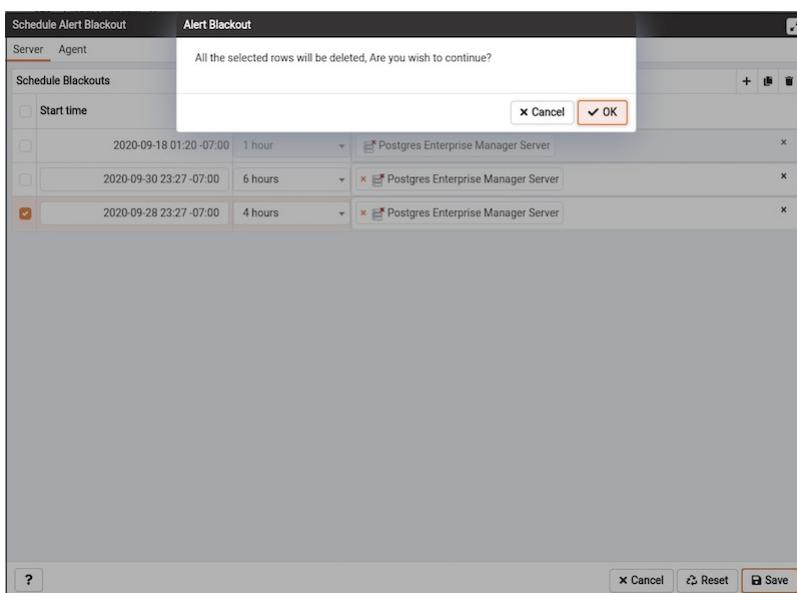
After providing details, you can save the details by clicking on **Save** button on the right bottom corner of the dialog. Once saved, it cannot be edited. The alerts will not be displayed on the **Alert** dashboard for the scheduled interval for that PEM agent.

You can use **Clone** button from the top right corner of dialog, to clone the scheduling of alert blackout. Select the servers or agents you want to clone and then click on **Clone** button to create the cloned copy of all the selected servers or agents. You can edit newly created schedules as needed, and then click **Save**.

You can use **Delete** button from the top right corner of dialog to remove a scheduled alert blackout. Select the servers or agents and then click on highlighted **Delete** button in the right top corner to remove the scheduled alerts associated with that server or agent.



Select a server for which you wish to delete the scheduled alert blackout and then click on the **Delete** button. The server will ask for confirmation before deleting that row.



You can use the **Reset** button to reset the details on the **Alert Blackout** dialog to the default settings. Please note that all saved blackouts will remain unaffected after resetting the current dialog values.

Notifications

PEM can send a notification or execute a script if an alert is triggered, or if an alert is cleared. The Notifications can be send via following options:

- SMTP
- Webhooks
- SNMP
- Nagios

Use the **Notification** tab to specify how PEM will behave if an alert is raised.

SMTP

Please note that you must configure the PEM Server to use an SMTP server to deliver email before PEM can send email notifications.

Creating an Email Group

Postgres Enterprise Manager monitors your system for conditions that require user attention. You can use an email group to specify the email addresses of users that the server will notify if current values deviate from threshold values specified in an alert definition. An email group has the flexibility to notify multiple users, or target specific users during user-defined time periods.

Use the **Email Groups** tab to configure groups of SMTP email recipients. To access the **Email Groups** tab, select **Manage Alerts...** from the PEM client's **Management** menu; when the **Manage Alerts** tab opens, select **Email Groups** from the Quick Links toolbar.

Email Groups	
	Group name
<input checked="" type="checkbox"/>	acctg_admin
<input checked="" type="checkbox"/>	hr_resources
<input checked="" type="checkbox"/>	sales
<input checked="" type="checkbox"/>	<Default>

The **Email Groups** tab displays a list of the currently defined email groups. Highlight a group name and click the Edit icon (at the far left end of the row) to modify an existing group.

To define a new email group, click the Add icon (+) in the upper-right corner of the **Email Groups** table. The **Email Group** definition dialog opens.

The screenshot shows the 'Email Groups' dialog. At the top, there's a table with two rows: one for the 'Group Name' and another for the 'Email Group'. The 'Group Name' row contains the value 'sales'. Below this is a detailed description of 'Email group options' with various configuration fields like 'To addresses', 'From address', 'From time', 'To time', and several 'Options' fields for 'Reply to addresses', 'CC addresses', 'BCC addresses', 'Subject prefix', 'From time', and 'To time'.

Use the **Email Group** dialog to define an email group and its members:

- Provide a name for the email group in the **Group Name** field.

Each row within the email group definition will associate a unique set of email addresses with a specific time period. When an alert is triggered, the server will evaluate the times specified in each row and send the message to those group members whose definitions are associated with the time that the alert triggered.

Click the Add icon (+) in the group members table to open the **Options** tab, and add the member addresses that will receive notifications for the time period specified:

- Enter a comma-delimited list of recipient addresses in the **Reply to Addresses** field.
- Enter a comma-delimited list of addresses that will receive a copy of the email in the **CC Addresses** field.
- Enter a comma-delimited list of addresses that will receive a copy of the email (without the knowledge of other recipients) in the **Bcc Addresses** field.
- Enter the email address that messages to this group should be sent from in the **From Address** field.
- Use the **Subject prefix** field to provide a message that will be added to the start of each subject line when a notification is sent.
- Use the **From Time** and **To Time** time selectors to specify the time range for notifications to the group member(s) that are identified on this row. Provide the From Time and To Time values in the locale of the PEM client host, and the PEM server will translate the time into other time zones as required.

When you've identified the member or members that will receive an email during a specific time period, click the Add icon to add a row to the table, and specify another time period and the email addresses that will be notified during those hours. When you've finished defining the email group, click the Save icon.

Deleting an Email Group

To delete an email group, highlight the name of the group in the **Email Group** table and click the Delete icon (located to the left of the group name).

Email Groups	
	Group Name
	sales
	<Default>

The group name will be displayed in the **Email Group** table in red; click the Save icon to make the change persistent and remove the group from the table.

After creating the email group, you can use the **Manage Alerts** tab to set up the **Notification** details for an alert that will direct notifications to the group.

Webhook

Please note that you must configure the PEM Server to use webhooks to receive notification of alert events on threshold value violations in your configured applications.

Creating Webhook

Postgres Enterprise Manager monitors your system for conditions that require user attention. You can use a webhook to create the endpoints that will receive a notification if current values deviate from threshold values specified in an alert definition. PEM sends a notification to multiple webhook endpoints, or to specific target webhook endpoints based on the events triggered.

Use the **Webhooks** tab to configure endpoint recipients. To access the **Webhooks** tab, select **Manage Alerts...** from the PEM client's **Management** menu; when the **Manage Alerts** tab opens, select **Webhooks** from the **Quick Links** toolbar.

Dashboard	Properties	SQL	Statistics	Dependencies	Dependents	Monitoring	Manage Alerts	Webhooks	x																								
Description																																	
Webhooks: Webhooks are user defined HTTP callbacks which are triggered by specific events. You can use webhooks to receive notifications of alert events on threshold values violations, in your configured applications.																																	
Webhooks																																	
<table border="1"> <thead> <tr> <th>Name</th> <th>URL</th> <th>Enable?</th> <th colspan="4">Alert notifications</th> </tr> <tr> <th></th> <th></th> <th></th> <th>High alerts?</th> <th>Medium alerts?</th> <th>Low alerts?</th> <th>Cleared alerts?</th> </tr> </thead> <tbody> <tr> <td colspan="10">No data</td></tr> </tbody> </table>										Name	URL	Enable?	Alert notifications							High alerts?	Medium alerts?	Low alerts?	Cleared alerts?	No data									
Name	URL	Enable?	Alert notifications																														
			High alerts?	Medium alerts?	Low alerts?	Cleared alerts?																											
No data																																	

The **Webhooks** tab displays a list of the currently defined recipient applications as endpoints. Highlight an endpoint and click the edit icon (at the far left end of the row) to modify an existing endpoint.

To define a new webhook, click the **Add** icon (+) in the upper-right corner of the table.

The screenshot shows the 'Webhooks' section of the EDB Postgres Enterprise Manager. At the top, there's a table for 'Alert notifications' with columns for Name, URL, Enable?, High alerts?, Medium alerts?, Low alerts?, and Cleared alerts?. Below this is a 'General' tab with fields for Name (empty), URL (empty), Request method (set to POST), and Enable? (set to Yes). A note at the bottom states: 'Note that to enable TSL certification, configuration has to be done at agent side. For more information [Click here](#)'. A red error message box at the bottom left says: '⚠ Please specify webhook name.'

Use the **General** tab to define the basic details of the webhook:

- Provide a name for the webhook in the **Name** field.
- Specify a webhook URL where all the notifications will be delivered in the **URL** field.
- Set the request method type used to make the call in the **Request Method** field i.e. **POST** or **PUT**.
- By default **webhooks** will be enabled; to disable a webhook set **Enable?** to **No**.

!!! Note The above **Enable?** setting will work only if **enable_webhook** parameter is set to true in **agent.cfg** file. By default, **enable_webhook** parameter is set to true only for the Agent running on the PEM Server Host. For all other Agents running on other hosts, it needs to be set to true manually.

Defining a Webhook SSL configurations

You can define the Webhook SSL parameters in the respective agent configuration file or registry in windows. You can find the list of Webhook SSL parameters **PEM Agent Configuration Parameters** section. If you add or remove any of the agent configuration parameters, you must restart the agent to apply them.

- On 32 bit Windows systems, PEM registry entries for Webhooks are located in **HKEY_LOCAL_MACHINE\\Software\\EnterpriseDB\\PEM\\agent\\WEBHOOK**
- On 64 bit Windows systems, PEM registry entries for Webhooks are located in **HKEY_LOCAL_MACHINE\\Software\\Wow6432Node\\EnterpriseDB\\PEM\\agent\\WEBHOOK**
- On Linux systems, PEM configuration options for Webhooks are stored in the **agent.cfg** file, located (by default) in **/usr/edb/pem/agent/etc**

```
[WEBHOOK/Django]
webhook_ssl_key=<webhook_client_ssl_key_path>
webhook_ssl_crt=<webhook_client_ssl_certificate_path>
webhook_ssl_ca_crt=<webhook_server_ca_certificate_path>
webhook_ssl_crl=<crl_file_path_to_validate_webhook_server>
allow_insecure_webhooks=<true|false>
```

Name	Type	Data
(Default)	REG_SZ	(value not set)
AllowInsecureWebhooks	REG_SZ	true
WebhookSSLCAcrt	REG_SZ	server.ca.pem
WebhookSSLCrt	REG_SZ	client.crt
WebhookSSLKey	REG_SZ	client.key

The screenshot shows the 'Webhooks' section of the EDB Postgres Enterprise Manager. A specific webhook named 'Django' is selected. The 'HTTP Headers' tab is active, showing a table with one row: 'Content-type' set to 'application/json'. Other tabs like 'General', 'Payload', and 'Notifications' are also visible.

Use the **HTTP Headers** tab to define the header parameters to pass while calling the webhook endpoints:

- All the values will be specified as a key and value pair.
- Specify a key parameter in the **Key** field and a value in the **Value** field.
- To add multiple **HTTP Headers**, click the **Add** icon (+) in the upper-right corner of the **HTTP Headers** table.
- To delete the **HTTP Headers**, click on **Delete** icon to the left of the **Key**; the alert will remain in the list, but in strike-through font. Click the **Save** button to reflect the changes.
- To edit the **HTTP Headers**, click on the **Edit** icon to the left of **Key**.

The screenshot shows the 'Webhooks' section with the 'Payload' tab selected. It displays a JSON template with many placeholders, such as %AlertID%, %AlertName%, %ObjectNames%, etc. Below the template, a 'Placeholder' section provides a detailed list of these variables and their descriptions.

Placeholder	Description
%AlertID%	the id of the triggered alert.
%AlertName%	the name of the triggered alert.
%ObjectName%	the name of the server or agent on which the alert was triggered.
%ObjectType%	the type on which alert was generated.
%ThresholdValue%	the threshold value reached by the metric when the alert triggered.
%CurrentValue%	the current value of the metric that triggered the alert.
%CurrentState%	the current state of the alert.
%OldState%	the previous state of the alert.
%AlertRaisedTime%	the time that the alert was raised, or the most recent time that the alert state was changed.
%AgentID%	the id of the agent by which alert was generated.
%AgentName%	the name of the agent by which alert was generated.
%ServerID%	the id of the server on which alert was generated.
%ServerName%	the name of the server on which alert was generated.
%ObjectID%	the id of the object on which alert was generated.
%ObjectName%	the name of the object on which alert was generated.
%ObjectTypeName%	the type of the object on which alert was generated.
%MetricName%	the name of the metric on which alert was generated.
%MetricType%	the type of the metric on which alert was generated.
%MetricValue%	the value of the metric on which alert was generated.
%SchemaName%	the name of the schema on which alert was generated.
%PackageName%	the name of the package on which alert was generated.
%FunctionName%	the name of the function or procedure on which alert was generated.
%VariableName%	the list of custom parameters used to generate the alert.
%AlertInfo%	the detailed database object level information of the alert.

Use the **Payload** tab to define the JSON data to be sent to the endpoint when an alert is triggered:

- **Type** specifies data to be sent in format type (i.e. JSON).
- Use **Template** to configure JSON data sent to endpoints. Within the **Template**, you can use placeholders for the following:
 - **%AlertID%** - the id of the triggered alert.
 - **%AlertName%** - the name of the triggered alert.
 - **%ObjectName%** - the name of the server or agent on which the alert was triggered.
 - **%ObjectType%** - the type on which alert was generated.
 - **%ThresholdValue%** - the threshold value reached by the metric when the alert triggered.
 - **%CurrentValue%** - the current value of the metric that triggered the alert.
 - **%CurrentState%** - the current state of the alert.
 - **%OldState%** - the previous state of the alert.
 - **%AlertRaisedTime%** - the time that the alert was raised, or the most recent time that the alert state was changed.
 - **%AgentID%** - the id of the agent by which alert was generated.
 - **%AgentName%** - the name of the agent by which alert was generated.
 - **%ServerID%** - the id of the server on which alert was generated.
 - **%ServerName%** - the name of the server on which alert was generated.
 - **%ObjectID%** - the id of the object on which alert was generated.
 - **%ObjectName%** - the name of the object on which alert was generated.
 - **%ObjectTypeName%** - the type of the object on which alert was generated.
 - **%MetricName%** - the name of the metric on which alert was generated.
 - **%MetricType%** - the type of the metric on which alert was generated.
 - **%MetricValue%** - the value of the metric on which alert was generated.
 - **%SchemaName%** - the name of the schema on which alert was generated.
 - **%PackageName%** - the name of the package on which alert was generated.
 - **%FunctionName%** - the name of the function or procedure on which alert was generated.
 - **%VariableName%** - the list of custom parameters used to generate the alert.
 - **%AlertInfo%** - the detailed database object level information of the alert.

- **%ServerID%** - the id of the server on which alert was generated.
- **%ServerName%** - the name of the server on which alert was generated.
- **%ServerIP%** - the ip or address of the server on which alert was generated.
- **%ServerPort%** - the port of the server on which alert was generated.
- **%DatabaseName%** - the name of the database on which alert was generated.
- **%SchemaName%** - the name of the schema on which alert was generated.
- **%PackageName%** - the name of the package on which alert was generated.
- **%DatabaseObjectName%** - the name of the database object name like table name, function name etc on which alert was generated.
- **%Parameters%** - the list of custom parameters used to generate the alert.
- **%AlertInfo%** - the detailed database object level information of the alert.

- Click on the **Test Connection** button, to test notification delivery to the mentioned endpoint.

Webhooks						
Name	URL	Enable?	High alerts?	Medium alerts?	Low alerts?	Cleared alerts?
Django	http://192.168.1.2:8000/public-w...	<input checked="" type="checkbox"/>				

General HTTP Headers Payload Notifications

Select Yes to enable the alert notifications based on threshold value violations. You can override these notification setting from manage alert tab specific to alert.

All alerts? Select Yes to enable all the alert notifications.

Alert notifications

High alerts?	<input checked="" type="checkbox"/>
Medium alerts?	<input checked="" type="checkbox"/>
Low alerts?	<input checked="" type="checkbox"/>
Cleared alerts?	<input checked="" type="checkbox"/>

Use the **Notifications** tab to specify an alert level for webhook endpoints:

- Set **All alerts** to **Yes** to enable all alert levels to send notifications.
- To instruct PEM to send a notification when a specific alert level is reached, set the slider next to an alert level to **Yes**. Please note that you must set **All alerts** to **No** to configure an individual alert level.

Deleting a Webhook

To mark a webhook for deletion, highlight the webhook name in the **Webhooks** table and click the delete icon to the left of the name; the alert will remain in the list, but in strike-through font.

Webhooks						
Name	URL	Enable?	High alerts?	Medium alerts?	Low alerts?	Cleared alerts?
Django	http://192.168.1.2:8000/public-w...	<input checked="" type="checkbox"/>				
Slack	https://hooks.slack.com/services/T0...	<input checked="" type="checkbox"/>				

The delete icon acts as a toggle; you can undo the deletion by clicking the delete icon a second time; when you save your work (by clicking the save icon), the webhook definition will be permanently deleted.

Using PEM with Nagios

The PEM server can send a passive alert result to Nagios network-alerting software when a user-defined alert is triggered. To instruct the PEM server to notify Nagios of a triggered alert, you must:

- Enable Nagios notification for each alert that will trigger a notification from the PEM server to Nagios. Please note that PEM alerting must be configured before you create the host.cfg file and services.cfg file.
- Configure Nagios-related behaviors of the PEM server.
- Create the host.cfg and services.cfg configuration files.
- If necessary, modify the Nagios configuration file and restart the Nagios server.
- Install the PEM Agent on the system where Nagios server is installed and register it with the PEM Server. Set `enable_nagios` configuration to `true` in the agent.cfg for that agent, and restart the agent service.

After configuring the server to enable Nagios alerting, any triggered alerts will send a passive check result to the Nagios service. The syntax of a passive alert is:

```
<timestamp> PROCESS_SERVICE_CHECK_RESULT; <host_name> ; <service_name> ;
<service_status> ;
```

Where:

`timestamp` is the date and time that the alert was triggered.

`host_name` is the name of the server or agent.

`service_name` is the name of the alert.

`service_status` is the numeric service status value:

0 if the service status is OK 1 if the service status is WARNING 2 if the service status is CRITICAL 3 if the service status is UNKNOWN

The PEM server uses the following rules to evaluate the service status:

- If the PEM alert level is CLEARED, the warning message will read OK.
- If the PEM alert level is LOW, the warning message will read WARNING.
- If the `is_nagios_medium_alert_as_critical` flag (specified in the PEM server configuration dialog) is set to FALSE and the alert level MEDIUM, the warning message will read WARNING.
- If the `is_nagios_medium_alert_as_critical` flag (specified in the PEM server configuration dialog) is set to TRUE and the alert level is MEDIUM, the warning message will read CRITICAL.
- If the PEM alert level is `HIGH`, the warning message will read `CRITICAL`.

Enabling Nagios Notification for an Alert

The PEM server maintains a unique set of notification properties for each enabled alert. Use the `Notification` tab of the `Manage Alerts` tab to specify that (when triggered), a given alert will send an alert notice to Nagios.

To modify the notification properties of an alert, right-click on the name of the object monitored by the alert, and select `Manage Alerts...` from the `Management` menu. When the `Manage Alerts` tab opens, locate the alert, and then click the edit button to the left of the alert name in the `Alerts` list. When the edit pane opens, select the `Notification` tab.

General **Notification**

Email notification

All alerts? No

Low alerts? No

Medium alerts? No

High alerts? No

To configure notifications for an alert, use the fields in the Email notification box to specify the user or user group that will receive an email notification if the alert is triggered at the specified level. Use the drop-down listbox to select a pre-defined group that will be sent a notification if an alert of the selected level is triggered. Please note that you must configure the PEM Server to use an SMTP server to deliver email before PEM can send email notifications.

Trap notification

Send trap? No

Medium alert? No

High alert? No

Use the Trap notification options to configure trap notifications for this alert. Note that you must configure the PEM Server to send notifications to an SNMP trap/notification receiver before notifications can be sent.

Nagios notification

Submit passive service check result to Nagios? Yes

Set "Submit passive service check result to Nagios" to "Yes" to instruct the PEM server to notify Nagios when the alert is triggered or cleared.

Script execution

Execute script? No

Execute script on PEM Server Monitored Server

Code

Use the fields in the Script execution box to (optionally) define a script that will be executed if an alert is triggered, and to specify details about the script execution.

- Set the Execute script slider to Yes to instruct PEM to execute the provided script if an alert is triggered.
- Set the Execute on alert cleared slider to Yes to instruct PEM to execute the provided script when the situation that triggered the the alert has been resolved.
- Use the selector to indicate if the script should execute on the PEM Server or the Monitored Server.
- Provide the script that PEM should execute in the Code field. You can provide a batch/shell script, or SQL code. Within the script you can use the placeholders to replace the following:
 - %AlertName% - the name of the triggered alert.
 - %ObjectName% - the name of the server or agent on which the alert was triggered.
 - %ThresholdValue% - the threshold value reached by the metric when the alert triggered.
 - %CurrentValue% - the current value of the metric that triggered the alert.
 - %CurrentState% - the current state of the alert.
 - %OldState% - the previous state of the alert.
 - %AlertRaisedTime% - the time that the alert was raised, or the most recent time that the alert state was changed.

To enable Nagios notification, move the slider next to **Submit passive service check result** to Nagios to **Yes**; before exiting the **Manage Alerts** tab, click the save icon to preserve your changes.

Configuring Nagios-related behavior of the PEM Server

You can use the **Server Configuration** dialog to provide information about your Nagios configuration to the PEM server. To open **Server Configuration** dialog, select **Server Configuration...** from the PEM client's **Management** menu.

Server Configuration

		Search by parameter name
flapping_detection_state_change	3	
job_failure_notification	<input type="checkbox"/> False	t/f
job_notification_email_group	default	
job_retention_time	30	days
job_status_change_notification	<input type="checkbox"/> False	t/f
long_running_transaction_minutes	5	minutes
max_metrics_per_group_chart	16	
nagios_cmd_file_name	/usr/local/nagios/var/rw/nagios.cmd	
nagios_enabled	<input checked="" type="checkbox"/> True	t/f
nagios_medium_alert_as_critical	<input type="checkbox"/> False	t/f
nagios_spool_retention_time	7	days
probe_log_retention_time	30	days
reminder_notification_interval	24	hours
server_log_retention_time	30	days
show_data_points_on_graph	<input type="checkbox"/> False	t/f
show_data_tab_on_graph	<input type="checkbox"/> False	t/f
show_unmanaged_servers	<input checked="" type="checkbox"/> True	t/f

?

Cancel Reset Save

Four server configuration parameters specify information about your Nagios installation and PEM server behavior related to Nagios:

- Use the `nagios_cmd_file_name` parameter to specify the location of the Nagios pipeline file that will receive passive check alerts from PEM. The default value of this parameter is `/usr/local/nagios/var/rw/nagios.cmd`. If your `nagios.cmd` file resides in an alternate location, specify the file location in the Value field.
- Move the slider in the `nagios_enabled` parameter to `Yes` to instruct the PEM server to send passive check alerts to Nagios.
- Use the `nagios_medium_alert_as_critical` slider to specify the warning severity that the PEM server will pass to Nagios if a medium alert is triggered:

If the `is_nagios_medium_alert_as_critical` flag is set to FALSE and the alert level is MEDIUM, the warning message will read WARNING.

If the `is_nagios_medium_alert_as_critical` flag is set to TRUE and the alert level is MEDIUM, the warning message will read CRITICAL.

- Use the `nagios_spool_retention_time` parameter to specify the number of days of notification history that will be stored on the PEM server. The default value is 7 days.

After modifying parameter values, click the save icon (in the upper-right corner of the `Server Configuration` dialog) to preserve your changes.

Creating the hosts.cfg and services.cfg File

The `templates.cfg` file (by default, located in `/usr/local/nagios/etc/objects`) specifies the properties of a generic-host and generic-service. The properties specify the parameters used in the `hosts.cfg` and `services.cfg` files.

In most cases (when PEM is installed in a default configuration), you will not be required to modify the `templates.cfg` file before creating the `hosts.cfg` and `services.cfg` files. If necessary, you can modify the

`templates.cfg` file to specify alternate values for parameters or to create new templates.

Before modifying the Nagios configuration file, use the following command to create a `hosts.cfg` file that contains information about the PEM hosts that reside on the local system:

```
psql -U postgres -p 5433 -d pem -A -t -c "select
pem.create_nagios_host_config('generic-host')"
> /usr/local/nagios/etc/objects/hosts.cfg
```

Then, use the following command to create a `services.cfg` file that contains information about the PEM services that reside on the local system:

```
psql -U postgres -p 5433 -d pem -A -t -c "select
pem.create_nagios_service_config('generic-service')"
> /usr/local/nagios/etc/objects/services.cfg
```

If you wish to use a `custom template.cfg` file entry, specify the entry name in place of generic-host or generic-service in the above commands.

Modifying the Nagios Configuration File

After creating the `host.cfg` and `services.cfg` files, you must specify their location in the Nagios configuration file (by default, `/usr/local/nagios/etc/nagios.cfg`). Modify the configuration file, adding entries that specify the location of the files:

```
cfg_file=/usr/local/etc/objects/hosts.cfg
```

```
cfg_file=/usr/local/etc/objects/services.cfg
```

You can use the following command to confirm that Nagios is properly configured:

```
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
```

After confirming that Nagios is configured correctly, restart the Nagios service:

```
/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
```

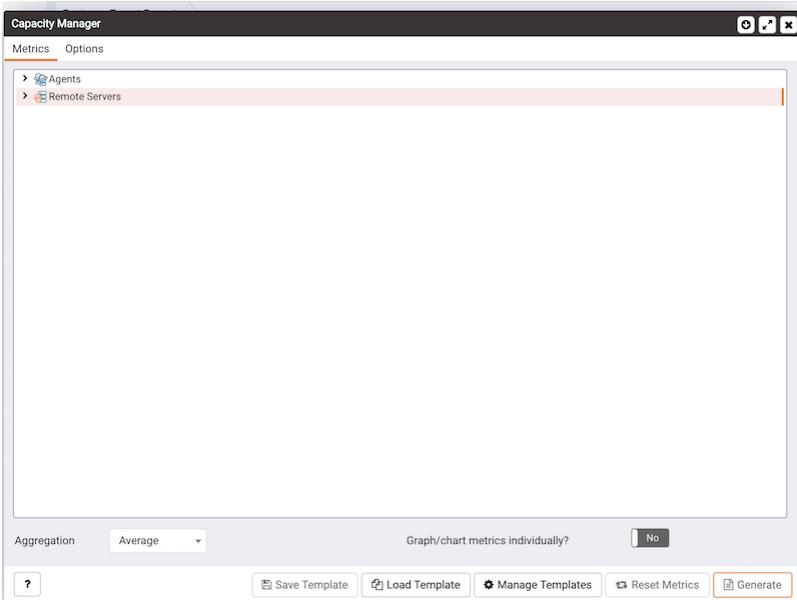
4.5 Capacity Manager

PEM's Capacity Manager analyzes collected statistics (metrics) to generate a graph or table that displays the historical usage statistics of an object, and can project the anticipated usage statistics for an object. You can configure Capacity Manager to collect and analyze metrics for a specific host, server, database, or database object.

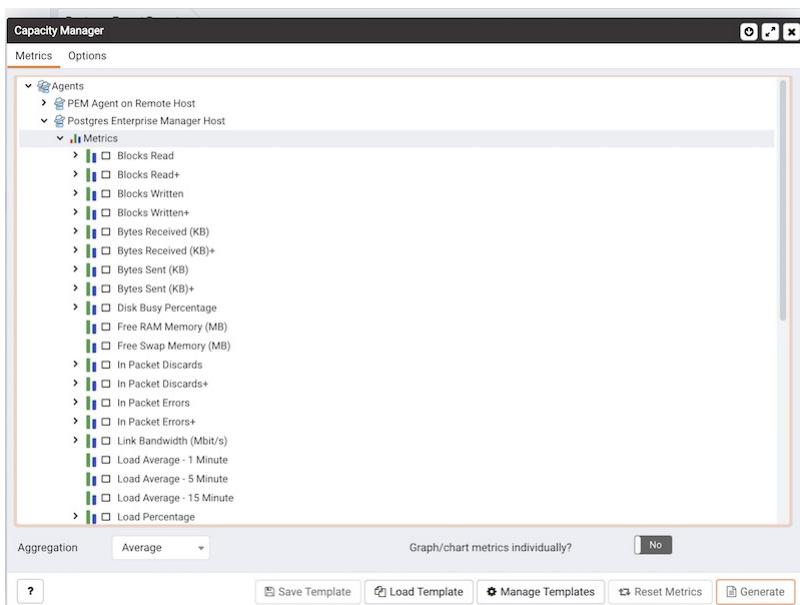
You can tailor the content of the Capacity Manager report by choosing a specific metric (or metrics) to include in the report, the time range over which the metrics were gathered, and a high or low threshold for the metrics analyzed. You can also specify a start and end date for the Capacity Manager report. If the end date of the report specifies a time in the future, Capacity Manager will analyze the historical usage of the selected object to extrapolate the projected object usage in the future.

To open Capacity Manager, select the `Capacity Manager...` option from the PEM client `Management` menu; the

Capacity Manager wizard opens, displaying a tree control on the **Metrics** tab.



Expand the tree control on the **Metrics** tab to review the metrics for the node that you wish to analyze. Check the box to the left of the name of the metric to include the metric in your report.



Capacity Manager will use the aggregation method specified with the Aggregation drop-down listbox (located at the bottom of the Metrics tab). The aggregation method instructs Capacity Manager how to evaluate and plot the metric values. Select from:

- **Average**: Use the average of the values recorded during the time period.
- **Maximum**: Use the maximum value recorded during the time period.
- **Minimum**: Use the minimum value recorded during the time period.
- **First**: Use the first value recorded during the time period.

To remove a metric from the Capacity Manager report, uncheck the box to the left of the name of a metric.

Move the slider next to **Graph/chart metrics individually?** to **Yes** to instruct Capacity Manager to produce a separate report for each metric selected on the Metrics tab. If the option is set to No, all selected metrics will be merged into a single graph or table.

Click the **Generate** button to display the report onscreen (accepting the default configuration options), or use the Options tab to customize sampling boundaries, report type and report destination. Please note that the times displayed on the Options tab are the time zone in which the PEM client resides.



Use the fields within the **Time Period** box to define the boundaries of the Capacity Manager report:

- Use the **Period** drop-down listbox to select the type of time period you wish to use for the report. You can select:

Start time and end time	Specify a start date and an end date/time for the report.
Start time and threshold	Specify a start date and time, and a threshold to determine the end time and date for the report.
Historical days and extrapolated days	Specify a start date for the report that is a number of days in the past, and an end date that is a number of days in the future. This option is useful for report templates that do not specify fixed dates.
Historical days and threshold	Specify a start date that is a number of days in the past, and end it when a threshold value is reached.

After specifying the type of time period for the report, select from other options in the Time Period box to define the time period for the report:

- Use the date and time selectors next to the **Start time** field to specify the starting date and time of the sampling period, or select the number of Historical day(s) of data to include in the report. The date and time specified in the Start time field must not be later than the current date/time.

By default, Capacity Manager will select a start time that is one week prior to the current date and time.

- The end boundary for the report can be a time, a number of days in the future, or the point at which a selected metric reaches a user-specified threshold value. Use the date and time selectors next to the **End time** field to specify an end boundary for the report, or select the number of Extrapolated day(s) of data to include in the report. The time specified in the End time field must be later than the time specified in the Start time field.

Note that if you select an end date and time in the future, Capacity Manager will use historical usage information to extrapolate anticipated future usage. Since the projected usage is based on the sampling of historical data, the accuracy of the future usage trend will improve with a longer sampling period.

To specify a threshold value, use the drop-down listbox in the Threshold field to select a metric, an operator (Exceeds

or Falls below), and enter a target value for the metric. If you choose to define the end of the report using a threshold, the Capacity Manager report will terminate when the value for the selected metric exceeds or falls below the specified value.

The `cm_max_end_date_in_years` configuration parameter defines a default time value for the end boundary of a Capacity Manager report. If you specify a threshold value as the end boundary of a report, and the anticipated usage of the boundary is not met before the maximum time has passed, the report will terminate at the time specified by the `cm_max_date_in_years` parameter. By default, `cm_max_end_date_in_years` is 5; you can use the Server Configuration dialog to modify the value of `cm_max_end_date_in_years`.

The fields in the `Report` box specify the report type and destination. Use the `Include on report` radio buttons to specify the type of report produced by Capacity Manager:

- Select `Graph` to instruct Capacity Manager to display the report in the form of a line graph in the PEM client window.
- Select `Table of data` to instruct Capacity Manager to display a table containing the report data in the PEM client window.
- Select `Graph and table of data` to instruct Capacity Manager to display both a line graph and a data table in the PEM client window.

Use the `Report destination` radio buttons to instruct Capacity Manager where to display or save the report:

- Select `New` tab to instruct Capacity Manager to display the report on a new tab in the PEM client. You must select `New` tab to display the first generation of a Capacity Manager report; for subsequent reports, you may select `Previous` tab.
- Select `Previous` tab to instruct Capacity Manager to re-use a previously opened tab when displaying the report.
- Select `Download` the report as a file and specify a file name to instruct Capacity Manager to write the report to the specified file.

When you have specified the report boundaries and selected the type and destination of the Capacity Manager report, click the `Generate` button to create the report.



Reports saved to file are stored in HTML format. You can review a Capacity Manager report with any web browser that supports Scalable Vector Graphics (SVG). Browsers that do not support SVG will be unable to display a Capacity Manager graph and may include unwanted characters.

Capacity Manager Templates

After defining a report, you can save the definition as a template for future reports. Capacity Manager report templates may be accessed by all PEM users. To save a report definition as a template:

1. Use the Metrics and Options tabs to define your report.
2. Click the Save button to open the Save Template dialog.



1. Provide a report name in the Title field, select a location to store the template in the tree control.
2. Click OK.

When creating a report, you can use the Load Template button to browse and open an existing template. Once opened, the report definition may be modified if required, and optionally saved again, either as a new template, or overwriting the original template.

Use the Manage Templates button open a dialog that allows you to rename or remove unwanted templates.

4.6 Audit Manager

You can use the PEM Audit Manager to simplify audit log configuration for Advanced Server instances. With the Audit Manager, you can configure logging attributes such as:

- How often log files are to be collected by PEM
- The type of database activities that are included in the log files
- How often (and when) log files are to be rotated

Audit logs may include the following activities:

- All connections made to the database instance
- Failed connection attempts
- Disconnections from the database instance
- All queries (SELECT statements)

- All DML statements (INSERT, UPDATE, DELETE)
- All DDL statements (e.g., CREATE, DROP, ALTER)

Once the audit logs are stored on the PEM server, you can use the Audit Log dashboard to review the information in an easy-to-read form. The Audit Log dashboard allows you to filter the log file by timestamp range (when an activity occurred), the database on which the activity occurred, the user performing the activity, or the type of command being invoked.

Setting the Advanced Server Instance Service ID

To configure logging for an Advanced Server instance, the server must be a PEM-managed server with a bound agent, and the server registration must include the name of a service script. When registering a new server, include the service name in the Service ID field on the Advanced tab of the New Server dialog.

Before adding a service name to an existing (registered and connected) server, you must disconnect the server. Right click on the server name, and select **Disconnect server** from the context menu. Then, right click on the server name and select **Properties** from the context menu. Select the **Advanced** tab, and add a service name to the **Service ID** field.



The Service ID field allows the PEM server to stop and start the service.

- The name of the Advanced Server 11 service script is **edb-as-12**.
- The name of the Advanced Server 11 service script is **edb-as-11**.
- The name of the Advanced Server 10 service script is **edb-as-10**.
- The name of the Advanced Server 9.6 service script is **edb-as-9.6**.
- The name of the Advanced Server 9.5 (or prior) service script is **ppas-9.x**, where **x** specifies the version.
- The name of the PostgreSQL 9.6 service script is **postgresql-11**.
- The name of the PostgreSQL 9.6 service script is **postgresql-10**.
- The name of the PostgreSQL 9.6 service script is **postgresql-9.6**.

Setting the EDB Audit Configuration Probe

Before configuring audit logging of Advanced Server servers, you must ensure that the EDB Audit Configuration probe is enabled. To open the **Manage Probes** tab and check the status of the probe, right click on the name of a registered Advanced Server server in the tree control, and select **Manage Probes...** from the **Management** menu.

Ensure that the **Enabled** column in the **Probe Configuration** dialog is set to **Yes** for the **EDB Audit**

Configuration probe.

Probes							
Probe name	Execution Frequency			Enabled?		Data Retention	
	Default?	Minutes	Seconds	Default?	Probe Enable?	Default?	Days
Background Writer Statistics	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Blocked Session Information	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Data and Log File Analysis	<input checked="" type="checkbox"/>	0	10	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Database Frozen XID	<input checked="" type="checkbox"/>	720	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Database Size	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Database Statistics	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	90
EDB Audit Configuration	<input checked="" type="checkbox"/>	2	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Failover Manager Cluster Info	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	7
Failover Manager Node Status	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	7

If EDB Audit Configuration is not enabled, use the **Enabled?** switch on the Manage Probes tab to enable it.

Configuring Audit Logging with the Audit Manager

To open the **Audit manager** wizard, select **Audit Manager...** from the **Management** menu. The **Audit manager - Welcome** dialog opens.



Click **Next** to continue.



Use the Select servers tree control to specify the servers to which the auditing configuration will be applied. To make a server available in the tree control, you must provide the **Service ID** on the **Advanced** tab of the **Create – Server** dialog when registering a server for monitoring by PEM. Note that only EDB Postgres Advanced Server supports auditing; PostgreSQL servers will not be included in the tree control.

Click **Next** to continue.

The **Auditing Parameters Configuration** dialog lets you enable or disable auditing and choose how often log records are collected into PEM.



Use the fields on the **Auditing parameters configuration** dialog to specify auditing preferences:

- Use the **Auditing** switch to Enable or Disable auditing on the specified servers.
- Use the **Audit destination** drop-down to select a destination for the audit logs; select File or Syslog. Please note this feature is supported on Advanced Server 10 and newer releases only.
- Use the **Import logs to PEM** switch to instruct PEM to periodically import log records from each server to the PEM Server. Set the switch to Yes to import log files; the default is No.
- Use the **Collection frequency** drop-down listbox to specify how often PEM will collect log records from monitored servers when log collection is enabled.

- Use the **Log format** drop-down listbox to select the raw log format that will be written on each server. If log collection is enabled, the PEM server will use CSV format.
- Use the **File name** field to specify the format used when generating log file names. By default, the format is set to **audit-%Y-%m-%d_%H%M%S** where:
 - audit** is the file name specified in the Audit Directory Name field
 - Y** is the year that the log was stored
 - m** is the month that the log was stored
 - d** is the day that the log was stored
 - H** is the hour that the log was stored
 - M** is the minute that the log was stored
 - S** is the second that the log was stored
- Check the box next to **Change Log Directory for selected servers?** and use the **Audit Directory Name** field to specify a directory name to which the audit logs will be written. The directory will reside beneath the data directory on the PEM server.
- Use fields in the **Log directory** box to specify information about the directory in which the log files will be saved:

Move the **Change log directory for selected servers?** switch to Yes to enable the Directory name field.

Use the **Directory name** field to specify the name of the directory on each server into which audit logs will be written. The directory specified will be created as a sub-directory of the data directory on the server.

Click **Next** to continue.

The **Audit log configuration** dialog is only available if you have enabled auditing on the Auditing parameters configuration dialog.



Use the controls on the **Audit log configuration** dialog to specify log configuration details that will be applied to each server:

- Use the **Connection attempts** switch to specify if connection attempts should be logged:

None to disable connection logging.

All to indicate that all connection attempts will be logged.

Failed to log any connection attempts that fail.

- Use the **Disconnection attempts** switch to specify if disconnections should be logged. Specify:

None to specify that disconnections should not be logged.

All to enable disconnection logging.

- Use the **Log statements** field to specify the statement types that will be logged. Click within the field, and select from:

Select - All statements that include the **SELECT** keyword will be logged.

Error - All statements that result in an error will be logged.

DML - All DML (Data Modification Language) statements will be logged.

DDL - All DDL (Data Definition Language) statements (those that add, delete or alter data) will be logged.

Check the box next to **Select All** to select all statement types.

Check the box next to **Unselect All** to deselect all statement types.

- Use the **Audit tag** field to specify a tracking tag for the collected logs. Please note that audit tagging functionality is available only for Advanced Server versions 9.5 and later. If you are defining auditing functionality for multiple servers, and one or more of the servers are version 9.5 or later, this field will be enabled, but if selected, tagging functionality will only apply to those servers that are version 9.5 or later.

- Use the fields in the **Log rotation** box to specify how the log files are managed on each server:

Use the **Enable?** switch to specify that logfiles should be rotated. Please note that a new log file should be used periodically to prevent a single file becoming unmanageably large.

Use the **Day drop-down** listbox to select a day or days on which the log file will be rotated.

Use the **Size (MB)** field to specify a size in megabytes at which the log file will be rotated.

Use the **Time (seconds)** field to specify the number of seconds between log file rotations.

Click **Next** to continue:



Use the **Schedule Auditing Changes** dialog to determine when auditing configuration changes are to take effect.

- Select **Configure logging now?** if you want the auditing configuration changes to take place immediately. The affected database servers will be restarted so the auditing changes can take effect.
- Use the **Time?** selector to schedule the auditing configuration changes to take place at some point in the future. Select the desired date and time from the drop-down lists. The affected database servers will be restarted at the specified date/time to put the auditing changes into effect.

Click **Finish** to complete the auditing configuration process.

The Audit Manager will schedule a job to apply the configuration to each server. The job will consist of two tasks: one to update the audit logging configuration on the server, and one to restart the server with the new configuration.

You can use the **Scheduled Tasks** tab to review a list of Scheduled jobs. To open the **Scheduled Tasks** tab, highlight the name of a server or agent and select **Scheduled Tasks...** from the **Management** menu.

Viewing the Log with the Audit Log Dashboard

Use the Audit Log dashboard to view the audit log from Advanced Server database instances.

To open the **Audit Log** dashboard, right click on a server or agent node, and select **Audit Log Analysis** from the **Dashboards** menu. You can also open the Audit Log dashboard by navigating through the **Dashboards** menu (located on the **Management** menu).

ID	Server	Timestamp	User Name	Database Name	Process ID	Session ID	Transaction ID	Connection From	Command	Message
7879	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26391	Sea902cf.6717	0	127.0.0.1:46780	idle	disconnection: session time: 0:00:00.014 user=enterprisedb database=postgres host=127.0.0.1 port=46780
7878	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26391	Sea902cf.6717	0	127.0.0.1:46780	idle	statement: SELECT setting FROM pg_settings WHERE name = 'edb_audit_rotation_seconds'
7877	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26391	Sea902cf.6717	0	127.0.0.1:46780	idle	statement: SELECT version();
7876	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26391	Sea902cf.6717	0	127.0.0.1:46780	authentication	connection authorized: user=enterprisedb database=postgres
7875	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26388	Sea902cf.6714	0	127.0.0.1:46774	idle	disconnection: session time: 0:00:00.007 user=enterprisedb database=postgres host=127.0.0.1 port=46774
7874	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26388	Sea902cf.6714	0	127.0.0.1:46774	authentication	connection authorized: user=enterprisedb database=postgres
7873	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	Sea902cf.6710	0	127.0.0.1:46766	idle	disconnection: session time: 0:00:00.029 user=enterprisedb database=postgres host=127.0.0.1 port=46766
7872	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	Sea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT setting FROM pg_settings WHERE name='log_temp_files'
7871	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	Sea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT setting FROM pg_settings WHERE name='log_autovacuum_min_duration'
7870	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	Sea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT setting FROM pg_settings WHERE name='log_min_duration_statement'
7869	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	Sea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT (setting::int/(24*60))::int FROM pg_settings WHERE name = 'log_rotation_agr'
7868	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	Sea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT (setting::int/1024)::int FROM pg_settings WHERE name = 'log_rotation_size'
7867	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	Sea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT upper(setting) FROM pg_settings WHERE name='syslog_facility';

The Audit Log dashboard displays the audit records in reverse chronological order (newest records at the top, oldest records towards the bottom).

To view older audit records that do not appear in the window, use the vertical scroll bar controlling the list of audit records (the innermost scroll bar of the two located on the right-hand side of the window). As you move the scroll bar towards the bottom of the window, older audit records are continuously loaded and displayed.

You can use filtering to limit the number of audit records that are displayed. Click **Show Filters** to expose the filters panel.

ID	Timestamp	User Name	Database Name	Process ID	Session ID	Transaction ID	Connection From	Command	Message
7879	2020-04-29 08:36:42	enterprisedb	edb	26391	Sea902cf.6717	0	127.0.0.1:46780	idle	disconnection: session time: 0:00:00.014 user=enterprisedb database=postgres host=127.0.0.1 port=46780

Use the fields in the **filters panel** to provide certain selection criteria for the audit records you wish to display.

- Use the **Start** field to specify a start date for the report. Click the mouse button in the field to open a calendar and select a start date.
- Use the **End** field to specify an end date for the report. Click the mouse button in the field to open a calendar and select an end date.
- Use the **User** field to display only those entries where the activity was initiated by the given Postgres user.
- Use the **Database** field to display only those entries where the activity was issued on the given database.
- Use the **Command type** field to display only those entries where the activity was of the given type. Command types you can specify are idle, authentication, and SELECT. (For viewing SQL statements from user applications, specify the idle command type.)

Click **Filter** to apply the filtering criteria to the log entries.

4.7 Log Manager

You can use the PEM Log Manager to simplify server log configuration for Postgres instances. With the Log Manager, you can modify all of your server log parameters with a click:

- Where log files are written
- How often log files are written
- The type of information written to log files
- The format of log file entries
- Log rotation properties

To configure logging for a Postgres instance, the server must be registered as a PEM-managed server, and the registration information must include the name of a service script.

To open the **Log Manager**, select the **Log Manager...** option from the **Management** menu of the PEM client. The wizard opens, welcoming you to the Log Manager.



Click **Next** to continue to the **Server selection** dialog.



The **Server selection** dialog displays a list of the server connections monitored by PEM. Check the box next to the name of a server (or servers) to which the Log Manager wizard will apply the specified configuration. Log Manager is disabled for any server displaying a red exclamation mark to the left of its name in the Server selection tree control; there are several reasons that a server may not be enabled:

- Only a server that specifies a **Service ID** on the **Advanced** tab of the **Properties** dialog can be configured by Log Manager.

To provide a service ID, right click on the server name in the tree control, and select **Disconnect Server** from the context menu; if prompted, provide a password. Then, open the context menu for the server, and select **Properties**. Navigate to the **Advanced** tab, and provide the name of the service in the **Service ID** field; click **Save** to save your change and exit the dialog.

- If the PEM agent bound to the server does not have sufficient privileges to restart the server, the server will be disabled.
- If the PEM agent bound to the server is an older version than the associated PEM server, the server will be disabled.

Click **Next** to continue.



Use the options on the **Log configuration** dialog to specify how often log files will be imported to PEM and to specify log rotation details:

Options within the **Import Logs** box specify how often log files will be imported to PEM:

- Use the switch next to the **Import logs to PEM** label to specify if log files will be imported to PEM and displayed on the Server Log Analysis dashboard.
- Use the **Import Frequency** drop-down list box to specify how often log files are imported to PEM.

Use the fields in the **Log rotation configuration** box to specify the maximum length (lifespan or size) of a log file:

- Use the **Rotation Size** field to specify the maximum size in megabytes of an individual log file. The default value is 10 MB; when set to 0, no limit is placed on the maximum size of a log file.
- Use the **Rotation Time** field to specify the number of whole days that should be stored in each log file. The default value is 1 day.

Use the **Truncation on Rotation** switch to specify server behavior for time-based log file rotation:

- Select **ON** to specify that the server should overwrite any existing log file that has the same name that a new file would take.
- Select **OFF** to specify that the server should append any new log file entries to an existing log file with the same name that a new log file would take. This is the default behavior.

Click **Next** to continue.



Use the fields on the **Where to log** dialog to specify where log files should be written.

- Select an option from the **Log Destination** box to specify a destination for the server log output:
 - Set the **stderr** switch to **Yes** to specify that log files should be written to stderr.
 - Set the **csvlog** switch to **Yes** to specify that log files should be written to file in a comma-separated value format. This option is automatically enabled (and no longer editable) if you have selected **Import logs to PEM** on the **Schedule** dialog; if you are not importing server log files to PEM, this option is editable.
 - Set the **syslog** switch to **Yes** to specify that log files should be written to the system log files.
 - On Windows, set the **eventlog** switch to Yes to specify that log files should be written to the event log.
- Use the options within the **Log collection** box to specify your collection preferences:
 - Set the **Log Collector** switch to **Enable** to instruct the server to re-direct captured log messages (directed to STDERR) into log files.
 - Set the **Log Silent Mode** switch to **Enable** to instruct the server to run silently in the background, disassociated from the controlling terminal.
- Use options in the **Log Directory** box to specify log file location preferences:
 - Set the **Change log directory for selected servers?** switch to **Yes** to specify that each set of log files should be maintained in a separate directory.
 - Use the **Directory name** field to specify the directory to which log files will be written. The directory will reside beneath the pg_log directory under the installation directory of the monitored server.
- Use the **Log File Name** field to specify a format for the log file name. If set to **DEFAULT**, the format is **enterprisedb-%Y-%m-%d_%H%M%S**, where:
 - **enterprisedb** is the file name prefix
 - **Y** is the year that the log was stored
 - **m** is the month that the log was store

- **d** is the day that the log was stored
- **H** is the hour that the log was stored
- **M** is the minute that the log was stored
- **S** is the second that the log was stored

- When logging to syslog is enabled:

- Use the **Syslog Facility** drop-down list box to specify which syslog facility should be used.
- Use the **Syslog Ident** field to specify the program name that will identify Advanced Server entries in system logs.

Click **Next** to continue.



Use the fields on the **When to log** dialog to specify which events will initiate a log file entry. The severity levels (in order of severity, from most severe to least severe) are:

- **panic** - Errors that cause all database sessions to abort.
- **fatal** - Errors that cause a session to abort.
- **log** - Information messages of interest to administrators.
- **error** - Errors that cause a command to abort.
- **warning** - Error conditions in which a command will complete but may not perform as expected.
- **notice** - Items of interest to users. This is the default.
- **info** - Information implicitly requested by the user.
- **debug5** through **debug1** - Detailed debugging information useful to developers.
- Use the **Client min messages** drop-down list box to specify the lowest severity level of message sent to the client application.
- Use the **Log min messages** drop-down list box to specify the lowest severity level that will be written to the server log.
- By default, when an error message is written to the server log, the text of the SQL statement that initiated the log entry is not included. Use the **Log min error statement** drop-down list box to specify a severity level that will trigger SQL statement logging. If a message is of the specified severity or higher, the SQL statement that produced the message will be written to the server log.
- Use the **Log min duration statement** drop-down list box to specify a statement duration (in milliseconds); any statements that exceed the specified number of milliseconds will be written to the server log. A value of **-1** disables all duration-based logging; a value of **0** logs all statements and their duration.
- Use the **Log temp files** field to specify a file size in kilobytes; when a temporary file reaches the specified size, it will be logged. A value of **-1** (the default) disables this functionality.
- Use the **Log autoVacuum min duration** field to specify a time length in milliseconds; if auto-vacuuming

exceeds the length of time specified, the activity will be logged. A value of `-1` (the default) disables this functionality.

Click **Next** to continue.



Use the fields on the **What to log** dialog to specify log entry options that are useful for debugging and auditing.

The switches in the **Debug options** box instruct the server to include information in the log files related to query execution that may be of interest to a developer:

- Set the **Parse tree** switch to Yes to instruct the server to include the parse tree in the log file.
- Set the **Rewriter output** switch to Yes to instruct the server to include query rewriter output in the log file.
- Set the **Execution plan** switch to Yes to instruct the server to include the execution plan for each executed query in the log file.

When the **Indent Debug Options Output in Log** switch is set to **Yes**, the server indents each line that contains a parse tree entry, a query rewriter entry or query execution plan entry. While indentation makes the resulting log file more readable, it does result in a longer log file.

Use the switches in the **General Options** box to instruct the server to include auditing information in the log file:

- Set the **Checkpoints** switch to **Yes** to include checkpoints and restartpoints in the server log.
- Set the **Connections** switch to **Yes** to include each attempted connection to the server (as well as successfully authenticated connections) in the server log.
- Set the **Disconnections** switch to **Yes** to include a server log entry for each terminated session that provides the session information and session duration.
- Set the **Duration** switch to **Yes** to include the amount of time required to execute each logged statement in the server log.
- Set the **Hostname** switch to **Yes** to include both the IP address and host name in each server log entry (by default, only the IP address is logged). Please note that this may cause a performance penalty.
- Set the **Lock Waits** switch to **Yes** to instruct the server to write a log entry for any session that waits longer than the time specified in the deadlock_timeout parameter to acquire a lock. This is useful when trying to determine if lock waits are the cause of poor performance.

Use the **Error verbosity** drop-down list box to specify the detail written to each entry in the server log:

- Select **default** to include the error message, DETAIL, HINT, QUERY and CONTEXT in each server log entry.
- Select **terse** to log only the error message.

- Select **verbose** to include the error message, the DETAIL, HINT, QUERY and CONTEXT error information, SQLSTATE error code and source code file name, the function name, and the line number that generated the error.

Use the **Prefix string** field to specify a printf-style string that is written at the beginning of each log file entry. For information about the options supported, please see the `log_line_prefix` documentation (in the Postgres core documentation), available at:

<http://www.postgresql.org/docs/current/static/runtime-config-logging.html>

Use the **Statements** drop-down list box to specify which SQL statements will be included in the server log. The default is none; valid options are:

- Specify **none** to disable logging of SQL statements.
- Specify **ddl** to instruct the server to log ddl (data definition language) statements, such as CREATE, ALTER, and DROP.
- Specify **mod** to instruct the server to log all ddl statements, as well as all dml (data modification language) statements, such as INSERT, UPDATE, DELETE, TRUNCATE and COPY FROM.
- Specify **all** to instruct the server to log all SQL statements.

Click **Next** to continue.



Use options on the **Schedule logging changes** dialog to specify when logging configuration changes will be applied:

- Set the **Configure logging now** switch to **Yes** to specify that your configuration preferences will be enabled, and the server will restart when you have completed the Log Manager wizard.
- Set **Configure logging now** to **No** to use the Schedule it for some other time calendar selector to specify a convenient time for logging configuration preferences to be applied, and the server to restart.

Note that when you apply the configuration changes specified by the Log Manager wizard, the server restart will temporarily interrupt use of the database server for users.

Click **Finish** to exit the wizard, and either restart the server, or schedule the server restart for the time specified on the scheduling dialog.

Reviewing the Server Log Analysis Dashboard

After invoking the Log Manager wizard, and importing your log files to PEM, you can use the **Server Log Analysis** dashboard to review the log files for a selected server. To open the **Server Log Analysis** dashboard, right-click on the name of a monitored server in the PEM client tree control, and navigate through the **Dashboards** menu, selecting **Server Log Analysis**.



The screenshot shows the 'Server Log' tab of the PEM interface. At the top, there's a header bar with tabs for Properties, SQL, Statistics, Dependencies, Dependents, Monitoring, and Server Log. Below the header, the title 'Postgres Enterprise Manager Host > Postgres Enterprise Manager Server > Server Log' is displayed, along with status information: Object Type Server, Status UP (Since: 27/04/2020, 15:47:09), Generated On 29/04/2020, 10:20:00, No. of alerts 6 (Acknowledged: 0). The main area is titled 'Server Logs' and contains a table with columns: id, Timestamp, User Name, Database Name, Process ID, Session ID, Transaction ID, Connection From, Command, and Message. The table lists several log entries from 27/04/2020 to 29/04/2020, with various command types like COPY, COMMIT, and UPDATE.

id	Timestamp	User Name	Database Name	Process ID	Session ID	Transaction ID	Connection From	Command	Message
1870601	29/04/2020, 10:15:06	agent1	pem	64373	5ea8fb6.fb75	5002111	127.0.0.1:55512	COPY	duration: 0.187 ms statement: BEGIN;COPY pemdata.server_logs(server_id, log_time, user_name, database_name, process_id, connection_from, session_id, session_line_num, command_tag, session_start_time, virtual_transaction_id, transaction_id, error_severity, sql_state_code, message, detail, hint, internal_query, internal_query_pos, context, query, query_pos, location, application_name) FROM STDIN WITH NULL AS 'NULL' QUOTE " CSV;
1870600	29/04/2020, 10:15:06	agent1	pem	64373	5ea8fb6.fb75	0	127.0.0.1:55512	COMMIT	duration: 0.572 ms statement: END;
1870599	29/04/2020, 10:15:06	agent1	pem	64373	5ea8fb6.fb75	5002110	127.0.0.1:55512	UPDATE	duration: 0.129 ms statement: UPDATE pem.log_configuration SET (last_read_filename, file_offset) = ('/var/lib/pgsql/12/data/log/postgresql-2020-04-29_004252.csv', 3135795) WHERE server_id = 1;
1870598	29/04/2020, 10:15:06	agent1	pem	64373	5ea8fb6.fb75	5002110	127.0.0.1:55512	COPY	duration: 0.289 ms statement: BEGIN;COPY pemdata.server_logs(server_id, log_time, user_name, database_name, process_id, connection_from, session_id, session_line_num, command_tag, session_start_time, virtual_transaction_id, transaction_id, error_severity, sql_state_code, message, detail, hint, internal_query, internal_query_pos, context, query, query_pos, location, application_name) FROM STDIN WITH NULL AS 'NULL' QUOTE " CSV;
1870597	29/04/2020, 10:15:06	agent1	pem	64373	5ea8fb6.fb75	0	127.0.0.1:55512	COMMIT	duration: 0.521 ms statement: END;
1870596	29/04/2020, 10:15:06	agent1	pem	64373	5ea8fb6.fb75	5002109	127.0.0.1:55512	UPDATE	duration: 0.132 ms statement: UPDATE pem.log_configuration SET (last_read_filename, file_offset) = ('/var/lib/pgsql/12/data/log/postgresql-2020-04-29_004252.csv', 3134305) WHERE server_id = 1;
1870595	29/04/2020, 10:15:06	agent1	pem	64373	5ea8fb6.fb75	5002109	127.0.0.1:55512	COPY	duration: 0.200 ms statement: BEGIN;COPY pemdata.server_logs(server_id, log_time, user_name, database_name, process_id, connection_from, session_id, session_line_num, command_tag, session_start_time, virtual_transaction_id, transaction_id, error_severity, sql_state_code, message, detail, hint, internal_query, internal_query_pos, context, query, query_pos, location, application_name) FROM STDIN WITH NULL AS 'NULL' QUOTE " CSV;
1870594	29/04/2020, 10:15:06	agent1	pem	64372	5ea8fb6.fb75	0	127.0.0.1:55512	COMMIT	duration: 0.382 ms statement: END;

The header information on the **Server Log Analysis** dashboard displays the date and time that the server was started, the date and time that the page was last updated, and the current number of triggered alerts.

Entries in the **Server Log** table are displayed in chronological order, with the most-recent log entries first. Use the scroll bars to navigate through the log entries, or to view columns that are off of the display.

Headings at the top of the server log table identify the information stored in each column; hover over a column heading to view a tooltip that contains a description of the content of each column.

You can use filtering to limit the number of server log records that are displayed. Click **Show Filters** to expose the filters panel and define a filter.



The screenshot shows the 'Server Log' tab with the 'Show Filters' button highlighted. Below it, a filter definition panel is visible with fields for Start (2020-04-29 09:22:00), End (2020-04-29 10:22:21), User (postgres), Database (pem), and Command type. There are also 'Apply' and 'Reset' buttons.

Use the fields within the **filter definition** box to describe the selection criteria that PEM will use to select a subset of a report for display:

- Use the **From** field to specify a starting date for the displayed server log.
- Use the **To** field to specify an ending date for the displayed server log.
- Enter a role name in the **Username** field display only transactions performed by that user.
- Enter a database name in the **Database field** to specify that the server should limit the displayed records to only those transactions that were performed against the specified database.
- Use the **Command Type** field to specify a selection criteria for the commands that will be displayed in the filtered report.

When you've described the criteria by which you wish to filter the server logs, click **Filter** to display the filtered server log in the **Server Log** table.

Postgres Log Analysis Expert

The PEM Log Analysis Expert analyzes the log files of servers that are registered with Postgres Enterprise Manager, and produces a report that provides an analysis of your Postgres cluster's usage based on log file entries. You can use information on the Log Analysis Expert reports to make decisions about optimizing your cluster usage and configuration to improve performance.

Before using the PEM Log Analysis Expert, you must specify the Service ID on the Advanced tab of the Server Properties dialog, and use the Log Manager wizard to enable log collection by the PEM server.

To open the **Postgres Log Analysis Expert** wizard, select the **Postgres Log Analysis Expert...** option from the **Management** menu of the PEM client. The wizard's **Welcome** dialog opens; click **Next** to continue:



The wizard's **Analyzer selection** dialog displays a list of Analyzers from which you can select. Each Analyzer generates a corresponding table, chart, or graph that contains information gleaned from the log files.



Check the box to the left of an Analyzer to indicate that the Log Analysis Expert should prepare the corresponding table, chart or graph. After making your selections, click **Next** to continue to the Server selection tree control.



Use the tree control to specify which servers you would like the Postgres Log Analysis Expert to analyze. If you select multiple servers, the resulting report will contain the corresponding result set for each server in a separate (but continuous) list. Click **Next** to continue to the Report options dialog.



Use the fields in the **Options** section to specify the analysis method and the maximum length of any resulting tables:

- Use the **Aggregate method** drop-down to select the method used by the Log Analysis Expert to consolidate data for the selected time span. You can select from:
 - **SUM** instructs the analyzer to calculate a value that is the sum of the collected values for the specified time span.
 - **AVG** instructs the analyzer to calculate a value that is the average of the collected values for the specified time span.
 - **MAX** instructs the analyzer to use the maximum value that occurs within a specified time span.
 - **MIN** instructs the analyzer to use the minimum value that occurs within a specified time span.
- Use the **Time span** field to specify the number of minutes that the analyzer will incorporate into each calculation for a point on a graph. For example, if the Time span is 5 minutes, and the Aggregate method is AVG, each point on the given graph will contain the average value of the activity that occurred within a five minute time span.
- Use the **Rows limit** field to specify the maximum number of rows to include in a table.

Use the fields in the **Time Intervals** section to specify the time range that the Log Analysis Expert will analyze:

- Set **Relative days** to Yes to enable the (+/-)From date field and specify the number of days before or after the date and time selected in the From field.
- Use the **From** field to specify the starting date and time for the analysis.
- Use the **To** field to specify the ending date and time for the analysis.
- Use the **(+/-)** From date selector to specify the number of days before or after the From date that should be included in the analysis.

When you've specified the report options, click **Next** to continue to the Report destination dialog.



You can choose the default option and select **Finish** to view the Log Analysis Expert report in the PEM client's tabbed browser, or click the radio button next to Download the report to save a copy of the report to an HTML file for later use.

If you have specified that the report should be saved to a file, the report will be downloaded.

Reviewing the Postgres Log Analysis Expert Report

If you've elected to review the report immediately, the Postgres Log Analysis Expert report will be displayed in the PEM Client window. The report header displays the date and time that the report was generated, the time period that the report spans, and the aggregation method specified when defining the report. The name of the server for which information is displayed is noted at the start of each section of the report.

The report displays the tables, graphs and charts that were selected in the Log Analysis Expert wizard. Use the **Jump To** button (located in the lower-right hand corner of the screen) to navigate to a specific graphic.

The screenshot shows the 'Postgres Log Analysis Expert' interface. At the top, it displays the analysis period (Interval: 2020-04-20 15:55:19 - 2020-04-27 15:55:19), generation time (Generated: 2020-04-27 16:00:51), span (Span: 5 Minutes), aggregate (Aggregate: SUM), and a 'Go to:' dropdown set to 'Postgres Enterprise Manager Server'. Below this is a tree view node 'Postgres Enterprise Manager Server(192.168.1.18:5432)'. Under this node, there are two sections: 'Summary Statistics' and 'Hourly DML Statistics'.

Summary Statistics:

Settings	Values
Number of unique queries	151649
Total queries	155045
Total queries duration	
First query	27/04/2020 15:47:09.86 IST
Last query	27/04/2020 15:53:59.611 IST
Queries peak time	27/04/2020 15:49:38 IST queries 2821
Number of events	155045
Number of unique events	1
Total number of sessions	348
Total duration of sessions	
Average sessions duration	
Total number of connections	0
Total number of databases	0

Hourly DML Statistics:

Time	Database name	Statement	Count	Min duration	Max duration	Avg duration
27/04/2020 15:00	db01	SELECT	80	0.05	105.40	4.90
27/04/2020 15:00	edbstore_temp	SELECT	58	0.02	66.58	4.64
27/04/2020 15:00	hr	SELECT	48	0.01	29.26	1.87
27/04/2020 15:00	pem	COPY	1641	0.65	43.81	2.08
27/04/2020 15:00	pem	DELETE	73	0.44	8.74	1.04
27/04/2020 15:00	pem	INSERT	190	0.06	9.50	2.19

If the report contains an analysis of more than one monitored server, charts and tables will be displayed in sets; first the graphs, tables and charts that display statistics for one server, then the graphics for the next server in the report.

4.8 SQL Profiling and Analysis

Most RDBMS experts agree that inefficient SQL code is the leading cause of most database performance problems. The challenge for DBAs and developers is to locate the poorly-running SQL code in large and complex systems, and then optimize that code for better performance.

The SQL Profiler component allows a database superuser to locate and optimize poorly-running SQL code. Users of Microsoft SQL Server's Profiler will find PEM's SQL Profiler very similar in operation and capabilities. SQL Profiler is installed with each Advanced Server instance; if you are using PostgreSQL, you must download the SQL Profiler installer, and install the SQL Profiler product into each managed database instance you wish to profile.

For each database monitored by SQL Profiler, you must:

1. Edit the `postgresql.conf` file; you must include the SQL Profiler library in the `shared_preload_libraries` configuration parameter.

For Linux installations, the parameter value should include:

```
$libdir/sql-profiler
```

on Windows, the parameter value should include:

```
$libdir/sql-profiler.dll
```

2. Create the functions used by SQL Profiler in your database. The SQL Profiler installation program places a SQL script (named `sql-profiler.sql`) in the `share/postgresql/contrib` subdirectory of the main PostgreSQL installation

directory on Linux systems. On Windows systems, this script is located in the **share** subdirectory. You must invoke this script on the maintenance database specified when registering the server with PEM.

3. Stop and re-start the server for the changes to take effect.

Please note: if you have connected to the PEM server with the PEM client before configuring SQL Profiler, you must disconnect and reconnect with the server to enable SQL Profiler functionality. For more detailed information about installing and configuring the SQL Profiler plugin, please refer to the PEM Installation Guide, available from the EDB website at:

<http://enterprisedb.com/products-services-training/products/documentation>

Creating a New SQL Trace

SQL Profiler captures and displays a specific SQL workload for analysis in a SQL trace. You can start and review captured SQL traces immediately, or save captured traces for review at a later time. You can use SQL Profiler to create and store up to 15 named traces; use menu options to create and manage traces.

Creating a Trace

You can use the **Create trace...** dialog to define a SQL Trace for any database on which SQL Profiler has been installed and configured. To access the dialog, highlight the name of the database in the PEM client tree control; navigate through the Management menu to the SQL Profiler pull-aside menu, and select Create trace....



Use the fields on the **Trace options** tab to specify details about the new trace:

- Provide a name for the trace in the Name field.
- Click in the **User filter** field to specify the roles whose queries will be included the trace; optionally, check the box next to Select All to include queries from all roles.
- Click in the **Database filter** field to specify which databases to trace; optionally, check the box next to Select All to include queries against all databases.
- Specify a **trace size in the Maximum Trace File Size** field; SQL Profiler will terminate the trace when it reaches approximately the size specified.
- Specify Yes in the **Run Now** field to start the trace when you select the Create button; select No to enable fields on the Schedule tab.



Use the fields on the **Schedule** tab to specify scheduling details for the new trace:

- Use the **Start time** field to specify the starting time for the trace.
- Use the **End time** field to specify the ending time for the trace.
- Specify Yes in the **Repeat?** field to indicate that the trace should be repeated every day at the times specified; select No to enable fields on the Periodic job options tab.



Fields on the **Periodic job options** tab specify scheduling details about a recurring trace. Use fields in the Days section to specify the days on which the job will execute:

- Click in the **Week days** field to select the days of the week on which the trace will execute.
- Click in the **Month days** field to select the days of the month on which the trace will execute.
- Click in the **Months** field to select the months in which the trace will execute.

Use fields in the **Times** section to specify a time schedule for the trace execution:

- Click in the **Hours** field to select the hours at which the trace will execute.
- Click in the **Minutes** field to select the hours at which the trace will execute.

When you've completed the **Create trace...** dialog, click **Create** to start the newly defined trace or to schedule the trace for a later time.

The screenshot shows the SQL Profiler interface with the 'Trace: test_trace' tab selected. The main area displays a table of trace events. The columns are: #, Start Time, Duration (ms), Query, Rows Affected, User, Database, PID, File System Read, File System Write, and Page Fault. The table contains numerous rows of event data.

If you elect to execute the trace immediately, the trace results will display in the PEM client.

Opening an Existing Trace

To view a previous trace, highlight the name of the profiled database in the PEM client tree control; navigate through the **Management** menu to the SQL Profiler pull-aside menu, and select **Open trace....**. You can also use the **SQL Profiler toolbar** menu to open a trace; select the **Open trace...** option. The Open trace... dialog opens.



Highlight an entry in the trace list and click Open to open the selected trace. The selected trace opens in the SQL Profiler tab.

Filtering a Trace

A filter is a named set of (one or more) rules, each of which can hide events from the trace view. When you apply a filter to a trace, the hidden events are not removed from the trace, but are merely excluded from the display.

Click the Filter icon to open the **Trace Filter** dialog and create a rule (or set of rules) that define a filter. Each rule will screen the events within the current trace based on the identity of the role that invoked the event, or the query type invoked during the event.

To open an existing filter, select the **Open** button; to define a new filter, click the **Add (+)** icon to add a row to the table displayed on the General tab and provide rule details:

- Use the **Type** drop-down listbox to specify the trace field that the filter rule will apply to.
- Use the **Condition** drop-down listbox to specify the type of operator that SQL Profiler will apply to the Value when it filters the trace:

- Select **Matches** to filter events that contain the specified Value.
 - Select **Does not match** to filter events that do not contain the specified Value.
 - Select **Is equal** to to filter events that contain an exact match to the string specified in the Value field.
 - Select **Is not equal** to to filter events that do not contain an exact match to the string specified in the Value field.
 - Select **Starts with** to filter events that begin with the string specified in the Value field.
 - Select **Does not start with** to filter events that do not begin with the string specified in the Value field.
 - Select **Less than** to filter events that have a numeric value less than the number specified in the Value field.
 - Select **Greater than** to filter events that have a numeric value greater than the number specified in the Value field.
 - Select **Less than or equal to** to filter events that have a numeric value less than or equal to the number specified in the Value field.
 - Select **Greater than or equal to** to filter events that have a numeric value greater than or equal to the number specified in the Value field.
- Use the **Value** field to specify the string, number or regular expression that SQL Profiler will search for.

When you've finished defining a rule, click the Add (+) icon to add another rule to the filter. To delete a rule from a filter, highlight the rule and click the Delete icon.

Click the **Save** button to save the filter definition to a file without applying the filter; to apply the filter, click **OK**. Select **Cancel** to exit the dialog and discard any changes to the filter.

Deleting a Trace

To delete a trace, highlight the name of the profiled database in the PEM client tree control; navigate through the **Management** menu to the SQL Profiler pull-aside menu, and select **Delete trace(s)...**. You can also use the SQL Profiler toolbar menu to delete a trace; select the **Delete trace(s)...** option. The **Delete traces** dialog opens.



Click the icon to the left of a trace name to mark one or more traces for deletion and click **Delete**. The PEM client will acknowledge that the selected traces have been deleted.

Viewing Scheduled Traces

To view a list of scheduled traces, highlight the name of the profiled database in the PEM client tree control; navigate through the **Management** menu to the SQL Profiler pull-aside menu, and select **Scheduled traces...**. You can also use the SQL Profiler toolbar menu to the list; select the **Scheduled traces...** option.



The **Scheduled traces...** dialog displays a list of the traces that are awaiting execution. Click the edit button to the left of a trace name to access detailed information about the trace:

- The **Status** field lists the status of the current trace.
- The **Enabled?** switch displays Yes if the trace is enabled; No if it is disabled.
- The **Name** field displays the name of the trace.
- The **Agent** field displays the name of the agent responsible for executing the trace.
- The **Last run** field displays the date and time of the last execution of the trace.
- The **Next run** field displays the date and time of the next scheduled trace.
- The **Created** field displays the date and time that the trace was defined.

Using the Index Advisor

Index Advisor is distributed with Advanced Server 9.0 and above. Index Advisor works with SQL Profiler by examining collected SQL statements and making indexing recommendations for any underlying tables to improve SQL response time. The Index Advisor works on all DML (INSERT, UPDATE, DELETE) and SELECT statements that are invoked by a superuser.

Diagnostic output from the Index Advisor includes:

- Forecasted performance benefits from any recommended indexes
- The predicted size of any recommended indexes
- DDL statements you can use to create the recommended indexes

Before using Index Advisor, you must:

1. Modify the **postgresql.conf** file on each Advanced Server host, adding the `index_advisor` library to the `shared_preload_libraries` parameter.
2. Install the **Index Advisor contrib** module. To install the module, use the `psql` client or PEM Query Tool to connect to the database, and invoke the following command:

```
\i <complete_path>/share/contrib/index_advisor.sql
```

3. Restart the server for your changes to take effect.

Index Advisor can make indexing recommendations based on trace data captured by SQL Profiler. Simply highlight one or more queries in the **SQL Profiler Trace Data** pane, and click the **Index Advisor** toolbar button (or select Index Advisor from the View menu). For detailed usage information about Index Advisor, please see the EDB Postgres Advanced Server Guide.

Please note: Index Advisor cannot analyze statements invoked by a non-superuser. If you attempt to analyze statements

invoked by a non-superuser, the server log will include the following error:

```
ERROR: access to library "index_advisor" is not allowed
```

For more information about configuring and using Index Advisor, please see the EDB Postgres Advanced Server Guide, available from EDB at:

<https://www.enterprisedb.com/docs>

4.9 Tuning Wizard

The Tuning Wizard reviews your PostgreSQL or Advanced Server installation, and recommends a set of configuration options that will help tune the installation to best suit its anticipated workload. Please note that benchmarking systems or systems with a high work load may require additional manual tuning to reach optimum performance.

Before using the Tuning Wizard, you must specify the name of the service in the Service ID field on the Advanced tab of the server's Properties dialog. PEM will use the service name when restarting the service after tuning.

The Tuning Wizard can only make recommendations for those servers that reside on the same server as their bound PEM agent. If you have specified a value of Yes in the Remote monitoring field when defining your server, the server will not be displayed in the Tuning Wizard tree control.

To open the Tuning Wizard, select **Tuning Wizard...** from the **Management** menu of the PEM client. The Tuning Wizard opens, welcoming you.



Click **Next** to continue to the server selection dialog.



Expand the **Servers** node of the tree control to view a list of the servers that are currently monitored by PEM that are available for tuning. Check a box to the left of a server name to select the server for tuning.

!!! Note The Tuning Wizard displays a red warning symbol to the left of a server name in the tree control if the service name for that server is not provided on the server's Properties dialog.

Click **Next** to continue to the Configuration dialog.



Select an option in the **Machine utilization** field to specify the type of work performed by the selected servers. The type of work performed by the server determines how the tuning wizard will allocate system resources:

- Select **Dedicated** to dedicate the majority of the system resources to the database server.
- Select **Mixed use** to dedicate a moderate amount of system resources to the database server.
- Select **Developer workstation** to dedicate a relatively small amount of system resources to the database server.

Select an option in the **Workload Selection** field to specify the type of workload typically performed on the selected server:

- Select **OLTP** if the selected server is used primarily to process online transaction workloads.
- Select **Mixed** if the selected server provides a mix of transaction processing and data reporting.
- Select **Data warehouse** if the server is used for heavy data reporting.

Click **Next** to continue to the **Tuning Changes Summary** dialog.



The tree control on the **Tuning Changes Summary** dialog displays the parameter setting modifications recommended for each server analyzed by the Tuning Wizard. Use the checkboxes next to a server or parameter name to select the recommendations that tuning wizard will either include in a preview report or apply:

- A checked box to the left of a parameter name specifies that the Tuning Wizard will include the parameter setting.
- A checked box to the left of a server name specifies that the Tuning Wizard will include all parameter setting recommendations for the specified server.

Specify which Tuning Wizard recommendations you wish to include in a report or apply, and click **Next** to continue.

Use the **Schedule or Run?** dialog to either specify a time that PEM will apply the changes, or generate a report that details the recommended changes.

The selected actions will apply to all of the changes noted on the Tuning Changes Summary. If you opt to generate a report, PEM will create a report that contains a list of the current values and recommended modifications to the configuration parameters selected on the Tuning Changes Summary dialog. Note that to implement changes, you will need to invoke the Tuning Wizard a second time, specifying the parameters you wish to modify on the **Tuning Changes Summary** dialog.

Select **Schedule changes** to view and specify your scheduling options.



You can:

- Set the **Configuration now?** slider to **Yes** to apply the tuning wizard's recommendations and restart the server now.
- Set the **Configuration now?** slider to **No** to enable the **Time?** field and use the calendar selector to specify a time for PEM to apply the tuning wizard's recommendations and restart the server. Note that if you schedule a time for the changes to be applied, you will not be provided with a preview of the change recommendations.

Select **Generate report** to view your report options.



You can:

- Set the **View report now?** slider to **Yes** to display the Tuning Wizard report onscreen.
- Set the **View report now?** slider to **No** to enable the **Save the report to file** field and use the calendar selector to specify a file name and location to which PEM will write the Tuning Wizard report.

Click the **Finish** button to either apply the Tuning Wizard's modifications or generate a report and exit the Tuning Wizard.

GUC Parameter	Original Value	Recommended Value
effective_cache_size	4096MB	943MB
maintenance_work_mem	64MB	66MB
random_page_cost	4	2
shared_buffers	128MB	503MB
wal_buffers	4MB	16MB
work_mem	4MB	7MB

You can confirm that Tuning Wizard has implemented the recommended changes by reviewing the **postgresql.conf** file for the modified server. The Tuning Wizard adds a comment above each modified parameter in the **postgresql.conf** file when the change is applied.



```

root@localhost:/opt/PostgresPlus/9.5AS/data
File Edit View Search Terminal Help
# The value for shared_buffers was changed by the Postgres Enterprise Manager Tuning Wizard at 2017-03-06 09:15:32
#shared_buffers = 196MB
# The value for shared_buffers was changed by the Postgres Enterprise Manager Tuning Wizard at 2017-03-06 09:18:23
shared_buffers = 196MB
    # (change requires restart)
    # on, off, or try
    # (change requires restart)
#temp_buffers = 8MB
    # min 800KB
#max_prepared_transactions = 0
    # zero disables the feature
    # (change requires restart)
# Note: Increasing max_prepared_transactions costs ~600 bytes of shared memory
# per transaction slot, plus lock space (see max_locks_per_transaction).
# It is not advisable to set max_prepared_transactions nonzero unless you
# actively intend to use prepared transactions.
#work_mem = 4MB
    # min 64kB
# The value for work_mem was changed by the Postgres Enterprise Manager Tuning Wizard at 2017-03-06 09:15:32
#work_mem = 3MB
# The value for work_mem was changed by the Postgres Enterprise Manager Tuning Wizard at 2017-03-06 09:18:23
work_mem = 3MB
#maintenance_work_mem = 64MB
    # min 1MB
# The value for maintenance_work_mem was changed by the Postgres Enterprise Manager Tuning Wizard at 2017-03-06 09:15:32
#maintenance_work_mem = 25MB
--More-- (21%)

```

You can also confirm a parameter value by querying the server. For example, to confirm the value of the `shared_buffers` parameter, open a SQL command line using either the Query Tool (accessed through the Tools menu) or the `psql` client, and issue the command:

```
SHOW shared_buffers;
```

The value returned by the server will confirm that the parameter has been modified.

4.10 Postgres Expert - Best Practice Enforcement

The Postgres Expert utility provides expert advice on how to best configure your Postgres servers for optimal performance, security, and more. Postgres Expert serves as a PostgreSQL 'DBA in a box' by analyzing your servers for deviations in best practices. Postgres Expert contains three specialized Experts:

- The Configuration Expert.
- The Schema Expert.
- The Security Expert.

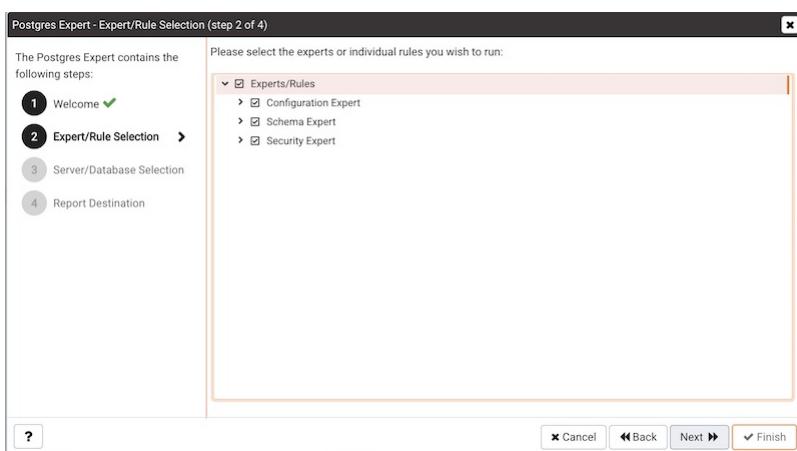
You can select specific rules for each Expert to analyze, or accept all rules, and then review Postgres Expert reports detailing any best practice issues that require your attention.

Using the Postgres Expert Wizard

To use the Postgres Expert wizard select the `Postgres Expert` option from the `Management` menu in the PEM client. When the wizard's `Welcome` window opens, click `Next` to continue.



The wizard displays a tree control that allows you to choose the **Experts and Rules** with which Postgres Expert will evaluate the specified server or database.



The tree control categorizes the rules under three Expert headings:

- Select from the **Configuration Expert rules** to analyze the parameter settings of the server or operating system to find any adjustments that might improve system performance.
- Select from the **Schema Expert rules** to analyze schema objects (locating missing primary keys, foreign keys without indexes, etc).
- Select from the **Security Expert rules** to review the system to find security vulnerabilities.

Use the checkmark indicator to the left of an expert or rule to indicate that the Postgres Expert should analyze the configuration of the selected servers for any best practice deviations related to the checked item.

You can:

- Check the box next to the name of an expert to select or deselect all of the configuration items listed under that node of the tree control.
- Check the box next to **Servers/Databases** to instruct Postgres Expert to review the selected server for all of the items in the tree control.
- Deselect the box next to **Servers/Databases** to un-check all of the rules; then, navigate through the tree control, specifying only the items that you wish Postgres Expert to evaluate.

After making your selections, click **Next** to continue to the Server/Databases tree control.



Select or de-select the servers and databases that you would like Postgres Expert to analyze. If you select multiple servers or databases, the resulting report will contain a separate analysis of each target. When you've finished, click **Next** to select a report destination.



You can select the default option and click **Finish** to view an onscreen report from Postgres Expert, or check the box next to Download the report to save a copy of the report to an HTML file for later use. If you choose to save the report to a file, the download will begin immediately. The file will be saved in your default download directory.

Reviewing Postgres Expert Recommendations

Postgres Expert produces an easily navigated report that contains an analysis of the selected rules, categorized by high, medium, and low severities, for the selected servers.

Rule	Database	Severity
> Check checkpoint_completion_target	-	Medium
> Check effective_cache_size	-	Medium
> Check effective_io_concurrency	-	Low
> Check reducing random_page_cost	-	Low

Rule	Database	Severity
> Check data and transaction log on same drive	-	High
> Check for missing foreign key indexes	db01	Medium

The report header contains a summary of the report, and includes the date and time that the report was generated, the number of rules analyzed, and the number of deviations from best practices found by Postgres Expert. Use the [Jump to](#) drop-down listbox to select a server to navigate to the section of the report that targets recommendations for that server.

The body of the report contains the detailed findings for each server selected for analysis. The findings are sorted by Expert; within each Expert heading, any rule violations are ranked by Severity.

The screenshot shows the 'Postgres Expert Report' interface. At the top, it displays 'Generated On: 2020-04-27 16:57:47' and a 'Go to:' dropdown set to 'Postgres Enterprise Manager Server'. Below this is a 'Summary' section with metrics: Servers Tested: 1, Rules Checked: 31, High Alerts: 1, Medium Alerts: 3, Low Alerts: 2. A tree view shows 'Server: Postgres Enterprise Manager Server (192.168.1.18:5432)'. Under 'Advisor: Configuration Expert', there's a table for 'Check checkpoint_completion_target' with columns 'Rule', 'Database', and 'Severity'. The rule has a trigger of 'checkpoint_completion_target != 0.9', a medium severity, and a description explaining the purpose of checkpoints. Other rules listed are 'Check effective_cache_size', 'Check effective_ioConcurrency', and 'Check reducing random_page_cost', all with low severity. Under 'Advisor: Schema Expert', there's a table for 'Check data and transaction log on same drive' and 'Check for missing foreign key indexes', both with medium severity.

Click on each rule in the Postgres Expert report to display details and recommendations for that rule. Within each rule, section headings display:

- The **Advisor** section lists the name of the Postgres Expert advisor that prompted the recommendation.
- The **Trigger** section displays a description of the rule that raised the alert.
- The **Recommended Value** section displays the value to which Postgres Expert recommends setting the selected parameter.
- The **Description** section displays information and advice about the parameter that caused the alert.
- The **Current Values** section displays the current value(s) of any parameter(s) that influence the Postgres Expert's evaluation.

4.11 Reports

You can generate the System Configuration report and Core Usage report for all locally and remotely managed servers. To generate this report, select **Reports** from the **Management** Menu.

Reports has following options:

- System Configuration Report (JSON)
- System Configuration Report (HTML)
- Core Usage Report (JSON)
- Core Usage Report (HTML)

Please note that only superusers or the users with the pem_admin role permission can download the System Configuration or Core Usage reports.

Also note that information in these reports will reflect the latest probe run time.

System Configuration Report

The System Configuration Report provides detailed information about the PEM Agents group, PEM Server directory group and custom groups listed under browser tree. These groups can contain Postgres Enterprise Manager, PEM Agents and Database servers. You can download this report in HTML as well as in JSON format.

The **Postgres Enterprise Manager Summary** provides details about:

- The Postgres Enterprise Manager backend database server version
- Application version
- User name accessing the application
- Python version
- Flask version
- Platform specific information

The **Summary** provides information about the number of agents and servers.

The screenshot shows the 'System Configuration Report' interface. At the top, it displays the generation date and time: 'Generated On: 2020-04-28 14:30:49'. A 'Go to:' dropdown menu is set to 'PEM Agents'. Below this, there are two expandable sections: 'Postgres Enterprise Manager Summary' and 'Summary'.

Postgres Enterprise Manager Summary:

Parameter	Value												
Name	Postgres Enterprise Manager												
Backend version	PostgreSQL 12.1 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-39), 64-bit												
App version	7.14.0-dev (schema: 202003031)												
User	postgres												
Python version	3.7.5												
Flask version	1.0.4												
Platform	<table border="1"> <tr> <td>System</td> <td>: Darwin</td> </tr> <tr> <td>Node</td> <td>: Laptop358.bn.in</td> </tr> <tr> <td>Release</td> <td>: 18.7.0</td> </tr> <tr> <td>Version</td> <td>: Darwin Kernel Version 18.7.0: Thu Jan 23 06:52:12 PST 2020; root:xnu-4903.278.25~1RELEASE_X86_64</td> </tr> <tr> <td>Machine</td> <td>: x86_64</td> </tr> <tr> <td>Processor</td> <td>: i386</td> </tr> </table>	System	: Darwin	Node	: Laptop358.bn.in	Release	: 18.7.0	Version	: Darwin Kernel Version 18.7.0: Thu Jan 23 06:52:12 PST 2020; root:xnu-4903.278.25~1RELEASE_X86_64	Machine	: x86_64	Processor	: i386
System	: Darwin												
Node	: Laptop358.bn.in												
Release	: 18.7.0												
Version	: Darwin Kernel Version 18.7.0: Thu Jan 23 06:52:12 PST 2020; root:xnu-4903.278.25~1RELEASE_X86_64												
Machine	: x86_64												
Processor	: i386												

Summary:

Parameter	Value												
Agents	<table border="1"> <tr> <td>Windows</td> <td>: 0</td> </tr> <tr> <td>Linux</td> <td>: 2</td> </tr> </table>	Windows	: 0	Linux	: 2								
Windows	: 0												
Linux	: 2												
Servers	<table border="1"> <tr> <td>PG</td> <td>: 2</td> </tr> <tr> <td>EPAS</td> <td>: 2</td> </tr> <tr> <td>Unknown</td> <td>: 0</td> </tr> <tr> <td>Locally Managed</td> <td>: 3</td> </tr> <tr> <td>Remotely Managed</td> <td>: 1</td> </tr> <tr> <td>Unmanaged</td> <td>: 0</td> </tr> </table>	PG	: 2	EPAS	: 2	Unknown	: 0	Locally Managed	: 3	Remotely Managed	: 1	Unmanaged	: 0
PG	: 2												
EPAS	: 2												
Unknown	: 0												
Locally Managed	: 3												
Remotely Managed	: 1												
Unmanaged	: 0												

At the bottom left, there are two links: 'Group: PEM Agents' and 'Group: PEM Server Directory'.

The **Group: PEM Agents** panel provides details about the PEM agent, CPU cores, Disk Utilization, and Memory information.

System Configuration Report

Generated On: 2020-04-28 14:30:49

Go to: PEM Agents

> Postgres Enterprise Manager Summary

> Summary

> Group: PEM Agents

Agent: Postgres Enterprise Manager Host

> Agent Details

Parameter	Value
Platform	Linux-x64
OS	CentOS Linux release 7.5.1804 (Core)
Version	7.14.0-dev
Active	True
Hostname	localhost.localdomain
Domain Name	(none)
Bound Local Servers	<ul style="list-style-type: none"> » Postgres Enterprise Manager Server » EDB Postgres Advanced Server 11 » EPAS_12
Bound Remote Servers	(none)

> CPU

Total CPU Cores: 2

Average CPU Utilization (%): 25.31

Core ID	Load Percentage
CPU0	25.267327
CPU1	25.353135

> Disk Utilization

Total Disk Size (MB): 32098

Disk Space Used (MB): 13791

Disk Space Available (MB): 16669

Disk Utilization (%): 42.97

Mount Point	File System	Size (MB)	Space Used (MB)	Space Available (MB)
/	/dev/sda3	31622	13657	16352
/boot	/dev/sda1	476	134	317

> Memory Details

Parameter	Value
Free RAM (MB)	1050
Memory Usage Percentage	72.17
Total Swap Memory (MB)	7999
Free Swap Memory (MB)	6589
Swap Usage Percentage	17.63

Agent: localhost.localdomain

> Agent Details

> CPU

> Disk Utilization

> Memory Details

> Group: PEM Server Directory

The **Group: PEM Server Directory**, provides details about:

- Database server version
- Host
- Port
- Database name
- Database size
- Tablespace size

System Configuration Report

Generated On: 2020-04-28 14:30:49 Go to: PEM Agents

> Postgres Enterprise Manager Summary

> Summary

> Group: PEM Agents

> Group: PEM Server Directory

Server: Postgres Enterprise Manager Server

Server Details

Parameter	Value
Agent	Postgres Enterprise Manager Host
Host	192.168.1.19
Port	5432
Database	postgres
Version	PostgreSQL 12.1 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-39), 64-bit
Service Id	postgresql-12
Remote Monitored?	False
Active	True

Database Details

Name	Size (MB)	Tablespace Name
postgres	8	pg_default
edbstore_temp	8	pg_default
hr	8	pg_default
pem	2407	pg_default
testdb	27	pg_default
db01	8	pg_default

Tablespace Details

Name	Size (MB)
pg_global	0
pg_default	2482

Server: EPAS_12

> Server Details

> Database Details

> Tablespace Details

Server: EDB Postgres Advanced Server 11

> Server Details

> Database Details

> Tablespace Details

Server: PGSQL12_Centos7_1

> Server Details

> Database Details

> Tablespace Details

Please note that here Group Server Name depends on the group name to which the server is added.

Core Usage Report

The Core Usage report provides detailed information about number of cores specific to:

- The server type
 - Database version
 - Platform and group name

The report also gives detailed information about locally managed servers:

- Type
 - Host
 - Port
 - Platform
 - Cores
 - RAM

Core Usage Report

Generated On: 2020-02-18 16:08:47 IST Using: Postgres Enterprise Manager Version: 7.13.0-dev (schema: 202001011)

Core Summary

Total Number of Cores: 13

Server Type	Number of Servers	Number of Cores
EDB Postgres Advanced Server	1	4
PostgreSQL	2	5
BART	1	4

Database Version	Number of Servers	Number of Cores
PostgreSQL 10	1	4
PostgreSQL 11	1	1
Advanced Server 11	1	4

Platform	Number of Servers	Number of Cores
Windows-x64	1	1
Linux-x64	3	12

Group Name	Number of Servers	Number of Cores
PEM Server Directory	3	9

Server Core Summary

Locally Managed Servers: 3

Name	Type	Host:Port	Platform	Cores	Total RAM (MB)
Windows PostgreSQL 11	PostgreSQL	127.0.0.1:5432	Windows-x64	1	2047
PostgreSQL-10-Local	PostgreSQL	localhost:5432	Linux-x64	4	5786
Postgres Enterprise Manager Server	EDB Postgres Advanced Server	127.0.0.1:5444	Linux-x64	4	5786
				9	13619

Remotely Managed Servers: 1

Name	Type	Host:Port
PostgreSQL-11-Remote	PostgreSQL	172.19.12.3:5432

Unmanaged Servers: 1

Name	Host:Port
Performance Diagnostics Server	172.16.254.22: 5444

4.12 Monitoring Failover Manager

If you are using EDB Failover Manager to monitor your replication scenario, you must manually install and configure Failover Manager. For detailed information about installing Failover Manager, visit the EDB website at:

<https://www.enterprisedb.com/products/edb-postgres-platform/edb-postgres-failover-manager>

To monitor the status of a Failover Manager cluster on the Streaming Replication dashboard, you must provide the following information on the **Advanced** tab of the **server Properties** dialog for each node of the cluster:

- Use the **EFM Cluster Name** field to specify the name of the Failover Manager cluster. The cluster name is the prefix of the name of the cluster properties file. For example, if your cluster properties file is named efm.properties, your cluster name is efm.
- Use the **EFM Installation Path** field to specify the location of the Failover Manager binary file. By default, the Failover Manager binary file is installed in `/usr/efm-<X>/bin`.

Where `<X>` is the EFM Version.

After registering your servers, the **Streaming Replication Analysis** dashboard will display status information about your EFM cluster near the bottom of the dashboard.

The **Failover Manager Cluster Status** section of the Streaming Replication Analysis dashboard displays information about the monitored cluster:

The **Failover Manager Cluster Information** table provides information about the Failover Manager cluster:

- The **Properties** column displays the name of the cluster property.
- The **Values** column displays the current value of the property.

The **Failover Manager Node Status** table displays information about each node of the Failover Manager cluster:

- The **Agent Type** column displays the type of agent that resides on the node; the possible values are Primary, Replica, Witness, Idle, and Promoting.
- The **Address** column displays the IP address of the node.
- The **Agent** column displays the status of the agent that resides on the node.
- The **DB** column displays the status of the database that resides on the node.
- The **XLog Location** column displays the transaction log location of the database.
- The **Status Information** column displays any error-related information about the node.
- The **XLog Information** column displays any error-related information about the transaction log.
- The **VIP** column displays the VIP address that is associated with the node.
- The **VIP Status** column displays True if the VIP is active for the node, False if the VIP is not.

Replacing a Primary Node

You can use the PEM client to replace the Primary node of a Failover Manager cluster with a replica node. To initiate the failover process, select **Replace Cluster Primary** from **Server** under the **Tools** menu. A dialog opens, asking you to confirm that you wish to replace the current primary node.



Select **Yes** to remove the current primary node from the Failover Manager cluster and promote a replica node to the role of read/write primary node within a Failover Manager cluster. The node with the highest promotion priority (defined in Failover Manager) will become the new primary node. PEM will display a dialog, reporting the job status.



When the job completes and the [Streaming Replication Analysis](#) dashboard refreshes, you can review the [Failover Manager Node Status](#) table to confirm that a replica node has been promoted to the role of primary within the Failover Manager cluster.

Switchover EFM Cluster

You can use the PEM client to replace the primary node of a Failover Manager cluster with a replica node. To initiate the switchover process, select [Switchover EFM Cluster](#) from the [Tools](#) menu. A dialog opens, asking you to confirm that you wish to switchover EFM cluster.



Select [Yes](#) to start the Failover Manager switchover, and promote a replica node to the role of read/write primary node and reconfigure the primary database as a new replica within a Failover Manager cluster. The node with the highest promotion priority (defined in Failover Manager) will become the new primary node. PEM will display a dialog, reporting the job status.



When the job completes and the [Streaming Replication Analysis](#) dashboard refreshes, you can review the [Failover Manager Node Status](#) table to confirm that a switchover occurred.

4.13 Monitoring an xDB Replication Cluster

Before configuring PEM to retrieve statistics from an Advanced Server or PostgreSQL database that is part of an xDB replication scenario, you must manually install and configure xDB Replication. For more information about xDB replication solutions and documentation, please visit the EDB website at:

<http://www.enterprisedb.com/products-services-training/products-overview/xdb-replication-server-multi-master>

The PEM xDB Replication probe monitors lag data for clusters that use xDB multi-primary or single-primary replication that have a publication database that is an EDB Postgres Advanced Server or PostgreSQL database. Please note that if you have configured replication between other proprietary database hosts (i.e. Oracle or SQL Server) and Advanced Server or PostgreSQL, the probe cannot return lag information.

The screenshot shows the 'Manage Probes' tab of the EDB Postgres Enterprise Manager interface. The tab includes sections for 'Description', 'Quick Links' (Manage Custom Probes, Copy Probes, Help), and 'Probes'. The 'Probes' section displays a table with columns for Probe name, Execution Frequency (Default? Minutes Seconds), Enabled? (Default? Probe Enable?), and Data Retention (Default? Days). The table lists various probes such as Background Writer Statistics, Blocked Session Information, Data and Log File Analysis, Database Frozen XID, Database Size, Database Statistics, Failover Manager Cluster Info, Failover Manager Node Status, Lock Information, Number of Prepared Transactions, Number of WAL Files, Object Catalog: Database, Object Catalog: Tablespace, PG HBA Conf, Server Information, Server log Configuration, Session Information, Settings, Streaming Replication, Streaming Replication Database Conflicts, Streaming Replication Lag Time, Tablespace Size, User Information, and WAL Archive Status. Most probes have a default execution frequency of 5 minutes and 0 seconds, and a data retention of 180 days. Some probes like 'Streaming Replication' and 'Streaming Replication Lag Time' have a default frequency of 5 minutes and 0 seconds, but a probe enable setting of 'No'.

Probe name	Execution Frequency			Enabled?		Data Retention	
	Default?	Minutes	Seconds	Default?	Probe Enable?	Default?	Days
Background Writer Statistics	Yes	5	0	Yes	Yes	Yes	180
Blocked Session Information	Yes	5	0	Yes	Yes	Yes	180
Data and Log File Analysis	Yes	0	10	Yes	Yes	Yes	180
Database Frozen XID	Yes	720	0	Yes	Yes	Yes	180
Database Size	Yes	30	0	Yes	Yes	Yes	180
Database Statistics	Yes	30	0	Yes	Yes	Yes	90
Failover Manager Cluster Info	Yes	5	0	Yes	No	Yes	7
Failover Manager Node Status	Yes	5	0	Yes	No	Yes	7
Lock Information	Yes	5	0	Yes	Yes	Yes	180
Number of Prepared Transactions	Yes	5	0	Yes	Yes	Yes	180
Number of WAL Files	Yes	5	0	Yes	Yes	Yes	180
Object Catalog: Database	Yes	5	0	Yes	Yes	Yes	180
Object Catalog: Tablespace	Yes	5	0	Yes	Yes	Yes	180
PG HBA Conf	Yes	30	0	Yes	Yes	Yes	180
Server Information	Yes	5	0	Yes	Yes	Yes	180
Server log Configuration	Yes	0	10	Yes	Yes	Yes	180
Session Information	Yes	5	0	Yes	Yes	Yes	180
Settings	Yes	5	0	Yes	Yes	Yes	180
Streaming Replication	Yes	5	0	Yes	No	Yes	180
Streaming Replication Database Conflicts	Yes	5	0	Yes	No	Yes	180
Streaming Replication Lag Time	Yes	5	0	Yes	No	Yes	180
Tablespace Size	Yes	30	0	Yes	Yes	Yes	180
User Information	Yes	30	0	Yes	Yes	Yes	180
WAL Archive Status	Yes	30	0	Yes	No	Yes	180

By default, the **xDB Replication** probe is disabled. To enable the **xDB Replication** probe, right click on the name of the server, and select **Connect** from the context menu; if prompted, provide authentication information. After connecting, expand the server node of the tree control, and highlight the name of the replicated database. Then, select **Manage Probes...** from the **Management** menu.

Use fields on the **Manage Probes** tab to configure the xDB Replication probe:

- Move the **Default** slider to **No** to modify the Minutes and Seconds between probe executions.
- Use the **Enabled?** slider to instruct PEM to execute the xDB Replication probe.
- Set the **Default** slider in the **Data Retention** field to **No** to modify the number of days that PEM will store the information retrieved by the probe.

After enabling the probe, you can use the metrics returned to create custom charts and dashboards in the PEM client.

4.14 Performance Diagnostic

You can use the Performance Diagnostic dashboard to analyze the database performance for Postgres instances by monitoring the wait events. To display the diagnostic graphs, PEM uses the data collected by EDB Wait States module.

Performance Diagnostic feature is supported for Advanced Server databases from PEM 7.6 version onwards and for PostgreSQL databases it is supported from PEM 8.0 onwards.

!!! Note For PostgreSQL databases, Performance Diagnostics is supported only for versions 10, 11, and 12 installed on the supported CentOS or RHEL platforms.

For more information on EDB Wait States, see [EDB Postgres Advanced Server Guide](#).

You can analyze the Wait States data on multiple levels by narrowing down your selection of data. Each level of the graph is populated on the basis of your selection of data at the higher level.

Prerequisite:

- For PostgreSQL, you need to install `edb_wait_states_<X>` package from `edb.repo` where `<X>` is the version of PostgreSQL Server. You can refer to [EDB Build Repository](#) for the steps to install this package. For Advanced Server, you need to install `edb-as<X>-server-edb-modules`, Where `<X>` is the version of Advanced Server.
- Once you ensure that EDB Wait States module of EDB Postgres Advanced Server is installed, then configure the list of libraries in the `postgresql.conf` file as below:

```
shared_preload_libraries = '$libdir/edb_wait_states'
```

Restart the database server, and then create the following extension in the maintenance database:

```
CREATE EXTENSION edb_wait_states;
```

- You must have super user privileges to access the Performance Diagnostic dashboard.

You get the following error while accessing the Performance Diagnostic dashboard if the above prerequisites are not met:



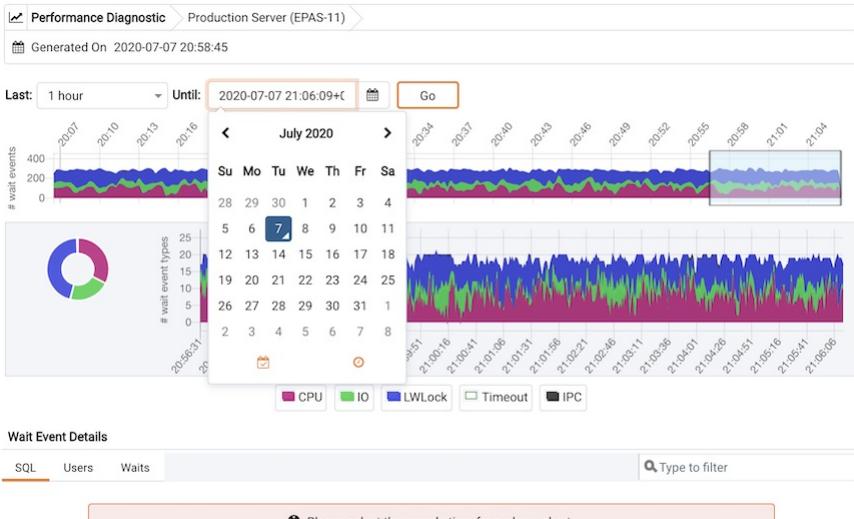
To open the Performance Diagnostic dashboard, select `Server` and then `Performance Diagnostic...` from the `Tools` menu of the PEM client.



By default, the top most Performance Diagnostic graph pulls the data of last one hour, starting from current date and time. This graph shows the time series containing the number of active sessions. Each point of this time series represents the active sessions and wait events at a particular time and last 15 seconds. These sessions may or may not be waiting for an wait event, or using the CPU at a particular point in time. This time series is generated based on the wait event samples collected by the `edb_wait_states` extension.

You can also use the [Preferences](#) dialog to display Performance Diagnostic in a new browser tab. Use [Open in New Browser Tab?](#) to display the Performance Diagnostics dashboard in a new browser tab.

The range selection in the first graph is 10 minutes. You can use the **Last** drop-down list box to select the duration for which you want to see the graph: select the last 1 hour, last 4 hours, last 12 hours, or last 24 hours. You can also select the date and time through which you want the data to be displayed.



The first graph displays the number of active sessions (and - wait event types) for the selected time interval. You can narrow down the timeline in the first graph to analyze the data for a specific time period.

Next section plots the following graphs based on the selected time interval in the first graph:

1. Donut graph - It shows total wait event types according to the time range selection in the first graph. It helps you understand how much time was spent by those session on waiting for an event.
2. Line graph - It plots a time series with each point representing the active sessions for each sample time.

To differentiate each wait event types and the CPU usage clearly, the graph for each wait event type is displayed in a

different color.

Select a particular time on the **Line graph** for which you wish to analyze the wait events; the third section displays the wait event details in the Performance Diagnostics dashboard on the basis of your selected particular time in the second graph. The third section displays wait event details on three tabs:

- The **SQL** tab displays the list of SQL queries having wait events for the selected sample time.
- The **Users** tab displays the details of the wait events grouped by users for selected sample time.
- The **Waits** tab displays the number of wait events belonging to each wait event type for the selected sample time.



You can click on the graph legends to show or hide a particular wait event type in all the graphs. This will make the analysis of a specific wait event type easier.



You can filter the data displayed in the rows under all the three tabs. You can also sort the data alphabetically by clicking on the column headers.

SQL tab

Wait Event Details

SQL	Users	Waits	
			Q year
Load By Waits ▾		SQL	Number of sessions
eye		select nation, o_year, sum(amount) as sum_profit from 2	2
eye		select o_year, sum(case when nation = \$1 then volume	1
eye		select l_shipmode, sum(case when o_orderpriority = \$1	1
eye		select supp_nation, cust_nation, l_year, sum(volume)	1

Users tab

Wait Event Details

SQL	Users	Waits	
			Q Type to filter
Load By Waits ▾	Users	Number of Events	Execution Count
enterprisedb	6	6	6
test2	6	6	6
test1	5	6	6
test3	5	7	7
test4	3	5	5

Waits tab

Wait Event Details

SQL	Users	Waits	
			Q Type to filter
Load By Wait ▾	Wait Event Type	Wait Event	Number of Events
LWLock	buffer_mapping	21	
IO	DataFileRead	3	
LWLock	buffer_io	1	

Click on the Eye icon in any row of the SQL tab to display a new tab with details of the query of that particular row. This page displays Query ID and its corresponding session IDs in a dropdown list at that particular selected sample time in the Query information section. You can select the session ID for the selected query for which you want to analyze the data. You will see the details corresponding to the selected session ID and query ID. The Query information table also displays the SQL query. If the SQL query is being displayed partially, click the down arrow at the bottom of the section to view the complete SQL query.

The **Wait event types** section displays the total number of wait event types for the selected session ID and query ID. It shows two type of graphs:

1. Donut graph - It shows the proportions of categorical data, with the size of each piece representing the proportion of each wait event type.
2. Timeline bar graph - It can be used to visualize trends in counts of wait event types over time.

To differentiate clearly, each wait event type is represented by a different color in the bar graph.



The **Wait events** section has a table displaying all the wait events occurred during the query execution. It displays data in decreasing order by number of the wait events. Second table displays the wait event with sample time occurred over the period of whole query execution. It allows to analyze the wait events during the query execution over the period of time. It shows the actual samples collected by the EDB Wait States extension for that particular query ID and session ID.

4.15 Reference

The following sections are provided for reference; please note that the items referred to in the following tables are subject to change.

PEM Server Configuration Parameters - Reference

You can use global configuration options to modify aspects of the PEM Server's behavior. Please note that the list of configuration parameters is subject to change.

Parameter name	Value and Unit	Description
audit_log_retention_time	30 days	Specifies the number of days that an audit log will be retained on the PEM server.
auto_create_agent_alerts	true	Specifies whether to create default agent level alerts automatically when an agent is registered.

Parameter name	Value and Unit	Description
auto_create_server_alerts	true	Specifies whether to create default server level alerts automatically when a server is bound to an agent.
chart_disable_bullets	false	Enable/disable bullets on line charts on dashboards and Capacity Manager reports.
cm_data_points_per_report	50	Specifies the number of data points to plot on charts on Capacity Manager reports.
cm_max_end_date_in_years	5 years	Specifies the maximum amount of time that the Capacity Manager will extrapolate data for. Ensures that threshold-based end dates of on reports do not get extrapolated indefinitely.
dash_alerts_timeout	60 seconds	Specifies the number of seconds after which the components of the Alerts dashboard are auto-refreshed.
dash_db_comrol_span	7 days	Specifies the number of days worth of data to plot on the Commit/Rollback Analysis chart on the Database Analysis dashboard and Server Analysis dashboard.
dash_db_comrol_timeout	1800 seconds	Specifies the number of seconds after which the Commits/Rollbacks line chart is auto-refreshed on the Database Analysis dashboard and Server Analysis dashboard.
dash_db_connovervw_timeout	300 seconds	Specifies the number of seconds after which the Connection Overview pie chart is auto-refreshed in the Database Analysis dashboard.
dash_db_eventlag_span	7 days	Specifies the number of days worth of data to plot on the Number of Events Lag chart for slony replication on the Database Analysis dashboard.
dash_db_eventlag_timeout	1800 seconds	Specifies the number of seconds after which the Number of Events Lag line chart for slony replication is auto-refreshed on the Database Analysis dashboard.
dash_db_hottable_rows	25 rows	Specifies the number of rows to show on the HOT Table Analysis table on the Database Analysis dashboard.
dash_db_hottable_timeout	300 seconds	Specifies the number of seconds after which the Hot Tables table is auto-refreshed in the Database Analysis dashboard.
dash_db_io_span	7 days	Specifies the number of days worth of data to plot on the Database I/O Analysis chart on the Database Analysis dashboard and I/O Analysis dashboard.
dash_db_io_timeout	1800 seconds	Specifies the number of seconds after which the Database I/O line chart is auto-refreshed on the Database Analysis dashboard and I/O Analysis dashboard.
dash_db_rowact_span	7 days	Specifies the number of days worth of data to plot on the Row Activity Analysis chart on the Database Analysis dashboard, the I/O Analysis dashboard, and the Server Analysis dashboard.
dash_db_rowact_timeout	1800 seconds	Specifies the number of seconds after which the Row Activity line chart is auto-refreshed on the Database Analysis dashboard, the I/O Analysis dashboard, and the Server Analysis dashboard.

Parameter name	Value and Unit	Description
dash_db_storage_timeout	300 seconds	Specifies the number of seconds after which the Storage bar chart is auto-refreshed in the Database Analysis dashboard.
dash_db_timelag_span	7 days	Specifies the number of days worth of data to plot on the Time Lag chart for Slony replication on the Database Analysis dashboard.
dash_db_timelag_timeout	1800 seconds	Specifies the number of seconds after which the Time Lag line chart for Slony replication is auto-refreshed on the Database Analysis dashboard.
dash_db_useract_span	7 days	Specifies the number of days worth of data to plot on the User Activity Analysis chart on the Database Analysis dashboard.
dash_db_useract_timeout	1800 seconds	Specifies the number of seconds after which the User Activity line chart is auto-refreshed in the Database Analysis dashboard.
dash_efm_timeout	300 seconds	Specifies the number of seconds after which the Failover Manager Node Status and Failover Manager Cluster Info line chart is auto-refreshed on the Streaming Replication dashboard.
dash_global_overview_timeout	30 seconds	Specifies the number of seconds after which the components of the Global Overview dashboard are auto-refreshed.
dash_header_timeout	60 seconds	Specifies the number of seconds after which the information on the header of all the dashboards are auto-refreshed.
dash_io_chkpt_span	7 days	Specifies the number of days worth of data to plot on the Checkpoints chart on the I/O Analysis dashboard.
dash_io_chkpt_timeout	1800 seconds	Specifies the number of seconds after which the Checkpoints line chart is auto-refreshed on the I/O Analysis dashboard.
dash_io_hotindx_timeout	300 seconds	Specifies the number of seconds after which the Hot Indexes bar chart is auto-refreshed on the I/O Analysis dashboard.
dash_io_hottbl_timeout	300 seconds	Specifies the number of seconds after which the Hot Tables bar chart is auto-refreshed on the I/O Analysis dashboard.
dash_io_index_objectio_rows	25 rows	Specifies the number of rows displayed on the Index Activity table on the I/O Analysis dashboard and the Object Activity Analysis dashboard.
dash_io_index_objectio_timeout	60 seconds	Specifies the number of seconds after which the Index Activity table is auto-refreshed on the I/O Analysis dashboard and the Object Activity Analysis dashboard.
dash_io_objectio_rows	25 rows	Specifies the number of rows displayed in the Object I/O Details table on the I/O Analysis dashboard and Object Activity Analysis dashboard.
dash_io_objectio_timeout	300 seconds	Specifies the number of seconds after which the Object I/O Details table is auto-refreshed on the I/O Analysis dashboard and Object Activity Analysis dashboard.

Parameter name	Value and Unit	Description
dash_memory_hostmemact_span	7 days	Specifies the number of days worth of data to plot on the Host Memory Activity Analysis chart on the Memory Analysis dashboard.
dash_memory_hostmemact_timeout	1800 seconds	Specifies the number of seconds after which the Host Memory Activity line chart is auto-refreshed on the Memory Analysis dashboard.
dash_memory_hostmemconf_timeout	300 seconds	Specifies the number of seconds after which the Host Memory Configuration pie chart is auto-refreshed on the Memory Analysis dashboard and Server Analysis dashboard.
dash_memory_servmemact_span	7 days	Specifies the number of days worth of data to plot on the server Memory Activity Analysis chart on the Memory Analysis dashboard.
dash_memory_servmemact_timeout	1800 seconds	Specifies the number of seconds after which the Server Memory Activity line chart is auto-refreshed on the Memory Analysis dashboard.
dash_memory_servmemconf_timeout	300 seconds	Specifies the number of seconds after which the Server Memory Configuration pie chart is auto-refreshed on the Memory Analysis dashboard.
dash_objectact_objstorage_rows	15 rows	Specifies the number of rows to show on the Object Storage table on the Object Activity Analysis dashboard.
dash_objectact_objstorage_timeout	300 seconds	Specifies the number of seconds after which the Object Storage table is auto-refreshed in the Object Activity Analysis dashboard.
dash_objectact_objtopindexes_timeout	300 seconds	Specifies the number of seconds after which the Top 5 Largest Indexes bar chart is auto-refreshed in the Object Activity Analysis dashboard.
dash_objectact_objtoptables_timeout	300 seconds	Specifies the number of seconds after which the Top 5 Largest Tables bar chart is auto-refreshed in the Object Activity Analysis dashboard.
dash_os_cpu_span	7 days	Specifies the number of days worth of data to plot on the CPU chart on the Operating System Analysis dashboard.
dash_os_cpu_timeout	1800 seconds	Specifies the number of seconds after which the CPU line chart is auto-refreshed on the Operating System Analysis dashboard.
dash_os_data_span	7 days	Specifies the number of days worth of data to plot on the I/O line chart on the Operating System Analysis dashboard.
dash_os_disk_span	7 days	Specifies the number of days worth of data to plot on the Utilisation chart on the Operating System Analysis dashboard.
dash_os_hostfs_timeout	1800 seconds	Specifies the number of seconds after which the Host File System Details table is auto-refreshed on the Operating System Analysis dashboard.
dash_os_io_timeout	1800 seconds	Specifies the number of seconds after which the I/O line chart is auto-refreshed on the Operating System Analysis dashboard.

Parameter name	Value and Unit	Description
dash_os_memory_span	7 days	Specifies the number of days worth of data to plot on the Memory chart on the Operating System Analysis dashboard.
dash_os_memory_timeout	1800 seconds	Specifies the number of seconds after which the Memory line chart is auto-refreshed on the Operating System Analysis dashboard.
dash_os_packet_span	7 days	Specifies the number of days worth of data to plot on the Packet chart on the Operating System Analysis dashboard.
dash_os_packet_timeout	1800 seconds	Specifies the number of seconds after which the Network Packets line chart is auto-refreshed on the Operating System Analysis dashboard.
dash_os_process_span	7 days	Specifies the number of days worth of data to plot on the Process chart on the Operating System Analysis dashboard.
dash_os_process_timeout	1800 seconds	Specifies the number of seconds after which the Process line chart is auto-refreshed on the Operating System Analysis dashboard.
dash_os_storage_timeout	1800 seconds	Specifies the number of seconds after which the Storage pie chart is auto-refreshed on the Operating System Analysis dashboard.
dash_os_traffic_span	7 days	Specifies the number of days worth of data to plot on the Traffic chart on the Operating System Analysis dashboard.
dash_os_traffic_timeout	1800 seconds	Specifies the number of seconds after which the Traffic line chart is auto-refreshed on the Operating System Analysis dashboard.
dash_os_util_timeout	1800 seconds	Specifies the number of seconds after which the Utilisation line chart is auto-refreshed on the Operating System Analysis dashboard.
dash_probe_log_timeout	300 seconds	Specifies the number of seconds after which the Probe Log table is auto-refreshed on
dash_replication_archivestat_span	7 days	Specifies the number of days worth of data to plot on the WAL Archive Status chart on the Streaming Replication Analysis dashboard.
dash_replication_archivestat_timeout	1800 seconds	Specifies the number of seconds after which the WAL Archive Status line chart is auto-refreshed on the Streaming Replication dashboard.
dash_replication_pagelag_span	7 days	Specifies the number of days worth of data to plot on the WAL Lag Pages chart on the Streaming Replication dashboard.
dash_replication_pagelag_timeout	1800 seconds	Specifies the number of seconds after which the WAL Lag Pages line chart is auto-refreshed on the Streaming Replication dashboard.
dash_replication_segmentlag_span	7 days	Specifies the number of days worth of data to plot on the WAL Lag Segments chart on the Streaming Replication dashboard.
dash_replication_segmentlag_timeout	1800 seconds	Specifies the number of seconds after which the WAL Lag Segments line chart is auto-refreshed on the Streaming Replication dashboard.

Parameter name	Value and Unit	Description
dash_replication_timelag_span	7 days	Specifies the number of days worth of data to plot on the Replication Lag Time chart on the Streaming Replication dashboard.
dash_replication_timelag_timeout	1800 seconds	Specifies the number of seconds after which the Replication Lag Time line chart is auto-refreshed on the Streaming Replication dashboard.
dash_server_buffers_written	168 hours	Specifies the number of days worth of data to plot on the Background Writer Statistics chart on the Server Analysis dashboard.
dash_server_buffers_written_timeout	300 seconds	Specifies the number of seconds after which the Background Writer Statistics line chart is auto-refreshed on the Server Analysis dashboard.
dash_server_connovervw_timeout	300 seconds	Specifies the number of seconds after which the Connection Overview pie chart is auto-refreshed in the Server Analysis dashboard.
dash_server_database_timeout	300 seconds	Specifies the number of seconds after which the Databases table is auto-refreshed in the Server Analysis dashboard.
dash_server_dbsize_span	7 days	Specifies the number of days worth of data to plot on the Database Size Analysis on the Server Analysis dashboard.
dash_server_dbsize_timeout	1800 seconds	Specifies the number of seconds after which the Database Size line chart is auto-refreshed in the Server Analysis dashboard.
dash_server_disk_timeout	1800 seconds	Specifies the number of seconds after which the Disk line chart is auto-refreshed in the Server Analysis dashboard.
dash_server_global_span	7 days	Specifies the number of days worth of data to plot on the Disk line chart on the Server Analysis dashboard.
dash_server_sharedbuff_span	7 days	Specifies the number of days worth of data to plot on the Shared Buffer chart on the Server Analysis dashboard.
dash_server_sharedbuff_timeout	1800 seconds	Specifies the number of seconds after which the Shared Buffers line chart is auto-refreshed in the Server Analysis dashboard.
dash_server_tabspaceysize_span	7 days	Specifies the number of days worth of data to plot on the Tablespace Size chart on the Server Analysis dashboard.
dash_server_tabspaceysize_timeout	1800 seconds	Specifies the number of seconds after which the Tablespace Size line chart is auto-refreshed in the Server Analysis dashboard.
dash_server_useract_span	7 days	Specifies the number of days worth of data to plot on the User Activity chart on the Server Analysis dashboard.
dash_server_useract_timeout	1800 seconds	Specifies the number of seconds after which the User Activity line chart is auto-refreshed in the Server Analysis dashboard.
dash_sessact_lockact_timeout	300 seconds	Specifies the number of seconds after which the Session Lock Activity table is auto-refreshed in the Session Activity Analysis dashboard.

Parameter name	Value and Unit	Description
dash_sessact_workload_timeout	300 seconds	Specifies the number of seconds after which the Session Workload table is auto-refreshed in the Session Activity Analysis dashboard.
dash_sess_waits_nowaits_timeout	300 seconds	Specifies the number of seconds after which the Session Waits By Number Of Waits pie
dash_sess_waits_timewait_timeout	300 seconds	Specifies the number of seconds after which the Session Waits By Time Waited pie chart is auto-refreshed in the Session Waits Analysis dashboard.
dash_sess_waits_waitdtl_timeout	300 seconds	Specifies the number of seconds after which the Session Waits Details table is auto-refreshed in the Session Waits Analysis dashboard.
dash_storage_dbdtls_timeout	300 seconds	Specifies the number of seconds after which the Database Details table is auto-refreshed in the Storage Analysis dashboard.
dash_storage_dbovervw_timeout	300 seconds	Specifies the number of seconds after which the Database Overview pie chart is auto-refreshed in the Storage Analysis dashboard.
dash_storage_hostdtls_timeout	300 seconds	Specifies the number of seconds after which the Host Details table is auto-refreshed
dash_storage_hostovervw_timeout	300 seconds	Specifies the number of seconds after which the Host Overview pie chart is auto-refreshed in the Storage Analysis dashboard.
dash_storage_tblspcdtls_timeout	300 seconds	Specifies the number of seconds after which the Tablespace Details table is auto-refreshed in the Storage Analysis dashboard.
dash_storage_tblspcovervw_timeout	300 seconds	Specifies the number of seconds after which the Tablespace Overview pie chart is auto-refreshed in the Storage Analysis dashboard.
dash_sys_waits_nowaits_timeout	300 seconds	Specifies the number of seconds after which the System Waits By Number Of Waits pie chart is auto-refreshed in the System Waits Analysis dashboard.
dash_sys_waits_timewait_timeout	300 seconds	Specifies the number of seconds after which the System Waits By Time Waited pie chart is auto-refreshed in the System Waits Analysis dashboard.
dash_sys_waits_waitdtl_timeout	300 seconds	Specifies the number of seconds after which the System Waits Details table is auto-refreshed in the System Waits Analysis dashboard.
deleted_charts_retention_time	7 days	Specifies the number of days that a custom chart (displayed on a user-defined dashboard) is stored.
deleted_probes_retention_time	7 days	Specifies the number of days that a custom probe (displayed on a user-defined dashboard) is stored.
download_chart_format	jpeg	Specifies the format in which a downloaded chart will be stored. May be jpeg or png.

Parameter name	Value and Unit	Description
flapping_detection_state_change	3	Specifies the number of state changes detected within a specified interval to define a given alert as flapping. Flapping starts when more than N state changes have occurred over $[N + 1 * \text{min(probe_interval)} * 2]$ minutes and the fine state is not None. Where the default value of N is 2 or 3, and $\text{min(probe_interval)}$ is the smallest interval for all the probes used by the alert. Flapping ends when ZERO state changes have occurred over $[2 N * \text{min(probe_interval)}]$ minutes.
job_retention_time	30 days	Specifies the number of days that non-recurring scheduled tasks and their associated
long_running_transaction_minutes	5 minutes	Specifies the number of minutes a query executes for before being considered long running.
nagios_cmd_file_name	<file_name>	Specifies nagios command file to which passive service check result will be sent.
nagios_enabled	t	Specifies whether alert notification will be submitted to nagios or not.
nagios_medium_alert_as_critical	f	Specifies whether medium level PEM alert will be considered as critical in nagios.
nagios_spool_retention_time	7 days	Specifies the number of days to retain nagios messages in the spool table before they are discarded.
probe_log_retention_time	30 days	Specifies the number of days that probe log records are retained.
reminder_notification_interval	24 hours	Specifies the number of hours after which a reminder email is sent in case an alert has not been cleared.
server_log_retention_time	30 days	Specifies the number of days that the server log is retained on the PEM server.
show_data_tab_on_graph	false	If 'true', a Data tab is added to each graph. Select the Data tab to review the data that is plotted on the graph.
smtp_authentication	false	Specifies whether to enable/disable authentication over SMTP.
smtp_enabled	true	Specifies whether to enable/disable sending of emails.
smtp_encryption smtp_password	false	Specifies whether to send SMTP email using an encrypted connection. Specifies the password to be used to connect to the SMTP server.
smtp_port	25	Specifies the SMTP server port to be used for sending email.
smtp_server	127.0.0.1	Specifies the SMTP server host address to be used for sending email.
smtp_spool_retention_time smtp_username	7 days	Specifies the number of days to retain sent email messages in the spool table before they are discarded. Specifies the username to be used to connect to SMTP server.
snmp_community	public	Specifies the SNMP community used when sending traps. Used only with SNMPv1 and SNMPv2.
snmp_enabled	true	Specifies whether to enable/disable sending SNMP traps.

Parameter name	Value and Unit	Description
snmp_port	162	Specifies the SNMP server port to be used for sending SNMP traps.
snmp_server	127.0.0.1	Specifies the SNMP server host address to be used for sending SNMP traps.
snmp_spool_retention_time snmp_security_name snmp_security_engine_id	7 days	Specifies the number of days to retain sent traps in the spool table before they are discarded. Specifies the user name or security name for sending SNMP traps. Used only with SNMPv3. Specifies the Engine id of the SNMP Agent on the SNMP Server. Used only with SNMPv3.
snmp_security_level snmp_context_name snmp_context_engine_id	NOAUTH_NOPRIV	Specifies Security level and its possible values can be: AUTH_NOPRIV - Authentication, No Privacy or AUTH_PRIV - Authentication, Privacy or NOAUTH_NOPRIV - no Authentication, no Privacy. Used only with SNMPv3. Specifies the Context name, the identifier for MIB objects when sending SNMP traps. Used only with SNMPv3. Specifies the Context engine id, the identifier for MIB objects when sending SNMP traps. If not specified, snmp_security_engine_id will be used. Used only with SNMPv3.
snmp_authentication_protocol	NONE	Specifies the authentication type for SNMP traps. Its possible values can be NONE, HMACMD5 or HMACSHA. Used only with SNMPv3.
snmp_privacy_protocol snmp_authentication_password snmp_privacy_password	NONE	Specifies the privacy protocol for SNMP traps. Its possible values can be NONE, DES, AES128, IDEA, AES192, or AES256. Used only with SNMPv3. Specifies the authentication password associated with security name mentioned in snmp_security_name. Used only for SNMPv3. Specifies the privacy password associated with security name mentioned in snmp_security_name. Used only for SNMPv3.
webclient_help_pg	EDB hosted documentation	Specifies the location of the online PostgreSQL core documentation.

Capacity Manager Metrics - Reference

Please Note that the Capacity Manager metrics available will vary by platform, and are subject to change. The available metrics may include the metrics described in the table below.

Metric Name	Description
# Dead Tuples	The number of dead tuples in the selected table.
# Dead Tuples+	The cumulative number of dead tuples in the selected table.
# Heap Tuples Fetched by Index Scans	The number of heap tuples fetched by index scans.
# Heap Tuples Fetched by Index Scans	The cumulative number of heap tuples fetched by index scans.
# Idle Backends+	The cumulative number of currently idle backend clients.

Metric Name	Description
# Index Scans	The number of index scans performed on the specified object.
# Index Scans+	The cumulative number of index scans performed on the specified object.
# Index Tuples Read	The number of index tuples read.
# Index Tuples Read+	The cumulative number of index tuples read.
# Live Tuples	The number of tuples visible to transactions.
# Live Tuples+	The cumulative number of tuples visible to transactions.
# Pages Estimated by ANALYZE	The number of pages estimated by ANALYZE.
# Pages Estimated by ANALYZE+	The cumulative number of pages estimated by ANALYZE.
# Sequential Scans	The number of sequential scans performed on the specific table.
# Sequential Scans+	The cumulative number of sequential scans performed on the specific table.
# Sequential Scan Tuples	The number of tuples sequentially scanned in the specific table.
# Sequential Scan Tuples+	The cumulative number of tuples sequentially scanned in the specific table.
# Tuples Deleted	The number of tuples deleted.
# Tuples Deleted+	The cumulative number of tuples deleted.
# Tuples Estimated by ANALYZE	The number of live (visible) tuples estimated by ANALYZE.
# Tuples Estimated by ANALYZE+	The cumulative number of live tuples estimated by ANALYZE.
# Tuples HOT Updated	The number of tuples HOT updated. In a HOT update, the new tuple resides in the same block as the original tuple and the tuples share an index entry.
# Tuples HOT Updated+	The cumulative number of tuples HOT updated.
# Tuples Inserted	The number of tuples inserted into the specified table.
# Tuples Inserted+	The cumulative number of tuples inserted into the specified table.
# Tuples Updated	The number of tuples updated in the selected table.
# Tuples Updated+	The cumulative number of tuples updated in the selected table.
Blocks Hit	The number of blocks found in the cache.
Blocks Hit+	The cumulative number of blocks found in the cache.
Blocks Read	The number of blocks read.
Blocks Read+	The cumulative number of blocks read.
Blocks Read from InfiniteCache	The number of blocks read from InfiniteCache.
Blocks Read from InfiniteCache+	The cumulative number of blocks read from InfiniteCache.
Blocks Written	The number of blocks written.
Blocks Written+	The cumulative number of blocks written.
Buffers Allocated	The number of buffers allocated.
Buffers Allocated+	The cumulative number of buffers allocated.
Buffers Written - Backends	The number of buffer blocks written to disk by server processes (processes connected to a client application).
Buffers Written - Backends+	The cumulative number of buffer blocks written to disk by server processes.

Metric Name	Description
Buffers Written - Checkpoint	The number of blocks written to disk by the checkpoint process.
Buffers Written - Checkpoint+	The cumulative number of blocks written to disk by the checkpoint process.
Buffers Written - Cleaning Scan	The number of blocks written to disk by the autovacuum process.
Buffers Written - Cleaning Scan+	The cumulative number of blocks written to disk by the autovacuum process.
Bytes Received (KB)	The number of bytes received from the client (in kilobytes).
Bytes Received (KB)+	The cumulative number of bytes received (in kilobytes).
Bytes Sent (KB)	The number of bytes sent to the client (in kilobytes).
Bytes Sent (KB)+	The cumulative number of bytes sent (in kilobytes).
Checkpoints - Timed	The number of checkpoint operations triggered by the checkpoint interval.
Checkpoints - Timed+	The cumulative number of checkpoint operations triggered by the checkpoint interval.
Checkpoints - Untimed	The number of checkpoint operations triggered by checkpoint size.
Checkpoints - Untimed+	The cumulative number of checkpoint operations triggered by checkpoint size.
Database Size (MB)	The size of the specified database (in megabytes).
Free RAM Memory	The amount of free RAM memory (in megabytes).
Free Swap Memory	The amount of free swap space on disk (in megabytes).
Heap Blocks Hit	The number of heap blocks found in the cache.
Heap Blocks Hit+	The cumulative number of heap blocks found in the cache.
Heap Blocks Read	The number of heap blocks read.
Heap Blocks Read+	The cumulative number of heap blocks read.
Index Blocks Hit	The number of index blocks found in the cache.
Index Blocks Hit+	The cumulative number of index blocks found in the cache.
Index Blocks Read	The number of index blocks read.
Index Blocks Read+	The cumulative number of index blocks read.
Index Size (MB)	The size of the specified index (in megabytes).
In Packets Discards	The number of inbound packets discarded.
In Packets Discards+	The cumulative number of inbound packets discarded.
In Packets Errors	The number of inbound packets that contain errors.
In Packets Errors+	The cumulative number of inbound packets that contain errors.
Link Bandwidth (Mbit/s)	The speed of the network adapter (in megabits per second).
Load Average - 15 Minute	CPU saturation (in percent) - 15 minute sampling average.
Load Average - 1 Minute	CPU saturation (in percent) - 1 minute sampling average.
Load Average - 5 Minute	CPU saturation (in percent) - 5 minute sampling average.
Load Percentage	CPU saturation in percent.

Metric Name	Description
Number of Prepared Transactions+	The cumulative number of prepared transactions.
Number of WAL Files+	The cumulative number of write-ahead log files.
Out Packets Discards	The number of outbound packets discarded.
Out Packets Discards+	The cumulative number of outbound packets discarded.
Out Packets Errors	The number of outbound packets that contain errors.
Out Packets Errors+	The cumulative number of outbound packets that contain errors.
Packets Received	The number of packets received.
Packets Received+	The cumulative number of packets received.
Packets Sent	The number of packets sent.
Packets Sent+	The cumulative number of packets sent.
Size (MB)	The total size of the disk (in megabytes).
Size of Indexes (MB)	The size of indexes on the specified table (in megabytes).
Space Available (MB)	The current disk space available (in megabytes).
Space Used (MB)	The current disk space used (in megabytes).
Table Size (MB)	The size of the specified table (in megabytes).
Tablespace Size (MB)	The size of the specified tablespace (in megabytes).
Temp Buffers (MB)	The size of temporary buffers (in megabytes).
Toast Blocks Hit	The number of TOAST blocks found in the cache.
Toast Blocks Hit+	The cumulative number of TOAST blocks found in the cache.
Toast Blocks Read	The number of TOAST blocks read.
Toast Blocks Read+	The cumulative number of TOAST blocks read.
Total RAM Memory	The total amount of RAM memory on the system (in megabytes).
Total Swap Memory	The total amount of swap space on the system (in megabytes).
Total Table Size w/Indexes and Toast	The total size of the specified table (including indexes and associated oversized attributes).
Transactions Aborted	The number of aborted transactions.
Transactions Aborted+	The cumulative number of aborted transactions.
Transactions Committed	The number of committed transactions.
Transactions Committed+	The cumulative number of committed transactions.
Tuples Deleted	The number of tuples deleted from the specified table.
Tuples Deleted+	The cumulative number of tuples deleted from the specified table.
Tuples Estimated by ANALYZE	The number of visible tuples in the specified table.
Tuples Estimated by ANALYZE+	The cumulative number of visible tuples in the specified table.
Tuples Fetched	The number of tuples fetched from the specified table.
Tuples Fetched+	The cumulative number of tuples fetched from the specified table.
Tuples HOT Updated	The number of tuples HOT updated. In a HOT update, the new tuple resides in the same block as the original tuple and the tuples share an index entry.

Metric Name	Description
Tuples HOT Updated+	The cumulative number of tuples HOT updated. In a HOT update, the new tuple resides in the same block as the original tuple and the tuples share an index entry.
Tuples Inserted	The number of tuples inserted into the specified table.
Tuples Inserted+	The cumulative number of tuples inserted into the specified table.
Tuples Returned	The number of tuples returned in result sets.
Tuples Returned+	The cumulative number of tuples returned in result sets.
Tuples Updated	The number of tuples updated in the specified table.
Tuples Updated+	The cumulative number of tuples updated in the specified table.
WAL Segment Size (MB)	The segment size of the write-ahead log (in megabytes).

!!! Note The '+' following the name of a metric signifies that the data for the metric is gathered cumulatively; those metrics that are not followed by the '+' sign are collected as a 'point-in-time' value.

PEM Probes – Reference

A probe is a scheduled task that retrieves information about the database objects that are being monitored by the PEM agent. PEM uses the collected information to build the graphs displayed on each dashboard. The Manage Probes tab (accessed via the Management menu) allows you to modify the data collection schedule and the length of time that PEM will retain information returned by a specific probe.

Probe Name	Information Monitored by Probe	Level
Background Writer Statistics	This probe monitors information about the background writer. The information includes: The number of timed checkpoints The number of requested checkpoints The number of buffers written (by checkpoint) The number of buffers written (by background writer) The number of background writer cycles The number of background buffers written The number of buffers allocated	Server
Blocked Session Information	This probe provides information about blocked sessions.	Server
CPU Usage	This probe monitors CPU Usage information.	Agent
Data and Log File Analysis	This probe monitors information about log files. The information includes: The name of the log file The directory in which the log file resides	Server
Database Statistics	This probe monitors database statistics. The information includes: The number of backends The number of transactions committed The number of transactions rolled back The number of blocks read The number of blocks hit The number of rows returned The number of rows fetched The number of rows inserted The number of rows updated The number of rows deleted	Server

Probe Name	Information Monitored by Probe	Level
Disk Busy Info	This probe monitors information about disk activity. Note: This probe is not supported on Mac OS X, Solaris or HP-UX	Agent
Disk Space	This probe monitors information about disk space usage. The information includes: The amount of disk space used The amount of disk space available	Agent
EDB Audit Configuration	This probe monitors the audit logging configuration of EDB Postgres Advanced Server.	Server
Failover Manager Cluster Info	This probe monitors a Failover Manager cluster, returning information about the cluster. This probe is disabled unless a cluster name and path of the Failover Manager binary is provided on the Server Properties dialog.	Server
Failover Manager Node Status	This probe monitors a Failover Manager cluster, returning detailed about each node within the cluster. This probe is disabled unless a cluster name and path of the Failover Manager binary is provided on the Server Properties dialog.	Server
Function Statistics	This probe monitors a database, retrieving information about functions. The information includes: Function names Argument types Return values	Database
Index Size	This probe monitors a database, retrieving information about indexes. The information includes: The name of the index The time the data was gathered The size of the index (in MB's)	Database
Index Statistics	This probe monitors index statistics. The information includes: The number of index scans The number of rows read The number of rows fetched The number of blocks read The number of blocks hit	Database
Installed Packages	This probe monitors the packages that are currently installed. The information gathered includes: The name of the installed package The version of the installed package The date and time that the probe executed	Agent
IO Analysis	This probe monitors disk I/O information in. The information includes: The number of blocks read The number of blocks written The date and time that the probe executed Note: This probe is not supported on Mac OS X	Agent
Load Average	This probe monitors CPU load averages. The information includes: The 1-minute load average The 5-minute load average The 15-minute load average Note: This probe is not supported on Windows	Agent
Lock Information	This probe monitors lock information. The information includes: The database name The lock type The lock mode The process holding the lock	Server
Memory Usage	This probe monitors information about system memory usage.	Agent

Probe Name	Information Monitored by Probe	Level
Network Statistics	<p>This probe monitors network statistics. The information includes:</p> <ul style="list-style-type: none"> The interface IP address The number of packets sent The number of packets received The number of bytes sent The number of bytes received The link speed (in MB/second) 	Agent
Number of Prepared Transactions	This probe stores the number of prepared transactions.	Server
Number of WAL Files	This probe monitors the number of WAL files.	Server
Object Catalog: Database	<p>This probe monitors a list of databases and their properties. The information includes:</p> <ul style="list-style-type: none"> The database name The database encoding type If the database allows user connections or system connections 	Server
Object Catalog: Foreign Key	<p>This probe monitors a list of foreign keys and their properties. The information includes:</p> <ul style="list-style-type: none"> The name of the table that contains the foreign key The name of the table that the foreign key references The name of the database in which the table resides The name of the schema in which the table resides 	Schema
Object Catalog: Function	<p>This probe monitors a list of functions and their properties. The information includes:</p> <ul style="list-style-type: none"> The name of the function The name of the schema in which the function resides The name of the database in which the function resides 	Schema
Object Catalog: Index	<p>This probe monitors a list of indexes and their properties. The information includes:</p> <ul style="list-style-type: none"> The name of the index The name of the table that the index is associated with The name of the database in which the indexed table resides 	Schema
Object Catalog: Schema	This probe monitors a list of schemas and their associated databases and servers.	Database
Object Catalog: Sequence	This probe monitors a list of sequences and their properties.	Schema
Object Catalog: Table	<p>This probe monitors a list of table information. The information includes:</p> <ul style="list-style-type: none"> The table name The name of the schema in which the table resides The name of the database in which the schema resides A Boolean indicator that indicates if the table has a primary key 	Schema
Object Catalog: Tablespace	This probe monitors a list of tablespaces.	Server
Operating System Information	This probe monitors the operating system details and boot time.	Agent
Package Catalog	<p>This probe monitors the packages that are currently available for installation. The information gathered includes:</p> <ul style="list-style-type: none"> The package name The package version 	Agent
PG HBA Conf	This probe monitors authentication configuration information from the pg_hba.conf file.	Server
Server Information	This probe monitors server information.	Server

Probe Name	Information Monitored by Probe	Level
Session Information	<p>This probe monitors session information. The information includes:</p> <p>The name of the session user</p> <p>The date and time that the session connected to the server</p> <p>The status of the session at the time that the information was gathered (idle, waiting, etc)</p> <p>The client address and port number</p>	Server
Settings	This probe monitors the values currently assigned to GUC variables.	Server
SQL Protect	This probe monitors a server, retrieving information about SQL injection attacks.	Server
Slony Replication	This probe monitors lag data for clusters replicated using Slony.	Database
Streaming Replication	<p>This probe monitors a cluster that is using streaming replication, retrieving information about:</p> <p>The sent Xlog location (in bytes)</p> <p>The write Xlog location (in bytes)</p> <p>The flush Xlog location (in bytes)</p> <p>The replay Xlog location (in bytes)</p> <p>The Xlog lag (in segments)</p> <p>The Xlog lag (in pages)</p>	Server
Streaming Replication Lag Time	<p>This probe monitors a cluster that is using streaming replication, retrieving lag information about:</p> <p>Replication lag time (in seconds)</p> <p>Current status of replication (running/paused)</p>	Server
Streaming Replication Database Conflicts	<p>This probe monitors a database that is using streaming replication, retrieving information about any conflicts that arise. This includes information about queries that have been canceled due to:</p> <p>The # of drop tablespace conflicts</p> <p>The # of lock timeout conflicts</p> <p>The # of old snapshot conflicts</p> <p>The # of pinned buffer conflicts</p> <p>The # of deadlock conflicts</p>	Server
Table Bloat	<p>This probe monitors information about the current table bloat. The information includes:</p> <p>The name of the table</p> <p>The name of the schema in which the table resides</p> <p>The estimated number of pages</p> <p>The estimated number of wasted pages</p> <p>The estimated number of bytes per row</p>	Database
Table Frozen XID	This probe monitors the frozen XID of each table.	Schema

Probe Name	Information Monitored by Probe	Level
Table Size	This probe monitors table statistics. The information includes: The number of sequential scans The number of sequential scan rows The number of index scans The number of index scan rows The number of rows inserted The number of rows updated The number of rows deleted The number of live rows The number of dead rows The last VACUUM The last auto-vacuum The last ANALYZE The last auto-analyze The number of pages estimated by ANALYZE The number of rows estimated by ANALYZE	Database
Table Statistics	This probe monitors a list of tablespaces and their sizes.	Server
Tablespace Size	This probe monitors a list of tablespaces and their sizes.	Server
User Information	This probe monitors a list of the current users. The stored information includes: The user name The user type (superuser vs. non-superuser) The server to which the user is connected	Server
WAL Archive Status	This probe monitors the status of the WAL archive. The stored information includes: The # of WAL archives done The # of WAL archives pending The last archive time The # of WAL archives failed The time of the last failure	Server
xDB Replication	This probe monitors lag data for clusters replicated using xDB replication.	Database

PEM Pre-defined Alert Templates – Reference

An alert definition contains a system-defined or user-defined set of conditions that PEM compares to the system statistics; if the statistics deviate from the boundaries specified for that statistic, the alert triggers, and the PEM client displays a warning on the *Alerts Overview* page, and optionally sends a notification to a monitoring user.

The tables that follow list the system-defined alert templates that you can use to create an alert; please note that this list is subject to change, and may vary by system:

Templates applicable on Agent

Template Name	Description
Load Average (1 minute)	1-minute system load average.
Load Average (5 minutes)	5-minute system load average.
Load Average (15 minutes)	15-minute system load average.
Load Average per CPU Core (1 minutes)	1-minute system load average per CPU core.
Load Average per CPU Core (5 minutes)	5-minute system load average per CPU core.

Template Name	Description
Load Average per CPU Core (15 minutes)	15-minute system load average per CPU core.
CPU utilization	Average CPU consumption.
Number of CPUs running higher than a	Number of CPUs running at greater than K% utilization threshold
Free memory percentage	Free memory as a percent of total system memory.
Memory used percentage	Percentage of memory used.
Swap consumption	Swap space consumed (in megabytes).
Swap consumption percentage	Percentage of swap area consumed.
Disk Consumption	Disk space consumed (in megabytes).
Disk consumption percentage	Percentage of disk consumed.
Disk Available	Disk space available (in megabytes).
Disk busy percentage	Percentage of disk busy.
Most used disk percentage	Percentage used of the most utilized disk on the system.
Total table bloat on host	The total space wasted by tables on a host, in MB.
Highest table bloat on host	The most space wasted by a table on a host, in MB.
Average table bloat on host	The average space wasted by tables on host, in MB.
Table size on host	The size of tables on host, in MB.
Database size on host	The size of databases on host, in MB.
Number of ERRORS in the logfile on agent N in last X hours.	The number of ERRORS in the logfile on agent N in last X hours
Number of WARNINGS in the logfile on agent N in last X hours	The number of WARNINGS in the logfile on agent N in last X hours.
Number of WARNINGS or ERRORS in the logfile on agent N in last X hours	The number of WARNINGS or ERRORS in the logfile on agent N in last X hours.
Package version mismatch	Check for package version mismatch as per catalog.
Total materialized view bloat on host	The total space wasted by materialized views on a host, in MB.
Highest materialized view bloat on host	The most space wasted by a materialized view on a host, in MB.
Average materialized view bloat on host	The average space wasted by materialized views on host, in MB.
Materialized view size on host	The size of materialized views on host, in MB.
Agent Down	Specified agent is currently down.

Templates applicable on Server

Template Name	Description
Total table bloat in server	The total space wasted by tables in server, in MB.
Largest table (by multiple of unbloated size)	Largest table in server, calculated as a multiple of its own estimated unbloated size; exclude tables smaller than N MB.
Highest table bloat in server	The most space wasted by a table in server, in MB.
Average table bloat in server	The average space wasted by tables in server, in MB.
Table size in server	The size of tables in server, in MB.
Database size in server	The size of databases in server, in MB.

Template Name	Description
Number of WAL files	Total number of Write Ahead Log files.
Number of prepared transactions	Number of transactions in prepared state.
Total connections	Total number of connections in the server.
Total connections as percentage of max_connections	Total number of connections in the server as a percentage of maximum connections allowed on server, settings.
Unused, non-superuser connections	Number of unused, non-superuser connections on the server, user_info, settings.
Unused, non-superuser connections as percentage of max_connections	Number of unused, non-superuser connections on the server as a percentage of max_connections of max_connections, user_info, settings.
Ungranted locks	Number of ungranted locks in server.
Percentage of buffers written by backends	The percentage of buffers written by backends vs. the total buffers written.
Percentage of buffers written by checkpoint	The percentage of buffers written by the checkpoints vs. the total buffers written.
Buffers written per second	Number of buffers written per second, over the last two probe cycles.
Buffers allocated per second	Number of buffers allocated per second, over the last two probe cycles.
Connections in idle state	Number of connections in server that are in idle state.
Connections in idle-in-transaction state	Number of connections in server that are in idle-in-transaction state.
Connections in idle-in-transaction state, as percentage of max_connections	Number of connections in server that are in idle-in-transaction state, as a percentage of maximum connections allowed on server, settings.
Long-running idle connections	Number of connections in the server that have been idle for more than N seconds.
Long-running idle connections and idle transactions	Number of connections in the server that have been idle or transactions idle-in-transaction for more than N seconds.
Long-running idle transactions	Number of connections in the server that have been idle in transaction for more than N seconds.
Long-running transactions	Number of transactions in server that have been running for more than N seconds.
Long-running queries	Number of queries in server that have been running for more than N seconds.
Long-running vacuums	Number of vacuum operations in server that have been running for more than N seconds.
Long-running autovacuums	Number of autovacuum operations in server that have been running for more than N seconds.
Committed transactions percentage	Percentage of transactions in the server that committed vs. that rolled-back over last N minutes.
Shared buffers hit percentage	Percentage of block read requests in the server that were satisfied by shared buffers, over last N minutes.
Tuples inserted	Tuples inserted into server over last N minutes.
InfiniteCache buffers hit percentage	Percentage of block read requests in the server that were satisfied by InfiniteCache, over last N minutes.
Tuples fetched	Tuples fetched from server over last N minutes.
Tuples returned	Tuples returned from server over last N minutes.
Dead Tuples	Number of estimated dead tuples in server.
Tuples updated	Tuples updated in server over last N minutes.
Tuples deleted	Tuples deleted from server over last N minutes.

Template Name	Description
Tuples hot updated	Tuples hot updated in server, over last N minutes.
Sequential Scans	Number of full table scans in server, over last N minutes.
Index Scans	Number of index scans in server, over last N minutes.
Hot update percentage	Percentage of hot updates in the server over last N minutes.
Live Tuples	Number of estimated live tuples in server.
Dead tuples percentage	Percentage of estimated dead tuples in server.
Last Vacuum	Hours since last vacuum on the server.
Last AutoVacuum	Hours since last autovacuum on the server.
Last Analyze	Hours since last analyze on the server.
Last AutoAnalyze	Hours since last autoanalyze on the server.
Percentage of buffers written by backends over the last N minutes	The percentage of buffers written by backends vs. the total buffers backends over last N
Table Count	Total number of tables in server.
Function Count	Total number of functions in server.
Sequence Count	Total number of sequences in server.
A user expires in N days	Number of days before a user's validity expires.
Index size as a percentage of table size	Size of the indexes in server, as a percentage of their tables' size.
Largest index by table-size percentage oc_index, table_size.	Largest index in server, calculated as percentage of its table's size.
Number of ERRORS in the logfile on server M in the last X hours	The number of ERRORS in the logfile on server M in last X hours.
Number of WARNINGS in the logfile on server M in the last X hours	The number of WARNINGS in logfile on server M in the last X hours.
Number of WARNINGS or ERRORS in the logfile on server M in the last X hours	The number of WARNINGS or ERRORS in the logfile on server M in the last X hours.
Number of attacks detected in the last N minutes	The number of SQL injection attacks occurred in the last N minutes.
Number of attacks detected in the last N minutes by username	The number of SQL injection attacks occurred in the last N minutes by username.
Number of replica servers lag behind the primary by write location	> Streaming Replication: number of replica servers lag behind the primary by write location.
Number of replica servers lag behind the primary by flush location	> Streaming Replication: number of replica servers lag behind the primary by flush location.
Number of replica servers lag behind the primary by replay location	> Streaming Replication: number of replica servers lag behind the primary by replay location.
Replica server lag behind the primary by write location	> Streaming Replication: replica server lag behind the primary by write location in MB.
Replica server lag behind the primary by flush location	> Streaming Replication: replica server lag behind the primary by flush location in MB.
Replica server lag behind the primary by replay location	> Streaming Replication: replica server lag behind the primary by replay location in MB.
Replica server lag behind the primary by size (MB)	> Streaming Replication: replica server lag behind the primary by size in MB.
Replica server lag behind the primary by WAL segments	> Streaming Replication: replica server lag behind the primary by WAL segments.

Template Name	Description
Replica server lag behind the primary by WAL pages	> Streaming Replication: replica server lag behind the primary by WAL pages.
Total materialized view bloat in server	The total space wasted by materialized views in server, in MB.
Largest materialized view (by multiple of unbloated size)	Largest materialized view in server, calculated as a multiple of its own estimated unbloated size; exclude materialized views smaller than N MB.
Highest materialized view bloat in server	The most space wasted by a materialized view in server, in MB.
Average materialized view bloat in server	The average space wasted by materialized views in server, in MB.
Materialized view size in server	The size of materialized view in server, in MB.
View Count	Total number of views in server.
Materialized View Count	Total number of materialized views in server.
Audit config mismatch	Check for audit config parameter mismatch
Server Down	Specified server is currently inaccessible.
Number of WAL archives pending	Streaming Replication: number of WAL files pending to be replayed at replica.
Number of minutes lag of replica server from primary server	> Streaming Replication: number of minutes replica node is lagging behind the primary node.
Log config mismatch	Check for log config parameter mismatch.

Templates applicable on Database

Template Name	Description
Total table bloat in database	The total space wasted by tables in database, in MB.
Largest table (by multiple of unbloated size)	Largest table in database, calculated as a multiple of its own estimated unbloated size; exclude tables smaller than N MB.
Highest table bloat in database	The most space wasted by a table in database, in MB.
Average table bloat in database	The average space wasted by tables in database, in MB.
Table size in database	The size of tables in database, in MB.
Database size	The size of the database, in MB.
Total connections	Total number of connections in the database.
Total connections as percentage of max_connections	Total number of connections in the database as a percentage of maximum connections allowed on server, settings.
Ungranted locks	Number of ungranted locks in database.
Connections in idle state	Number of connections in database that are in idle state.
Connections in idle-in-transaction state	Number of connections in database that are in idle-in-transaction state
Connections in idle-in-transaction state,as percentage of max_connections	Number of connections in database that are in idle-in-transaction state, as a percentage of maximum connections allowed on server, settings.
Long-running idle connections	Number of connections in the database that have been idle for more than N seconds.
Long-running idle connections and idle transactions	Number of connections in the database that have been idle or idle-in-transaction for more than N seconds.
Long-running idle transactions	Number of connections in the database that have been idle in transaction for more than N seconds.

Template Name	Description
Long-running transactions	Number of transactions in database that have been running for more than N seconds.
Long-running queries	Number of queries in database that have been running for more than N seconds.
Long-running vacuums	Number of vacuum operations in database that have been running for more than N seconds.
Long-running autovacuums	Number of autovacuum operations in database that have been running for more than N seconds.
Committed transactions percentage	Percentage of transactions in the database that committed vs. that rolled-back over last N minutes.
Shared buffers hit percentage	Percentage of block read requests in the database that were satisfied by shared buffers, over last N minutes.
InfiniteCache buffers hit percentage	Percentage of block read requests in the database that were satisfied by InfiniteCache, over last N minutes.
Tuples fetched	Tuples fetched from database over last N minutes.
Tuples returned	Tuples returned from database over last N minutes.
Tuples inserted	Tuples inserted into database over last N minutes.
Tuples updated	Tuples updated in database over last N minutes.
Tuples deleted	Tuples deleted from database over last N minutes.
Tuples hot updated	Tuples hot updated in database, over last N minutes.
Sequential Scans	Number of full table scans in database, over last N minutes.
Index Scans	Number of index scans in database, over last N minutes.
Hot update percentage	Percentage of hot updates in the database over last N minutes.
Live Tuples	Number of estimated live tuples in database.
Dead Tuples	Number of estimated dead tuples in database.
Dead tuples percentage	Percentage of estimated dead tuples in database.
Last Vacuum	Hours since last vacuum on the database.
Last AutoVacuum	Hours since last autovacuum on the database.
Last Analyze	Hours since last analyze on the database.
Last AutoAnalyze	Hours since last autoanalyze on the database.
Table Count	Total number of tables in database.
Function Count	Total number of functions in database.
Sequence Count	Total number of sequences in database.
Index size as a percentage of table size	Size of the indexes in database, as a percentage of their tables' size.
Largest index by table-size percentage	Largest index in database, calculated as percentage of its table's size, oc_index, table_size.
Database Frozen XID	The age (in transactions before the current transaction) of the database's frozen transaction ID.
Number of attacks detected in the	The number of SQL injection attacks occurred in the last N minutes. last N minutes
Number of attacks detected in the	The number of SQL injection attacks occurred in the last N minutes by last N minutes by username.
Queries that have been cancelled due to dropped tablespaces	Streaming Replication: number of queries that have been cancelled due to dropped tablespaces.
Queries that have been cancelled due to lock timeouts	Streaming Replication: number of queries that have been cancelled due to lock timeouts.

Template Name	Description
Queries that have been cancelled due to old snapshots	Streaming Replication: number of queries that have been cancelled due to old snapshots.
Queries that have been cancelled due to pinned buffers	Streaming Replication: number of queries that have been cancelled due to pinned buffers.
Queries that have been cancelled due to deadlocks	Streaming Replication: number of queries that have been cancelled due to deadlocks.
Total events lagging in all slony clusters	Slony Replication: total events lagging in all slony clusters.
Events lagging in one slony cluster	Slony Replication: events lagging in one slony cluster.
Lag time (minutes) in one slony cluster	Slony Replication: lag time (minutes) in one slony cluster.
Total rows lagging in xdb single primary replication	> xDB Replication: Total rows lagging in xdb single primary replication
Total rows lagging in xdb multi primary replication	> xDB Replication: Total rows lagging in xdb multi primary replication.
Total materialized view bloat in database	The total space wasted by materialized views in database, in MB.
Largest materialized view (by multiple of unbloated size)	Largest materialized view in database, calculated as a multiple of its estimated unbloated size; exclude materialized views smaller than N MB.
Highest materialized view bloat in database	The most space wasted by a materialized view in database, in MB.
Average materialized view bloat in database	The average space wasted by materialized views in database, in MB.
Materialized view size in database	The size of materialized view in database, in MB.
View Count	Total number of views in database.
Materialized View Count	Total number of materialized views in database.

Templates applicable on Schema

Template Name	Description
Total table bloat in schema	The total space wasted by tables in schema, in MB.
Largest table (by multiple of unbloated size)	Largest table in schema, calculated as a multiple of its own estimated unbloated size; exclude tables smaller than N MB.
Highest table bloat in schema	The most space wasted by a table in schema, in MB.
Average table bloat in schema	The average space wasted by tables in schema, in MB.
Table size in schema	The size of tables in schema, in MB.
Tuples inserted	Tuples inserted in schema over last N minutes.
Tuples updated	Tuples updated in schema over last N minutes.
Tuples deleted	Tuples deleted from schema over last N minutes.
Tuples hot updated	Tuples hot updated in schema, over last N minutes.
Sequential Scans	Number of full table scans in schema, over last N minutes.
Index Scans	Number of index scans in schema, over last N minutes.
Hot update percentage	Percentage of hot updates in the schema over last N minutes.
Live Tuples	Number of estimated live tuples in schema.
Dead Tuples	Number of estimated dead tuples in schema.
Dead tuples percentage	Percentage of estimated dead tuples in schema.
Last Vacuum	Hours since last vacuum on the schema.

Template Name	Description
Last AutoVacuum	Hours since last autovacuum on the schema.
Last Analyze	Hours since last analyze on the schema.
Last AutoAnalyze	Hours since last autoanalyze on the schema.
Table Count	Total number of tables in schema.
Function Count	Total number of functions in schema.
Sequence Count	Total number of sequences in schema.
Index size as a percentage of table size	Size of the indexes in schema, as a percentage of their table's size.
Largest index by table-size percentage	Largest index in schema, calculated as percentage of its table's size, <code>oc_index, table_size</code>
Materialized View bloat	Space wasted by the materialized view, in MB.
Total materialized view bloat in schema	The total space wasted by materialized views in schema, in MB.
Materialized view size as a multiple of unbloated size	Size of the materialized view as a multiple of estimated unbloated size.
Largest materialized view (by multiple of unbloated size)	Largest materialized view in schema, calculated as a multiple of its own estimated unbloated size; exclude materialized view smaller than N MB.
Highest materialized view bloat in schema	The most space wasted by a materialized view in schema, in MB.
Average materialized view bloat in schema	The average space wasted by materialized views in schema, in MB.
Materialized view size	The size of materialized view, in MB.
Materialized view size in schema	The size of materialized views in schema, in MB.
View Count	Total number of views in schema.
Materialized View Count	Total number of materialized views in schema.
Materialized View Frozen XID	The age (in transactions before the current transaction) of the materialized view's frozen transaction ID.

Templates applicable on Table

Template Name	Description
Table bloat	Space wasted by the table, in MB.
Table size	The size of table, in MB.
Table size as a multiple of ubloated size	Size of the table as a multiple of estimated unbloated size.
Tuples inserted	Tuples inserted in table over last N minutes.
Tuples updated	Tuples updated in table over last N minutes.
Tuples deleted	Tuples deleted from table over last N minutes.
Tuples hot updated	Tuples hot updated in table, over last N minutes.
Sequential Scans	Number of full table scans on table, over last N minutes.
Index Scans	Number of index scans on table, over last N minutes.
Hot update percentage	Percentage of hot updates in the table over last N minutes.
Live Tuples	Number of estimated live tuples in table.

Template Name	Description
Dead Tuples	Number of estimated dead tuples in table.
Dead tuples percentage	Percentage of estimated dead tuples in table.
Last Vacuum	Hours since last vacuum on the table.
Last AutoVacuum	Hours since last autovacuum on the table.
Last Analyze	Hours since last analyze on the table.
Last AutoAnalyze	Hours since last autoanalyze on the table.
Row Count	Estimated number of rows in a table.
Index size as a percentage of table size	Size of the indexes on table, as a percentage of table's size.
Table Frozen XID	The age (in transactions before the current transaction) of the table's frozen transaction ID.

Global Templates

Template Name	Description
Agents Down	Number of agents that haven't reported in recently.
Servers Down	Number of servers that are currently inaccessible.
Alert Errors	Number of alerts in an error state.

5 PEM Installation Guide on Linux

Postgres Enterprise Manager (PEM) is designed to assist database administrators, system architects, and performance analysts when administering, monitoring, and tuning PostgreSQL and Advanced Server database servers. PEM has been designed to manage and monitor a single server or multiple servers from a single console, allowing complete control over monitored databases.

This document provides step-by-step instructions to guide you through the installation of Postgres Enterprise Manager on a Linux host.

Throughout this guide, the term *Postgres* refers to either a PostgreSQL or an Advanced Server installation, where either is appropriate.

5.1 What's New

The following changes have been made to Postgres Enterprise Manager to create version 8.0.1:

Improved the overall performance, usability, and stability of the product by:

- Allowing the dash (-) character in the superuser name that is provided during configuration.

- Gracefully closing the operating system resources during batch probe and command execution to avoid errors.
- Improved support for unicode string handling in pemAgent.
- Fixed a regression restoring BART backup when agent is not bound with the BART server.
- In addition to the above-listed items, a few other minor bugs have also been fixed.

PEM 8.0.1 provides an option to hide or unhide the Dashboard, Statistics, Dependents, and Dependencies tabs.

Features and Functionalities removed in recent versions of PEM:

- PEM 8.0 and later is not supported on CentOS/RHEL 6.x.
- EDB ARK UI Management is not supported by PEM 7.16 and later.
- PEM 7.16 and later is not supported on Internet explorer version 11 and lesser.

5.2 PEM - Hardware and Software Requirements

Hardware Prerequisites

For optimum performance when monitoring servers and rendering dashboards, we recommend installing PEM on a system with at least:

- 4 CPU cores
- 8 GB of RAM
- 100 GB of Storage

Additional disk space is required for data storage. Please note that resource usage will vary based on which probes are defined and enabled, and the activity level on the monitored databases. Monitoring server resources (as you use PEM) will let you know when you need to expand your initial system configuration.

Software Prerequisites

Platforms and Versions Support

For information about the platforms and versions supported by PEM, visit the EnterpriseDB website at:

<https://www.enterprisedb.com/services-support/edb-supported-products-and-platforms>

Please note: PEM 8.0.1 is no longer supported on CentOS/RHEL/OEL 6.x platforms. It is strongly recommended that EDB products running on these platforms be migrated to a supported platform.

Modifying the pg_hba.conf File

The `pg_hba.conf` file manages connections for the Postgres server. You must ensure that the `pg_hba.conf` file on each monitored server allows connections from the PEM server, the monitoring PEM Agent, and the host of the PEM-HTTPD server.

For information about modifying the `pg_hba.conf` file, see the [PEM Administrator's Guide](#).

Information about managing authentication is also available in the [Postgres core documentation](#).

Firewall Restrictions

Please note that you must adjust your firewall to allow communication between PEM components.

Supported Locales

Currently, the PEM server and web interface support a locale of **English(US) en_US** and use of a period (.) as a language separator character. Using an alternate locale, or a separator character other than a period may result in errors.

5.3 PEM Architecture

Postgres Enterprise Manager (PEM) is a tool designed to monitor and manage multiple Postgres servers through a single GUI interface. PEM is capable of monitoring the following areas of the infrastructure:

Note: The term Postgres refers to either PostgreSQL or EDB Postgres Advanced Server.

- Hosts - One or more servers (physical or virtual) and their operating systems.
- Servers - One or more instances of PostgreSQL or EDB Postgres Advanced Server running on a host.
- Databases - One or more databases and the schema objects (tables, indexes, etc.) within them.

PEM consists of a number of individual software components; the individual components are described below.

- PEM Server - The PEM Server is used as the data repository for monitoring data and as a server to which both Agents and Clients connect. The PEM server consists of an instance of PostgreSQL and an associated database for storage of monitoring data, and a server that provides web services.
- PEM Agent - The PEM Agent is responsible for executing tasks and reporting statistics from the Agent host and monitored Postgres instances to the PEM server. A single PEM Agent can monitor multiple installed instances of Postgres that reside on one or many hosts.
- PEM Web Client - The PEM web interface allows you to manage and monitor Postgres servers and utilize PEM extended functionality. The web interface software is installed with the PEM server and is accessed via any supported web browser.
- SQL Profiler - SQL Profiler is a Postgres server plugin to record the monitoring data and query plans to be analysed by the SQL Profiler tool in PEM. This is an optional component of PEM, but the plugin must be installed into each instance of Postgres with which you wish to use the SQL Profiler tool. The SQL Profiler may be used with any supported version of an EDB distribution of a PostgreSQL server or Advanced Server (not just those managed through the PEM server). See the [PEM SQL Profiler Configuration Guide](#) for details and supported versions.

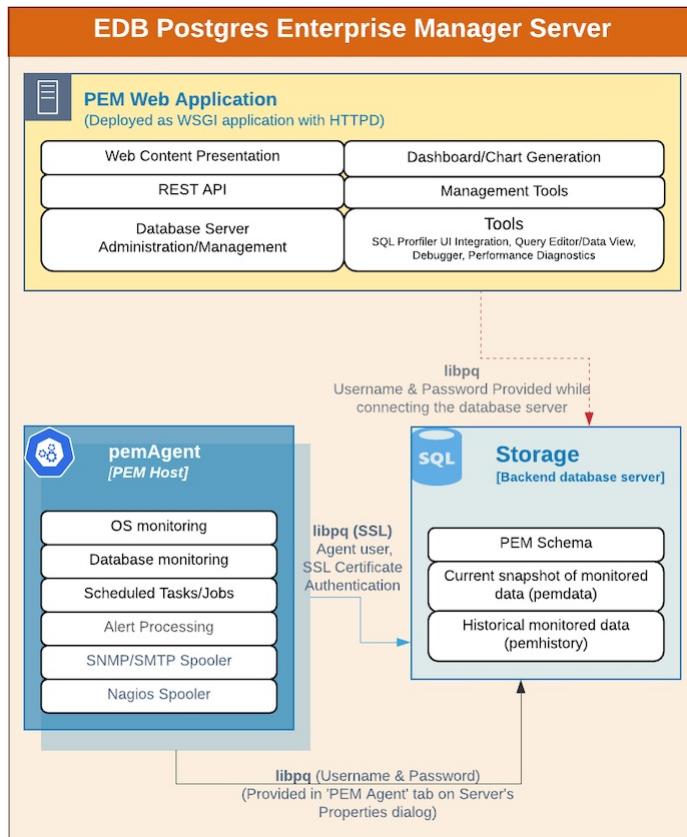
PEM architecture

The following architectural diagram illustrates the relationships between the PEM server, clients, and managed as well as unmanaged Postgres servers.



PEM Architecture

The PEM Server



PEM Server

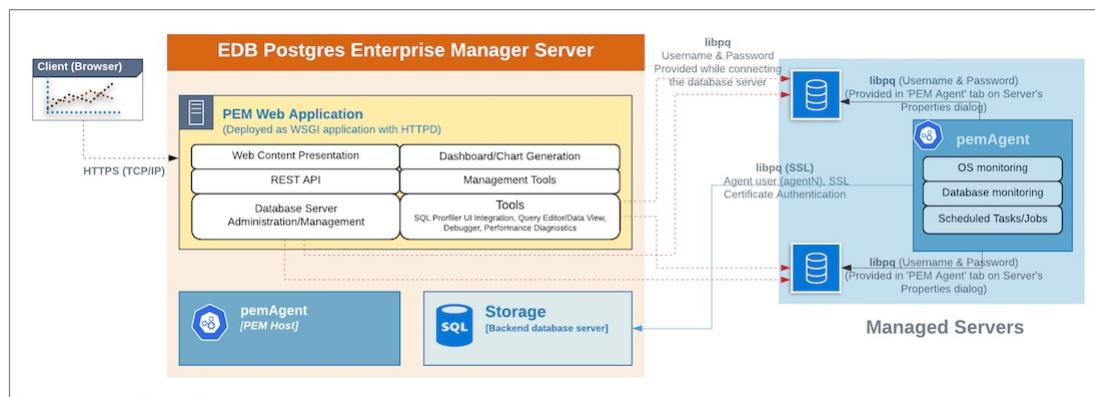
The PEM server consists of an instance of Postgres, an instance of the Apache web-server providing web services to the client, and a PEM Agent. PEM utilizes a server-side cryptographic plugin to generate authentication certificates.

The instance of Postgres (a database server) and an instance of the Apache web-server (HTTPD) can be on the same host or on separate hosts.

- **Postgres Instance (Database server)** - This is the backend database server. It hosts a database named pem which acts as the repository for PEM Server. The pem database contains several schemas that store metric data collected from each monitored host, server, and database.
 - pem - This schema is the core of the PEM application. It contains the definitions of configuration functions, tables, or views required by the application.
 - pemdata - This schema stores the current snapshot of the monitored data.
 - pemhistory - This schema stores the historical monitored data.
- **Apache Web Server (HTTPD)** - The PEM Web Application is deployed as a WSGI application with HTTPD to provide web services to the client. It is comprised of the following:
 - Web content presentation - The presentation layer is created by the Web Application (for example Browser, login page,..).
 - Rest API - The REST API allows integration with other apps and services.
 - Database Server Administration/Management - Database server administration and management activities like CREATE, ALTER, DROP, etc. can be performed for managed as well as unmanaged servers.
 - Dashboard/Chart generation - Internally, the web application includes functionality that generates Dashboards and Charts.
 - Management Tools - The Audit Manager, Capacity Manager, Log Manager, Postgres Expert, Postgres Log Analysis Expert, and the Tuning Wizard are made available in the Web Application.
 - Other tools provide functionality on managed or unmanaged servers:
 - SQL Profiler UI Integration - SQL Profiler generates easily analyzed traces of session content.
 - Query Editor/Data View - The Query editor allows you to query, edit, and view data.
 - Debugger - The Debugger helps you debug queries.
 - Performance Diagnostics - Performance Diagnostics help you analyze the performance of Postgres instances.

We recommend that you use a dedicated machine to host production instances of the PEM backend database. The host may be subject to high levels of data throughput, depending on the number of database servers that are being monitored and the workloads the servers are processing.

The PEM Agent



PEM Agent

The PEM Agent is responsible for the collection of monitoring data from the machine and operating system, as well as from each of the Postgres instances to which they are bound. Each PEM Agent can monitor one physical or virtual machine and is capable of monitoring multiple database servers locally - installed on the same system, or remotely - installed on other systems. It is also responsible for executing other tasks that may be scheduled by the user (for example, server shutdowns, SQL Profiler traces, user-defined jobs).

A PEM Agent is installed by default on the PEM Server along with the installation of the PEM Server. It is generally referred to as a PEM Agent on the PEM Host. Separately, the PEM Agent can also be installed on the other servers hosting the Postgres instances to be monitored using PEM.

Whether monitoring locally or remotely, the PEM Agent connects to the PEM Server using PostgreSQL's libpq, using SSL certificate-based authentication. The PEM Agent installer in Windows and pemworker CLI in Linux is responsible for registering each agent with the PEM Server, and generating and installing the required certificates.

Please note that there is only one-way traffic between the PEM Agent and PEM Server; the PEM Agent always connects to the PEM Server.

The PEM Agent must be able to connect to each database server that it monitors. This connection is made over a TCP/IP connection (or optionally a Unix Domain Socket on Unix hosts), and may optionally use SSL. The user must configure the connection and authentication to the monitored server.

Once configured, each agent collects statistics and other information on the host and each database server and database that it monitors. Each piece of information is known as a metric and is collected by a probe. Most probes will collect multiple metrics at once for efficiency. Examples of the metrics collected include:

- Disk I/O statistics
- Network statistics
- Database server version string
- Database server configuration option (GUC) values
- Table access statistics
- Table and index sizes

A list of PEM probes can be found [here](#).

By default, the PEM Agent bound to the database server collects the OS/Database monitoring statistics and also runs any scheduled tasks/jobs for that particular database server, storing data in the pem database on the PEM server.

The Alert processing, SNMP/SMTP spoolers, and Nagios Spooler data is stored in the `pem` database on the PEM server and is then processed by the PEM Agent on the PEM Host by default. However, processing by other PEM Agents can be enabled by adjusting the SNMP/SMTP and Nagios parameters of the PEM Agents.

To see more information about these parameters see [Server Configuration](#).

The PEM Web Client

The PEM client is a web-based application that runs in supported browsers. The client's web interface connects to the PEM server and allows direct management of managed or unmanaged servers, and the databases and schemas that reside on them.

The client allows you to use PEM functionality that makes use of the data logged on the server through features such as the dashboards, the Postgres Log Analysis Expert, and Capacity Manager.

The SQL Profiler Plugin

You are not required to install the SQL Profiler plugin on every server, but you must install and configure the plugin on each server on which you wish to use the SQL Profiler. You may also want to install and configure SQL Profiler on un-monitored development servers. For ad-hoc use also, you may temporarily install the SQL Profiler plugin.

The plugin is installed with the EDB Postgres Advanced Server distribution but must be installed separately for use with

PostgreSQL. The SQL Profiler installer is available from the [EDB website](#).

SQL Profiler may be used on servers that are not managed through PEM, but to perform scheduled traces, a server must have the plugin installed, and must be managed by an installed and configured PEM agent.

For more information about using SQL Profiler, see the [PEM SQL Profiler Guide](#).

5.4 Installing Postgres Enterprise Manager

The `edb-pem` package for Linux platforms installs the PEM Server, a PEM Agent, and the required software to connect to the PEM web interface with a supported browser.

The PEM server uses a Postgres installation and backend database to manage data. The `pem` backend database is created when you configure PEM.

For detailed information about installing the PEM Server, see [Installing the PEM Server on Linux](#). For information about configuring a PEM Server see [Configuring the PEM Server on Linux](#).

The PEM Agent that is installed with the PEM server is capable of monitoring multiple servers that reside on the same host, or on remote hosts. Please note that the PEM functionality on servers monitored by a remote Agent may be limited. For detailed information about remote monitoring functionality see the [PEM Agent Privileges](#).

For detailed information about installing and configuring a PEM Agent, see [Installing the PEM Agent on Linux](#).

5.4.1 Prerequisites for Installing the PEM Server on Linux Platforms

1. Install a backend database.

When installing a PEM server on a Linux host, you must first install a backend database cluster which will hold the `pem` database. The PEM server's backend database may be installed via package for Linux. The backend database must be one of the following versions:

- EDB Postgres Advanced Server version 11 or above
- PostgreSQL version 11 or above

For detailed information about installing an Advanced Server or PostgreSQL database, please see the product documentation at the EDB website.

2. Configure Postgres authentication on the backend database.

The `pg_hba.conf` file on the backend database can be configured to use any supported authentication methods (for example: md5, trust,...) for connections. For information about modifying the `pg_hba.conf` file, see the [PostgreSQL core documentation](#).

3. If you are using a PostgreSQL database, use the following command to install the `hstore contrib module`:

```
yum install postgresql<x>-contrib
```

Where, `x` is the server version.

4. Ensure that the `sslutils` extension is installed.

- On an Advanced Server backend database, the `sslutils` extension is installed by default.
- If you are using a PostgreSQL backend database, ensure you have access to the PostgreSQL community repository, and use the command:

```
yum install sslutils_<x>
```

Where, `x` is the server version.

Please note that Debian 10 and Ubuntu 20 has increased the requirements to accept the certificates due to security reason. If a user wants to install the PEM Agent on any of the machines, they must upgrade `sslutils` to 1.3 where 4096 bit RSA key and sha256 signature algorithm support has added. If the user does not upgrade `sslutils` to 1.3, then PEM Agent may fail to connect to the PEM backend database server, and it might log the error `ca md too weak`.

5. Adjust your firewall restrictions.

If you are using a firewall, you must allow access to port `8443` on the PEM backend database:

```
firewall-cmd --permanent --zone=public --add-port=8443/tcp
firewall-cmd --reload
```

6. Request credentials that allow you to access the EDB repositories: To install the PEM Server, you must have credentials that allow access to the EDB repository. To request credentials for the repository, contact [EDB](#). When using commands in the sections that follow, replace the `username` and `password` placeholders with the credentials provided by EDB
7. PEM is dependent on third-party components from the vendor repository, including the `python3`, `libboost`, `openssl`, `snmp++`, `libcurl`, etc. To ensure these components are up to date, you should update your operating system using following platform-specific commands. Minimum version require for `openssl` is 1.0.2k.

To upgrade packages on a CentOS or RHEL 7.x host

```
yum upgrade
```

To upgrade packages on a CentOS or RHEL 8.x host

```
dnf upgrade
```

To upgrade packages on a Debian or Ubuntu host

```
apt-get update
```

To upgrade packages on a SLES host

```
zypper update
```

5.4.2 Web Server Hosting Preferences

During the PEM server installation, you can specify your hosting preferences for the Apache Web Server(PEM-HTTPD):

To install the PEM Server and Apache Web Server (PEM-HTTPD) on the same host

Follow the installation steps; while running the configuration script, select the **Web Services and Database** option to install PEM Server and Apache Web Server on the same host.

To install the PEM Server and Apache Web Server (PEM-HTTPD) on separate hosts

Follow the installation steps on both the hosts. While running the configuration script, first configure the PEM Server host by selecting the **Database** option on first host and then configure an Apache Web Server (PEM-HTTPD) by selecting the **Web Services** option on the second host.

For detailed information about configuring a PEM Server, see [Configuring the PEM Server on Linux Platforms](#).

5.4.3 Installing the PEM Server on Linux Platforms

Before following the detailed instructions that install the PEM server on your specific platform, you must perform the prerequisite steps detailed in [Prerequisites for installing PEM Server](#).

Installing the PEM Server on a CentOS or RHEL Host

On a CentOS or RHEL system, you can use the **yum** package manager or **dnf** command to install a PEM Server; the installation tool you use will be dependent on the version of the host operating system. Before installing the server, you must ensure that your system contains the required prerequisite software listed below.

Install Version-Specific Software

Follow the version-specific instructions listed below to prepare your host system.

- On a CentOS or RHEL 7.x host:

```
yum -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
```

- On a CentOS or RHEL 8.x host:

```
yum -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm
```

!!! Note You may need to enable the **[extras]** repository definition in the **CentOS-Base.repo** file (located in **/etc/yum.repos.d**).

If you are a Red Hat Network user

You must enable the **rhel-<x>-server-extras-rpms** repository, where **x** specifies the RHEL version.

You must also enable the `rhel-<x>-server-optional-rpms` repository to use EPEL packages, where `x` specifies the RHEL version. You can make the repository accessible by enabling the `RHEL optional subchannel` for `RHN-Classic`. If you have a certificate-based subscription, then you must also enable `rhel-<x>-server-eus-optional-rpms` repository to use EPEL packages; please see the [Red Hat Subscription Management Guide](#) for more information about the required repositories.

Install and Configure the `edb.repo` File

To create an EnterpriseDB repository configuration file, assume superuser privileges and invoke the following command:

```
yum -y install https://yum.enterprisedb.com/edb-repo-rpms/edb-repo-latest.noarch.rpm
```

The repository configuration file is named `edb.repo`. The file resides in `/etc/yum.repos.d`. After creating the `edb.repo` file, use the following command to replace the `USERNAME` and `PASSWORD` placeholders in the `baseurl` specification with the `<username>` and `<password>` of a registered EDB user:

```
sed -i "s@<username>:<password>@USERNAME:PASSWORD@" /etc/yum.repos.d/edb.repo
```

Install the PEM Server

After meeting the platform-specific prerequisites listed above, you can use `yum` or `dnf` to install the PEM Server:

- On CentOS or RHEL 7.x or 8.x, you can use `yum` to install the PEM server :

```
yum install edb-pem
```

- On CentOS or RHEL 8.x, you can use `dnf` to install the PEM Server:

```
dnf install edb-pem
```

If you are doing a fresh installation of the PEM Server on a CentOS or RHEL 7.x host, the installer will also install `edb-python3-mod_wsgi` packages along with the installation as per requirement of the operating system.

If you are upgrading the PEM Server on a CentOS or RHEL 7.x host, the `mod_wsgi system` package will be replaced by the `edb-python3-mod_wsgi` package as per the requirement of the operating system.

When you install an RPM package that is signed by a source that is not recognized by your system, yum may ask for your permission to import the key to your local server. If prompted, and you are satisfied that the packages come from a trustworthy source, enter `y`, and press `Return` to continue.

During the installation, yum may encounter a dependency that it cannot resolve. If it does, it will provide a list of the required dependencies that you must manually resolve.

If you want to install PEM server on a machine that is in isolated network, you must first create PEM repository on that machine. For more information about creating PEM repository on an isolated network, see [Creating a PEM repository in an Isolated Network](#).

After installing PEM Server using `yum` or `dnf`, you need to configure the PEM Server. For more detailed information see [Configuring the PEM Server on Linux platforms](#).

Installing the PEM Server on a Debian or Ubuntu Host

The following steps will walk you through using the EDB `apt` repository to install a Debian package.

!!! Note You can also visit <https://repos.enterprisedb.com/> and select the platform and product to view the steps for installation.

1. Log in as root on your Debian or Ubuntu host:

```
sudo su -
```

2. Configure the EDB repository:

- For Debian 9:

```
sh -c 'echo "deb https://username:password@apt.enterprisedb.com $(lsb_release -cs)-edb/ $(lsb_release -cs) main" > /etc/apt/sources.list.d/edb-$(lsb_release -cs).list'
```

Where `username:password` is to be replaced by the credentials provided by EDB.

- For Debian 10:

- a. Set up the EnterpriseDB repository:

```
sh -c 'echo "deb [arch=amd64] https://apt.enterprisedb.com/$(lsb_release -cs)-edb/ $(lsb_release -cs) main" > /etc/apt/sources.list.d/edb-$(lsb_release -cs).list'
```

- b. Substitute your EnterpriseDB credentials for the `username` and `password` placeholders in the following command:

```
sh -c 'echo "machine apt.enterprisedb.com login <username> password <password>" > /etc/apt/auth.conf.d/edb.conf'
```

3. Add support to your system for secure APT repositories:

```
apt-get install apt-transport-https
```

4. Add the EBD signing key:

```
wget -q -O -https://username:password@apt.enterprisedb.com/edb-deb.gpg.key | apt-key add -`
```

5. Update the repository metadata:

```
apt-get update
```

6. Use the following command to install the Debian package for the PEM server:

```
apt-get install edb-pem
```

When the installation completes, you must configure the PEM Server. For detailed information see [Configuring the PEM Server on Linux Platforms](#).

Installing the PEM Server on a SLES Host

Use the following command to add the EDB repository configuration files to your SLES host:

```
zypper addrepo https://zypp.enterpriseedb.com/suse/edb-sles.repo
```

The command creates a repository configuration file named `edb.repo` in the `/etc/zypp/repos.d` directory. Modify the repository configuration file, adding the username and password of a registered EDB user.

Before installing PEM, you must install prerequisite packages. Invoke the following commands, replacing `sp_no` with the service pack that you are using (i.e. SP4):

```
SUSEConnect -p sle-module-legacy/12/x86_64
```

```
SUSEConnect -p sle-sdk/12/x86_64
```

```
zypper addrepo  
https://download.opensuse.org/repositories/Apache:Modules/SLE_12_<sp_no>/Apache:Modul  
o
```

```
zypper addrepo  
http://download.opensuse.org/repositories/Cloud:/OpenStack:/Newton:/cisco-  
apic:/2.3.1/SLE_12_<sp_no>/ pem_opensuse_boost
```

Then, refresh the repository and install the PEM server:

```
zypper refresh
```

```
zypper install edb-pem
```

After installing PEM Server using `zypper`, you must configure the PEM Server. For detailed information see [Configuring the PEM Server on Linux Platforms](#).

5.4.4 Creating a PEM Repository on an Isolated Network

You can create a local repository to act as a host for the PEM RPM packages if the server on which you wish to upgrade PEM cannot directly access the EnterpriseDB repository. Please note that this is a high-level overview of the steps required; you may need to modify the process for your individual network. To create and use a local repository, you must:

1. Use the following commands on a system with Internet access to download the dependencies for PEM:

```
yum install yum-plugin-downloaddir  
  
mkdir <pem_dir>  
  
yum install --downloaddir=<pem_dir>/ edb-pem  
  
mkdir <epel_dir>  
  
yum install --downloaddir=<epel_dir>/ epel-release*
```

Where `<pem_dir>` and `<epel_dir>` are the local directories that you create for downloading the RPMs.

2. Copy the directories `<pem_dir>` and `<epel_dir>` to the machine that is in the isolated network.

3. Create the repositories:

```
createrepo /<pem_dir>
createrepo /<epel_dir>
```

4. Create a repository configuration file called `/etc/yum.repos.d/pem.repo` with connection information that specifies:

```
[pemrepo]
name=PEM Repository
baseurl=file:///pem7/
enabled=1
gpgcheck=0
```

5. Create a repository configuration file called `/etc/yum.repos.d/epel.repo` with connection information that specifies:

```
[epelrepo]
name=epel Repository
baseurl=file:///pem7/
enabled=1
gpgcheck=0
```

6. After specifying the location and connection information for your local repository, you can use `yum` commands to install or upgrade PEM server:

To install PEM server:

```
yum install edb-pem
```

To upgrade PEM server:

```
yum upgrade edb-pem
```

For more information about creating a local yum repository, visit: <https://wiki.centos.org/HowTos/CreateLocalRepos>

5.4.5 Configuring the PEM Server on Linux Platforms

Before configuring the PEM server, ensure that the `sslutils` extension and `hstore` contrib module are installed on your backend database.

- For an Advanced Server backend database, the `sslutils` extension and `hstore` contrib module are by default installed along with Advanced Server.
- For a PostgreSQL backend database, ensure you have access to the PostgreSQL community repository, and then install `sslutils` extension and `hstore` contrib module with the command:

```
yum install sslutils_<x> postgresql<X>-contrib
```

Where, `x` is the server version.

The PEM server package includes a script (`configure-pem-server.sh`) to help automate the configuration process for Linux platform installations. The script is installed in the `/usr/edb/pem/bin` directory. To invoke the script, use the command:

```
/usr/edb/pem/bin/configure-pem-server.sh
```

When invoking the script, you can include command line options to specify configuration properties; the script will prompt you for values that you omit on the command line. The accepted options are:

!!! Note If you are using the SSL certificates then make sure that all the SSL certificates are inside the data directory the backend database server. If the certificates are not inside the data directory then the PEM Server's configure script may fail as it looks in to the data directory while configuring the PEM Server.

Option Description

<code>-acp</code>	Defines PEM Agent certificate path. The default is <code>/root/.pem</code> .
<code>-ci</code>	CIDR formatted network address range that Agents will connect to the server from, to be added to the server's <code>pg_hba.conf</code> file. For example, <code>192.168.1.0/24</code> . The default is <code>0.0.0.0/0</code> .
<code>-dbi</code>	The directory for the database server installation. For example, <code>/usr/edb/as12</code> for Advanced Server or <code>/usr/pgsql-12</code> for PostgreSQL.
<code>-ds</code>	The unit file name of the PEM database server. For Advanced Server, the default file name is <code>edb-as-12</code> ; for PostgreSQL, it is <code>postgresql-12</code> .
<code>-ho</code>	The host address of the PEM database server.
<code>-p</code>	The port number of the PEM database server.
<code>-ps</code>	The service name of the pemagent; the default value is <code>pemagent</code> .
<code>-sp</code>	The superuser password of the PEM database server. This value is required.
<code>-su</code>	The superuser name of the PEM database server.
<code>-t</code>	The installation type: Specify 1 if the configuration is for web services and backend database, 2 if you are configuring web services, or 3 if you are configuring the backend database. If you specify 3, please note that the database must reside on the local host.
<code>-un</code>	To unregister the PEM Server.

If you do not provide configuration properties on the command line, you will be prompted for values by the script. When you invoke the script, choose from:

1. **Web Services and Database** -Select this option if the web server and database both reside on the same host as the PEM server.
2. **Web Services** -Select this option if the web server resides on a different host than the PEM server.
3. **Database** -Select this option to configure the PEM backend database for use by the PEM server. Please note that the specified database must reside on the local host.

!!! Note If the web server (PEM-HTTPD) and the backend database (PEM Server) reside on separate hosts, configure the database server first (option 3), and then web services (option 2). The script will exit if the backend database is not configured before web services.

After selecting a configuration option, the script will proceed to prompt you for configuration properties. When the script completes, it will create the objects required by the PEM server, or perform the configuration steps required. 1 To view script-related help, use the command:

```
/usr/edb/pem/bin/configure-pem-server.sh -help
```

After configuring the PEM server, you can access the PEM web interface in your browser. Navigate to:

```
https://<ip_address_of_PEM_server>:8443/pem
```

5.4.6 Installing a PEM Agent on Linux Platforms

A PEM Agent may monitor one or more servers on one or more hosts. For comprehensive information about managing a PEM Agent, see the [PEM Agent User Guide](#).

Installing a PEM Agent on a CentOS or RHEL Host

On a CentOS or RHEL system, you can use the `yum` package manager or `dnf` command to install a PEM Agent; the installation tool you use will be dependent on the version of the host operating system. Before installing the agent, you must ensure that your system contains the required prerequisite software listed below.

Install Platform-Specific Software

- On a CentOS or RHEL 7.x host:

```
yum -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
```

- On a CentOS or RHEL 8.x host:

```
yum -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm
```

!!! Note You may need to enable the `[extras]` repository definition in the `CentOS-Base.repo` file (located in `/etc/yum.repos.d`).

If you are a Red Hat Network user

You must enable the `rhel-<x>-server-extras-rpms` repository, where `x` specifies the RHEL version.

You must also enable the `rhel-<x>-server-optional-rpms` repository to use EPEL packages, where `x` specifies the RHEL version. You can make the repository accessible by enabling the `RHEL optional subchannel` for `RHN-Classic`. If you have a certificate-based subscription, then you must also enable `rhel-<x>-server-eus-optional-rpms` repository to use EPEL packages; please see the [Red Hat Subscription Management Guide](#) for more information about the required repositories.

Install and Configure the `edb.repo` File

To create an EnterpriseDB repository configuration file, assume superuser privileges and invoke the following command:

```
yum -y install https://yum.enterprisedb.com/edb-repo-rpms/edb-repo-latest.noarch.rpm
```

The repository configuration file is named `edb.repo`. The file resides in `/etc/yum.repos.d`. After creating the `edb.repo` file, use the following command to replace the `USERNAME` and `PASSWORD` placeholders in the `baseurl` specification with the `<username>` and `<password>` of a registered EDB user:

```
sed -i "s@<username>:<password>@USERNAME:PASSWORD@" /etc/yum.repos.d/edb.repo
```

Installing the PEM Agent

- Use the `yum install` command to install the PEM Agent on CentOS or RHEL 7.x or 8.x:

```
yum install edb-pem-agent
```

- Also, you can use `dnf` command to install the PEM Agent on CentOS or RHEL 8.x:

```
dnf install edb-pem-agent
```

When the installation is complete, you can review a list of the installed packages and dependencies.

```
root@localhost:/etc/yum.repos.d
File Edit View Search Terminal Help
Is this ok [y/N]: y
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : edb-as12-server-libs-12.2.3-1.rhel7.x86_64 1/8
  Installing : libcurl-pem-7.61.1-2.rhel7.x86_64 2/8
  Installing : boost-atomic-1.53.0-27.el7.x86_64 3/8
  Installing : boost-program-options-1.53.0-27.el7.x86_64 4/8
  Installing : snmp++-3.3.8-1.rhel7.x86_64 5/8
  Installing : boost-regex-1.53.0-27.el7.x86_64 6/8
  Installing : boost-chrono-1.53.0-27.el7.x86_64 7/8
  Installing : edb-pem-agent-7.12.0-2.rhel7.x86_64 8/8
  Verifying : boost-chrono-1.53.0-27.el7.x86_64 1/8
  Verifying : boost-regex-1.53.0-27.el7.x86_64 2/8
  Verifying : snmp++-3.3.8-1.rhel7.x86_64 3/8
  Verifying : boost-program-options-1.53.0-27.el7.x86_64 4/8
  Verifying : boost-atomic-1.53.0-27.el7.x86_64 5/8
  Verifying : edb-pem-agent-7.12.0-2.rhel7.x86_64 6/8
  Verifying : libcurl-pem-7.61.1-2.rhel7.x86_64 7/8
  Verifying : edb-as12-server-libs-12.2.3-1.rhel7.x86_64 8/8

Installed:
  edb-pem-agent.x86_64 0:7.12.0-2.rhel7

Dependency Installed:
  boost-atomic.x86_64 0:1.53.0-27.el7           boost-chrono.x86_64 0:1.53.0-27.el7
  boost-program-options.x86_64 0:1.53.0-27.el7    boost-regex.x86_64 0:1.53.0-27.el7
  edb-as12-server-libs.x86_64 0:12.2.3-1.rhel7   libcurl-pem.x86_64 0:7.61.1-2.rhel7
  snmp++.x86_64 0:3.3.8-1.rhel7

Complete!
[root@localhost yum.repos.d]#
[root@localhost yum.repos.d]#
```

When you install an RPM package that is signed by a source that is not recognized by your system, yum may ask for your permission to import the key to your local server. If prompted, and you are satisfied that the packages come from a trustworthy source, enter `y`, and press `Return` to continue.

During the installation, yum may encounter a dependency that it cannot resolve. If it does, it will provide a list of the required dependencies that you must manually resolve.

After installing PEM Agent using `yum` or `dnf`, you need to register the PEM Agent. For detailed information see

Registering a PEM Agent.

Installing a PEM Agent on a Debian or Ubuntu Host

To install PEM on a Debian or Ubuntu host, you must have credentials that allow access to the EDB repository. To request credentials for the repository, [contact EDB](#).

The following steps will walk you through using the EDB apt repository to install a Debian package. When using the commands, replace the username and password with the credentials provided by EDB.

1. Log in as root:

```
sudo su -
```

2. Configure the EDB repository:

- For Debian 9:

```
sh -c 'echo "deb https://username:password@apt.enterprisedb.com $(lsb_release -cs)-edb/ $(lsb_release -cs) main" > /etc/apt/sources.list.d/edb-$(lsb_release -cs).list'
```

Where `username:password` is to be replaced by the credentials provided by EDB.

- For Debian 10:

- a. Set up the EnterpriseDB repository:

```
sh -c 'echo "deb [arch=amd64] https://apt.enterprisedb.com/$(lsb_release -cs)-edb/ $(lsb_release -cs) main" > /etc/apt/sources.list.d/edb-$(lsb_release -cs).list'
```

- b. Substitute your EnterpriseDB credentials for the `username` and `password` placeholders in the following command:

```
sh -c 'echo "machine apt.enterprisedb.com login <username> password <password>" > /etc/apt/auth.conf.d/edb.conf'
```

3. Add support to your system for secure APT repositories:

```
apt-get install apt-transport-https
```

4. Add the EBD signing key:

```
wget -q -O -https://username:password@apt.enterprisedb.com/edb-deb.gpg.key | apt-key add -
```

5. Update the repository metadata:

```
apt-get update
```

6. Use the following command to install the Debian package for the PEM Agent:

```
apt-get install edb-pem-agent
```

After installing PEM Agent using `apt-get`, you need to register the PEM Agent. For more detailed information see [Registering a PEM Agent](#).

Installing a PEM Agent on a SLES Host

Use the following command to add the EDB repository configuration files to your SLES host:

```
zypper addrepo https://zypp.enterprisedb.com/suse/edb-sles.repo
```

The command creates a repository configuration file named `edb.repo` in the `/etc/zypp/repos.d` directory. Modify the repository configuration file, adding the username and password of a registered EDB user.

Before installing PEM, you must install prerequisite packages. Use the following commands replacing `sp_no` with the service pack that you are using (i.e. SP2 or SP3):

```
SUSEConnect -p sle-module-legacy/12/x86_64
```

```
SUSEConnect -p sle-sdk/12/x86_64
```

```
zypper addrepo  
https://download.opensuse.org/repositories/Apache:Modules/SLE_12_<sp_no>/Apache:Modul  
o
```

```
zypper addrepo  
http://download.opensuse.org/repositories/Cloud:/OpenStack:/Newton:/cisco-  
apic:/2.3.1/SLE_12_<sp_no>/ pem_opensuse_boost
```

Then, you can refresh the repository and add a PEM agent:

```
zypper refresh
```

```
zypper install edb-pem-agent
```

After installing the PEM Agent, you must register the agent. For more detailed information see [Registering a PEM Agent](#).

5.4.7 Registering a PEM Agent

Each PEM Agent must be *registered* with the PEM Server. The registration process provides the PEM server with the information it needs to communicate with the Agent. You can use the `pemworker` utility to register the Agent if you use the package to install a PEM Agent.

The PEM Agent package places the PEM Agent in the `/usr/edb/pem/agent/bin` directory. To register an Agent, include the `--register-agent` keywords along with registration details when invoking the `pemworker` utility:

```
pemworker --register-agent
```

Append command line options to the command string when invoking the `pemworker` utility. Each option should be followed by a corresponding value:

Option	Description
--pem-server	Specifies the IP address of the PEM backend database server. This parameter is required.
--pem-port	Specifies the port of the PEM backend database server. The default value is 5432.
--pem-user	Specifies the name of the Database user (having superuser privileges) of the PEM backend database server. This parameter is required.
--pem-agent-user	Specifies the Agent user to connect the PEM server backend database server.
--cert-path	Specifies the complete path to the directory in which certificates will be created. If you do not provide a path, certificates will be created in: On Linux, <code>~/.pem</code> On Windows, <code>%APPDATA%/pem</code>
--config-dir	Specifies the directory path where configuration file can be found. The default is the <code><pemworker path>/..etc</code> .
--display-name	Specifies a user-friendly name for the Agent that will be displayed in the PEM Browser tree control. The default is the system hostname.
--force-registration	Include the force_registration clause to instruct the PEM server to register the Agent with the arguments provided; this clause is useful if you are overriding an existing Agent configuration. The default value is Yes.
--group	The name of the group in which the Agent will be displayed.
--team	The name of the database role, on the PEM backend database server, that should have access to the monitored database server.
--owner	The name of the database user, on the PEM backend database server, who will own the Agent.
--allow_server_restart	Enable the allow-server_restart parameter to allow PEM to restart the monitored server. The default value is True.
--allow-batch-probes	Enable the allow-batch-probes parameter to allow PEM to run batch probes on this Agent. The default value is False.
--batch-script-user	Specifies the operating system user that should be used for executing the batch/shell scripts. The default value is none; the scripts will not be executed if you leave this parameter blank or the specified user does not exist.
--enable-heartbeat-connection	Enable the enable-heartbeat-connection parameter to create a dedicated heartbeat connection between PEM Agent and server to update the active status. The default value is False.
--enable-smtp	When set to true for multiple PEM Agents (7.13 or lesser) it may send more duplicate emails. Whereas for PEM Agents (7.14 or higher) it may send lesser duplicate emails.
--enable-snmp	When set to true for multiple PEM Agents (7.13 or lesser) it may send more duplicate traps. Whereas for PEM Agents (7.14 or higher) it may send lesser duplicate traps.
-o	Specify if you want to override the configuration file options.

If you want to use any PEM feature for which a database server `restart` is required by the `pemagent` (such as Audit Manager, Log Manager, or the Tuning Wizard), then you must set the value for `allow_server_restart` to `true` in the `agent.cfg` file.

!!! Note When configuring a shell/batch script run by a PEM Agent that has PEM version 7.11 or later installed, the user for the `batch_script_user` parameter must be specified. It is strongly recommended that a non-root user is used to run the scripts. Using the root user may result in compromising the data security and operating system security. However, if you want to restore the `pemagent` to its original settings using the `root` user to run the scripts, then the `batch_script_user` parameter value must be set to `root`.

Before any changes are made on the PEM database, the connecting agent is authenticated with the PEM database

server. When invoking the `pemworker` utility, you must provide the password associated with the PEM server administrative user role (`postgres`). There are three ways to specify the administrative password; you can:

- set the `PEM_MONITORED_SERVER_PASSWORD` environment variable.
- provide the password on the command line with the `PGPASSWORD` keyword.
- create an entry in the `.pgpass` file.

Failure to provide the password will result in a password authentication error; you will be prompted for any other required but omitted information. When the registration is complete, the server will confirm that the Agent has been successfully registered.

Setting PEM Agent Configuration Parameters

The PEM Agent native package creates a sample configuration file named `agent.cfg.sample` in the `/usr/edb/pem/agent/etc` directory. When you register the PEM Agent, the `pemworker` program creates the actual Agent configuration file (named `agent.cfg`). Modify the `agent.cfg` file, changing the following configuration parameter value to `true`:

```
heartbeat_connection = true
```

By default, `heartbeat_connection` value is `false` but you can override the value during pemagent registration with `pemworker` utility using the `--enable-heartbeat-connection` option.

Then, use a platform-specific command to start the PEM Agent service;

On a CentOS or RHEL 7.x or 8.x host, use `systemctl` to start the service:

```
systemctl start pemagent
```

The service will confirm that it is starting the Agent; when the Agent is registered and started, it will be displayed on the [Global Overview](#) and in the [Object browser](#) of the PEM web interface.

For information about using the `pemworker` utility to register a server, please see the [PEM Administrator's Guide](#)

5.5 The PEM Web Interface

After installing a PEM server and Agent, you can configure PEM to start monitoring and managing PostgreSQL or Advanced Server instances. The PEM server installer installs the PEM web interface. You can use the interface to review information about objects that reside on monitored servers, or to review statistical information gathered by the PEM server.

After installing and configuring PEM, you can use your browser to access the PEM web interface. Open your browser, and navigate to:

```
https://<ip_address_of_PEM_host>:8443/pem
```

Where `ip_address_of_PEM_host` specifies the IP address of the host of the PEM server. The [Postgres Enterprise Manager Web Login](#) window opens:



Use the fields on the [Postgres Enterprise Manager Login](#) window to authenticate yourself with the PEM server:

- Provide the name of a [pem](#) database user in the [Username](#) field. For the first user connecting, this will be the name provided when installing the PEM server.
- Provide the password associated with the user in the [Password](#) field.

Click the [Login](#) button to connect to the PEM server.

Blackout	Status	Name	Connections	Alerts	Version	Processes	Threads	CPU Utilisation (%)	Memory Utilisation (%)	Swap Utilisation (%)	Disk Utilisation
<input type="checkbox"/>	✔ UP	Postgres Enterprise Manager Host	0	7.14.0-dev	309	810	24.85	77.18	17.88	45.84	
<input type="checkbox"/>	✔ UP	PEM Agent on Remote Host	0	7.13.0	207	524	0.35	51.73	3.03	24.30	

Blackout	Status	Name	Connections	Alerts	Version						Remotely Monitored?
<input type="checkbox"/>	✔ UP	Postgres Enterprise Manager Server	12	6	PostgreSQL 12.1 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-39), 64-bit						No
<input type="checkbox"/>	✔ UP	EDB Postgres Advanced Server 11	3	3	PostgreSQL 11.7 (EnterpriseDB Advanced Server 11.7.14) on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-36), 64-bit						No
<input type="checkbox"/>	⚠ DOWN	PGSQL12_Centos7_1	0	0	PostgreSQL 12.2 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-39), 64-bit						Yes
<input type="checkbox"/>	✔ UP	EPAS_12	6	5	PostgreSQL 12.2 (EnterpriseDB Advanced Server 12.2.3) on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-36), 64-bit						No

Alarm Type	Object Description	Alert Name	Value	Database	Schema	Package	Object	Alerting Since
▶ ● High	EDB Postgres Advanced Server 11	Last Vacuum	Never ran					2020-04-21 21:26:54
▶ ● High	EDB Postgres Advanced Server 11	Last AutoVacuum	177.03 hrs					2020-04-22 12:04:05
▶ ● High	EDB Postgres Advanced Server 11	Database size in server	113 MB					2020-04-22 11:50:00
▶ ● High	EPAS_12	Server Down	1					2020-04-29 09:11:09
▶ ● High	EPAS_12	Table size in server	427 MB					2020-04-09 15:53:51
▶ ● High	EPAS_12	Last Vacuum	15.39 hrs					2020-04-29 08:19:11
▶ ● High	EPAS_12	Database size in server	473 MB					2020-04-09 15:52:50
▶ ● High	EPAS_12	Last AutoVacuum	15.38 hrs					2020-04-29 08:19:11
▶ ● High	N/A	Alert Errors	3					2020-01-21 14:26:04
▶ ● High	PGSQL12_Centos7_1	Server Down	1					2020-04-29 08:54:02
▶ ● High	PGSQL12_Centos7_1	Last Vacuum	Never ran					2020-04-03 14:58:57
▶ ● High	PGSQL12_Centos7_1	Last AutoVacuum	Never ran					2020-04-03 14:58:57
▶ ● High	Postgres Enterprise Manager Server	Largest index by table-size percentage	100 %					2020-04-21 22:07:52
▶ ● High	Postgres Enterprise Manager Server	Database size in server	2.748046875 GB					2020-02-05 18:26:49
▶ ● Medium	Postgres Enterprise Manager Server	Total table bloat in server	88.28 MB					2020-04-29 08:36:18
▶ ● High	Postgres Enterprise Manager Server	Table size in server	2.6591796875 GB					2020-02-20 11:29:45
▶ ● High	Postgres Enterprise Manager Server	Connections in idle state	17					2020-04-29 09:05:07
▶ ● High	Postgres Enterprise Manager Server	Last Vacuum	41.46 hrs					2020-04-28 09:38:02

Before you can use the PEM web interface to manage or monitor a database server, you must *register* the server with the PEM server. When you register a server, you describe the connection to the server, provide authentication information for the connection, and specify any management preferences (optionally binding an Agent).

A server may be managed or unmanaged:

- A **managed** server is bound to a PEM Agent. The PEM Agent will monitor the server to which it is bound, and perform tasks or report statistics for display on the PEM dashboards. A managed server has access to extended PEM functionality such as Custom Alerting; when registering a server, you can also allow a managed server to be restarted by PEM as required.
- An **unmanaged** server is not bound to a PEM Agent; you can create database objects on an unmanaged server, but extended PEM functionality (such as Custom Alerting) is not supported on an unmanaged server.

You must also ensure the **pg_hba.conf** file of the server that you are registering allows connections from the host of the PEM web interface.

To access online help information about the PEM web interface, select **Help** from the menu bar. Additional information is available from the [EDB website](#).

5.6 Uninstalling Postgres Enterprise Manager Components

The process of uninstalling the PEM server or Agent is platform-specific. The name of the package for PEM server is `edb-pem` and for PEM Agent is `edb-pem-agent`.

If you uninstall the PEM server package from a host, the PEM Agent package installed on the same host doesn't get uninstalled. But if you uninstall the PEM Agent package, then the PEM server package installed on the same host also gets uninstalled.

Uninstalling PEM components from CentOS or RHEL Hosts

You can use variations of the `rpm`, `yum remove`, or `yum erase`, commands to remove the installed packages from CentOS or RHEL 7.x or 8.x hosts. Also you use `dnf remove` command to remove the installed package from CentOS or RHEL 8.x host. Note that removing a package does not damage the PEM data directory.

- Include the `-e` option when invoking the `rpm` command to remove an installed package; the command syntax is:

```
rpm -e package_name
```

- You can use the `yum remove` command to remove the PEM Server or Agent package installed by yum. To remove a package, open a terminal window, assume superuser privileges, and enter the command:

```
yum remove package_name
```

- You can use the `yum erase` command to remove the pem server or Agent package along with the `edb-pem` and `edb-pem-docs` dependencies. To remove a package, open a terminal window, assume superuser privileges, and enter the command:

```
yum erase package_name
```

Where `package_name` is the name of the package that you would like to remove.

- You can use `dnf remove` command to remove the pem server or Agent along with the `edb-pem` and `edb-pem-docs` dependencies on CentOS or RHEL 8.x hosts. To remove a package, open a terminal window, assume superuser privileges, and enter the command:

```
dnf remove package_name
```

Uninstalling PEM components from Debian or Ubuntu hosts

You can use `apt-get remove` or `apt-get purge` command to uninstall the PEM server or Agent package from a Debian or Ubuntu host:

- To uninstall PEM server or Agent from a Debian or Ubuntu host without impacting the configuration files and data directories, invoke the following command:

```
apt-get remove package_name
```

- To uninstall PEM server or Agent along with the configuration files and data directory, invoke the following

command:

```
apt-get purge package_name
```

Where `package_name` is the name of the package that you would like to remove.

Uninstalling PEM components from SLES hosts

To uninstall PEM server or Agent from a SLES host, invoke the following command:

```
zypper remove package_name
```

Where `package_name` is the name of the package that you would like to remove.

5.7 Reference - Linux Service Script

- A service script allows the PEM server to start, stop or restart the server if necessary when performing configuration management, certificate management, or other administrative tasks.
- The Postgres server on which the PEM server resides must contain a service script. Postgres installers in Windows generated by EDB create a service script for you; if you are using a Postgres server from another source like native packages, you must provide a service script.

On CentOS or RHEL 7.x or 8.x, the service script resides in the `/usr/lib/systemd/system` directory.

- Service scripts are platform-specific.
- For information about customizing a Postgres service, visit:

<https://www.postgresql.org/docs/current/static/server-start.html>

6 PEM Installation Guide on Windows

Postgres Enterprise Manager (PEM) is designed to assist database administrators, system architects, and performance analysts when administering, monitoring, and tuning PostgreSQL and Advanced Server database servers. PEM has been designed to manage and monitor a single server or multiple servers from a single console, allowing complete control over monitored databases.

This document provides step-by-step instructions to guide you through the installation of Postgres Enterprise Manager.

Throughout this guide, the term *Postgres* refers to either a PostgreSQL or an Advanced Server installation, where either is appropriate.

Language pack installers contain supported languages that may be used with EDB Postgres Advanced Server and EDB

PostgreSQL database installers. The language pack installer allows you to install Perl, TCL/TK, and Python without installing supporting software from third party vendors. For more information about installing and using Language Pack, please see the *EDB Postgres Language Pack Guide*, available from the EDB Website.

6.1 What's New

The following changes have been made to Postgres Enterprise Manager to create version 8.0.1:

Improved the overall performance, usability, and stability of the product by:

- Allowing the dash (-) character in the superuser name that is provided during configuration.
- Gracefully closing the operating system resources during batch probe and command execution to avoid errors.
- Improved support for unicode string handling in pemAgent.
- Fixed a regression restoring BART backup when agent is not bound with the BART server.
- In addition to the above-listed items, a few other minor bugs have also been fixed.

PEM 8.0.1 provides an option to hide or unhide the Dashboard, Statistics, Dependents, and Dependencies tabs.

Features and Functionalities removed in recent versions of PEM:

- PEM 8.0 and later is not supported on CentOS/RHEL 6.x.
- EDB ARK UI Management is not supported by PEM 7.16 and later.
- PEM 7.16 and later is not supported on Internet explorer version 11 and lesser.

6.2 PEM - Hardware and Software Requirements

Hardware Prerequisites

For optimum performance when monitoring servers and rendering dashboards, we recommend installing PEM on a system with at least:

- 4 CPU cores
- 8 GB of RAM
- 100 GB of Storage

Additional disk space is required for data storage. Please note that resource usage will vary based on which probes are defined and enabled, and the activity level on the monitored databases. Monitoring server resources (as you use PEM) will let you know when you need to expand your initial system configuration.

Software Prerequisites

Supported Platforms and Versions

For information about the platforms and versions supported by PEM, visit the EnterpriseDB website at:

<https://www.enterprisedb.com/services-support/edb-supported-products-and-platforms#pem>

Please note: PEM 8.0.1 is no longer supported on CentOS/RHEL/OEL 6.x platforms. It is strongly recommended that EDB products running on these platforms be migrated to a supported platform.

Modifying the pg_hba.conf File

The `pg_hba.conf` file manages connections for the Postgres server. You must ensure that the `pg_hba.conf` file on each monitored server allows connections from the PEM server, the monitoring PEM agent, and the host of the PEM-HTTPD server.

For information about modifying the `pg_hba.conf` file, see the [PEM Administrator's Guide](#).

Information about managing authentication is also available in the [PostgreSQL core documentation](#).

Firewall Restrictions

Please note that you must adjust your firewall to allow communication between PEM components.

Supported Locales

Currently, the PEM server and web interface support a locale of `English(US) en_US` and use of a period (.) as a language separator character. Using an alternate locale, or a separator character other than a period may result in errors.

6.3 PEM Architecture

Postgres Enterprise Manager (PEM) is a tool designed to monitor and manage multiple Postgres servers through a single GUI interface. PEM is capable of monitoring the following areas of the infrastructure:

!!! Note The term Postgres refers to either PostgreSQL or EDB Postgres Advanced Server.

- Hosts - One or more servers (physical or virtual) and their operating systems.
- Servers - One or more instances of PostgreSQL or EDB Postgres Advanced Server running on a host.
- Databases - One or more databases and the schema objects (tables, indexes, etc.) within them.

PEM consists of a number of individual software components; the individual components are described below.

- PEM Server - The PEM Server is used as the data repository for monitoring data and as a server to which both Agents and Clients connect. The PEM server consists of an instance of PostgreSQL and an associated database for storage of monitoring data, and a server that provides web services.
- PEM Agent - The PEM Agent is responsible for executing tasks and reporting statistics from the Agent host and monitored Postgres instances to the PEM server. A single PEM Agent can monitor multiple installed instances of Postgres that reside on one or many hosts.
- PEM Web Client - The PEM web interface allows you to manage and monitor Postgres servers and utilize PEM extended functionality. The web interface software is installed with the PEM server and is accessed via any supported web browser.
- SQL Profiler - SQL Profiler is a Postgres server plugin to record the monitoring data and query plans to be analysed by the SQL Profiler tool in PEM. This is an optional component of PEM, but the plugin must be installed into each instance of Postgres with which you wish to use the SQL Profiler tool. The SQL Profiler may be used with any

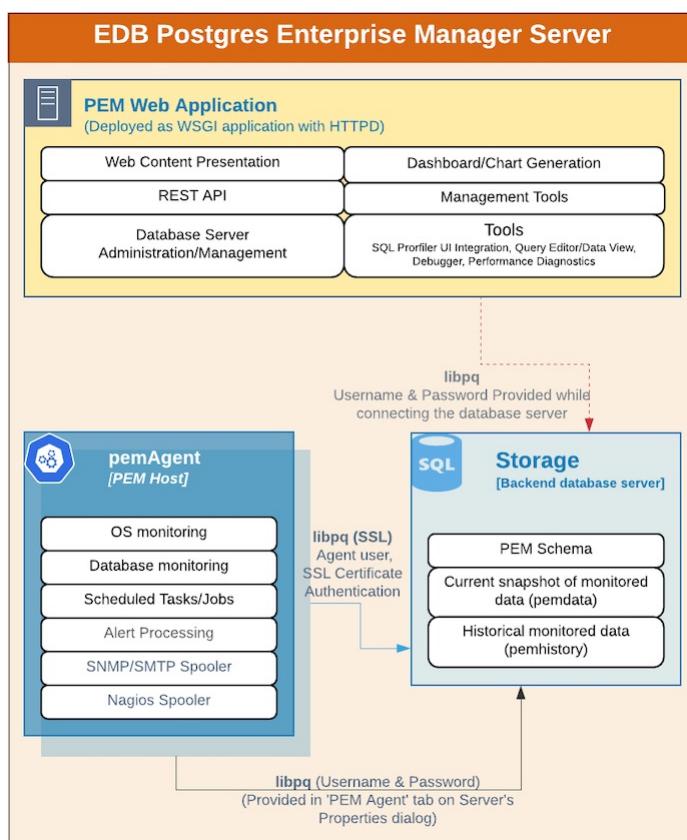
supported version of an EDB distribution of a PostgreSQL server or Advanced Server (not just those managed through the PEM server). See the [PEM SQL Profiler Guide](#) for details and supported versions.

PEM architecture

The following architectural diagram illustrates the relationships between the PEM server, clients, and managed as well as unmanaged Postgres servers.



The PEM Server



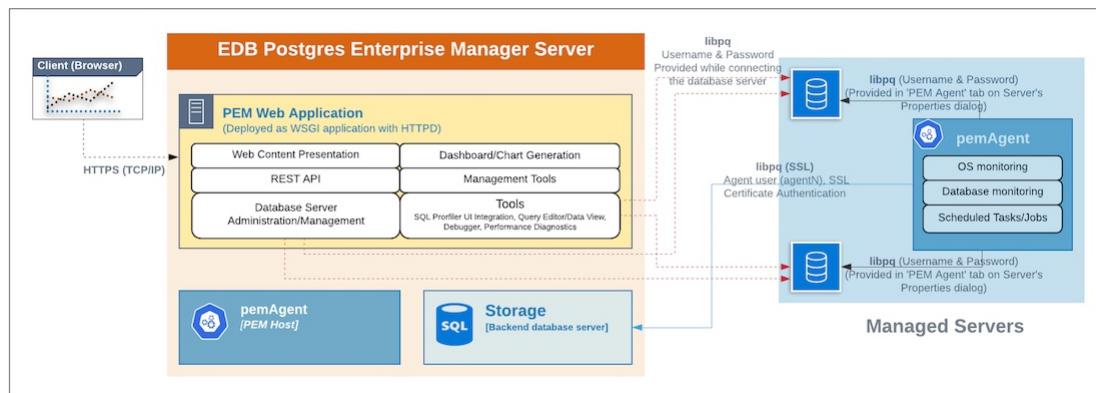
The PEM server consists of an instance of Postgres, an instance of the Apache web-server providing web services to the client, and a PEM Agent. PEM utilizes a server-side cryptographic plugin to generate authentication certificates.

The instance of Postgres (a database server) and an instance of the Apache web-server (HTTPD) can be on the same host or on separate hosts.

- **Postgres Instance (Database server)** - This is the backend database server. It hosts a database named pem which acts as the repository for PEM Server. The pem database contains several schemas that store metric data collected from each monitored host, server, and database.
 - pem - This schema is the core of the PEM application. It contains the definitions of configuration functions, tables, or views required by the application.
 - pemdata - This schema stores the current snapshot of the monitored data.
 - pemhistory - This schema stores the historical monitored data.
- **Apache Web Server (HTTPD)** - The PEM Web Application is deployed as a WSGI application with HTTPD to provide web services to the client. It is comprised of the following:
 - Web content presentation - The presentation layer is created by the Web Application (for example Browser, login page,..).
 - Rest API - The REST API allows integration with other apps and services.
 - Database Server Administration/Management - Database server administration and management activities like CREATE, ALTER, DROP, etc. can be performed for managed as well as unmanaged servers.
 - Dashboard/Chart generation - Internally, the web application includes functionality that generates Dashboards and Charts.
 - Management Tools - The Audit Manager, Capacity Manager, Log Manager, Postgres Expert, Postgres Log Analysis Expert, and the Tuning Wizard are made available in the Web Application.
 - Other tools provide functionality on managed or unmanaged servers:
 - SQL Profiler UI Integration - SQL Profiler generates easily analyzed traces of session content.
 - Query Editor/Data View - The Query editor allows you to query, edit, and view data.
 - Debugger - The Debugger helps you debug queries.
 - Performance Diagnostics - Performance Diagnostics help you analyze the performance of Postgres instances.

We recommend that you use a dedicated machine to host production instances of the PEM backend database. The host may be subject to high levels of data throughput, depending on the number of database servers that are being monitored and the workloads the servers are processing.

The PEM Agent



The PEM Agent is responsible for the collection of monitoring data from the machine and operating system, as well as from each of the Postgres instances to which they are bound. Each PEM Agent can monitor one physical or virtual machine and is capable of monitoring multiple database servers locally - installed on the same system, or remotely - installed on other systems. It is also responsible for executing other tasks that may be scheduled by the user (for example, server shutdowns, SQL Profiler traces, user-defined jobs).

A PEM Agent is installed by default on the PEM Server along with the installation of the PEM Server. It is generally referred to as a PEM Agent on the PEM Host. Separately, the PEM Agent can also be installed on the other servers hosting the Postgres instances to be monitored using PEM.

Whether monitoring locally or remotely, the PEM Agent connects to the PEM Server using PostgreSQL's libpq, using SSL certificate-based authentication. The PEM Agent installer in Windows and pemworker CLI in Linux is responsible for registering each agent with the PEM Server, and generating and installing the required certificates.

Please note that there is only one-way traffic between the PEM Agent and PEM Server; the PEM Agent always connects to the PEM Server.

The PEM Agent must be able to connect to each database server that it monitors. This connection is made over a TCP/IP connection (or optionally a Unix Domain Socket on Unix hosts), and may optionally use SSL. The user must configure the connection and authentication to the monitored server.

Once configured, each agent collects statistics and other information on the host and each database server and database that it monitors. Each piece of information is known as a metric and is collected by a probe. Most probes will collect multiple metrics at once for efficiency. Examples of the metrics collected include:

- Disk I/O statistics
- Network statistics
- Database server version string
- Database server configuration option (GUC) values
- Table access statistics
- Table and index sizes

A list of PEM probes can be found [here](#).

By default, the PEM Agent bound to the database server collects the OS/Database monitoring statistics and also runs any scheduled tasks/jobs for that particular database server, storing data in the pem database on the PEM server.

The Alert processing, SNMP/SMTP spoolers, and Nagios Spooler data is stored in the `pem` database on the PEM server and is then processed by the PEM Agent on the PEM Host by default. However, processing by other PEM Agents can be enabled by adjusting the SNMP/SMTP and Nagios parameters of the PEM Agents.

To see more information about these parameters see [Server Configuration](#).

The PEM Web Client

The PEM client is a web-based application that runs in supported browsers. The client's web interface connects to the PEM server and allows direct management of managed or unmanaged servers, and the databases and schemas that reside on them.

The client allows you to use PEM functionality that makes use of the data logged on the server through features such as the dashboards, the Postgres Log Analysis Expert, and Capacity Manager.

The SQL Profiler Plugin

You are not required to install the SQL Profiler plugin on every server, but you must install and configure the plugin on each server on which you wish to use the SQL Profiler. You may also want to install and configure SQL Profiler on un-monitored development servers. For ad-hoc use also, you may temporarily install the SQL Profiler plugin.

The plugin is installed with the EDB Postgres Advanced Server distribution but must be installed separately for use with

PostgreSQL. The SQL Profiler installer is available from the [EDB website](#).

SQL Profiler may be used on servers that are not managed through PEM, but to perform scheduled traces, a server must have the plugin installed, and must be managed by an installed and configured PEM agent.

For more information about using SQL Profiler, see the [PEM SQL Profiler Guide](#)

6.4 Installing Postgres Enterprise Manager

The PEM server graphical installer for Windows installs and configures the PEM server, a PEM agent, and the software required to connect to the PEM web interface with your choice of browser.

The PEM server uses a Postgres installation and backing database to manage data. The `pem` backing database gets created while installing the PEM Server through installer.

For detailed information about using the PEM server graphical installer, see [Installing the PEM Server on Windows](#).

The PEM agent graphical installer for Windows installs and registers the PEM agent. The PEM agent that is installed with the PEM server is capable of monitoring multiple servers that reside on the same host, or on remote hosts. Please note that the PEM functionality on servers monitored by a remote agent may be limited.

For detailed information about using the PEM agent graphical installer, see [Installing a PEM Agent on Windows](#).

6.4.1 Installing the PEM Server on Windows

At the heart of each PEM installation is the server. In a production environment, the server will typically be a dedicated machine, monitoring a large number of Postgres servers or a smaller number of busy servers.

The PEM server backend database may be an EnterpriseDB distribution of the PostgreSQL or Advanced Server database server, or an existing Postgres server installed from another source. The Postgres backend database server must be version 11 or later, and will contain a database named `pem`, which is used by the PEM server as a repository.

- If you would like to use an existing Postgres server to host the PEM server, the PEM server installer can create the `pem` database on the Postgres host. You must manually satisfy the software pre-requisites if you choose to use an existing server.

For more information about using an existing Postgres server to host the PEM server backend database, see [Installing the PEM Server on an Existing Postgres Server](#) section.

- If you do not wish to use an existing installation of Postgres as the PEM server host, the PEM server installer can install PostgreSQL, satisfy the server host's software pre-requisites, and create an instance (a PostgreSQL database cluster) that contains the `pem` database. This is the simplest PEM server installation option.
- PEM-HTTPD is made available for Postgres installations through the PEM server installer or the StackBuilder utility. If PEM-HTTPD is already installed on the host, the PEM server installer will review and update the existing installation if required. If the PEM server host does not contain an existing PEM-HTTPD installation, the PEM server

installer will add it.

- Before installing the PEM server, you must decide if you wish to run PostgreSQL and PEM-HTTPD on the same host or on separate hosts. If you intend to run the PostgreSQL database server and PEM-HTTPD on different hosts, then you must run the PEM server installer twice – once on each host, as detailed in [Installing the PEM Server and PEM-HTTPD on Separate Hosts](#) section.

The PEM server installer will also install the software required to access the server via the PEM web interface. You can access the web interface with a supported version of your browser.

Blackout	Status	Name	Alerts	Version	Processes	Threads	CPU Utilisation (%)	Memory Utilisation (%)	Swap Utilisation (%)	Disk Utilisation	Remotely Monitored?
	● UP	Postgres Enterprise Manager Host	1	7.14.0-dev	316	811	26.52	72.83	23.48	57.66	No
	● UP	PEM Agent on Remote Host	1	7.13.0	210	525	99.91	52.06	3.03	24.34	No

Blackout	Status	Name	Connections	Alerts	Version	Remotely Monitored?
	● UP	Postgres Enterprise Manager Server	17	8	PostgreSQL 12.1 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-39), 64-bit	No
	● UP	EDB Postgres Advanced Server 11	2	3	PostgreSQL 11.7 (EnterpriseDB Advanced Server 11.7.14) on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-36), 64-bit	No
	● UP	EPAS_12	4	4	PostgreSQL 12.2 (EnterpriseDB Advanced Server 12.2.3) on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-36), 64-bit	No

You can use the web interface to review information about objects that reside on monitored servers, manage databases and database objects that reside on monitored servers, or review statistical information gathered by the PEM server.

!!! Note If you are using the SSL certificates then make sure that all the SSL certificates are inside the data directory the backend database server. If the certificates are not inside the data directory then the PEM Server's configuration may fail as it looks in to the data directory while configuring the PEM Server.

Installing the PEM Server and PEM-HTTPD on the Same Host

The easiest PEM server installation configuration consists of a PEM backend database server (hosted on a PostgreSQL database installed with the PEM server installer) and a PEM-HTTPD service that reside on the same host. In this configuration, the PEM server installer will provide the pre-requisite software for the backend host register the service (on Windows).

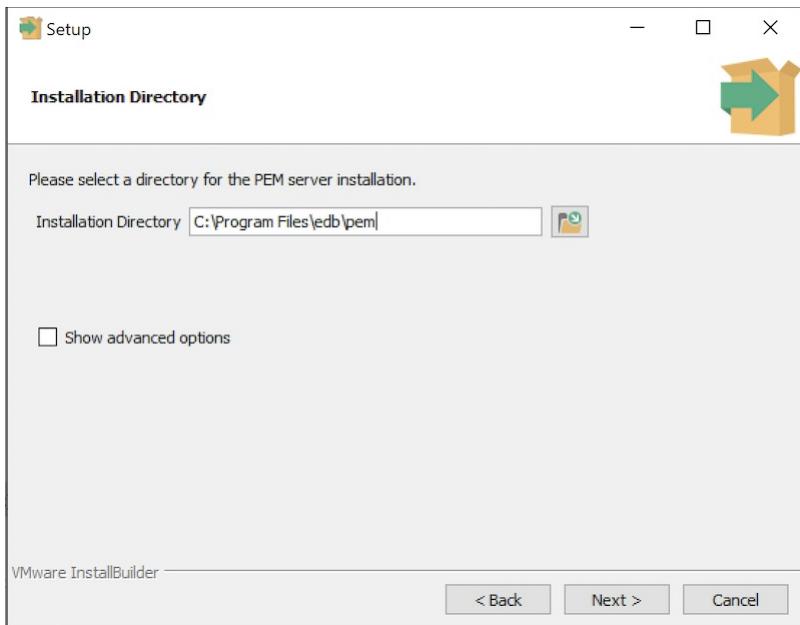
To invoke the PEM server installer on a Windows system, right click the installer icon and select **Run as Administrator**. The installer displays a **Welcome** dialog.



Click **Next** to continue to the **License Agreement** dialog.



Carefully review the license agreement before highlighting the appropriate radio button and accepting the agreement. Click **Next** to continue to the **Installation Directory** dialog.



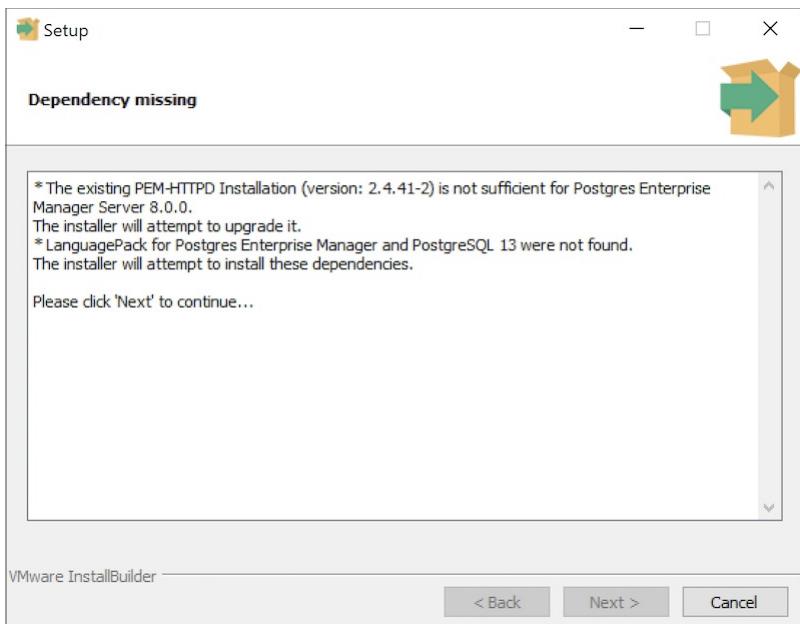
Use the **Installation Directory** dialog to specify the location of the PEM server:

- By default, the PEM server is installed in **C:\Program Files\edb\pem** on Windows. Accept the default location, or use the **Installation Directory** button to open a browser dialog and select the directory in which the PEM server will be installed.
- Use the **Show advanced options** check box to instruct the installer to open the **Advanced options** dialog.
- Use the **Advanced options** dialog when installing the Postgres database server and the PEM-HTTPD on different hosts, or if you wish the PEM server to reside on an existing Postgres server installation.

To install the PostgreSQL server packaged with the installer and PEM-HTTPD on the same host, leave the **Show advanced options** box unchecked and click **Next**.

The PEM server installer will perform a pre-installation check for PEM-HTTPD, Language Pack, and PostgreSQL 13. If the installer does not locate these packages, it will inform you in the **Dependency missing** dialog.

!!! Note By default EDB Language Pack is installed in **C:\edb\languagepack\v1**.



If the dependencies are missing, the PEM server installer will launch the respective installation wizards; follow the onscreen directions presented by the installation wizards for each package.

After installing any missing dependencies, the installation process continues by displaying the **Database Server Installation Details** dialog.



The information provided on the **Database Server Installation Details** dialog enables the installer to connect to the PostgreSQL server. Provide the **User name** and **Password** of a database superuser. After supplying the requested information, click **Next** to continue to the **pemAgent Service Account** dialog.



After providing the name and password of the Postgres database superuser, you may be prompted for the password to the user account under which the PEM agent will run. If prompted, provide the password, and press **Next** to continue to the **Network Details** dialog.



Use the **Network Details** dialog to specify the CIDR-style network address from which the PEM agents will connect to the server (the *client-side* address).

You may specify the address of a network host, or a network address range. For example, if you wish to monitor database servers with the addresses **192.168.10.23**, **192.168.10.76** and **192.168.10.184**, enter **192.168.10.0/24** to allow connections with hosts in that network.

The specified address will be added to the server's **pg_hba.conf** file. You can specify additional network addresses by manually adding entries to the **pg_hba.conf** file on the PostgreSQL server if required, using the initial entry as a template.

When you've added the **Network address**, click **Next** to continue to the **Agent Details** dialog.

The PEM server installer will install a PEM agent on the host on which the server resides, to monitor the server and provide alert processing and garbage collection services. A certificate will also be installed in the location specified in the **Agent certificate path** field.



Enter an alternate description or select an alternate agent certificate path for the PEM agent, or accept the defaults. Click **Next** to continue to the **Ready to Install** dialog.



The wizard is now ready to install the PEM server.

Click **Back** to modify any of the options previously selected, or **Next** to continue with the installation.



During the installation process, the installer will copy files to the system, and set up the database and web services required to run PEM. When the installation completes, a popup dialog opens confirming that the webservice has been configured, and is listening on port **8443**, and that the **pem** database has been created and configured.



Click **OK** to acknowledge that the webservice has been configured, and that the **pem** database has been created, and continue to the **Completed...** dialog.



Installing the PEM Server and PEM-HTTPD on Separate Hosts

To use separate hosts for the PEM server backend database and PEM-HTTPD, you must:

1. Invoke the PEM server installer on the host of the Postgres server that will contain the **pem** database. During the installation, select the **Database** option on the **Advanced options** dialog, and provide connection information for the Postgres server.
2. Modify the **pg_hba.conf** file of the Postgres installation on which the PEM server (and **pem** database) resides, allowing connections from the host of the PEM-HTTPD server.
3. Invoke the PEM server installer on the host of the PEM-HTTPD server, selecting the **Web Services** option on the **Installation Type** dialog.

To invoke the PEM server installer on a Windows system, right click the installer icon and select **Run as Administrator**. The installer displays a **Welcome** dialog.



Click **Next** to continue to the **License Agreement** dialog.



Carefully review the license agreement before highlighting the appropriate radio button and accepting the agreement; click **Next** to continue.



Use fields on the **Installation Directory** dialog to specify the directory in which the PEM server will reside, and to access the **Advanced options** dialog:

- By default, the PEM server is installed in the `C:\Program Files\edb\pem` on Windows. Accept the default location, or use the **Installation Directory** field to open a browser dialog and select the directory in which the PEM server will be installed.
- To install the PEM server and PEM-HTTPD on separate hosts, you must use the **Advanced options** dialog to specify the installation type (**Web Services** or **Database**). Select the **Show advanced options** check box to instruct the installer to include the **Advanced options** dialog in the installation process.

Click **Next** to continue to the **Advanced Options** dialog.



Use the radio buttons on the **Advanced options** dialog to specify the component or components that you would like to install:

- Select **Web Services and Database** to indicate that the Postgres server and PEM-HTTPD will both reside on the current host. If you select the **Web Services and Database** option, the PEM server installer will allow you to specify which Postgres server you wish to use for the PEM server before checking for a PEM-HTTPD installation.
- Select **Web Services** to install PEM-HTTPD on the current host, while using a Postgres database server that

resides on another host to host the PEM server and `pem` database.

!!! Note You must complete the PEM server installation process on the host of the PEM server (and `pem` backend database), selecting **Database** on the **Advanced options** dialog, and modifying the connection properties of the `pg_hba.conf` file on the PEM server before using the **Web Services** option to install PEM-HTTPD.

This option invokes the installation steps documented in [Installing Web Services](#) section.

- Select **Database** to use an existing Postgres server (version 11 or greater), or to install only the database server that is distributed with the PEM server installer. This option invokes the installation steps documented in [Specifying a Database Host](#).

After selecting an installation option, click **Next** to continue.

Specifying a Database Host

Selecting the **Database** option on the **Advanced options** dialog allows you to specify connection information for the host on which the PEM server backend database (named `pem`) will reside.



Click **Next** to continue to the **Database Server Selection** dialog.



Use the drop-down listbox on the **Database Server Selection** dialog to select a host for the PEM server backend database. You can:

- Select a host from existing Postgres installations that reside on the current host.

!!! Note You may be required to add the **sslutils** package to your installation.

- Select the **PostgreSQL x** option to install the Postgres server that is distributed with the PEM server installer where **x** is the PostgreSQL database server version. If you decide to use the version of PostgreSQL that is bundled with the PEM server installer, the EnterpriseDB one-click PostgreSQL installer will open and walk you through the installation.
- Select **Other Database Server** to specify connection information for a Postgres server that was not installed using a one-click graphical installer from EnterpriseDB. For information about the software pre-requisites for the PEM server database host, please see [Preparing the Postgres Server](#) section.

Click **Next** to continue.

If the PEM server will reside on an existing Postgres server, the **Database Server Installation Details** dialog shown in opens.



The information required on the **Database Server Installation Details** dialog may vary; the PEM server installer will ask you to provide only the information about the selected installation that it cannot locate:

- Specify the name of a Postgres database superuser in the **User** field.
- Specify the password associated with that user in the **Password** field.

Click **Next** to continue.

If prompted, provide the system password for the service account under which the PEM agent will run.



click **Next** to continue to the **Network Details** dialog.



Use the **Network Details** dialog to specify the CIDR-style network address from which PEM agents will connect to the server (the **client-side** address). The specified address will be added to the server's **pg_hba.conf** file.

!!! Note You can specify additional network addresses by manually adding entries to the **pg_hba.conf** file on the PostgreSQL server.

Accept the default (specifying the localhost), or specify a **Network address** range, and click **Next** to continue to

the **Agent Details** dialog.

The PEM server installer will install a PEM agent on the host on which the server resides, to monitor the server and provide alert processing and garbage collection services. A certificate will also be installed in the location specified in the **Agent certificate path** field.



You can enter an alternate description or an alternate agent certificate path for the PEM agent, or accept the defaults. Click **Next** to continue.



The wizard is now ready to install the PEM server. Click **Back** to modify any of the options previously selected, or **Next** to proceed with the installation.



During the installation process, the installer will copy files to the system, and set up the PEM server's backend database. A popup dialog opens confirming that the **pem** database has been created and configured.



Click **OK** to acknowledge that the **pem** database has been created, and continue to the **Completed...** dialog.



When the database portion of the PEM server installation is complete, you can invoke the PEM server on another host to install (or upgrade) PEM-HTTPD.

Installing Web Services

Selecting the **Web Services** radio button on the **Advanced options** dialog instructs the PEM server installer to either install PEM-HTTPD on the current host or update an existing PEM-HTTPD installation.



!!! Note The current host may not be the host of the PEM backing database.

Before selecting this option, you must have:

- Completed an installation of the PEM server installer on a host system, during which you specified a backing database for the PEM server.
- Modified the `pg_hba.conf` file on the PEM server database host to allow connections from the PEM-HTTPD host, and restarted the database server.

When you select the **Web Services** option and click **Next**, the PEM server installer will check the current host for existing PEM-HTTPD and LanguagePack installations.

If the installer does not locate the components, the installer will inform you that one or more dependencies are missing.



Click **Next** to instruct the server to install LanguagePack



After installing language pack, the installer will invoke the PEM-HTTPD setup wizard.



Follow the onscreen instructions of the [PEM-HTTPD Setup Wizard](#). When the wizard completes the installation, click [Finish](#) to open the [Database Server Installation Details](#) dialog.



Use the fields on the [Database Server Installation Details](#) dialog to provide connection information for the Postgres installation that is hosting the PEM server installation:

- Enter the name or IP address of the PEM server host in the [Host](#) field.
- Enter the port number on which the Postgres server is listening in the [Port](#) field.
- Enter the name of a Postgres database superuser in the [User](#) field.
- Enter the password associated with the Postgres superuser in the [Password](#) field.

Click [Next](#) to continue. Before completing the PEM server installation, the installer will contact the database host. The `pg_hba.conf` file on the PEM database host must be configured to accept connections from the host of the httpd server and the firewall must allow a connection for the installation to continue. The PEM server installer will complete the PEM server installation, adding only those items that must reside on the host of the PEM-HTTPD server.

Installing the PEM Server on an Existing Postgres Server

You can use an existing Postgres server (version 11 or later) to host the PEM server and the `pem` database. Postgres installers and pre-requisite software extensions are freely available on the [EnterpriseDB website](#).

This section provides information about configuring an existing Postgres server for a PEM server installation.

!!! Note The steps that follow should be considered guidelines only; the actual steps required to configure your Postgres installation will vary depending on the configuration of your Postgres server.

The following versions of Postgres are pre-configured to contain the `sslutils` extension and a service script; no additional preparation is required to use the following Postgres versions as a PEM backend database server:

- PostgreSQL 11 or later (as bundled with the PEM Server installer)
- Advanced Server 11 or later

Preparing the Postgres Server

Before installing the PEM server on an existing Postgres server, you must:

- Ensure that the Postgres server contains an installation of the `sslutils` extension. For more information, see the [Installing the sslutils Extension](#) section.
- Register the server with the Windows service manager. For more information, see the [Registering the Service](#) section.

After preparing the server, you can use the PEM server installer to install PEM on the existing Postgres server.

Installing the `sslutils` Extension

The Postgres server on which the PEM server will reside must contain the `sslutils` extension. The `sslutils` package is freely available for download from the [EDB website](#)

When the web page opens, select the link for the `SRC- SSL Utils 1.3` package. When the download completes, extract the file, and copy it into the Postgres installation directory.

Remember: You are *not* required to manually add the `sslutils` extension when using the following Postgres installations:

- PostgreSQL 9.6 or later (as distributed with the PEM server installer)
- Advanced Server 9.6 or later

`sslutils` must be built with the same compiler that was used to compile the backend Postgres installation. If you are using a backend Postgres database that was installed on a Windows platform using a PostgreSQL one-click installer (from EnterpriseDB) or an Advanced Server installer, you must use Visual Studio to build `sslutils`.

While specific details of the installation process will vary by platform and compiler, the basic steps are the same. You must:

1. Copy the `sslutils` package to the Postgres installation directory.
2. Open the command line of the appropriate compiler, and navigate into the `sslutils` directory.
3. Use the following commands to build `sslutils`:

```
SET USE_PGXS=1
```

```

SET GETTEXTPATH=<path_to_gettext>
SET OPENSSLPATH=<path_to_openssl>
SET PGPATH=<path_to_pg_installation_dir>
SET ARCH=x86
REM Set ARCH x64 for 64 bit
msbuild sslutils.proj /p:Configuration=Release

```

Where:

`path_to_gettext` specifies the location of the `gettext` library and header files.

`path_to_openssl` specifies the location of the `openssl` library and header files.

`path_to_pg_installation_dir` specifies the location of the Postgres installation.

4. Copy the compiled `sslutils` files to the appropriate directory for your installation. The `sslutils` directory will contain the following files:

```

sslutils--1.3.sql
sslutils--unpackaged--1.3.sql
sslutils--pemagent.sql.in
sslutils.dll

```

Copy the `.dll` libraries and `.sql` files into place:

```

COPY sslutils*.sql* "%PGPATH%\share\extension\
COPY sslutils.dll "%PGPATH%\lib\

```

Registering the Service

When you install a PostgreSQL or an Advanced Server database using an installer from EnterpriseDB, the installer will register the service for you.

If you are using Windows to host the PEM backend database, you must register the name of the Postgres server with the Windows service manager. If you are using a Postgres server that was created using an EnterpriseDB installer, the service will be registered automatically. If you are manually building the installation, you can use the `register` clause of the Postgres `pg_ctl` command to register the service. The syntax of the command is:

```

pg_ctl register [-N <service_name>] [-U <user_name>]
| [-P <password>] [-D <data_directory>]

```

Where:

`service_name` specifies the name of the Postgres cluster.

`user_name` specifies the name of an operating system user with sufficient privileges to access the Postgres installation directory and start the Postgres service.

`password` specifies the operating system password associated with the user.

`data_directory` specifies the location of the Postgres data directory.

For more information about using the `pg_ctl` command and the available command options, see the [Postgres core documentation](#)

Invoking the PEM Server Installer

After preparing the existing Postgres server, invoke the PEM server installer. Assume Administrative privileges and navigate into the directory that contains the installer. Then, invoke the installer with the command:

```
pem_server-7.<x>.<x>-<x>-<platform>
```

Where *x* is the major and minor versions of PEM and platform is the platform.

The installer displays a `Welcome` dialog.



Click `Next` to continue to the `License Agreement` dialog.



Carefully review the license agreement before highlighting the appropriate radio button and accepting the agreement; click **Next** to continue to the **Installation Directory** dialog.



Use the **Installation Directory** dialog to specify the location of the PEM server and access the **Advanced options** dialog:

- Use the **Installation Directory** field to open a browser dialog and select the directory in which the PEM server will be installed.
- If you are installing the PEM server on an existing server, check the box next to **Show advanced options** to instruct the installer to include the **Advanced options** dialog in the installation process.

Click **Next** to continue.



Use the radio buttons on the **Advanced options** dialog to specify an installation type. Select:

- **Web Services and Database** if both the Postgres server and the PEM-HTTPD server will reside on the current host. This option is valid if you are using an existing Postgres server to host the PEM server, or using the PEM server installer to install the Postgres server on which the PEM server will reside.

If you select **Web Services and Database**, the PEM server installer will check the current host for a PEM-HTTPD installation, and upgrade or install PEM-HTTPD if necessary.

- **Web Services** if only the PEM-HTTPD server will reside on the current host. See [Installing Web Services](#) section for more information about invoking this option.
- **Database** if you are installing only the PEM server (and creating the **pem** backend database) on the current host. This option is valid if you are using an existing Postgres server to host the PEM server, or using the PEM server installer to install the PostgreSQL server on which PEM will reside.

After selecting an installation option, click **Next** to continue.



Use the drop-down listbox on the **Database Server Selection** dialog to select a backend database for the PEM

server:

- Select the name of a Postgres server on the current host that was installed using a Postgres one-click installer or Advanced Server installer.
- Select the **PostgreSQL x (Packaged)** option to instruct the installation wizard to install and use the PostgreSQL server that is packaged with the PEM server installer. Where **x** is the version of the PostgreSQL database server.
- Select **Other Database Server** to instruct the PEM server installer to use a Postgres database that was installed from a source other than an EnterpriseDB installer (i.e. from an rpm, or built from source).

!!! Note The selected database server must include an installation of the **sslutils** contrib module, and a registered service (on Windows).

For information about Preparing the Postgres Server, please see [this section](#).

If you selected **Web Services and Database** on the **Advanced options** dialog, the installation wizard will check the current host for an existing PEM-HTTPD installation, and upgrade or install the service as needed.

If you selected **Database** on the **Advanced options** dialog, the **Database Server Installation Details** dialog opens.



Use the fields on the **Database Server Installation Details** dialog to describe the connection to the Postgres server that will host the PEM server:

- Enter the name of a database superuser in the **User** field.
- Enter the password associated with the superuser in the **Password** field.

Click **Next** to continue.



Provide the administrators password under which PEM Agent service will run.

Click **Next** to continue.



Use the **Network Details** dialog to specify the CIDR-style network address from which the PEM agents will connect to the server (the **client-side** address). The specified address will be added to the server's **pg_hba.conf file**.

You can specify additional network addresses by manually adding entries to the **pg_hba.conf** file on the PostgreSQL server if required, using the initial entry as a template.

When you've added the **Network address**, click **Next** to continue to the **Agent Details** dialog.

The PEM server installer will install a PEM agent to the host on which the server resides, to monitor the server and provide alert processing and garbage collection services. A certificate will also be installed in the location specified in the **Agent certificate** path field.



You can enter an alternate description or an alternate agent certificate path for the PEM agent, or accept the defaults. Click **Next** to continue to the **Ready to Install** dialog.



The wizard is now ready to install the PEM server. Click **Back** to modify any of the options previously selected, or **Next** to continue with the installation.



During the installation process, the installer will copy files to the system, and set up the PEM server's backend database. A popup dialog opens confirming that the **pem** database has been created and configured.



Click **OK** to acknowledge that the pem database has been created, and continue to the **Completed...** dialog.



If you are using a PEM-HTTPD service that resides on a separate host, you must:

- Modify the `pg_hba.conf` file on the Postgres server host to allow connections between the hosts.
- Invoke the PEM server installer on the host of the PEM-HTTPD server. See [Installing Web Services](#) section for more information about installing PEM-HTTPD.

Invoking the Server Installer from Command Line

The command line options of the PEM server and PEM agent installers offer functionality in situations where a graphical installation may not work because of limited resources or system configuration. You can:

- Include the `--mode unattended` option when invoking the installer to perform an installation without additional user input.

Not all command line options are suitable for all platforms. For a complete reference guide to the command line options, include the `--help` option when you invoke the installer.

Invoking the PEM Server Installer in Unattended Mode

You can perform an unattended PEM server installation by providing installation preferences on the command line when invoking the installer. Please note that the system on which you are installing the PEM server must have internet access.

You must have Administrative privileges to install the PEM server. Before invoking the PEM server installer, you must install the following dependencies:

- PostgreSQL
- pem-httpd
- Language Pack

You can use the PEM server installer to satisfy the dependencies of the PEM server; use the following command to extract the dependencies. Navigate to the location of the installer, and use the following command to extract the dependencies:

```
pem-server-7.<x>.<x>-windows-x64.exe --extract-dependents C:\
```

In our example, the files are extracted to the `C:\` directory. After extracting the files, you must install each program. Navigate into the directory that contains the files (in our example, `C:\`) , and enter:

```
edb-languagepack-<version>-windows-x64.exe --mode unattended
pem-httpd-<version>-windows-x64.exe --mode unattended
postgresql-<version>-windows-x64.exe --mode unattended
```

Then, you can invoke the PEM server installer:

```
pem-server-7.<x>.<x>-windows-x64.exe --mode unattended
--existing-user <registered_edb_user> --existing-password
<edb_user_password> --pgport <port> --pguser postgres
--agent_description pem-agent --systempassword <windows_password>
--agent-crt-path C:\edb\`
```

Where:

- `registered_edb_user` specifies the name of a registered EnterpriseDB user. To register, visit the [EDB website](#)
- `edb_user_password` specifies the password associated with the EDB user account.
- `port` specifies the port used by the backing PostgreSQL database; by default, the PostgreSQL database uses port 5432.
- `cdr_address_range` specifies the address range that will be added to the `pg_hba.conf` file of the PEM server's backing database to allow connections from the agents that will be monitored by the server. You may wish to specify a network range (for example, 192.168.2.0/24) to provide server access to agents that reside on the same network.
- `windows_password` specifies the password associated with the Windows Administrator's account.

!!! Note when invoked in unattended mode, the PostgreSQL installer creates a user named `postgres`, with a password of `postgres`.

6.4.2 Installing a PEM Agent on Windows

To invoke the PEM Agent installer, assume `Administrative` privileges and navigate into the directory that contains the installer. Then, invoke the installer with the command:

```
pem_agent-7.<x>.<x>-<x>-platform.exe
```

The `Setup...` page opens, welcoming you to the PEM Agent installer.



Click **Next** to continue to the **License Agreement**.



Carefully review the license agreement before highlighting the appropriate radio button and accepting the agreement; click **Next** to continue to the **Installation Directory** dialog.



By default, the PEM Agent is installed in the `C:\Program Files (x86)\edb\pem` directory. You can accept the default installation directory, or modify the contents of the **Installation Directory** field, specifying an alternate installation directory for the PEM agent.

By default, the PEM Agent installer places a certificate in the Administrator's `%APPDATA%\pem` directory. Check the **Show advanced options** box to indicate that you would like the PEM agent installer to include a dialog that allows you to specify an alternate path for the certificate file.

Check the box next to **Register now?** to instruct the installer to register the newly installed PEM Agent with the PEM server.

Click **Next** to continue to the **PEM Server Installation Details** dialog.



Enter the connection details for the PEM server on the **PEM server installation details** dialog:

- Specify the name or IP address of the system on which the PEM database server resides in the **Host** field. Please note: If the PEM-HTTPD web server and PEM database are hosted on different systems, you must specify *the host of the PEM database*.
- Specify the name of the database superuser in the **User Name** field.
- Specify the password associated with the database superuser in the **Password** field.
- Specify the port that PostgreSQL is monitoring in the **Port** field.

Click **Next** to continue to **pemAgent Service Account**. The installer will attempt to connect to the server to verify that the details are correct.

!!! Note The PEM server must allow connections from the PEM Agent installer. If you encounter a connection error, confirm the connection properties specified on the **PEM Server Installation Details** dialog are correct, and confirm that the **pg_hba.conf** file (on the PEM server) will allow a connection to the server described in the error message.



Provide the password for the edb account under which the pemAgent service will run. The Agent certificate and key files will be created in `C:\Users\edb\AppData\Roaming\pem` directory. Click [Next](#) to continue to [Agent Details](#) dialog.



The tree control displayed in the [Browser](#) panel of the PEM web interface displays the value entered in the

Description field to identify the PEM agent. Specify a descriptive name for the agent, such as the hostname of the machine the agent is installed on, or a name that reflects the host's functionality.

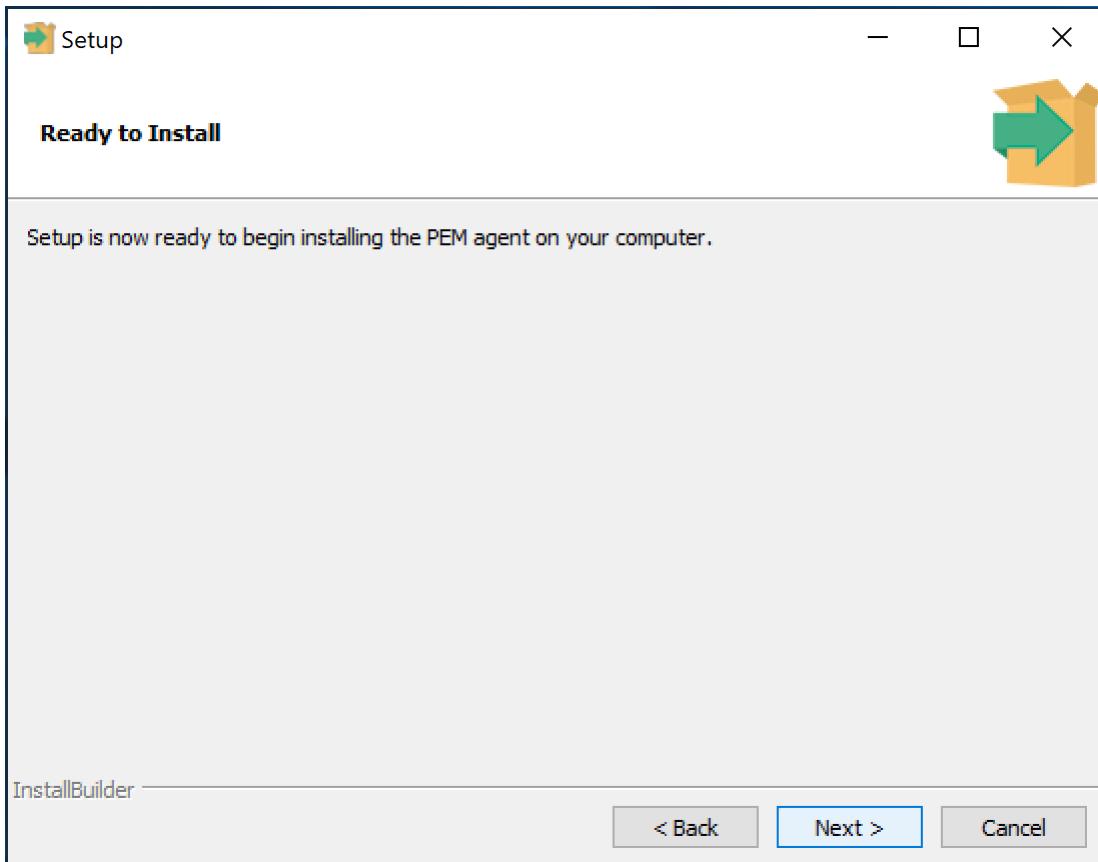
Provide a descriptive name, or accept the default provided by the PEM Agent host, and click **Next** to continue.

If you checked the **Show advanced options** checkbox, the **Advanced options** dialog opens.



By default, the PEM Agent installer places the certificate in the **C:\Program Files (x86)\edb\pem** directory. Specify an alternate path for the certificate or accept the default and click **Next**.

The wizard is now ready to install the PEM Agent; click **Back** to amend the installation directory, or **Next** to continue.



Click **Next** on the **Ready to Install** dialog to instruct the installer to copy files to the system and register the Agent on the PEM server.



The PEM Agent installer displays progress bars to mark the PEM Agent's installation progress.



When the installation has completed, the PEM Agent will be running and reporting operating system and host data to the PEM server. To start monitoring Postgres instances on the host of the PEM agent, they must now be added to PEM's enterprise directory and bound to the agent.

Invoking the Agent Installer from the Command Line

The command line options of the PEM Agent installers offer functionality in situations where a graphical installation may not work because of limited resources or system configuration. You can:

- Include the `--mode unattended` option when invoking the installer to perform an installation without additional user input.

Not all command line options are suitable for all platforms. For a complete reference guide to the command line options, include the `--help` option when you invoke the installer.

Invoking the PEM Agent Installer in Unattended Mode

You can perform an unattended PEM server installation by providing installation preferences on the command line when invoking the installer. Please note that the system on which you are installing the PEM server must have internet access.

Before invoking the PEM Agent installer in unattended mode, you must:

- Install the PEM server; the `pg_hba.conf` file of the PEM server must allow connections from the host of the PEM Agent.
- Ensure that the monitored Postgres database has SSL enabled, and is accepting connections.

You must have Administrator privileges to install the PEM Agent. Use the following command to invoke the PEM Agent installer in unattended mode:

```
pem-agent-7.<x>.<x>-windows-x64.exe --mode unattended
--pghost <pem_server_host_address> --pgport <pem_server_port>
--pguser postgres --pgpassword <pguser_password>
--agent_description <agent_name>
```

Where:

- `pem_server_host_address` specifies the IP address of the host of the PEM server.
- `pem_server_port` specifies the port used by the backing PEM database; by default, the database uses port 5432.
- `pguser_password` specifies the password associated with the PEM database superuser.
- `agent_name` specifies a descriptive name for the PEM Agent.

EnterpriseDB is the leading provider of value-added products and services for the Postgres community. Please visit our website at www.enterprisedb.com.

!!! Note When configuring a shell/batch script run by a Windows Agent that has PEM 7.11 or later version installed, the `AllowBatchJobSteps` parameter must be set to `True` in the `agent.cfg` file. The PEM agent will not execute any batch/shell script by default.

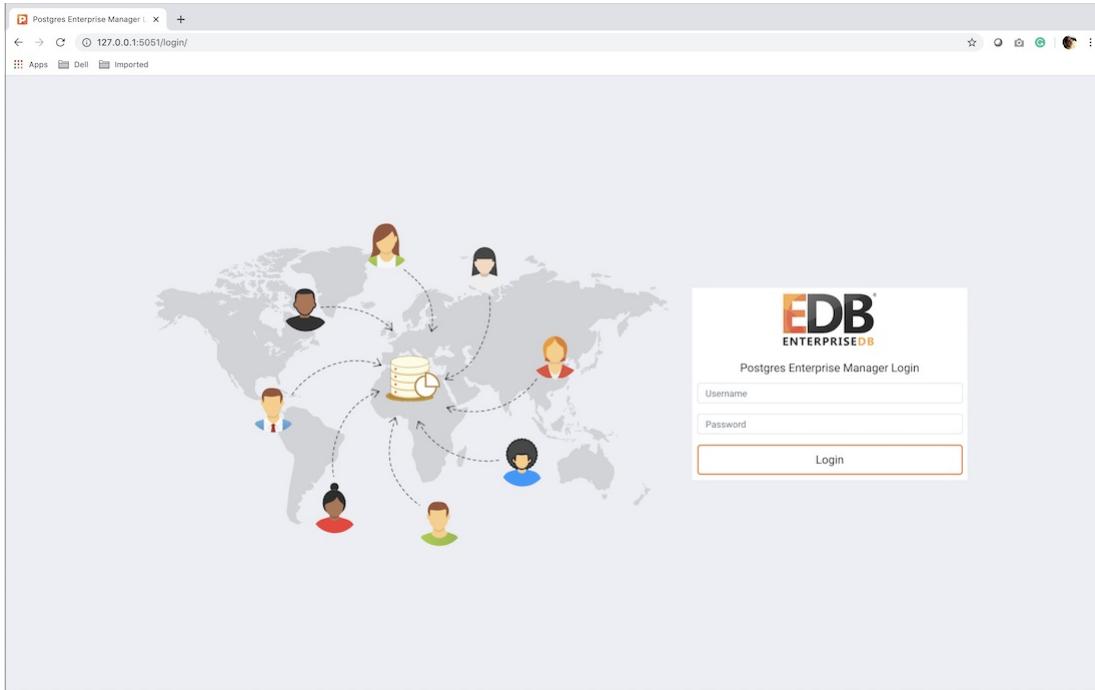
6.5 The PEM Web Interface

After installing a PEM Server and Agent, you can configure PEM to start monitoring and managing PostgreSQL or Advanced Server instances. The PEM Server installer installs the PEM web interface. You can use the interface to review information about objects that reside on monitored servers, or to review statistical information gathered by the PEM Server.

After installing and configuring PEM, you can use any supported browser to access the PEM web interface. Open your browser, and navigate to:

https://<ip_address_of_PEM_host>:8443/pem

Where `ip_address_of_PEM_host` specifies the IP address of the host of the PEM Server. The **Postgres Enterprise Manager Web Login** window opens:



Use the fields on the **Postgres Enterprise Manager Login** window to authenticate yourself with the PEM Server:

- Provide the name of a `pem` database user in the **Username** field. For the first user connecting, this will be the name provided when installing the PEM server.
- Provide the password associated with the user in the **Password** field.

Click the **Login** button to connect to the PEM Server.

Agent Status

Blackout	Status	Name	Alerts	Version	Processes	Threads	CPU Utilisation (%)	Memory Utilisation (%)	Swap Utilisation (%)	Disk Utilisation
<input type="checkbox"/>	UP	Postgres Enterprise Manager Host	0	7.14.0-dev	309	810	24.85	77.18	17.88	45.84
<input type="checkbox"/>	UP	PEM Agent on Remote Host	0	7.13.0	207	524	0.35	51.73	3.03	24.30

Server Status

Blackout	Status	Name	Connections	Alerts	Version	Remotely Monitored?
<input type="checkbox"/>	UP	Postgres Enterprise Manager Server	12	6	PostgreSQL 12.1 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-39), 64-bit	No
<input type="checkbox"/>	UP	EDB Postgres Advanced Server 11	3	3	PostgreSQL 11.7 (EnterpriseDB Advanced Server 11.7.14) on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-36), 64-bit	No
<input type="checkbox"/>	DOWN	PGSQL12_Centos7_1	0	0	PostgreSQL 12.2 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-39), 64-bit	Yes
<input type="checkbox"/>	UP	EPAS_12	6	5	PostgreSQL 12.2 (EnterpriseDB Advanced Server 12.2.3) on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-36), 64-bit	No

Alerts Status

Alarm Type	Object Description	Alert Name	Value	Database	Schema	Package	Object	Alerting Since
▶ ● High	EDB Postgres Advanced Server 11	Last Vacuum	Never ran					2020-04-21 21:26:54
▶ ● High	EDB Postgres Advanced Server 11	Last AutoVacuum	177.03 hrs					2020-04-22 12:04:05
▶ ● High	EDB Postgres Advanced Server 11	Database size in server	113 MB					2020-04-22 11:50:00
▶ ● High	EPAS_12	Server Down	1					2020-04-29 09:11:09
▶ ● High	EPAS_12	Table size in server	427 MB					2020-04-09 15:53:51
▶ ● High	EPAS_12	Last Vacuum	15.39 hrs					2020-04-29 08:19:11
▶ ● High	EPAS_12	Database size in server	473 MB					2020-04-09 15:52:50
▶ ● High	EPAS_12	Last AutoVacuum	15.38 hrs					2020-04-29 08:19:11
▶ ● High	N/A	Alert Errors	3					2020-01-21 14:26:04
▶ ● High	PGSQL12_Centos7_1	Server Down	1					2020-04-29 08:54:02
▶ ● High	PGSQL12_Centos7_1	Last Vacuum	Never ran					2020-04-03 14:58:57
▶ ● High	PGSQL12_Centos7_1	Last AutoVacuum	Never ran					2020-04-03 14:58:57
▶ ● High	Postgres Enterprise Manager Server	Largest index by table-size percentage	100 %					2020-04-21 22:07:52
▶ ● High	Postgres Enterprise Manager Server	Database size in server	2.748046875 GB					2020-02-05 18:26:49
▶ ● Medium	Postgres Enterprise Manager Server	Total table bloat in server	88.28 MB					2020-04-29 08:36:18
▶ ● High	Postgres Enterprise Manager Server	Table size in server	2.6591796875 GB					2020-02-20 11:29:45
▶ ● High	Postgres Enterprise Manager Server	Connections in idle state	17					2020-04-29 09:05:07
▶ ● High	Postgres Enterprise Manager Server	Last Vacuum	41.46 hrs					2020-04-28 09:38:02

Before you can use the PEM web interface to manage or monitor a database server, you must *register* the server with the PEM Server. When you register a server, you describe the connection to the server, provide authentication information for the connection, and specify any management preferences (optionally binding an agent).

A server may be managed or unmanaged:

- A **managed** server is bound to a PEM Agent. The PEM Agent will monitor the server to which it is bound, and perform tasks or report statistics for display on the PEM dashboards. A managed server has access to extended PEM functionality such as Package Management or Custom Alerting; when registering a server, you can also allow a managed server to be restarted by PEM as required.
- An **unmanaged** server is not bound to a PEM Agent; you can create database objects on an unmanaged server, but extended PEM functionality (such as Package Management or Custom Alerting) is not supported on an unmanaged server.

You must also ensure the **pg_hba.conf** file of the server that you are registering allows connections from the host of the PEM web interface.

To access online help information about the PEM web interface, select **Help** from the menu bar. Additional information is available in .pdf and .html format from the [EDB website](#)

- The [PEM Administrator's Guide](#) contains information about registering and managing servers, agents, and users.
 - The [PEM Enterprise Features Guide](#) contains information about using the tools and wizards that are part of the web interface.
 - The [PEM Agent User Guide](#) contains helpful information about managing your PEM Agents.
 - The [PEM Upgrade and Migration Guide](#) contains information about upgrading PEM to its latest version from a previous version.
 - The [PEM PgBouncer Configuration Guide](#) contains information about using PgBouncer with your PEM installation.
-

6.6 Uninstalling Postgres Enterprise Manager Components

If you uninstall the PEM server from a host, the PEM agent installed on the same host is uninstalled. But if you uninstall the PEM agent, then the PEM server installed on the same host will not be uninstalled.

You can use the Windows [Add/Remove Programs](#) application to remove PEM components from a Windows host. Select the [Add/Remove Programs](#) option from the Windows [Control Panel](#). When the [control panel](#) opens, locate the name of the PEM component in the program list. Click the [Remove](#) button to remove the component.

You can also invoke the uninstaller that resides at the following location:

For the PEM Server, [C:\Program Files\edb\pem\server\uninstall-pemserver](#)

For the PEM Agent, [C:\Program Files\edb\pem\agent\uninstall-pemagent](#)

7 Postgres Enterprise Manager

Welcome to Postgres Enterprise Manager (PEM). Postgres Enterprise Manager (PEM) consists of components that provide the management and analytical functionality for your EDB Postgres Advanced Server or PostgreSQL database. PEM is based on the Open Source pgAdmin IV project.

pgAdmin is the leading Open Source management tool for Postgres, the world's most advanced Open Source database. pgAdmin IV is a comprehensive [database](#) design and management system. pgAdmin 4 is designed to meet the needs of both novice and experienced Postgres users alike, providing a powerful graphical interface that simplifies the creation, maintenance and use of database objects.

Contents:

7.1 PEM Getting Started

You can use either a graphical installer or an RPM package to install the PEM server and PEM agent; for detailed installation instructions, please see the Postgres Enterprise Manager Installation Guide, available at www.enterprisedb.com.

Contents:

7.1.1 PEM Architecture

Postgres Enterprise Manager (PEM) is a tool designed to monitor and manage multiple Postgres servers through a single GUI interface. PEM is capable of monitoring the following areas of the infrastructure:

Note: The term Postgres refers to either PostgreSQL or EDB Postgres Advanced Server.

- Hosts - One or more servers (physical or virtual) and their operating systems.
- Servers - One or more instances of PostgreSQL or EDB Postgres Advanced Server running on a host.
- Databases - One or more databases and the schema objects (tables, indexes, etc.) within them.

PEM consists of a number of individual software components; the individual components are described below.

- PEM Server - The PEM Server is used as the data repository for monitoring data and as a server to which both Agents and Clients connect. The PEM server consists of an instance of PostgreSQL and an associated database for storage of monitoring data, and a server that provides web services.
- PEM Agent - The PEM Agent is responsible for executing tasks and reporting statistics from the Agent host and monitored Postgres instances to the PEM server. A single PEM Agent can monitor multiple installed instances of Postgres that reside on one or many hosts.
- PEM Web Client - The PEM web interface allows you to manage and monitor Postgres servers and utilize PEM extended functionality. The web interface software is installed with the PEM server and is accessed via any supported web browser.
- SQL Profiler - SQL Profiler is a Postgres server plugin to record the monitoring data and query plans to be analysed by the SQL Profiler tool in PEM. This is an optional component of PEM, but the plugin must be installed into each instance of Postgres with which you wish to use the SQL Profiler tool. The SQL Profiler may be used with any supported version of an EnterpriseDB distribution of a PostgreSQL server or Advanced Server (not just those managed through the PEM server). See the [PEM SQL Profiler Configuration Guide](#) for details and supported versions.

PEM architecture

The following architectural diagram illustrates the relationships between the PEM server, clients, and managed as well as unmanaged Postgres servers.



The PEM Server



The PEM server consists of an instance of Postgres, an instance of the Apache web-server providing web services to the client, and a PEM Agent. PEM utilizes a server-side cryptographic plugin to generate authentication certificates.

The instance of Postgres (a database server) and an instance of the Apache web-server (HTTPD) can be on the same host or on separate hosts.

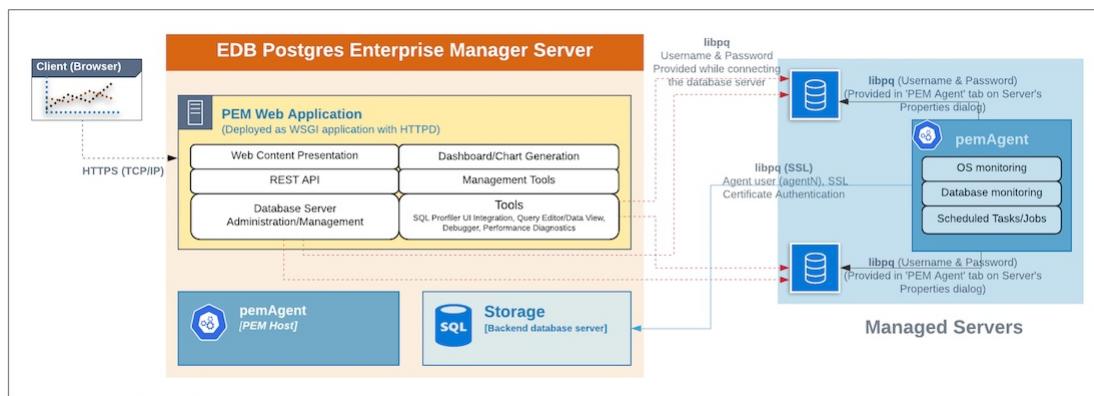
- Postgres Instance (Database server) - This is the backend database server. It hosts a database named pem which acts

as the repository for PEM Server. The pem database contains several schemas that store metric data collected from each monitored host, server, and database.

- pem - This schema is the core of the PEM application. It contains the definitions of configuration functions, tables, or views required by the application.
 - pemdata - This schema stores the current snapshot of the monitored data.
 - pemhistory - This schema stores the historical monitored data.
- Apache Web Server (HTTPD) - The PEM Web Application is deployed as a WSGI application with HTTPD to provide web services to the client. It is comprised of the following:
 - Web content presentation - The presentation layer is created by the Web Application (for example Browser, login page,..).
 - Rest API - The REST API allows integration with other apps and services.
 - Database Server Administration/Management - Database server administration and management activities like CREATE, ALTER, DROP, etc. can be performed for managed as well as unmanaged servers.
 - Dashboard/Chart generation - Internally, the web application includes functionality that generates Dashboards and Charts.
 - Management Tools - The Audit Manager, Capacity Manager, Log Manager, Postgres Expert, Postgres Log Analysis Expert, and the Tuning Wizard are made available in the Web Application.
 - Other tools provide functionality on managed or unmanaged servers:
 - SQL Profiler UI Integration - SQL Profiler generates easily analyzed traces of session content.
 - Query Editor/Data View - The Query editor allows you to query, edit, and view data.
 - Debugger - The Debugger helps you debug queries.
 - Performance Diagnostics - Performance Diagnostics help you analyze the performance of Advanced Server.

We recommend that you use a dedicated machine to host production instances of the PEM backend database. The host may be subject to high levels of data throughput, depending on the number of database servers that are being monitored and the workloads the servers are processing.

The PEM Agent



The PEM Agent is responsible for the collection of monitoring data from the machine and operating system, as well as from each of the Postgres instances to which they are bound. Each PEM Agent can monitor one physical or virtual machine and is capable of monitoring multiple database servers locally - installed on the same system, or remotely - installed on other systems. It is also responsible for executing other tasks that may be scheduled by the user (for example, server shutdowns, SQL Profiler traces, user-defined jobs).

A PEM Agent is installed by default on the PEM Server along with the installation of the PEM Server. It is generally referred to as a PEM Agent on the PEM Host. Separately, the PEM Agent can also be installed on the other servers hosting the Postgres instances to be monitored using PEM.

Whether monitoring locally or remotely, the PEM Agent connects to the PEM Server using PostgreSQL's libpq, using SSL certificate-based authentication. The PEM Agent installer in Windows and pemworker CLI in Linux is responsible for

registering each agent with the PEM Server, and generating and installing the required certificates.

Please note that there is only one-way traffic between the PEM Agent and PEM Server; the PEM Agent always connects to the PEM Server.

The PEM Agent must be able to connect to each database server that it monitors. This connection is made over a TCP/IP connection (or optionally a Unix Domain Socket on Unix hosts), and may optionally use SSL. The user must configure the connection and authentication to the monitored server.

Once configured, each agent collects statistics and other information on the host and each database server and database that it monitors. Each piece of information is known as a metric and is collected by a probe. Most probes will collect multiple metrics at once for efficiency. Examples of the metrics collected include:

- Disk I/O statistics
- Network statistics
- Database server version string
- Database server configuration option (GUC) values
- Table access statistics
- Table and index sizes

A list of PEM probes can be found [here](#).

By default, the PEM Agent bound to the database server collects the OS/Database monitoring statistics and also runs any scheduled tasks/jobs for that particular database server, storing data in the pem database on the PEM server.

The Alert processing, SNMP/SMTP spoolers, and Nagios Spooler data is stored in the pem database on the PEM server and is then processed by the PEM Agent on the PEM Host by default. However, processing by other PEM Agents can be enabled by adjusting the SNMP/SMTP and Nagios parameters of the PEM Agents.

To see more information about these parameters see [Server Configuration](#).

The PEM Web Client

The PEM client is a web-based application that runs in supported browsers. The client's web interface connects to the PEM server and allows direct management of managed or unmanaged servers, and the databases and schemas that reside on them.

The client allows you to use PEM functionality that makes use of the data logged on the server through features such as the dashboards, the Postgres Log Analysis Expert, and Capacity Manager.

The SQL Profiler Plugin

You are not required to install the SQL Profiler plugin on every server, but you must install and configure the plugin on each server on which you wish to use the SQL Profiler. You may also want to install and configure SQL Profiler on un-monitored development servers. For ad-hoc use also, you may temporarily install the SQL Profiler plugin.

The plugin is installed with the EDB Postgres Advanced Server distribution but must be installed separately for use with PostgreSQL. The SQL Profiler installer is available from the [EnterpriseDB website](#).

SQL Profiler may be used on servers that are not managed through PEM, but to perform scheduled traces, a server must have the plugin installed, and must be managed by an installed and configured PEM agent.

For more information about using SQL Profiler, see the [PEM SQL Profiler Configuration Guide](#)

7.1.2 PEM Server Logon

The PEM web interface uses Apache to connect to the PEM server on port 8080 of the IP address on which the PEM server is installed. To connect to PEM, open your browser of choice, and navigate to:

`<ip_address_of_PEM_host>:8080/pem`

Where `ip_address_of_PEM_host` specifies the IP address of the host of the PEM server.



Use the fields on the Login window to authenticate yourself with the PEM server:

- Provide the name of a `pem` database user in the `Username` field. Users logon to PEM using user credentials setup as `login roles` on the PostgreSQL database used by the PEM server. By default, the `postgres` superuser account will be used for the initial logon.

We strongly recommend you create an individual role for each user. You can create a login role with the `CREATE ROLE` SQL statement, or by defining a role with the PEM client `Create – Login/Group Role` dialog. To access the dialog, connect to the PEM server database; right-click the `Login/Group Roles` node in the tree control, and select `New Login Role...` from the `Create` pull-aside menu. Roles must be granted permissions and role memberships to properly use PEM:

- users that are members of the `pem_user` role are essentially `read-only` users; they may view dashboards, change the database server connection options, but they will not be able to install agents or configure the server directory, alerts, probes, or run any of the wizard/dialog based components of PEM.
- users that are members of the `pem_admin` role have the same read permissions as members of the `pem_user` role, plus sufficient privileges to configure the servers, directory, alerts and probes.
- `administrative` users must be added to the `pem_admin` role and explicitly granted the `create` role privilege. In addition to the permissions granted through membership in the `pem_admin` role, the `create` role privilege allows an administrator to create additional pem users, and to install and register new agents.
- users can be member of one of the `PEM roles` to give right to run a particular component, to manage, or to configure PEM.
- Provide the password associated with the user in the `Password` field.

After providing your credentials, click `Login` to connect to the PEM client. PEM opens, displaying the `Global Overview` Dashboard:



7.1.3 Managing Configuration Settings

There are multiple configuration files that are read at startup by Postgres Enterprise Manager. These are as follows:

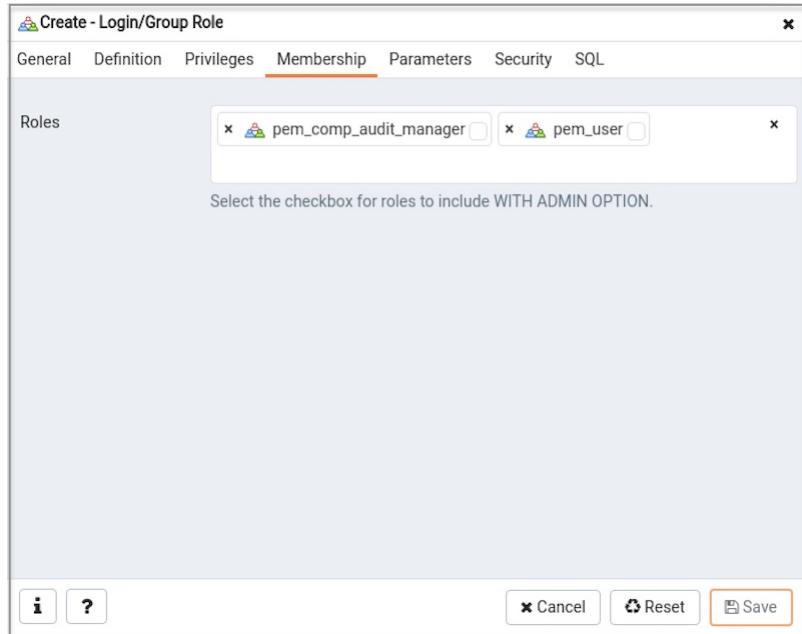
- **`config.py`**: This is the main configuration file, and should not be modified. It can be used as a reference for configuration settings, that may be overridden in one of the following files.
- **`config_distro.py`**: This file is read after `config.py` and is intended for packagers to change any settings that are required for their Postgres Enterprise Manager distribution. This may typically include certain paths and file locations. This file is optional, and may be created by packagers in the same directory as `config.py` if needed.
- **`config_local.py`**: This file is read after `config_distro.py` and is intended for end users to change any default or packaging specific settings that they may wish to adjust to meet local preferences or standards. This file is optional, and may be created by users in the same directory as `config.py` if needed.

The default `config.py` file is shown below for reference:

..../web/config.py

7.1.4 Roles for managing PEM

You can use the **Login/Group Role** dialog to allow a role with limited privileges to access PEM features such as the Audit Manager, Capacity Manager, or SQL Profiler. PEM pre-defined roles allow access to PEM functionality; roles that are assigned membership in these roles can access the associated feature.



When defining a user, use the **Membership** tab to specify the roles in which the new user is a member. The new user will share the privileges associated with each role in which it is a member. For a user to have access to PEM extended functionality, the role must be a member of the pem_user role and the pre-defined role that grants access to the feature. Use the **Roles** field to select pre-defined role names from a drop down list.

Check the checkbox to the right of the role name to allow administrative access to the functionality.

The **SQL** tab displays the SQL command that the server will execute when you click **Save**.

```

1 CREATE ROLE postgres WITH
2   NOLOGIN
3   NOSUPERUSER
4   NOCREATEDB
5   NOCREATEROLE
6   INHERIT
7   NOREPLICATION
8   CONNECTION LIMIT -1;
9
10 GRANT pem_comp_audit_manager, pem_user TO postgres;

```

Buttons: Cancel, Reset, Save

The examples shown above creates a login role named `acctg_clerk` that will have access to the Audit Manager; the role can make unlimited connections to the server at any given time.

You can use PEM pre-defined roles to allow access to the functionality listed in the table below:

Value	Parent Role	Description
pem_super_admin		Role for administration/management/configuration of all the objects within Postgres Enterprise Manager console.
pem_admin	pem_super_admin	Role for administration/management/configuration of all the agents, servers, or monitored objects that are visible to a user having pem_admin role. A user with pem_admin role can view and manage only those objects where this role has been mentioned in the Team field under the server's properties.
pem_user		Role for having read-only access to all the agents, servers, or monitored objects that are visible to a user having pem_user role. A user with pem_user role can view only those objects where this role has been mentioned in the Team field under the server's properties.
pem_config	pem_admin	Role for configuration management of Postgres Enterprise Manager.
pem_component	pem_admin	Role to run/execute all wizard/dialog based components.
pem_rest_api	pem_admin	Role to access the REST API.
pem_server_service_manager	pem_admin	Role for allowing to restart/reload the monitored database server (if server-id provided).
pem_manage_schedule_task	pem_admin	Role to configure the schedule tasks.
pem_manage_alert	pem_admin	Role for managing/configuring alerts, and its templates.
pem_config_alert	pem_config, pem_manage_alert	Role for configuring the alerts on any monitored objects.
pem_manage_probe	pem_admin	Role to create, update, delete the custom probes, and change custom probe configuration.

Value	Parent Role	Description
pem_config_probe	pem_config, pem_manage_probe	Role for probe configuration (history retention, execution frequency, enable/disable the probe) on all visible monitored objects.
pem_database_server_registration	pem_admin	Role to register a database server.
pem_comp_postgres_expert	pem_component	Role to run the Postgres Expert.
pem_comp_auto_discovery	pem_component	Role to run the Auto discovery of a database server dialog.
pem_comp_log_analysis_expert	pem_component	Role to run the Log Analysis Expert.
pem_comp_sqlprofiler	pem_component	Role to run the SQL Profiler.
pem_manage_efm	pem_admin	Role to manage Failover Manager functionalities.
pem_comp_capacity_manager	pem_component	Role to run the Capacity Manager.
pem_comp_log_manager	pem_component	Role to run the Log Manager.
pem_comp_audit_manager	pem_component	Role to run the Audit Manager.
pem_comp_tuning_wizard	pem_component	Role to run the Tuning Wizard.
pem_comp_bart	pem_component	Role to configure and manage BART server.

Note

The difference between pem_admin role and pem_super_admin role is that a user with pem_admin role can view and manage only those objects where the role has been mentioned in the Team field under the server's properties, while a user with pem_super_admin role can view and manage all the objects within Postgres Enterprise Manager console.

7.1.5 The Group Dialog

Use the **Group** dialog to add a new group to the PEM client tree control. You can use a group to simplify management of related servers or agents.



Use the **Name** field to specify a name that will identify the group in the **PEM** browser tree control.

- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

To add a server or agent to a group, right-click on the name of a server or agent, and select **Properties...** to open the properties dialog. Then, use the drop-down listbox in the **Group** field to select the group in which the object should reside.

7.1.6 Automatic Discovery of Servers

Use the **Auto Discovery** dialog to instruct a PEM agent to locate database servers that reside on a monitored system, and add a binding that allows the agent to monitor the selected server.

To enable auto discovery for a specific agent, you must enable the **Server Auto Discovery** probe. To access the **Manage Probes** tab, highlight the name of a PEM agent in the PEM client tree control, and select **Manage Probes...** from the **Management** menu. When the Manage Probes tab opens, confirm that the slider control in the **Enabled?** column is set to **Yes**.

To open the **Auto Discovery** dialog, highlight the name of a PEM agent in the PEM client tree control, and select **Auto Discovery...** from the **Management** menu.



When the **Auto Discovery** dialog opens, the **Discovered Database Servers** box will display a list of servers that are not currently monitored by a PEM agent. Check the box next to a server name to display information about the server in the **Server Connection Details** box, and provide any missing information to bind the server to the currently selected agent in the **Agent Connection Details** box.

Use the **Select All** button to select the box next to all of the displayed servers, or **Unselect All** to unselect all of the boxes to the left of the server names.

The fields in the **Server Connection Details** box provide information about the server that PEM will monitor:

- Accept or modify the name of the monitored server in the **Name** field. The specified name will be displayed in the tree control of the PEM client.
- Use the **Server group** drop-down listbox to select the server group under which the server will be displayed in the PEM client tree control.
- Use the **Host name/address** field to specify the IP address of the monitored server.
- The **Port** field displays the port that is monitored by the server; this field may not be modified.
- Provide the name of the service in the **Service ID** field. Please note that the service name must be provided to enable some PEM functionality.
- By default, the **Maintenance database** field indicates that the selected server uses a **postgres** maintenance database. Customize the content of the **Maintenance database** field for your installation.

The fields in the **Agent Connection Details** box specify the properties that the PEM agent will use when connecting to the server:

- The **Host** field displays the IP address that will be used for the PEM agent binding.
- The **Username** field displays the name that will be used by the PEM agent when connecting to the selected server.
- The **Password** field displays the password associated with the specified user name.
- Use the drop-down listbox in the **SSL mode** field to specify your SSL connection preferences.

When you've finished specifying the connection properties for the servers that you are binding for monitoring, click the **OK** button to save the properties. Click **Cancel** to exit without preserving any changes.



The dialog shown above displays the values required to bind an instance of Advanced Server for monitoring by PEM.

7.1.7 Defining a server

Use the **Create – Server** dialog to describe a new server connection, bind the server to a PEM agent, and display the server to the PEM tree control.



Use the fields on the **General** tab to describe the general properties of the server:

- Use the **Name** field to specify a user-friendly name for the server. The name specified will identify the server in the PEM client tree control.
- You can use **groups** to organize your servers and agents in the PEM client tree control. Using groups can help you manage large numbers of servers more easily. For example, you may want to have a production group, a test group, or LAN specific groups. Use the **Group** drop-down listbox to select the group in which the new server will be displayed.
- Use the **Team** field to specify a PostgreSQL role name. Only PEM users who are members of this role, who created the server initially, or have superuser privileges on the PEM server will see this server when they logon to PEM. If this field is left blank, all PEM users will see the server.
- Use the **Background** color selector to select the color that will be displayed in the PEM tree control behind database objects that are stored on the server.

- Use the **Foreground** color selector to select the font color of labels in the PEM tree control for objects stored on the server.
- Check the box next to **Connect now?** to instruct PEM to attempt a connection to the database server when you click the Save button on the Create - Server dialog. Leave the **Connect now?** checkbox unchecked if you do not want to establish a connection to the server immediately. If you do not select the **Connect now?** option, the connection parameters are not validated until you attempt a connection.
- Provide notes about the server in the **Comments** field.



Use fields on the **Connection** tab to specify connection details for the server:

- Specify the IP address of the server host, or the fully qualified domain name in the **Host name/address** field. On Unix based systems, the address field may be left blank to use the default PostgreSQL Unix Domain Socket on the local machine, or may be set to an alternate path containing a PostgreSQL socket. If you enter a path, the path must begin with a "/".
- Specify the port number of the host in the **Port** field.
- Use the **Maintenance database** field to specify the name of the initial database that PEM will connect to, and that will be expected to contain the **pgAgent** schema and adminpack objects if installed (both are optional). On an Advanced Server database, the maintenance database is named 'edb'. On PostgreSQL 8.1 and above, the maintenance DB for PostgreSQL is named 'postgres'; on earlier versions 'template1' is often used, though it is preferable to create a 'postgres' database for this purpose to avoid cluttering the template database.
- Specify the name that will be used when authenticating with the server in the **Username** field.
- Provide the password associated with the specified user in the **Password** field.
- Check the box next to **Save password?** to instruct the PEM server to save the password in encrypted format on the PEM server backend database server for later reuse. Password will be stored per server per user basis, hence - it won't be shared with other team members. To remove a password, disconnect from the server, click on the 'Clear Saved Password' menu item under Object/Context menu of the database server.
- Use the **Role** field to specify the name of the role that is assigned the privileges that the client should use after connecting to the server. This allows you to connect as one role, and then assume the permissions of another role when the connection is established (the one you specified in this field). The connecting role must be a member of the role specified.



Use the fields on the **SSL** tab to configure SSL.

- Use the drop-down list box in the **SSL mode** field to select the type of SSL connection the server should use. For more information about using SSL encryption, see [Section 33.18 of the Postgres documentation](#).

You can use the platform-specific File manager dialog to upload files that support SSL encryption to the server. To access the File manager dialog, click the icon that is located to the right of each of the following fields.

- Use the **Client certificate** field to specify the file containing the client SSL certificate. This file will replace the default `~/.postgresql/postgresql.crt` if PEM is installed in Desktop mode, and `<STORAGE_DIR>/<USERNAME>/postgresql/postgresql.crt` if PEM is installed in Web mode. This parameter is ignored if an SSL connection is not made.
- Use the **Client certificate key** field to specify the file containing the secret key used for the client certificate. This file will replace the default `~/.postgresql/postgresql.key` if PEM is installed in Desktop mode, and `<STORAGE_DIR>/<USERNAME>/postgresql/postgresql.key` if PEM is installed in Web mode. This parameter is ignored if an SSL connection is not made.
- Use the **Root certificate** field to specify the file containing the SSL certificate authority. This file will replace the default `~/.postgresql/root.crt`. This parameter is ignored if an SSL connection is not made.
- Use the **Certificate revocation list** field to specify the file containing the SSL certificate revocation list. This list will replace the default list, found in `~/.postgresql/root.crl`. This parameter is ignored if an SSL connection is not made.
- When **SSL compression?** is set to **True**, data sent over SSL connections will be compressed. The default value is **False** (compression is disabled). This parameter is ignored if an SSL connection is not made.

WARNING: certificates, private keys, and the revocation list are stored in the per-user file storage area on the server, which is owned by the user account under which the PEM server process is run. This means that administrators of the server may be able to access those files; appropriate caution should be taken before choosing to use this feature.



Use the fields on the **SSH Tunnel** tab to configure SSH Tunneling. You can use a tunnel to connect a database server (through an intermediary proxy host) to a server that resides on a network to which the client may not be able to connect directly.

- Set **Use SSH tunneling** to **Yes** to specify that PEM should use an SSH tunnel when connecting to the specified server.
- Specify the name or IP address of the SSH host (through which client connections will be forwarded) in the **Tunnel host** field.
- Specify the port of the SSH host (through which client connections will be forwarded) in the **Tunnel port** field.
- Specify the name of a user with login privileges for the SSH host in the **Username** field.
- Specify the type of authentication that will be used when connecting to the SSH host in the **Authentication** field.
 - Select **Password** to specify that PEM will use a password for authentication to the SSH host. This is the default.
 - Select **Identity file** to specify that PEM will use a private key file when connecting.
- If the SSH host is expecting a private key file for authentication, use the **Identity file** field to specify the location of the key file.
- If the SSH host is expecting a password, use the **Password** field to specify the password, or if an identity file is being used, the passphrase.



Use fields on the **Advanced** tab to specify details that are used to manage the server:

- Specify the IP address of the server host in the **Host Address** field.
- Use the **DB restriction** field to specify a SQL restriction that will be used against the `pg_database` table to limit the databases displayed in the tree control. For example, you might enter: `'live_db'`, `'test_db'` to instruct the PEM browser to display only the `live_db` and `test_db` databases.

- Use the **Password file** field to specify the location of a password file (.pgpass). The .pgpass file allows a user to login without providing a password when they connect, and it must be present on the PEM server. For more information, see [Section 33.15 of the Postgres documentation](#). Please note: Use of a password file is only supported when PEM is using libpq v10.0 or later to connect to the server.
- Use the **Service ID** field to specify parameters to control the database service process. For servers that are stored in the Enterprise Manager directory, enter the service ID. On Windows machines, this is the identifier for the Windows service. On *nix machines, this is the name of the init script used to start the server in /etc/init.d. An example of an ID on all platforms is **postgresql-9.0**. For local servers, the setting is operating system dependent:
 - If the PEM client is running on a Windows machine, it can control the postmaster service if you have enough access rights. Enter the name of the service. In case of a remote server, it must be prepended by the machine name (e.g. PSE1\pgsql-8.0). PEM will automatically discover services running on your local machine.
 - If the PEM client is running on a Unix machine, it can control processes running on the local machine if you have enough access rights. Enter a full path and needed options to access the pg_ctl program. When executing service control functions, PEM will append status/start/stop keywords to this. For example: `sudo /usr/local/pgsql/bin/pg_ctl -D /data/pgsql`
- If the server is a member of a [Failover Manager](#) cluster, you can use PEM to monitor the health of the cluster and to replace the primary node if necessary. To enable PEM to monitor Failover Manager, use the **EFM cluster name** field to specify the cluster name. The cluster name is the prefix of the name of the Failover Manager cluster properties file. For example, if the cluster properties file is named **efm.properties**, the cluster name is **efm**.
- If you are using PEM to monitor the status of a [Failover Manager](#) cluster, use the **EFM installation path** field to specify the location of the Failover Manager binary file. By default, the Failover Manager binary file is installed in **/usr/efm-2.x/bin**, where **x** specifies the Failover Manager version.



Use fields on the **PEM Agent** tab to specify connection details for the PEM agent:

- Specify **Yes** in the **Remote monitoring?** field to indicate that the PEM agent does not reside on the same host as the monitored server. When remote monitoring is enabled, agent level statistics for the monitored server will not be available for custom charts and dashboards, and the remote server will not be accessible by some PEM utilities (such as Audit Manager, Capacity Manager, Log Manager, Postgres Expert and Tuning Wizard).
- Use the drop-down listbox to the right of the **Bound agent** label to select a PEM agent. One agent can monitor multiple Postgres servers.
- Enter the IP address or socket path that the agent should use when connecting to the database server in the **Host** field. By default, the agent will use the host address shown on the **General** tab. On a Unix server, you may wish to specify a socket path, e.g. `/tmp`.
- Enter the **Port** number that the agent will use when connecting to the server. By default, the agent will use the port defined on the **Properties** tab.
- Use the drop-down listbox in the **SSL** field to specify an SSL operational mode; select from require, prefer, allow, disable, verify-ca or verify-full.

Mode Description:

require	To require SSL encryption for transactions between the server and the agent.
prefer	To use SSL encryption between the server and the agent if SSL encryption is available.
allow	To allow the connection to use SSL if required by the server.
disable	To disable SSL encryption between the agent and the server.
verify-ca	To require SSL encryption, and to require the server to authenticate using a certificate registered by a certificate authority.
verify-full	To require SSL encryption, and to require the server to authenticate using a certificate registered by a trusted certificate authority.

For information about using SSL encryption, see [Section 31.17 of the Postgres documentation](#).

- Use the **Database** field to specify the name of the Postgres Plus database to which the agent will initially connect.
- Specify the name of the user that agent should use when connecting to the server in the **User name** field. Note that if the specified user is not a database superuser, then some of the features will not work as expected. If you are using Postgres version 10 or above, you can use the **pg_monitor** role to grant the required privileges to a non-superuser. For information about **pg_monitor** role, see [Default Roles](#).
- Specify the password that the agent should use when connecting to the server in the **Password** field, and verify it by typing it again in the **Confirm password** field. If you do not specify a password, you will need to configure the authentication for the agent manually; you can use a **.pgpass** file for example.
- Specify **Yes** in the **Allow takeover?** field to specify that another agent may be signaled (for example, by a fencing script) to monitor the server. This feature allows an agent to take responsibility for the monitoring of the database server if, for example, the server is part of a [high availability](#) failover process.

If you experience connection problems, please visit the [connection problems](#) page.

To view the properties of a server, right-click on the server name in the PEM client tree control, and select the **Properties...** option from the context menu. To modify a server's properties, disconnect from the server before opening the **Properties** dialog.

7.1.8 Defining and Monitoring Postgres instances on AWS

There are two scenarios in which you can monitor a Postgres instance on an AWS host with PEM:

- Postgres Instance running on AWS EC2
- Postgres Instance running on AWS RDS

Monitoring a Postgres Instance Running on AWS EC2

After creating a Postgres instance on AWS EC2, you can use the PEM server to register and monitor your instance. The following scenarios are currently supported:

- Postgres instance and PEM Agent running on the same AWS EC2 and a PEM Server running on your local machine.
- Postgres instance and PEM Agent running on the same local machine and a PEM Server running on AWS EC2.
- Postgres instance and PEM Agent running on the same AWS EC2 and a PEM Server running in different AWS

EC2.

Note

In the first two scenarios, you must configure the VPN on AWS EC2 , so the AWS EC2 instance can access the **pem** database. Please contact your network administrator to setup the VPN if needed.

The PEM Agent running on AWS EC2 or on your local machine should be registered to the PEM Server. Please note that when registering the PEM Agent with the PEM Server you should use the hostname of AWS EC2 instance. For more details on registering the PEM Agent see, [PEM Self Registration](#).

You can register the Postgres instance running on AWS EC2 on PEM Server using the **Create – Server** dialog. For more details on registering the server using **Create – Server** dialog see, [Define a Server](#). Use the **PEM Agent** tab on the **Create – Server** dialog to bind the registered PEM Agent with the Postgres instance.

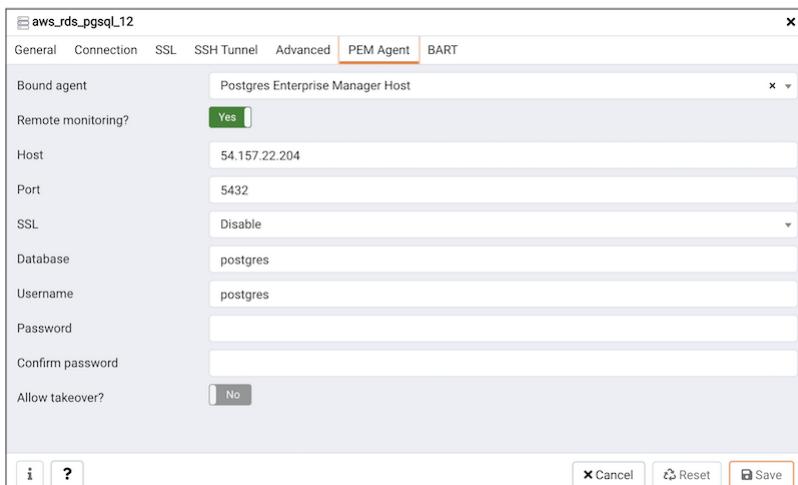
When the PEM Agent is registered to the PEM Server and your Postgres instance that is running on AWS EC2 is registered to the PEM Server, you can monitor your instance with PEM.

Monitoring a Postgres Instance Running on AWS RDS

While creating an AWS RDS database, choose **PostgreSQL** when prompted for **Engine options**. After creating a **Postgres(RDS)** instance on AWS, use **Create – Server** dialog to add the **Postgres(RDS)** instance to the PEM Server. Using this dialog you can describe a new server connection, bind the server to a PEM Agent, and display the server to the PEM browser tree control.

For detailed information on the **Create – Server** dialog and configuration details for each tab, see [Define a Server](#).

The **PEM Agent** tab in the **Create – Server** dialog must have the **Remote Monitoring** field set to **Yes** to monitor the **Postgres(RDS)** instance on AWS instance using PEM Server.



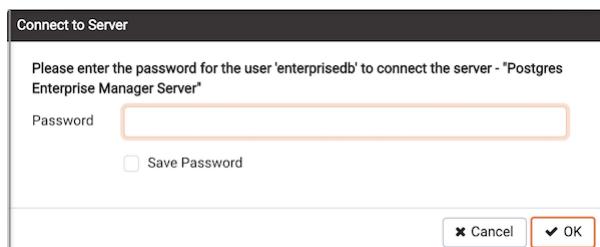
As the PEM Agent will be monitoring the Postgres(RDS) AWS instance remotely, the functionality will be limited as described below:

Feature Name	Works with remote PEM Agent	Comments
Audit Manager	No	

Feature Name	Works with remote PEM Agent	Comments
Capacity Manager	Limited	There will be no correlation between the database server and operating system metrics.
Log Manager	No	
Manage Alerts	Limited	When you run an alert script on the database server, it will run on the machine where the bound PEM Agent is running, and not on the actual database server machine.
Manage Charts	Yes	
Manage Dashboards	Limited	Some dashboards may not be able to show complete data. For example, the operating system information of the database server will not be displayed as it is not available.
Manage Probes	Limited	Some of the PEM probes will not return information, and some of the functionalities may be affected. For details about probe functionality, see the PEM Agent Guide .
Postgres Expert	Limited	The Postgres Expert will provide partial information as operating system information is not available.
Postgres Log Analysis Expert	No	The Postgres Log Analysis Expert will not be able to perform an analysis as it is dependent on the logs imported by log manager, which will not work as required.
Scheduled Tasks	Limited	Scheduled tasks will work only for database server; scripts will run on a remote Agent.
Tuning Wizard	No	
System Reports	Yes	
Core Usage Reports	Limited	The Core Usage report will not show complete information. For example, the platform, number of cores, and total RAM will not be displayed.
Managing BART	No	BART requires password less authentication between two machines, where database server and BART are installed. An AWS RDS instance doesn't allow to use host access.

7.1.9 Connect to server

After defining a server connection, use the **Connect to Server** dialog to authenticate with a server and access the objects stored on the server. To access the dialog, right click on the server name in the PEM client tree control, and select **Connect Server** from the context menu.



If prompted, provide authentication information for the selected server:

- Use the **Password** field to provide the password of the user that is associated with the defined server.

- Check the box next to **Save Password** to instruct the server to save the password for future connections; if you save the password, you will not be prompted when reconnecting to the database server with this server definition.

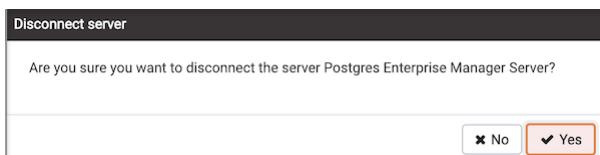
The browser displays a message in a green status bar in the lower right corner when the server connects successfully.

If you receive an error message while attempting a connection, verify that your network is allowing PEM and the host of the database server to communicate. For detailed information about a specific error message, please see the [Connection Error](#) help page.

To review or modify connection details, right-click on the name of the server, and select **Properties...** from the context menu.

Disconnecting from a Server

To disconnect from a server, right-click on the server name in the **Browser** tree control and select **Disconnect Server** from the context menu. A popup will ask you to confirm that you wish to disconnect the selected server.



7.1.10 Controlling a Server

If you provided a **Service ID** on the **Advanced** tab of the **Server** property dialogue, the PEM server can control the database service process.

- If the PEM client is running on a Windows machine, it can control the postmaster service if you have sufficient access rights. In case of a remote server, the service name must be prepended by the machine name (e.g. PSE1pgsql-8.0).
- If the PEM client is running on a Unix machine, it can control processes running on the local machine if you have sufficient access rights. When executing service control functions, PEM will append status/start/stop keywords to the service name provided.

7.1.11 Connection error

When connecting to a PostgreSQL server, you may get an error message. If you encounter an error message, please review the message carefully; each error message attempts to incorporate the information you'll need to resolve the problem. For more details about specific errors, please locate the error message in the list below:

Connection to the server has been lost



This error message indicates that the connection attempt has taken longer than the specified threshold; there may be a problem with the connection properties provided on the [Server](#) dialog, network connectivity issues, or the server may not be running.

could not connect to Server: Connection refused



If pgAdmin displays this message, there are two possible reasons for this:

- the database server isn't running - simply start it.
- the server isn't configured to accept TCP/IP requests on the address shown.

For security reasons, a PostgreSQL server "out of the box" doesn't listen on TCP/IP ports. Instead, it must be enabled to listen for TCP/IP requests. This can be done by adding `tcpip = true` to the `postgresql.conf` file for Versions 7.3.x and 7.4.x, or `listen_addresses='*' for Version 8.0.x and above`; this will make the server accept connections on any IP interface.

For further information, please refer to the PostgreSQL documentation about [runtime configuration](#).

FATAL: no pg_hba.conf entry



If pgAdmin displays this message when connecting, your server can be contacted correctly over the network, but is not configured to accept your connection. Your client has not been detected as a legal user for the database.

To connect to a server, the `pg_hba.conf` file on the database server must be configured to accept connections from the host of the pgAdmin client. Modify the `pg_hba.conf` file on the database server host, and add an entry in the form:

- host template1 postgres 192.168.0.0/24 md5 for an IPV4 network
- host template1 postgres ::ffff:192.168.0.0/120 md5 for an IPV6 network

For more information, please refer to the PostgreSQL documentation about [client authentication](#).

FATAL: password authentication failed



- The **password authentication failed for user** error message indicates there may be a problem with the password you entered. Retry the password to confirm you entered it correctly. If the error message returns, make sure that you have the correct password, that you are authorized to access the server, and that the access has been correctly configured in the server's postgresql.conf configuration file.

7.2 Managing a PEM Agent

The PEM agent is responsible for implementing scheduled tasks on the PEM server on behalf of the server. The agent runs as a service (on Windows) or as a daemon (on Linux). The PEM server installer automatically installs and configures an agent that is responsible for monitoring the PEM server; you can use the PEM agent installer to add additional agents.

Contents:

7.2.1 PEM Agent Properties

The **PEM Agent Properties** dialog provides information about the PEM agent from which the dialog was opened; to open the dialog, right-click on an agent name in the PEM client tree control, and select **Properties** from the context menu.



- The **Description** field displays a modifiable description of the PEM agent. This description is displayed in the tree control of the PEM client.
- You can use **groups** to organize your servers and agents in the PEM client tree control. Use the **Group** drop-down listbox to select the group in which the agent will be displayed.
- Use the **Team** field to specify the name of the group role that should be able to access servers monitored by the agent; the servers monitored by this agent will be displayed in the PEM client tree control to connected team members. Please note that this is a convenience feature. The **Team** field does not provide true isolation, and should not be used for security purposes.
- The **Heartbeat interval** fields displays the length of time that will elapse between reports from the PEM agent to the PEM server. Use the selectors next to the **Minutes** or **Seconds** fields to modify the interval.



Use the fields on the **Job Notifications** tab to configure the email notification settings on agent level:

- Use the **Override default configuration?** switch to specify if you want the agent level job notification settings to override the default job notification settings. If you select Yes for this switch, you can use the rest of the settings on this dialog to define when and to whom the job notifications should be sent. Please note that the rest of the settings on this dialog work only if you enable the **Override default configuration?** switch.
- Use the **Email on job completion?** switch to specify if the job notification should be sent on the successful

job completion.

- Use the **Email on a job failure?** switch to specify if the job notification should be sent on the failure of a job.
- Use the **Email group** field to specify the email group to whom the job notification should be sent.

Parameter	Value	Category
Agent Id	1	configuration
Running as root?	true	capability
Running as User	root	capability
Platform	"CentOS Linux 7 (Core)"	capability
Architecture	x64	capability
PEM host	127.0.0.1	configuration
PEM port	5444	configuration
Log level	debug1	configuration
Agent SSL key path	/root/.pem//agent1.key	configuration
Agent SSL crt path	/root/.pem//agent1.crt	configuration
Long wait	30	configuration
Short wait	10	configuration
Alert threads	1	configuration

Buttons: i ? Cancel Reset Save

Agent Configurations tab lists down all the current configurations and capabilities of a agent.

- The **Parameter** column displays list of parameters.
- The **Value** column displays current value of the corresponding parameter.
- The **Category** column displays category of the corresponding parameter, it can be either "configuration" or "capability".

7.2.2 Binding an Agent to a Server

The PEM agent runs as a service (on Windows) or as a daemon (on Linux), and is responsible for implementing scheduled tasks on the PEM server on behalf of the server. The PEM server installer automatically installs and configures an agent that is responsible for monitoring the PEM server. The PEM agent installer will setup and configure the agent to start automatically at boot time, however the agent can also be manually [started](#) if required.

To create a binding for a registered server, right click on the name of the server in the tree control, and select **Properties** from the context menu. Open the **PEM Agent** tab:



Use the fields on the **PEM Agent** tab to associate the server (defined on the Connection tab) with a PEM agent:

Use fields on the **PEM Agent** tab to specify connection details for the PEM agent:

- Specify **Yes** in the **Remote monitoring?** field to indicate that the PEM agent does not reside on the same host as the monitored server. When remote monitoring is enabled, agent level statistics for the monitored server will not be available for custom charts and dashboards, and the remote server will not be accessible by some PEM utilities (such as Audit Manager, Capacity Manager, Log Manager, Postgres Expert and Tuning Wizard).
- Select an Enterprise Manager agent using the drop-down listbox to the right of the **Bound agent** label. One agent can monitor multiple Postgres servers.
- Enter the IP address or socket path that the agent should use when connecting to the database server in the **Host** field. By default, the agent will use the host address shown on the **General** tab. On a Unix server, you may wish to specify a socket path, e.g. `/tmp`.
- Enter the **Port** number that the agent will use when connecting to the server. By default, the agent will use the port defined on the **Properties** tab.
- Use the drop-down listbox in the **SSL** field to specify an SSL operational mode; specify require, prefer, allow, disable, verify-ca or verify-full.
- Use the **SSL** field to specify an SSL operational mode.

Mode Specify:

require	To require SSL encryption for transactions between the server and the agent.
prefer	To use SSL encryption between the server and the agent if SSL encryption is available.
allow	To allow the connection to use SSL if required by the server.
disable	To disable SSL encryption between the agent and the server.
verify-ca	To require SSL encryption, and to require the server to authenticate using a certificate registered by a certificate authority.
verify-full	To require SSL encryption, and to require the server to authenticate using a certificate registered by a trusted certificate authority.

For information about using SSL encryption, see [Section 31.17 of the Postgres documentation](#).

- Use the **Database** field to specify the name of the Postgres Plus database to which the agent will initially connect.
- Specify the name of the user that agent should use when connecting to the server in the **User name** field. Note that if the specified user is not a database superuser, then some of the features will not work as expected. If you are using Postgres version 10 or above, you can use the **pg_monitor** role to grant the required privileges to a non-superuser. For information about **pg_monitor** role, see [Default Roles](#).
- Specify the password that the agent should use when connecting to the server in the **Password** field, and verify it by typing it again in the **Confirm password** field. If you do not specify a password, you will need to configure the

authentication for the agent manually; you can use a `.pgpass` file for example.

- Specify `Yes` in the `Allow takeover?` field to specify that the server may be "taken over" by another agent. This feature allows an agent to take responsibility for the monitoring of the database server if, for example, the server has been moved to another host as part of a [high availability failover process](#).

Contents:

7.2.2.1 PEM Agent Configuration Parameters

A number of user-configurable parameters and registry entries control the behavior of the PEM Agent. With the exception of the `PEM_MAXCONN` (or `pem_maxconn`) parameter, we strongly recommend against modifying any of the configuration parameters or registry entries listed below without first consulting EnterpriseDB support experts.

- On 32 bit Windows systems, PEM registry entries are located in `HKEY_LOCAL_MACHINE\Software\EnterpriseDB\PEM\agent`
- On 64 bit Windows systems, PEM registry entries are located in `HKEY_LOCAL_MACHINE\Software\Wow6432Node\EnterpriseDB\PEM\agent`
- On Linux systems, PEM configuration options are stored in the `agent.cfg` file, located (by default) in `/usr/edb/pem/agent/etc`

Parameter Name	Description	Value (if applicable)
<code>PEM_HOST</code> (on Windows) or <code>pem_host</code> (on Linux)	The IP address or hostname of the PEM server.	By default, set to 127.0.0.1.
<code>PEM_PORT</code> (on Windows) or <code>pem_port</code> (on Linux)	The database server port to which the agent connects to communicate with the PEM server.	By default, the PEM server monitors port 5432.
<code>AgentID</code> (on Windows) or <code>agent_id</code> (on Linux)	A unique identifier assigned to the PEM agent.	The first agent is assigned an identifier of '1', the second agent is assigned an identifier of '2', and so on.
<code>AgentUser</code> (on Windows) or <code>agent_user</code> (on Linux)	User to connect the PEM database server	If present, and not set to empty string, it will be used to connect the PEM database server.
<code>AgentCrtPath</code> (on Windows) or <code>agent_ssl_crt</code> (on Linux)	The complete path to the PEM agent's certificate file.	By default, on Windows, <code>C:\Users\user_name\AppData\Roaming\pem/agent.crt</code> . By default on Linux, <code>/root/.pem/agent.crt</code> .
<code>AgentKeyPath</code> (on Windows) or <code>agent_ssl_key</code> (on Linux)	The complete path to the PEM agent's key file.	By default, on Windows, <code>C:\Users\user_name\AppData\Roaming\pem/agent.key</code> . By default on Linux, <code>/root/.pem/agent.key</code> .

Parameter Name	Description	Value (if applicable)
AgentFlagDir (on Windows) or agent_flag_dir (on Linux)	Used for HA support. Specifies the directory path checked for requests to take over monitoring another server. Requests are made in the form of a file in the specified flag directory.	Not set by default. This option allows you to override the hard-coded d
LogLevel (on Windows) or log_level (on Linux)	Log level specifies the type of event that will be written to the PEM log files.	Log level may be set to: error, debug1, debug2, or warning By default, level is set to warning
log_location (on Linux only)	Specifies the location of the PEM worker log file.	On Linux, /var/log/pem/worker.log. On Windows, Logs & errors will be reported in the Application event log.
agent_log_location (on Linux only)	Specifies the location of the PEM agent log file.	On Linux, /var/log/pem/agent.log. On Windows, Logs & errors will be reported in the Application event log.
ShortWait (on Windows) or short_wait (on Linux)	The minimum length of time (in seconds) that the PEM agent will wait before checking which probes are next in the queue (waiting to run).	By default, 10 seconds.
LongWait (on Windows) or long_wait (on Linux)	The maximum length of time (in seconds) that the PEM agent will wait before attempting to connect to the PEM server if an initial connection attempt fails.	By default, 30 seconds.
AlertThreads (on Windows) or alert_threads (on Linux)	The number of alert threads to be spawned by the agent.	Set to 1 for the agent that resides on the host of the PEM server; should be 0 for all other agents.

Parameter Name	Description	Value (if applicable)
EnableSMTP (on Windows) or enable_smtp (on Linux)	When set to true for multiple PEM Agents (7.13 or lesser) and PEM backend database (9.4 or lesser) then it may send more duplicate emails. Whereas for PEM Agents (7.14 or higher) and PEM backend database (9.5 or higher) then it may send lesser duplicate emails.	By default, set to true for the agent that resides on the host of the PEM false for all other agents.
EnableSNMP (on Windows) or enable_snmp (on Linux)	When set to true for multiple PEM Agents (7.13 or lesser) and PEM backend database (9.4 or lesser) then it may send more duplicate traps. Whereas for PEM Agents (7.14 or higher) and PEM backend database (9.5 or higher) then it may send lesser duplicate traps.	By default, set to true for the agent that resides on the host of the PEM false for all other agents.
enable_nagios (on Linux only)	When set to true, Nagios alerting is enabled.	By default, set to true for the agent that resides on the host of the PEM false for all other agents.
EnableWebhook (on Windows) or enable_webhook (on Linux)	When set to true, Webhook alerting is enabled.	By default, set to true for the agent that resides on the host of the PEM false for all other agents.
MaxWebhookRetries (on Windows) or max_webhook_retries (on Linux)	Set maximum number of times pemAgent should retry to call webhooks on failure.	Default 3.
ConnectTimeout (on Windows) or connect_timeout (on Linux)	The maximum length of time (in seconds, written as a decimal integer string) that the agent will wait for a connection.	Not set by default. If set to 0, the agent will wait indefinitely.

Parameter Name	Description	Value (if applicable)
AllowServerRestart (on Windows) or allow_server_restart (on Linux)	If set to TRUE, the agent can restart the database server that it monitors. Some PEM features may be enabled/disabled, depending on the value of this parameter.	By default, set to TRUE.
MaxConnections (on Windows) or max_connections (on Linux)	The maximum number of probe connections used by the connection throttler.	By default, set to 0 (an unlimited number of connections).
ConnectionLifetime (on Windows) or connection_lifetime (on Linux)	Use ConnectionLifetime (or connection_lifetime) to specify the minimum number of seconds an open but idle connection is retained. This parameter is ignored if the value specified in MaxConnections is reached and a new connection (to a different database) is required to satisfy a waiting request.	By default, set to 0 (a connection is dropped when the connection is idle the agent's processing loop completes a cycle in which the connector has not been used).
HeartbeatConnection (on Windows) or heartbeat_connection (on Linux)	When set to TRUE, a dedicated connection used for sending the heartbeats.	By default, set to FALSE.
AllowBatchProbes (on Windows) or allow_batch_probes (on Linux)	If set to TRUE, the user will be able to create batch probes using custom probes feature.	By default, set to FALSE.
BatchScriptDir (on Windows) or batch_script_dir (on Linux)	Provide the path where script file (for alerting) will be stored.	On Windows, C:\Users\user_name\AppData\Local\Temp. On Linux, set to /tmp.
AllowBatchJobSteps (on Windows) or batch_script_user	Provide the username who will run the script.	On Windows, set to TRUE and restart PEM Agent. Entries located in HKEY_LOCAL_MACHINE\Software\Wow6432Node\EnterpriseDB\PEM. On Linux, Restart the agent after modifying the file. If you do not specify a user, or the specified user does not exist, then the script will not be executed.

Parameter Name	Description	Value (if applicable)
ConnectionCustomSetup (on Windows) or connection_custom_setup (on Linux)	Use this parameter to provide SQL code that will be invoked each time a new connection with the monitored server is established.	By default, no value is provided.
ca_file (Linux only)	Provide the path where the CA certificate resides.	By default, /opt/PEM/agent/share/certs/ca-bundle.crt
WebhookSSLKey (on Windows) or webhook_ssl_key (on Linux)	The complete path to the webhook's SSL client key file.	
WebhookSSLCrt (on Windows) or webhook_ssl_crt (on Linux)	The complete path to the webhook's SSL client certificate file.	
WebhookSSLCaCrt (on Windows) or webhook_ssl_ca_crt (on Linux)	The complete path to the webhook's SSL ca certificate file.	
WebhookSSLcrl (on Windows) or webhook_ssl_crl (on Linux)	The complete path of the CRL file to validate webhook server certificate.	
AllowInsecureWebhooks (on Windows) or allow_insecure_webhooks (on Linux)	When set to true, allow webhooks to call with insecure flag.	false

Note

If you add or remove any of the parameter in the `agent.cfg` file then agent must be restarted to apply the changes.

7.2.2.2 High Availability Integration

In high availability (HA) configurations, the database servers that are being monitored may be moved ("failed over") to a different host in the event of any problems, such as a hardware failure. There are numerous ways to maintain a backup server using features of Postgres and external tools. Please consult the Postgres documentation for further details.

In order to run in an HA environment, it is recommended that a PEM agent be installed on both the primary host machine, and any secondary machines that may be used as backups. The server is bound to the agent running on the primary host in the [normal fashion](#).

When the clustering solution initiates a failover of Postgres from one server to another, the PEM agent on the server that is taking over the running of the database may be instructed to take over the monitoring of the database server as well.

The server must first be configured to allow "takeovers" using the [Allow takeover? configuration option](#) on the [PEM Agent](#) tab of the server configuration dialogue.

To instruct the agent to takeover the monitoring of a server, the failover process must simply create a file in a special "flag" directory which will instruct the agent to take responsibility for the specified server. A command such as the following could be added to a failover script on a Linux server for example:

```
touch /tmp/pem/agent-AGENTID/takeover-server-SERVERID
```

where **AGENTID** is the numeric ID of the agent that should takeover the monitoring of the server, and **SERVERID** is the numeric ID of the server that should be taken over. The IDs may be found by logging into the PEM client, and selecting the Agent or Server and viewing the ID values on the [Properties](#) pane of the main window.

The agent will take over monitoring of the failed-over server within approximately 30 seconds in a standard configuration of PEM.

The flag directory used by the agent is `$TMPDIR/pem/agent-AGENTID` by default (where `$TMPDIR` is as set for the user account under which the agent runs, usually `root` on Linux/Unix, or `Administrator` on Windows). The directory path can be overridden using the [AgentFlagDir](#) configuration option in the registry on Windows, or the `agent_flag_dir` option in the agent configuration file on other platforms.

7.2.3 Controlling the PEM Agent

On Linux platforms, the name of the service script that controls a PEM agent is `pemagent`. You can use the `pemagent` service script to control the PEM agent. Enter:

```
/etc/init.d/pem_agent action
```

Where `action` specifies the action taken by the service. Specify:

- start to start the service.
- stop to stop the service.
- restart to stop and then start the service.
- status to check the status of the service.

To determine if a service is running on RHEL or CentOS version 7.x, open a command line, and issue the command:

```
systemctl pemagent action
```

Where `action` is the action taken by the service. You can specify:

- start to start the service.
- stop to stop the service.
- restart to stop and then start the service.
- status to inquire about the current status of the service.

Controlling the PEM Agent on Windows

The Windows operating system includes a graphical service controller (the Windows [Services](#) applet) that displays the server status, and offers point-and-click service control. The Services applet can be accessed through the Windows

Control Panel. When the utility opens, use the scroll bar to navigate through the listed services to highlight the **Postgres Enterprise Manager - pemAgent** service name.

- Use the Stop the service option to stop a service.
 - Use the Pause the service option to instruct Postgres to reload a service's configuration parameters.
 - Use the Start the service option to start a service.
-

7.2.4 High Availability Integration

In high availability (HA) configurations, the database servers that are being monitored may be moved ("failed over") to a different host in the event of any problems, such as a hardware failure. There are numerous ways to maintain a backup server using features of Postgres and external tools. Please consult the Postgres documentation for further details.

In order to run in an HA environment, it is recommended that a PEM agent be installed on both the primary host machine, and any secondary machines that may be used as backups. The server is bound to the agent running on the primary host in the [normal fashion](#).

When the clustering solution initiates a failover of Postgres from one server to another, the PEM agent on the server that is taking over the running of the database may be instructed to take over the monitoring of the database server as well. The server must first be configured to allow "takeovers" using the [Allow takeover? configuration option](#) on the **PEM Agent** tab of the server configuration dialogue.

To instruct the agent to takeover the monitoring of a server, the failover process must simply create a file in a special "flag" directory which will instruct the agent to take responsibility for the specified server. A command such as the following could be added to a failover script on a Linux server for example:

```
touch /tmp/pem/agent-AGENTID/takeover-server-SERVERID
```

where **AGENTID** is the numeric ID of the agent that should takeover the monitoring of the server, and **SERVERID** is the numeric ID of the server that should be taken over. The IDs may be found by logging into the PEM client, and selecting the Agent or Server and viewing the ID values on the **Properties** pane of the main window.

The agent will take over monitoring of the failed-over server within approximately 30 seconds in a standard configuration of PEM.

The flag directory used by the agent is `$TMPDIR/pem/agent-AGENTID` by default (where \$TMPDIR is as set for the user account under which the agent runs, usually `root` on Linux/Unix, or `Administrator` on Windows). The directory path can be overridden using the `AgentFlagDir` configuration option in the registry on Windows, or the `agent_flag_dir` option in the agent configuration file on other platforms.

7.2.5 PEM Agent Privileges

By default, the PEM agent is installed with `root` privileges for the operating system host and superuser privileges for the database server. These privileges allow the PEM agent to invoke unrestricted probes on the monitored host and database server about system usage, retrieving and returning the information to the PEM server.

Please note that PEM functionality diminishes as the privileges of the PEM agent decrease. For complete functionality,

the PEM agent should run as **root** and on the same host as the database server.

- If the PEM agent is run under the database server's service account, PEM probes will not have complete access to the statistical information used to generate reports, and functionality will be limited to the capabilities of that account.
- If the PEM agent is run under another lesser-privileged account, functionality will be limited even further.
- If the PEM agent is installed on a different host and is monitoring the database server remotely, then the functionality will be limited.

Feature Name	Works with root User		Works with remote PEM Agent
	Works with non-root User		
Audit Manager	yes	The Audit Log Manager may be unable to apply requested modifications if the service cannot be restarted. The user running PEM Agent may be different from the user who owns the data directory of the database server, so user running PEM Agent may not be able to change the configuration and also may not be able to restart the services of the database server.	no yes
Capacity Manager	yes	yes	NOTE: There will be no correlation between the database server and operating system metrics
Log Manager	yes	The Log Manager may be unable to apply requested modifications if the service cannot be restarted. The user running PEM Agent may be different from the user who owns the data directory of the database server, so user running the PEM Agent may not be able to change the configuration and also may not be able to restart the services of the database server.	no yes
Manage Alerts	yes	yes	NOTE: When run alert script on the database server is selected, it will run on the machine, where bound PEM Agent is running, and not on the actual database server machine.
Manage Charts	yes	yes	yes
Manage Dashboards	yes	Some dashboards may not be able to show complete data. For example, columns such as swap usage, CPU usage, IO read, and IO write will be displayed as 0 in the session activity dashboard.	Some dashboards may not be able to show complete data. For example, the operating system information of the database server will not be displayed as not available.
Manage Probes	yes	Some of the PEM probes will not return information, and some of functionalities may be affected. For details about probe functionality, see the PEM Agent Guide .	Some of the PEM probes will not return information, and some of the functionalities may be affected.

Feature Name	Works with root User	Works with non-root User	Works with remote PEM Agent
Postgres Expert	yes	The Postgres Expert will be able to access the configuration expert and schema expert, but not the security expert.	The Expert will provide partial information as operating system information is not available.
Postgres Log Analysis Expert	yes	The Postgres Log Analysis Expert may not be able to do the analysis as it is dependent on the logs imported by log manager, which will not work as required.	The Postgres Log Analysis Expert will not be able to do the analysis as it is dependent on the logs imported by log manager, which will not work as required.
Scheduled Tasks	yes	For Linux if user is the same as batch_script_user in agent.cfg then shell script will run.	Scheduled tasks will work only for database server; scripts will run on a remote Agent.
Tuning Wizard	yes	The Tuning Wizard will be unable to run if the service cannot be restarted. The user running PEM Agent may be different from the user who owns the data directory of the database server, so user running PEM Agent may not be able to change the configuration and also may not be able to restart the services of the database server.	no
System Reports	yes	yes	yes
Core Usage Reports	yes	yes	The Core Usage report will not show complete information. For example, the platform, number of cores, and total RAM will not be displayed.
Managing BART	yes	BART and the BART scanner may not be able to start/reload.	NOTE: BART requires password less authentication between two machines, where database server and BART are installed.

7.2.6 PEM Agent Configuration Parameters

A number of user-configurable parameters and registry entries control the behavior of the PEM Agent. With the exception of the PEM_MAXCONN (or pem_maxconn) parameter, we strongly recommend against modifying any of the configuration parameters or registry entries listed below without first consulting EnterpriseDB support experts.

- On 32 bit Windows systems, PEM registry entries are located in HKEY_LOCAL_MACHINE\Software\EnterpriseDB\PEM\agent
- On 64 bit Windows systems, PEM registry entries are located in

HKEY_LOCAL_MACHINE\Software\Wow6432Node\EnterpriseDB\PEM\agent

- On Linux systems, PEM configuration options are stored in the agent.cfg file, located (by default) in /usr/edb/pem/agent/etc

Parameter Name	Description	Value (if applicable)
PEM_HOST (on Windows) or pem_host (on Linux)	The IP address or hostname of the PEM server.	By default, set to 127.0.0.1.
PEM_PORT (on Windows) or pem_port (on Linux)	The database server port to which the agent connects to communicate with the PEM server.	By default, the PEM server monitors port 5432.
AgentID (on Windows) or agent_id (on Linux)	A unique identifier assigned to the PEM agent.	The first agent is assigned an identifier of '1', the second agent is assigned identifier of '2', and so on.
AgentUser (on Windows) or agent_user (on Linux)	User to connect the PEM database server	If present, and not set to empty string, it will be used to connect the PI database server.
AgentCrtPath (on Windows) or agent_ssl_crt (on Linux)	The complete path to the PEM agent's certificate file.	By default, on Windows, C:\Users\user_name\AppData\Roaming\pem/agent.crt. By default on /root/.pem/agent.crt.
AgentKeyPath (on Windows) or agent_ssl_key (on Linux)	The complete path to the PEM agent's key file.	By default, on Windows, C:\Users\user_name\AppData\Roaming\pem/agent.key. By default on /root/.pem/agent.key.
AgentFlagDir (on Windows) or agent_flag_dir (on Linux)	Used for HA support. Specifies the directory path checked for requests to take over monitoring another server. Requests are made in the form of a file in the specified flag directory.	Not set by default. This option allows you to override the hard-coded d
LogLevel (on Windows) or log_level (on Linux)	Log level specifies the type of event that will be written to the PEM log files.	Log level may be set to: error, debug1, debug2, or warning. By default, level is set to warning .
log_location (on Linux only)	Specifies the location of the PEM worker log file.	On Linux, /var/log/pem/worker.log. On Windows, Logs & errors will be reported in the Application event log.
agent_log_location (on Linux only)	Specifies the location of the PEM agent log file.	On Linux, /var/log/pem/agent.log. On Windows, Logs & errors will be reported in the Application event log.

Parameter Name	Description	Value (if applicable)
ShortWait (on Windows) or short_wait (on Linux)	The minimum length of time (in seconds) that the PEM agent will wait before checking which probes are next in the queue (waiting to run).	By default, 10 seconds.
LongWait (on Windows) or long_wait (on Linux)	The maximum length of time (in seconds) that the PEM agent will wait before attempting to connect to the PEM server if an initial connection attempt fails.	By default, 30 seconds.
AlertThreads (on Windows) or alert_threads (on Linux)	The number of alert threads to be spawned by the agent.	Set to 1 for the agent that resides on the host of the PEM server; should be 0 for all other agents.
EnableSMTP (on Windows) or enable_smtp (on Linux)	When set to true for multiple PEM Agents (7.13 or lesser) and PEM backend database (9.4 or lesser) then it may send more duplicate emails. Whereas for PEM Agents (7.14 or higher) and PEM backend database (9.5 or higher) then it may send lesser duplicate emails.	By default, set to true for the agent that resides on the host of the PEM false for all other agents.
EnableSNMP (on Windows) or enable_snmp (on Linux)	When set to true for multiple PEM Agents (7.13 or lesser) and PEM backend database (9.4 or lesser) then it may send more duplicate traps. Whereas for PEM Agents (7.14 or higher) and PEM backend database (9.5 or higher) then it may send lesser duplicate traps.	By default, set to true for the agent that resides on the host of the PEM false for all other agents.

Parameter Name	Description	Value (if applicable)
enable_nagios (on Linux only)	When set to true, Nagios alerting is enabled.	By default, set to true for the agent that resides on the host of the PEM false for all other agents.
EnableWebhook (on Windows) or enable_webhook (on Linux)	When set to true, Webhook alerting is enabled.	By default, set to true for the agent that resides on the host of the PEM false for all other agents.
MaxWebhookRetries (on Windows) or max_webhook_retries (on Linux)	Set maximum number of times pemAgent should retry to call webhooks on failure.	Default 3.
ConnectTimeout (on Windows) or connect_timeout (on Linux)	The maximum length of time (in seconds, written as a decimal integer string) that the agent will wait for a connection.	Not set by default. If set to 0, the agent will wait indefinitely.
AllowServerRestart (on Windows) or allow_server_restart (on Linux)	If set to TRUE, the agent can restart the database server that it monitors. Some PEM features may be enabled/disabled, depending on the value of this parameter.	By default, set to TRUE.
MaxConnections (on Windows) or max_connections (on Linux)	The maximum number of probe connections used by the connection throttler.	By default, set to 0 (an unlimited number of connections).
ConnectionLifetime (on Windows) or connection_lifetime (on Linux)	Use ConnectionLifetime (or connection_lifetime) to specify the minimum number of seconds an open but idle connection is retained. This parameter is ignored if the value specified in MaxConnections is reached and a new connection (to a different database) is required to satisfy a waiting request.	By default, set to 0 (a connection is dropped when the connection is idle the agent's processing loop completes a cycle in which the connector has not been used).

Parameter Name	Description	Value (if applicable)
HeartbeatConnection (on Windows) or heartbeat_connection (on Linux)	When set to TRUE, a dedicated connection used for sending the heartbeats.	By default, set to FALSE.
AllowBatchProbes (on Windows) or allow_batch_probes (on Linux)	If set to TRUE, the user will be able to create batch probes using custom probes feature.	By default, set to FALSE.
BatchScriptDir (on Windows) or batch_script_dir (on Linux)	Provide the path where script file (for alerting) will be stored.	On Windows, C:\Users\user_name\AppData\Local\Temp. On Linux, set to ,
AllowBatchJobSteps (on Windows) or batch_script_user	Provide the username who will run the script.	On Windows, set to TRUE and restart PEM Agent. Entries located in HKEY_LOCAL_MACHINE\Software\Wow6432Node\EnterpriseDB\PEM. On Linux, Restart the agent after modifying the file. If you do not specify user, or the specified user does not exist, then the script will not be ex
ConnectionCustomSetup (on Windows) or connection_custom_setup (on Linux)	Use this parameter to provide SQL code that will be invoked each time a new connection with the monitored server is established.	By default, no value is provided.
ca_file (Linux only)	Provide the path where the CA certificate resides.	By default, /opt/PEM/agent/share/certs/ca-bundle.crt
WebhookSSLKey (on Windows) or webhook_ssl_key (on Linux)	The complete path to the webhook's SSL client key file.	
WebhookSSLCrt (on Windows) or webhook_ssl_crt (on Linux)	The complete path to the webhook's SSL client certificate file.	
WebhookSSLCaCrt (on Windows) or webhook_ssl_ca_crt (on Linux)	The complete path to the webhook's SSL ca certificate file.	
WebhookSSLcrl (on Windows) or webhook_ssl_crl (on Linux)	The complete path of the CRL file to validate webhook server certificate.	
AllowInsecureWebhooks (on Windows) or allow_insecure_webhooks (on Linux)	When set to true, allow webhooks to call with insecure flag.	false

Note

If you add or remove any of the parameter in the `agent.cfg` file then agent must be restarted to apply the changes.

7.2.7 PEM Agent Self Registration

Each PEM agent must be `registered` with the PEM server. The registration process provides the PEM server with the information it needs to communicate with the agent. The PEM agent graphical installer supports agent self-registration, but you can use the `pemworker` utility to register the agent if you skip PEM agent registration during a graphical installation or use an RPM package to install a PEM agent.

The RPM installer places the PEM worker utility in the `/usr/edb/pem/agent/bin` directory. Use the following commands to register an agent:

- On Linux: `pemworker --register-agent [register-options]`
- On Windows: `pemworker.exe REGISTER [register-options]`

The following information is required when registering an agent with the PEM Server; you will be prompted for information if it is not provided on the command line:

Parameters	Command-line options	Optional	Description	Default Value
PEM Database Server Host	<code>--pem-server <hostname/address></code>	No	Address/Host name of the PEM database server	
PEM Admin User	<code>--pem-user <username></code>	No	<code>PEM Admin User</code> to connect to the PEM database server.	
PEM Database Server Port	<code>--pem-port <port number></code>	Yes	Port on which PEM database server is running.	5432
Agent Certificate Path	<code>--cert-path <certificate path></code>	Yes	Path, where certificates need to be created.	On Linux, " <code>~/.pem</code> " On Windows, " <code>%APPDATA%/pem</code> "
Agent Display Name	<code>--display-name <agent_name></code>	Yes	Display name of the PEM Agent.	System hostname
Agent Group	<code>--group <group_name></code>	Yes	The name of the group in which the agent will be displayed.	
Agent Team	<code>--team <team_name></code>	Yes	The name of the group role that may access the PEM Agent.	
Agent Owner	<code>--owner <owner_name></code>	Yes	The name of the owner of the PEM Agent.	
Force registration	<code>--force-registration</code>	Yes	Forcefully registers the agent to the PEM server with the arguments provided. It can be used to override the existing agent configuration.	
Enable Heartbeat	<code>--enable-heartbeat-connection</code>	Yes	Agent to use dedicated connection to update the heartbeat.	false

Parameters	Command-line options	Optional	Description	Default Value
Agent User	--pem-agent-user	Yes	<p>Use this user to connect the PEM database server. Specify it when you would like to use a connection pooler between PEM Agent and PEM database server. It will generate the SSL Certificates, which will be used by the pemworker to connect to the PEM database server instead, for this user instead of the default agent user.</p> <p>NOTE: Specified user must be a member of 'pem_agent' role.</p>	

!!! Note You can use the `PEM_SERVER_PASSWORD` environment variable to set the password of the `PEM Admin User`. If the `PEM_SERVER_PASSWORD` is not set, the server will use the `PGPASSWORD` or `pgpass` file when connecting to the PEM Database Server.

Example:

```
root@localhost:/usr/edb/pem/agent/etc - □ ×
File Edit View Search Terminal Help
[root@localhost yum.repos.d]# cd /usr/edb/pem/agent/bin
[root@localhost bin]# ls
pemagent pemworker pkgLauncher
[root@localhost bin]# export PEM_SERVER_PASSWORD=postgres
[root@localhost bin]# ./pemworker --register-agent
Postgres Enterprise Manager Server Hostname: 192.168.1.16
Postgres Enterprise Manager Server Username: postgres
Postgres Enterprise Manager Server Port: 5432
Postgres Enterprise Manager Agent registered successfully!
```

Refer the [PEM Worker Usage Guide](#).

7.2.8 Register/Unregister database server using PEM Agent

You can use the `pemworker` utility to register a database server for monitoring. The RPM installer places the utility in the `/usr/edb/pem/agent/bin` directory. Use the following commands to register a server:

- On Linux: `pemworker --register-server [register-server-options]`
- On Windows: `pemworker.exe REGISTER-SERVER [register-server-options]`

Use the parameters in the table that follow to provide connection information for a Postgres database server that you wish to register for monitoring by the PEM Server. Please note that the `pg_hba.conf` file on the database server must be configured to allow connections from the PEM server.

Properties that begin with `--asb` (agent server binding) define the binding for an agent that does not reside on the same host as the database server. These properties are optional if you have a PEM agent installed on the host of the database server. You will be prompted for required information if you do not include it on the command line.

Parameters	Command-line options	Optional	Description	Default Value
PEM Admin User	--pem-user <username>	No	The name of the <code>PEM Admin User</code> that will connect to the monitored server.	

Parameters	Command-line options	Optional	Description	Default Value
Server Host	--server-addr <host name/address>	No	Host name/address of the monitored server.	
Server Port	--server-port <port>	No	Port on which database server is running.	
Server Database	--server-database <name>	No	The database to which PEM will connect.	
Server User	--server-user <name>	No	The database user role that will be used by the agent for monitoring purposes.	
Server Service Name	--server-service-name <name>	Yes	Name of the system level service, which controls the operations like start, stop, restart, reload, etc. of the server.	
Remote Monitoring?	--remote-monitoring <yes/no>	No	<code>no</code> if the monitored server resides on the same machine as the bound PEM agent, <code>yes</code> if the agent is on another host.	no
- EDB Failover Manager Cluster Name	--efm-cluster-name <name>	Yes	Name of EDB Failover Manager Cluster associated with this server.	
EDB Failover manager Installation Path	--efm-install-path <path>	Yes	Installation path of EDB Failover Manager associated with this server.	
Server Display Name	--display-name <server_name>	Yes	Display name of the registered server.	System hostname
Host Name	--asb-host-name <name_of_host>	Yes	The name of the host to which the agent is connecting.	The value specified by the --server-addr property.
Host Port	--asb-host-port <port_number>	Yes	The port number that the agent will use when connecting to the database.	The value specified by the --server-port property.
Host DB	--asb-host-db <database_name>	Yes	The name of the database to which the agent will connect.	The value specified by the --server-database property.
Host User Name	--asb-host-user <database_user>	Yes	The database user name that the agent will supply when authenticating with the database.	The value specified by the --server-user property.
SSL Mode	--asb-ssl-mode <certificate path>	Yes	Type of SSL authentication that will be used for connections. Supported values include <code>prefer</code> , <code>require</code> , <code>disable</code> , <code>verify-CA</code> , <code>verify-full</code> .	prefer
Server Group	--group <group_name>	Yes	Specify the name of the server group in which the server will be displayed.	

Parameters	Command-line options	Optional	Description	Default Value
Server Team	--team <team_name>	Yes	Specify the name of the group role that will be allowed to access the server.	
Owner	--owner <owner_name>	Yes	Specify the name of the role that will own the monitored server.	

Use the PEM_MONITORED_SERVER_PASSWORD environment variable to set the password of the user of the database server which is to be registered. When registering the database server, the pemworker utility will bind the server to the **PEM Agent** associated with the pemworker utility. The PEM server will use the specified user name (**Server User**) and password specified in the PEM_MONITORED_SERVER_PASSWORD environment variable when monitoring the database server.

Use the PEM_SERVER_PASSWORD environment variable to provide the password of the user of the PEM database server. If the **PEM_SERVER_PASSWORD** is not set, the server will use the **PGPASSWORD** or **pgpass file** when connecting to the PEM Database Server.

To unregister a database server

You can use the pemworker utility to unregister a server:

- On Linux: pemworker --unregister-server [unregister-server-options]
- On Windows: pemworker.exe UNREGISTER-SERVER [unregister-server-options]

Include the following information when unregistering a database server from the **PEM Server**; you will be prompted for required information if you do not include it on the command line:

Parameters	Command-line options	Optional	Description	Default Value
PEM Admin User	--pem-user <username>	No	PEM Admin User to connect the PEM database server.	
Server Host	--server-addr <host name/address>	No	Host name/address of the database server.	
Server Port	--server-port <port>	No	Port on which database server is running.	

The command will unregister the server from the **PEM Server** for the specified combination of **Server Host** and **Server Port**, which is being monitored by the **PEM Agent**.

For more information, refer the [PEM Worker Usage Guide](#).

7.3 The PEM Client

The Postgres Enterprise Manager client provides a powerful and intuitive user interface that you can use to manage Advanced Server and PostgreSQL databases. The client interface is easily customized, and will preserve your preferences between sessions. Client features include:

- auto-detection and support for objects discovered at run-time
- a live SQL query tool with direct data editing
- support for administrative queries
- a syntax-highlighting SQL editor
- powerful graphical management dialogs and tools for common tasks
- a responsive, context-sensitive behavior
- supportive error messages
- helpful hints
- online help and information for dialogs and tools.

The PEM client features a highly-customizable display that features drag-and-drop panels that you can arrange to make the best use of your desktop environment. The application is installed during the PEM server installation; use your browser of choice to connect to the client.

The client tree control (the *Browser*) provides an elegant overview of the managed servers, and the objects that reside on each server. Right-click on a node within the tree control to access context-sensitive menus that provide quick access to management tasks for the selected object. The tabbed browser window provide quick access to statistical information about each object in the tree control, tools and utilities, and extended PEM features. The client opens an additional feature tab each time you access the extended functionality offered by PEM; you can open, close, and re-arrange tabs as needed.

You can search for objects in the database using the [Search objects](#)

Contents:

7.3.1 PEM Main Browser Window

The PEM client features a menu bar and a window divided into two panes: the **Browser** tree control in the left pane, and a tabbed browser in the right pane.



[Menus](#) displayed across the top of the browser window provide quick, context-sensitive access to PEM features and functionality.

The PEM Client Object Browser

The **Browser** tree control provides access to information and management options for the database objects that reside on each server. The tree control expands to display a hierarchical view of the servers and objects that are monitored by the PEM server. You can use context menu options (accessed by right-clicking on nodes of the tree control) to create new objects, and modify and delete existing objects if your role holds the required privileges.

Expand nodes in the tree control to display a hierarchical view of the database objects that reside on a selected server:

- Use the plus sign (+) to the left of a node to expand a segment of the tree control.
- Click the minus sign (-) to the left of a node to close that node.

Right-click on a node of the tree control to access a context-sensitive menu and perform common tasks. Context menu options may include one or more of the following selections:

Option	Action
Add named restore point	Click to create and enter the name of a restore point.
Backup...	Click to open the Backup... dialog to backup database objects.
Backup Globals...	Click to open the Backup Globals... dialog to backup cluster objects.
Backup Server...	Click to open the Backup Server... dialog to backup a server.
Connect Server	Click to establish a connection with the selected server.
Create	Click to access a context menu that provides context-sensitive selections. Your selection opens a Create dialog for creating a new object.
CREATE Script	Click to open the Query tool to edit or view the CREATE script.
CREATE Script	Click to open the Query tool to edit or view the CREATE script.
Dashboards	Click through for quick access to PEM dashboards.
Delete/Drop	Click to delete the currently selected object from the server.
Disconnect Database...	Click to terminate a database connection.
Disconnect Server...	Click to refresh the currently selected object.
Drop Cascade	Click to delete the currently selected object and all dependent objects from the server.
Debugging	Click to access the Debugger tool.
Grant Wizard	Click to access the Grant Wizard tool.
Maintenance...	Click to open the Maintenance... dialog to VACUUM, ANALYZE, REINDEX, or CLUSTER.
Management	Click to access management tasks that are relevant to the node.
Properties...	Click to review or modify the currently selected object's properties.
Refresh...	Click to refresh the currently selected object.
Reload Configuration...	Click to update configuration files without restarting the server.
Restore...	Click to access the Restore dialog to restore database files from a backup.
View Data	Use the View Data option to access the data stored in a selected table with the Data Output tab of the Query Tool .

The context-sensitive menus associated with [Tables](#) and nested [Table](#) nodes provides additional display options:

Option	Action
--------	--------

Option	Action
Import/Export...	Click open the Import/Export... <import_export_data> dialog to import data to or export data from the selected table.
Reset Statistics	Click to reset statistics for the selected table.
Scripts	Click to open the Query tool to edit or view the selected script from the flyout menu.
Truncate	Click to remove all rows from a table.
Truncate Cascade	Click to remove all rows from a table and its child tables.
View First 100 Rows	Click to access the data grid that displays the first 100 rows of the selected table.
View Last 100 Rows	Click to access the data grid that displays the last 100 rows of the selected table.
View All Rows	Click to access the data grid that displays all rows of the selected table.
View Filtered Rows...	Click to access the Data Filter popup to apply a filter to a set of data.

The PEM Tabbed Browser Window

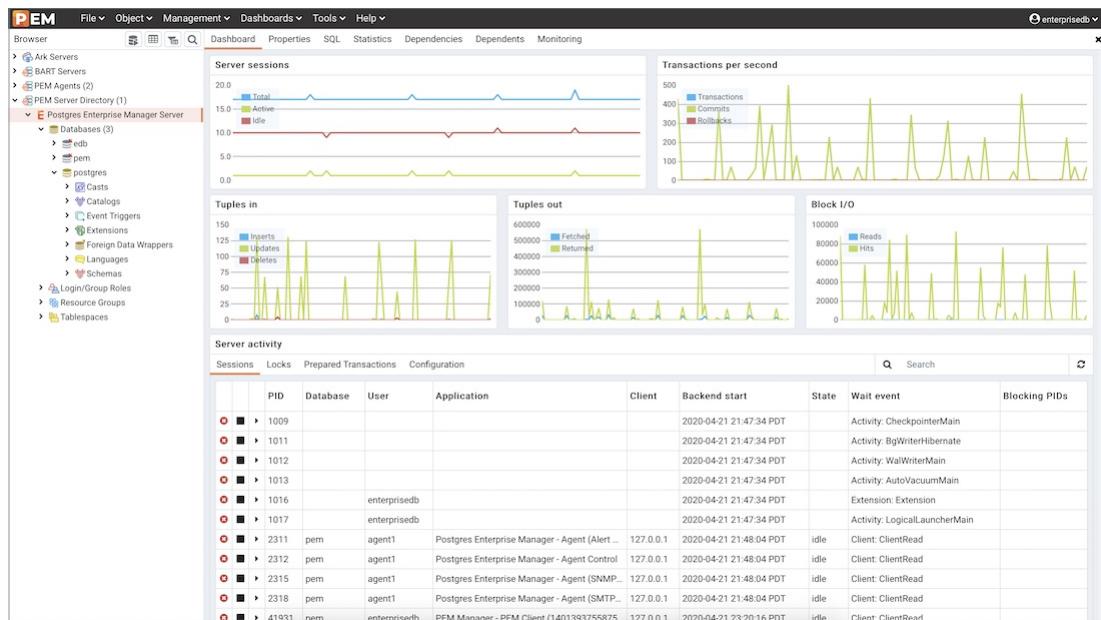
The main panel of the PEM client contains a collection of tabs that display information about the object currently selected in the tree control.



The [Dashboard](#) tab is context-sensitive; when you navigate to the [Dashboard](#) tab from a server group or the [PEM Agents](#) node, the EDB Postgres [Welcome](#) window opens, allowing you to:

- Click the [Add New Server](#) icon to open the [Create - Server dialog](#) to define a connection to a server.
- Click the [Configure PEM](#) icon to open the [Server Configuration dialog](#) and modify server parameters.
- Click the [Getting Started](#) icon to open a new tab, displaying the PEM Getting Started Guide at the EnterpriseDB website.
- Click the [EDB Website](#) icon to navigate to the home page of the EnterpriseDB website. The EnterpriseDB website features news about upcoming events and other projects.
- Click the [PostgreSQL Website](#) icon to navigate to the PostgreSQL project website. The PostgreSQL site features news about recent releases and other project information.
- Click the [EDB Blogs](#) icon to navigate to the EDB Blog page, where you can review the most-recent employee posts to Postgres related blogs.

Highlight the name of an agent or server and navigate to the **Dashboard** tab to review session or server activity for the currently selected object.



When opened from the name of an agent or server, the **Dashboard** tab provides a graphical analysis of usage statistics:

- The **Server sessions** or **Database sessions** graph displays the interactions with the server or database.
- The **Transactions per second** graph displays the commits, rollbacks, and total transactions per second that are taking place on the server or database.
- The **Tuples In** graph displays the number of tuples inserted, updated, and deleted on the server or database.
- The **Tuples out** graph displays the number of tuples fetched and returned from the server or database.
- The **Block I/O** graph displays the number of blocks read from the filesystem or fetched from the buffer cache (but not the operating system's file system cache) for the server or database.
- The **Server activity** tabbed panel displays tables that contain session information, session locks, prepared transactions and configuration.



Navigate to the **Properties** tab to review the properties of the item currently highlighted in the tree control.

```

1 -- Database: postgres
2
3 -- DROP DATABASE postgres;
4
5 CREATE DATABASE postgres
6   WITH
7     OWNER = enterpriseedb
8     ENCODING = 'UTF8'
9     LC_COLLATE = 'en_US.UTF-8'
10    LC_CTYPE = 'en_US.UTF-8'
11    TABLESPACE = pg_default
12    CONNECTION LIMIT = -1;
13
14 COMMENT ON DATABASE postgres
15 IS 'default administrative connection database';

```

The **SQL** tab displays the SQL code used to generate the object currently selected in the Browser tree control.

PID	User	Database	Backend start	Client	Application	Wait event type	Wait event name	Query	Query
1017	enterpriseedb		2020-04-21 21:47:34.635953+07	localhost:		Activity	LogicalLauncherMain		2020
110973	enterpriseedb	pem	2020-04-22 02:23:06.479197+07	127.0.0.1/32-34370	PEM Client - DB.pem	Client	ClientRead	SELECT rl.* , r.rולename AS user_name,...	2020
41931	enterpriseedb	pem	2020-04-21 23:20:16.310287+07	127.0.0.1/32-51298	PEM Manager - PEM Client (1401393..	Client	ClientRead	WITH chart.cfg AS (SELECT c.id AS ...	2020
121257	enterpriseedb	postgres	2020-04-22 02:35:33.532905+07	127.0.0.1/32-52688	Postgres Enterprise Manager - Agent..	Client	ClientRead	SELECT 1	2020
2318	agent1	pem	2020-04-21 21:48:04.543596+07	127.0.0.1/32-40248	Postgres Enterprise Manager - Agent..	Client	ClientRead	SELECT id FROM pem.smtp_spool_w...	2020
1012			2020-04-21 21:47:34.632452+07	localhost:		Activity	WalWriterMain		
1013			2020-04-21 21:47:34.632621+07	localhost:		Activity	AutoVacuumMain		
1016	enterpriseedb		2020-04-21 21:47:34.632849+07	localhost:		Extension			
1011			2020-04-21 21:47:34.629553+07	localhost:		Activity	BgWriterHibernat...		
2315	agent1	pem	2020-04-21 21:48:04.534893+07	127.0.0.1/32-40246	Postgres Enterprise Manager - Agent..	Client	ClientRead	SELECT id FROM pem.smtp_spool ...	2020
110861	enterpriseedb	postgres	2020-04-22 02:23:01.783554+07	127.0.0.1/32-34192	PEM Client - DB.postgres			SELECT pid AS "PID", username AS "U..."	2020
41934	enterpriseedb	pem	2020-04-21 23:20:16.419199+07	127.0.0.1/32-51300	PEM Manager - PEM Client (1401393..	Client	ClientRead	WITH chart.cfg AS (SELECT c.id AS ...	2020
70133	enterpriseedb	postgres	2020-04-22 00:50:44.407853+07	127.0.0.1/32-42718	Postgres Enterprise Manager - Agent..	Client	ClientRead	SELECT 1	2020
2311	agent1	pem	2020-04-21 21:48:04.507554+07	127.0.0.1/32-40242	Postgres Enterprise Manager - Agent..	Client	ClientRead	SELECT pem.process_one_alert()	2020
41945	enterpriseedb	pem	2020-04-21 23:20:16.592567+07	127.0.0.1/32-51302	PEM Manager - PEM Client (1401393..	Client	ClientRead	WITH chart.cfg AS (SELECT c.id AS ...	2020
1009			2020-04-21 21:47:34.63545+07	localhost:		Activity	CheckpointerMain		
2312	agent1	pem	2020-04-21 21:48:04.527354+07	127.0.0.1/32-40244	Postgres Enterprise Manager - Agent..	Client	ClientRead	UPDATE pem.probe_schedule SET cu...	2020
70131	enterpriseedb	postgres	2020-04-22 00:50:43.478092+07	127.0.0.1/32-42714	Postgres Enterprise Manager - Agent..	Client	ClientRead	SELECT 1	2020

The **Statistics** tab displays the statistics gathered for each object on the tree control; the statistics displayed in the table vary by the type of object that is highlighted. Click a column heading to sort the table by the data displayed in the column; click again to reverse the sort order. The following table lists some of the statistics that may be displayed:

Panel	Description
PID	The process ID associated with the row.
User	The name of the user that owns the object.
Database	displays the database name.
Backends	displays the number of current connections to the database.
Backend start	The start time of the backend process.
Xact Committed	displays the number of transactions committed to the database within the last week.
Xact Rolled Back	displays the number of transactions rolled back within the last week.
Blocks Read	displays the number of blocks read from memory (in megabytes) within the last week.
Blocks Hit	displays the number of blocks hit in the cache (in megabytes) within the last week.
Tuples Returned	displays the number of tuples returned within the last week.
Tuples Fetched	displays the number of tuples fetched within the last week.
Tuples Inserted	displays the number of tuples inserted into the database within the last week.
Tuples Updated	displays the number of tuples updated in the database within the last week.
Tuples Deleted	displays the number of tuples deleted from the database within the last week.
Last statistics reset	displays the time of the last statistics reset for the database.
Tablespace conflicts	displays the number of queries canceled because of recovery conflict with dropped tablespaces in database.

Panel	Description
Lock conflicts	displays the number of queries canceled because of recovery conflict with locks in database.
Snapshot conflicts	displays the number of queries canceled because of recovery conflict with old snapshots in database.
Bufferpin conflicts	displays the number of queries canceled because of recovery conflict with pinned buffers in database.
Temporary files	displays the total number of temporary files, including those used by the statistics collector.
Size of temporary files	displays the size of the temporary files.
Deadlocks	displays the number of queries canceled because of a recovery conflict with deadlocks in database.
Block read time	displays the number of milliseconds required to read the blocks read.
Block write time	displays the number of milliseconds required to write the blocks read.
Size	displays the size (in megabytes) of the selected database.

Type	Name	Restriction
Column	perm.agent_config_id	auto
Schema	perm	normal
Role	perm_admin	ACL
Role	perm_agent	ACL
Role	perm_user	ACL

The **Dependencies** tab displays the objects on which the currently selected object depends. To ensure the integrity of the database structure, the server makes sure that you do not accidentally drop objects that other objects depend on; you must use **DROP CASCADE** to remove an object on which another object depends.

The **Dependencies** table displays:

- The **Type** field specifies the parent object type.
- The **Name** field specifies the identifying name of the parent object.
- The **Restriction** field describes the dependency relationship between the currently selected object and the parent.

Type	Name	Restriction
Sequence	pem.agent_config_id_seq	auto
Function	nextval(pem.agent_config_id_seq::regclass)	auto
Foreign Key	pem.agent_config.agent_config_agent_id_fkey	auto
Primary Key	pem.agent_config.pkey	auto

The **Dependents** tab displays a table of objects that depend on the object currently selected in the **pgAdmin** browser. A dependent object can be dropped without affecting the object currently selected in the **pgAdmin** tree control.

- The **Type** field specifies the dependent object type.
- The **Name** field specifies the identifying name for the dependent object.
- The **Restriction** field describes the dependency relationship between the currently selected object and the parent.
- Navigate to the **Monitoring** tab to access information presented on **PEM dashboards**. Dashboards display statistical information about the objects monitored by the PEM server.



PEM will open additional tabs when you access PEM functionality through the **Management** or **Tools** dialogs. Right-click the current tab and select from a context menu that allows you to customize the display for your working style:

- Click **Remove Panel** to remove the currently selected panel.
- Click **Float Panel** to detach the currently selected panel, repositioning it for convenience.
- Click **Add Panel** and select from the context menu to display the pgAdmin or PostgreSQL project website.

The PEM client will preserve any adjustments when you exit the program; to reset the PEM client to its original format, select **Reset Layout** from the **File** menu.

Using Chart, Graph and Table Controls

Use the icons in the upper-right corner of each graphic on a PEM Client dashboard to control, download, and customize the charts, graphs and tables displayed in the PEM client.



Use the **Refresh** icon to display the most-recent content available from the PEM probes.

Select the **Download** icon to download a .jpeg or .png image of the chart or graph. By default, the file will be in .jpeg format; to save the file as a .png, use the **Personalize** icon to modify the download format.

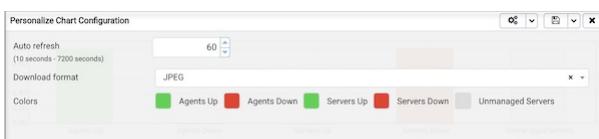
Select the **Fullscreen** icon to expand the chart or graph to fill the main pane of the PEM client.

Select the **Personalize** icon to modify the display properties of the chart or graph for your session only.

Use the **Information** icon to access information about the chart or graph.

Personalizing a Graphic

When you select the **Personalize** icon, the **Personalize chart configuration** dialog opens:



Use controls on the **Personalize chart configuration** dialog to modify the properties of the graphic:

- Use the **Auto Refresh** control to increase or decrease the number of seconds between refreshes.
- Use the **Auto Refresh** field to specify the number of seconds between updates of the data displayed in the table or chart.
- If applicable, use the **Download as** field to indicate if you would like a chart to be downloaded as a JPEG image or a PNG image.
- If applicable, use the **Colours** selectors to specify the display colors that will be used on a chart.
- If applicable, set the **Show Acknowledged Alerts** switch to **Yes** indicate that you would like the table to display alerts that you have acknowledged with a checkbox in the **Ack'ed** column. Set the field to **No** to indicate that the table should hide any acknowledged alerts. The switch acts as a toggle; acknowledged alerts are not purged from the table content until the time specified in the alert definition passes.

After personalizing the display properties, use the controls in the upper-right hand corner to apply your changes:

- Use the **Delete** icon to reset the properties of the graphic to their default settings; use the drop-down listbox to access a menu that allows you to apply the change to only this instance of the graphic, or to the same graphic when displayed on other dashboards.
- Use the **Save** icon to save your changes to the properties for the graphic; use the drop-down listbox to access a menu that allows you to apply the change to only this instance of the graphic, or to the same graphic when displayed on other dashboards.
- Click the X to close the dialog without changing the properties of the graphic.

7.3.2 Browser Toolbar

The browser toolbar provides shortcut buttons for frequently used features like View Data and the Query Tool which are most frequently used in PEM. This toolbar is visible on the Browser panel. Buttons get enabled/disabled based on the selected browser node.



- Use the [Query Tool](#) button to open the Query Tool in the current database context.
- Use the [View Data](#) button to view/edit the data stored in a selected table.
- Use the [Filtered Rows](#) button to access the Data Filter popup to apply a filter to a set of data for viewing/editing.

7.3.3 The PEM Menu Bar

The PEM menu bar provides access to commands and features that you can use to manage your database servers and the objects that reside on those servers. If an option is disabled:

- The database server to which you are currently connected may not support the selected feature.
- The selected menu option may not be valid for the current object (by design).
- The role that you have used to connect to the server may have insufficient privileges to change the selected object.

Context-sensitive menus across the top of the PEM web interface allow you to customize your environment and provide access to the enterprise management features of PEM.

The File Menu



Use the **File** menu to access the following options:

Menu Option	Action
Preferences	Click to open the Preferences dialog to customize your PEM client settings.
Lock Layout	Click to open a sub-menu to select the level for locking the UI layout. This can also be changed from the Browser -> Display settings tab preferences .
Server Configuration	Click to open the Server Configuration dialog and update your PEM server configuration settings.
Reset Layout	If a workspace panel is popped out by mistake or intentionally it can be reset back to default using Reset Layout.

The Object Menu



The **Object** menu is context-sensitive. Use the **Object** menu to access the following options:

Menu Option	Action
Create	Click Create to access a context menu that provides context-sensitive selections. Your selection opens a Create dialog for creating a new object.
Refresh...	Click to refresh the currently selected object.
Connect Server	Click to open the Connect to Server dialog to establish a connection with a server.
CREATE Script	Click to open the Query tool to edit or view the selected script.
Disconnect Server/Database	Click to disconnect the selected server.
Remove Server	Click to remove the selected server from the browser tree.
BART	Click to access a context menu that provides options for removing BART configuration, taking a BART backup, or revalidate the BART configuration.
Clear Saved Password	If you have saved the database server password, click to clear the saved password. Enabled only after password is saved.
Clear SSH Tunnel Password	If you have saved the ssh tunnel password, click to clear the saved password. Enabled only after password is saved.
Drop Cascade	Click to delete the currently selected object and all dependent objects from the server.
Properties...	Click to review or modify the currently selected object's properties
Delete/Drop	Click to delete the currently selected object from the server.

Menu Option	Action
Connect Database	Click to connect to selected database.
Trigger(s)	Click to Disable or Enable trigger(s) for the currently selected table.
Truncate	Click to remove all rows from a table (Truncate) or to remove all rows from a table and its child tables (Truncate Cascade).
View/Edit Data	Click to access a context menu that provides several options (All Rows, First 100 Rows, Last 100 Rows, Filtered Rows) for viewing data.
Count Rows	Click to count the number of rows of the selected table.
Reset Statistics	Click to reset the statistics of the selected table.
Scripts	Click to CREATE, DELETE, INSERT, SELECT and UPDATE script for the selected table.

The Management Menu



Use the **Management** menu to access the following PEM features:

Menu Option	Action
Audit Manager...	Click to open the Audit Manager and configure auditing on your monitored servers.
Auto Discovery...	Click to open the Auto Discovery dialog to instruct a PEM agent to locate and bind monitored database servers.
Capacity Manager...	Click to open the Capacity Manager dialog and analyze historical or project future resource usage.
Log Manager...	Click to open the Log Manager dialog and configure log collection for a server.
Manage Alerts...	Click to access the Manage Alerts tab and create or modify alerting behavior.
Manage Charts...	Click to open the Manage Charts tab to create or modify PEM charts.
Manage Dashboards...	Click to open the Manage Dashboards dialog to VACUUM, ANALYZE, REINDEX, or CLUSTER.
Manage Probes...	Click to open the Manage Probes dialog to VACUUM, ANALYZE, REINDEX, or CLUSTER.
Postgres Expert...	Click to open the Postgres Expert wizard and perform a static analysis of your servers and databases.
Postgres Log Analysis Expert...	Click to access the Postgres Log Analysis Expert dialog analyze log file contents for usage trends.
Scheduled Tasks...	Click to open the Scheduled Tasks tab and review tasks that are pending or recently completed.
Schedule Alert Blackout...	Click to open the Schedule Alert Blackout dialog and schedule the alerts blackout for your servers and agents.
Tuning Wizard...	Click to open the Tuning Wizard dialog to generate a set of tuning recommendations for your server.
Reports	Click to open the Reports dialog to generate the system configuration report and core usage report for your server.

The Dashboards Menu



The **Dashboards** menu is context-sensitive; use the **Dashboards** menu to access the following options:

Menu Option	Action
Alerts	Click to open the Alerts Dashboard for the selected node.
Audit Log	Click to open the Audit Log Analysis Dashboard for the selected node.
Database Server	Click to open the Database Analysis Dashboard for the selected node.
Memory	Click to open the Memory Analysis Dashboard for the selected node
Server Log	Click to open the Server Log Analysis Dashboard for the selected node.
Session Activity	Click to open the Session Activity Analysis Dashboard for the selected node.
Storage	Click to open the Storage Analysis Dashboard for the selected node.
Streaming Replication	Click to open the Streaming Replication Analysis Dashboard for the selected node.
System Wait	Click to open the System Wait Analysis Dashboard for the selected node.
I/O Analysis	Click to open the I/O Analysis Dashboard for the selected node.
Object Activity	Click to open the Object Activity Analysis Dashboard for the selected node.
Session Waits	Click to open the Session Waits Analysis DASHBOARD for the selected node.
Operating System	Click to open the Operating System Analysis Dashboard for the selected node.
Probe Log	Click to open the Probe Log Analysis Dashboard for the selected node.
Custom Dashboards	Click to open the Custom Dashboards. It will list custom dashboards configured by the user.

The Tools Menu

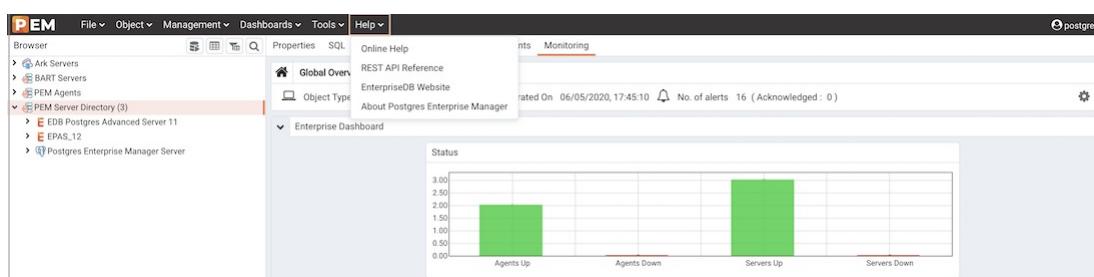


Use the **Tools** menu to access the following options:

Menu Option	Action
Schema Diff	Click to open the Schema Diff dialog to compare the schema objects between two database schemas.
Search objects	Click to open the Search Objects dialog to search the database objects within a database.

Menu Option	Action
Server	Click to access the various server related tools such as Add Named Restore Point, Performance Diagnostics, Queue Server Startup, Queue Server Shutdown, Replace Cluster Primary, Switchover EFM Cluster and SQL Profiler.
Query Tool	Click to open the Query tool for the currently selected object.
Storage Manager	Click to open the Storage manager to upload, delete or download the backup files.
Reload Configuration	Click to update configuration files without restarting the server.
Pause replay of WAL	Click to pause the replay of the WAL log.
Resume replay of WAL	Click to resume the replay of the WAL log.
Import/Export...	Click to open the Import/Export data... dialog to import or export data from a table.
Maintenance...	Click to open the Maintenance... dialog to VACUUM, ANALYZE, REINDEX, or CLUSTER.
Backup...	Click to open the Backup... dialog to backup database objects.
Backup Globals...	Click to open the Backup Globals... dialog to backup cluster objects.
Backup Server...	Click to open the Backup Server... dialog to backup a server.
Restore...	Click to access the Restore dialog to restore database files from a backup.
Grant Wizard...	Click to access the Grant Wizard tool.
Schedule Backup	Click to access the Schedule Backup dialog for BART backups.

The Help Menu



Use the options on the [Help](#) menu to access online help documents or to review information about the PEM installation:

Menu Option	Action
Online Help	Click to open documentation for Postgres Enterprise Manager.
REST API Reference	Click to open the REST API Reference.
EnterpriseDB Website	Click to open the EnterpriseDB website in a browser window.
About Postgres Enterprise Manager	Click to locate versioning and user information for Postgres Enterprise Manager.

7.3.4 PEM Client Preferences

Use options on the **Preferences** dialog to customize the behavior of the PEM web interface. To open the dialog, select **Preferences** from the **File** menu. The left pane of the **Preferences** dialog displays a tree control; each node of the tree control provides access to options that are related to the node under which they are displayed.

- Use the plus sign (+) to the left of a node name to expand a segment of the tree control.
- Use the minus sign (-) to the left of a node name to close that node.

The BART Servers Node

Use the **Nodes** panel to select the BART Servers that will be displayed in the Browser tree control of BART Servers:



- Slide the switch located next to BART Servers to show or hide the BART Servers in the browser tree.

The Browser Node

Use the fields on the **Browser** node of the tree control to personalize your workspace.



Use the fields on the **Display** panel to specify general display preferences:

- When the **Enable browser tree animation?** switch is set to **True**, the client will display the animated tree control; if the switch is **False**, the tree control will be unanimated.
- When the **Auto-expand sole children** switch is set to **True**, child nodes will be automatically expanded if a treeview node is expanded and has only a single child.

- Use the **Browser tree state saving interval** field to set the treeview state saving interval. A value of **-1** will disable the treeview state saving functionality.
- When the **Confirm before closing properties with unsaved changes** switch is set to **True**, pgAdmin will warn you before closing the properties dialog of an object if there are any unsaved changes. On user confirmation, the properties dialog will close.
- When the **Confirm on close or refresh** switch is set to **True**, pgAdmin will attempt to catch browser close or refresh events and prompt before allowing them to continue.
- When the **Show system objects?** switch is set to **True**, the client will display system objects such as system schemas (for example, `pg_temp`) or system columns (for example, `xmin` or `ctid`) in the tree control.
- When the **Enable dialogue/notification animation?** switch is set to **True**, the client will display the animated dialogues/notifications; if the switch is **False**, the tree control will be unanimated.
- Set **Show hidden groups?** to **True** to display hidden groups in the Browser tree control.
- Set **Show system objects?** to **True** to display system objects such as system schemas (for example, `pg_temp`) or system columns (for example, `xmin` or `ctid`) in the Browser tree control.
- Use the **Lock layout** field to lock the UI layout at different levels.

Option	Action
None	No locking. Every panel is resizable and dockable.
Prevent docking	This will disable the docking/undocking of the panels
Full	This will disable resizing, docking/undocking of the panels

- When the **Show system objects?** switch is set to **True**, the client will display system objects such as system schemas (for example, `pg_temp`) or system columns (for example, `xmin` or `ctid`) in the tree control.

Use the fields on the **Keyboard shortcuts** panel to configure shortcuts for the main window navigation:



- Use controls on the **Keyboard shortcuts** panel to specify the combination of modifier keys that define shortcuts for the PEM main window.

Use the fields on the **Nodes** panel to select the object types that will be displayed in the **Browser** tree control:



- The panel displays a list of database objects; slide the switch located next to each object type to **Show** or **Hide** the database object. When querying system catalogs, you can reduce the number of object types displayed to increase speed.

Use fields on the **Properties** panel to specify browser properties:



- Include a value in the **Count rows if estimated less than** field to perform a `SELECT count()` if the estimated number of rows in a table (as read from the table statistics) is below the specified limit. After performing the `SELECT count()`, pgAdmin will display the row count. The default is 2000.
- Provide a value in the **Maximum job history rows** field to limit the number of rows to show on the statistics tab for pgAgent jobs. The default is 250.

Use field on *Tab settings* panel to specify the tab related properties.



- Use *Debugger tab title placeholder* field to customize the Debugger tab title.
- When the *Dynamic tab size* If set to True, the tabs will take full size as per the title, it will also applicable for already opened tabs
- When the *Open in new browser tab* file is selected for Query tool, Schema Diff or Debugger, it will open in a new browser tab when invoked.
- Use the *Query tool tab title placeholder* field to customize the query tool tab title.
- Use *View/Edit tab title placeholder* field to customize the View/Edit Data tab title.

The Dashboards Node

Expand the **Dashboards** node to specify your dashboard display preferences.



- When the **Show activity?** switch is set to **True**, activity tables will be displayed on dashboards.
- When the **Show graph data points?** switch is set to **True**, data points will be visible on graph lines.
- When the **Show graphs?** switch is set to **True**, graphs will be displayed on dashboards.
- When the **Show mouse hover tooltip?** switch is set to **True**, a tooltip will appear on mouse hover on the graph lines giving the data point details.

Use the fields on the **Graphs** panel to specify your display preferences for the graphs on the **Dashboard** tab:



Use the fields on the **Graphs** panel to specify your display preferences for the graphs on the **Dashboard** tab:

- Use the **Block I/O statistics refresh rate** field to specify the number of seconds between block I/O statistic samples displayed in graphs.
- Use the **Session statistics refresh rate** field to specify the number of seconds between session statistic samples displayed in graphs.
- Use the **Transaction throughput refresh rate** field to specify the number of seconds between transaction throughput samples displayed in graphs.
- Use the **Tuples in refresh rate** field to specify the number of seconds between tuples-in samples displayed in graphs.
- Use the **Tuples out refresh rate** field to specify the number of seconds between tuples-out samples displayed in graphs.

The Debugger Node

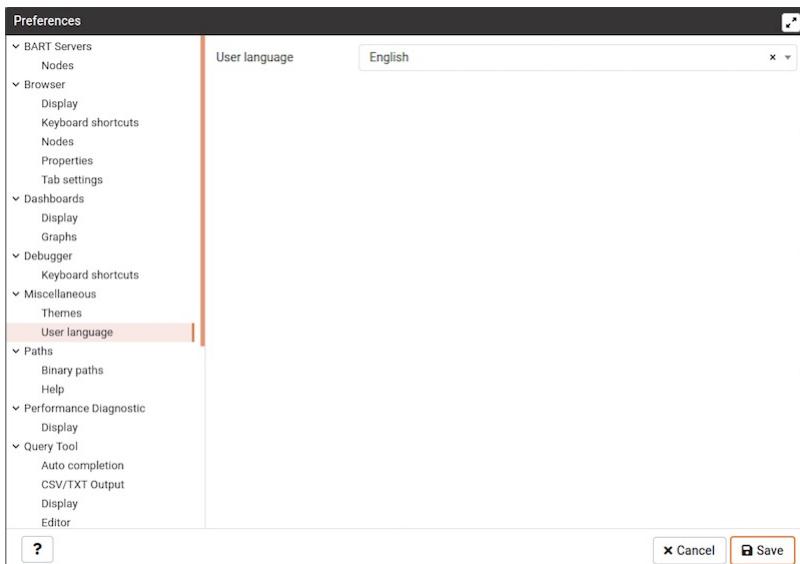
Expand the **Debugger** node to specify your debugger display preferences.

Use the fields on the **Keyboard shortcuts** panel to configure shortcuts for the debugger window navigation:

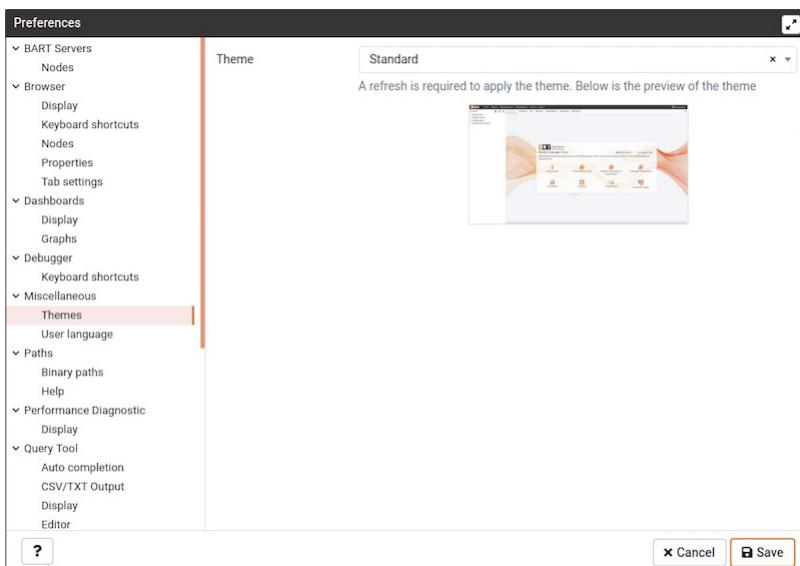


The Miscellaneous Node

Expand the **Miscellaneous** node to specify miscellaneous display preferences.



- Use the **User language** drop-down listbox to select the display language for the PEM web interface.



- Use the **Themes** drop-down listbox to select the theme for PEM. You'll also get a preview just below the drop down. Note that, to apply the theme you need to refresh the PEM page.

The Paths Node

Expand the **Paths** node to specify the locations of supporting utility and help files.



Use the fields on the **Binary paths** panel to specify the path to the directory that contains the utility programs (pg_dump, pg_restore, and pg_dumpall) for monitored databases:

- Use the **EDB Advanced Server Binary Path** field to specify the location of the EDB Postgres Advanced Server utility programs. If this path is not set, pgAdmin will attempt to find the utilities in standard locations used by EnterpriseDB.
- Use the **Greenplum Database Binary Path** field to specify the location of the Greenplum database utility programs. If this path is not set, pgAdmin will attempt to find the utilities in standard locations used by Greenplum.
- Use the **PostgreSQL Binary Path** field to specify the location of the PostgreSQL utility programs. If this path is not set, pgAdmin will attempt to find the utilities in standard locations used by PostgreSQL.



Use the fields on the **Help** panel to specify the location of help files.

- Use the **EDB Advanced Server Help Path** field to specify the path to EDB Postgres Advanced Server documentation.
- Use the **PostgreSQL Help Path** field to specify the path to PostgreSQL documentation.

Please note: the default help paths include the **VERSION** placeholder; the \$VERSION\$ placeholder will be replaced by the current database version.

The Performance Diagnostic Node

Expand the **Performance Diagnostic** node to specify your preferences for the Performance Diagnostic tool.



Use the fields on the **Performance Diagnostic** panel to control the Performance Diagnostic output.

- Use the **Default graph selection** field to specify the default selection range in hours for performance diagnostic graphs.
- When the **Open in new browser tab?** switch is set to True, the Performance Diagnostic tool will be opened in a new browser tab.

The Query Tool Node

Expand the **Query Tool** node to access panels that allow you to specify your preferences for the Query Editor tool.



Use the fields on the **Auto Completion** panel to set the auto completion options.

- When the **Keywords in uppercase** switch is set to **True**, keywords are displayed in upper case.



Use the fields on the **CSV Output** panel to control the CSV output.

- Use the **CSV field separator** drop-down listbox to specify the separator character that will be used in CSV/TXT output.
- Use the **CSV quote character** drop-down listbox to specify the quote character that will be used in CSV/TXT output.
- Use the **CSV quoting** drop-down listbox to select the fields that will be quoted in the CSV/TXT output; select **Strings**, **All**, or **None**.
- Use the **Replace null values with** option to replace null values with specified string in the output file. Default is set to 'NULL'.



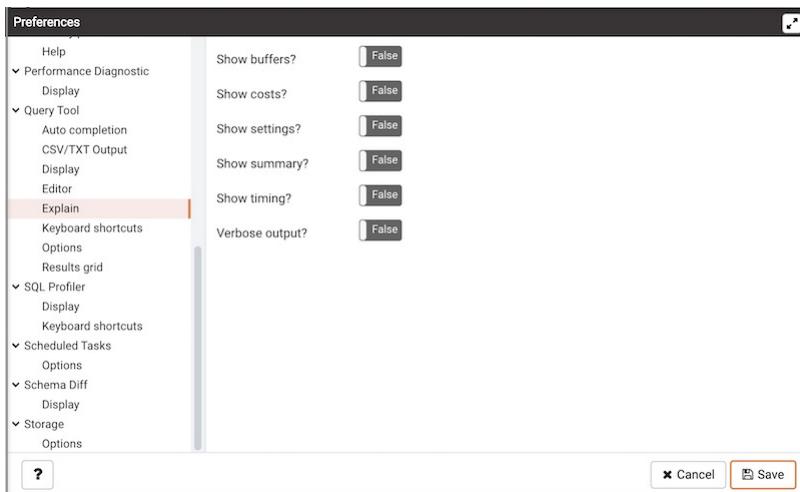
Use the fields on the **Display** panel to specify your preferences for the Query Tool display.

- When the **Connection status** switch is set to **True**, each new instance of the Query Tool will display connection and transaction status.
- Use the **Connection status refresh rate** field to specify the number of seconds between connection/transaction status updates.
- Use the **Query info notifier timeout** field to control the behaviour of the notifier that is displayed when query execution completes. A value of **-1** will disable the notifier, and a value of **0** will display it until clicked. If a positive value above zero is specified, the notifier will be displayed for the specified number of seconds. The default is **5**.



Use the fields on the **Editor** panel to change settings of the query editor.

- When the **Brace matching?** switch is set to **True**, the editor will highlight pairs of matched braces.
- When the **Code folding?** switch is set to **False**, the editor will disable code folding. Disabling will improve editor performance with large files.
- Use the **Font size** field to specify the font size that will be used in text boxes and editors.
- When the **Insert bracket pairs?** switch is set to **True**, the editor will automatically insert paired brackets.
- When the **Line wrapping** switch is set to **True**, the editor will implement line-wrapping behavior.
- When the **Plain text mode?** switch is set to **True**, the editor mode will be changed to text/plain. Keyword highlighting and code folding will be disabled. This will improve editor performance with large files.



Use the fields on the **Explain** panel to specify the level of detail included in a graphical EXPLAIN.

- When the **Show Buffers?** switch is set to **True**, graphical explain details will include information about buffer usage.
- When the **Show Costs?** switch is set to **True**, graphical explain details will include information about the estimated startup and total cost of each plan, as well as the estimated number of rows and the estimated width of each row.
- When the **Show Timing?** switch is set to **True**, graphical explain details will include the startup time and time spent in each node in the output.
- When the **Verbose output?** switch is set to **True**, graphical explain details will include extended information about the query execution plan.

Use the fields on the **Keyboard shortcuts** panel to configure shortcuts for the Query Tool.



Use the fields on the **Options** panel to manage Query Tool preferences.



- When the **Auto-Commit?** switch is set to **True**, each successful query is committed after execution.
- When the **Auto-Rollback?** switch is set to **True**, failed queries are rolled back.
- When the **Prompt to save unsaved data changes?** switch is set to **True**, the editor will prompt the user to saved unsaved data when exiting the data editor.
- When the **Prompt to save unsaved query changes?** switch is set to **True**, the editor will prompt the user to saved unsaved query modifications when exiting the query tool.
- When the **Prompt to commit/rollback active transactions?** switch is set to **True**, the editor will prompt the user to commit or rollback changes when exiting the Query Tool while the current transaction is not committed.
- When the **Sort View Data results by primary key columns?** is set to **True**, data returned when using the View/Edit Data - All Rows option will be sorted by the Primary Key columns by default. When using the First/Last 100 Rows options, data is always sorted.



Use the fields on the *SQL formatting* panel to specify your preferences for reformatting of SQL.

- Use the *Command-first notation* option to specify whether to place commas before or after column names.
- Use the *Identifier case* option to specify whether to change identifiers (object names) into upper, lower, or capitalized case.
- Use the *Keyword case* option to specify whether to change keywords into upper, lower, or capitalized case.
- Use the *Re-indent aligned?* option to specify that indentations of statements should be changed, aligned by keywords.
- Use the *Re-indent?* option to specify that indentations of statements should be changed.
- Use the *Spaces around operators?* option to specify whether or not to include spaces on either side of operators.
- Use the *Strip comments?* option to specify whether or not comments should be removed.
- Use the *Tab size* option to specify the number of spaces per tab or indent.
- Use the *Use spaces?* option to select whether to use spaces or tabs when indenting.
- Use the *Wrap after N characters* option to specify the column limit for wrapping column separated lists (e.g. of column names in a table). If set to 0 (zero), each item will be on its own line.



Use the fields on the *Results grid* panel to specify your formatting preferences for copied data.

- Use the *Result copy field separator* drop-down listbox to select the field separator for copied data.
- Use the *Result copy quote character* drop-down listbox to select the quote character for copied data.
- Use the *Result copy quoting* drop-down listbox to select which type of fields require quoting; select *All*, *None*, or *Strings*.

The SQL Profiler Node

Use fields on the *Display* panel to specify SQL Profiler preferences.



Set **Open in New Browser Tab?** to **True** to open SQL Profiler in a new browser tab when SQL Profiler is invoked.

Use the fields on the Keyboard shortcuts panel to configure shortcuts for toolbar buttons on SQL profiler trace window.



The Scheduled Tasks Node

Use fields on the **Options** panel to specify Scheduled Tasks preferences.



Use the **Auto refresh interval** field to specify the number of seconds between automatic refreshes; a value of 0 disables auto refresh.

The Schema Diff Node

Expand the **Schema Diff** node to specify your display preferences.



Use the *Ignore owner* switch to ignores the owner while comparing the objects.

Use the *Ignore whitespaces* switch to ignores the whitespaces while comparing the string objects. Whitespace includes space, tabs, and CRLF.

The Storage Node

Expand the **Storage** node to specify your storage preferences.



Use the fields on the **Options** panel to specify storage preferences.

- Use the **File dialog view** drop-down listbox to select the style of icons and display format that will be displayed when you open the file manager; select **List** to display a list view, or **Grid** to display folder icons.
- Use the **Last directory visited** field to specify the name of the folder in which the file manager will open.
- Use the **Maximum file upload size(MB)** field on the **Options** panel of the Storage node to specify the maximum file size for an upload.
- When the **Show hidden files and folders?** switch is set to **True**, the file manager will display hidden files and folders.

7.3.5 Keyboard Shortcuts

Keyboard shortcuts are provided in pgAdmin to allow easy access to specific functions. Alternate shortcuts can be configured through File > Preferences if desired."

Main Browser Window

When using main browser window, the following keyboard shortcuts are available:

Shortcut for all platforms	Function
Alt+Shift+F	Open the File menu
Alt+Shift+O	Open the Object menu
Alt+Shift+L	Open the Tools menu
Alt+Shift+H	Open the Help menu
Alt+Shift+B	Focus the browser tree
Alt+Shift+[Move tabbed panel backward
Alt+Shift+]	Move tabbed panel forward
Alt+Shift+Q	Open the Query Tool in the current database
Alt+Shift+V	View Data in the selected table/view
Alt+Shift+C	Open the context menu
Alt+Shift+N	Create an object
Alt+Shift+E	Edit object properties
Alt+Shift+D	Delete the object
Alt+Shift+G	Direct debugging

Dialog Tabs

Use the shortcuts below to navigate the tabs on dialogs:

Shortcut for all platforms	Function
Control+Shift+[Dialog tab backward
Control+Shift+]	Dialog tab forward

Property Grid Controls

Use the shortcuts below when working with property grid controls:

Shortcut for all platforms	Function
Control+Shift+A	Add row in Grid
Tab	Move focus to the next control
Shift+Tab	Move focus to the previous control
Return	Pick the selected an item in a combo box
Control+Shift+A	Add row in Grid

SQL Editors

When using the syntax-highlighting SQL editors, the following shortcuts are available:

Shortcut (Windows/Linux)	Shortcut (Mac)	Function
Alt + Left	Option + Left	Move to the beginning of the line
Alt + Right	Option + Right	Move to the end of the line
Ctrl + Alt + Left	Cmd + Option + Left	Move left one word
Ctrl + Alt + Right	Cmd + Option + Right	Move right one word
Ctrl + /	Cmd + /	Comment selected code (Inline)
Ctrl + .	Cmd + .	Uncomment selected code (Inline)
Ctrl + Shift + /	Cmd + Shift + /	Comment/Uncomment code (Block)
Ctrl + a	Cmd + a	Select all
Ctrl + c	Cmd + c	Copy selected text to the clipboard
Ctrl + r	Cmd + r	Redo last edit un-done
Ctrl + v	Cmd + v	Paste text from the clipboard
Ctrl + z	Cmd + z	Undo last edit
Tab	Tab	Indent selected text
Shift + Tab	Shift + Tab	Un-indent selected text
Alt + g	Option + g	Jump (to line:column)
Ctrl + Space	Ctrl + Space	Auto-complete
Ctrl + f	Cmd + f	Find
Ctrl + g	Cmd + g	Find next
Ctrl + Shift + g	Cmd + Shift + g	Find previous
Ctrl + Shift + f	Cmd + Shift + f	Replace

Query Tool

When using the Query Tool, the following shortcuts are available:

Shortcut (Windows/Linux)	Shortcut (Mac)	Function
F5	F5	Execute query
F6	F6	Save data changes
F7	F7	EXPLAIN query
Shift + F7	Shift + F7	EXPLAIN ANALYZE query
F8	F8	Execute query to CSV file
<accesskey> + o	<accesskey> + o	Open file
<accesskey> + s	<accesskey> + s	Save file
<accesskey> + n	<accesskey> + n	Find option drop down
<accesskey> + c	<accesskey> + c	Copy row(s)
<accesskey> + p	<accesskey> + p	Paste row(s)
<accesskey> + d	<accesskey> + d	Delete row(s)
<accesskey> + f	<accesskey> + f	Filter dialog
<accesskey> + i	<accesskey> + i	Filter options drop down

Shortcut (Windows/Linux)	Shortcut (Mac)	Function
<accesskey> + r	<accesskey> + r	Row limit
<accesskey> + q	<accesskey> + q	Cancel query
<accesskey> + l	<accesskey> + l	Clear option drop down
<accesskey> + x	<accesskey> + x	Execute option drop down
<accesskey> + t	<accesskey> + t	Display connection status
<accesskey> + y	<accesskey> + y	Copy SQL on history panel

Debugger

When using the Debugger, the following shortcuts are available:

Shortcut (Windows/Linux)	Shortcut (Mac)	Function
<accesskey> + i	<accesskey> + i	Step in
<accesskey> + o	<accesskey> + o	Step over
<accesskey> + c	<accesskey> + c	Continue/Restart
<accesskey> + t	<accesskey> + t	Toggle breakpoint
<accesskey> + x	<accesskey> + x	Clear all breakpoints
<accesskey> + s	<accesskey> + s	Stop
Alt + Shift + q	Option + Shift + q	Enter or Edit values in Grid

Inner Tab and Panel Navigation

When using the Query Tool and Debugger, the following shortcuts are available for inner panel navigation:

Shortcut (Windows/Linux)	Shortcut (Mac)	Function
Alt + Shift +]	Alt + Shift +]	Move to next tab within a panel
Alt + Shift + [Alt + Shift + [Move to previous tab within a panel
Alt + Shift + Tab	Alt + Shift + Tab	Move between inner panels

Access Key

<accesskey> is browser and platform dependant. The following table lists the default access keys for supported browsers.

Browser	Windows	Linux	Mac
Internet Explorer	Alt	Alt	
Chrome	Alt	Alt	Ctrl + Option
Firefox	Alt + Shift	Alt + Shift	Ctrl + Option
Safari	Alt		Ctrl + Option

7.3.6 Search objects

Object name	Type	Browser path
sslutils	Extensions	Extensions/sslutils
(...) openssl_csr_to_crt (...)	Functions	Schemas/public/Functions/openssl_csr_to_crt
(...) openssl_get_crt_expiry_dat...	Functions	Schemas/public/Functions/openssl_get_crt_expiry...
(...) openssl_is_crt_expire_on (...)	Functions	Schemas/public/Functions/openssl_is_crt_expire_on
(...) openssl_revoke_certificate (...)	Functions	Schemas/public/Functions/openssl_revoke_certific...
(...) openssl_rsa_generate_crl (...)	Functions	Schemas/public/Functions/openssl_rsa_generate_crl
(...) openssl_rsa_generate_key (...)	Functions	Schemas/public/Functions/openssl_rsa_generate_k...
(...) openssl_rsa_key_to_csr (...)	Functions	Schemas/public/Functions/openssl_rsa_key_to_csr
sslutils_version	Functions	Schemas/public/Functions/sslutils_version

9 matches found.

[?](#) [Close](#)

With this dialog, you can search for almost any kind of objects in a database.

You can access it by right clicking a database or any of its child nodes and select "Search objects". You can also access it by hitting the shortcut (default ALT+SHIFT+S).

The minimum pattern length are 3 characters. The search performed is non-casesensitive and will find all objets whose name contains the pattern. You can only search for object names currently. Examples are: abc, %ab%, ab%, %%%, etc.

The result is presented in the grid with object name, object type and the object tree path in the browser tree. You can double click on a result row to select the object in the browser tree. If the object is greyed out, this means that you have not enabled those object types in the [preferences](#), so you can't double click on it. You can click on the ellipsis appended to the function and procedure names to see there arguments.

You can filter based on a particular object type by selecting one from the object type dropdown. If the search button is hit when one of the object type is selected then only those types will be fetch from the database. An object type will not be visible in the dropdown if the database server does not support it or if it is not enabled from the [preferences](#).

7.4 Enterprise Management Features

Postgres Enterprise Manager offers a number of additional enterprise management features that will assist you in managing, analyzing, streamlining, and deploying Postgres functionality. PEM probes monitor managed servers, retrieving information that PEM then analyzes to create dashboards that display useful information and statistics about your hosts, servers and databases. PEM dialogs provide easy access to probe, server, and agent configurations so you can enable and customise the behaviour of PEM features.

Contents:

7.4.1 Dashboards

Postgres Enterprise Manager uses metrics (retrieved by probes) to generate the statistical information displayed on the dashboards. Dashboards are presented in a hierarchy comparable to the PEM client tree control; the dashboard for each object within the tree control displays the information for that object, as well as for any monitored object that resides below that level in the tree control, if appropriate.

Each dashboard header displays the date and time that the server was started (if relevant), the date and time that the dashboard was last updated, and the current number of triggered alerts. Navigation menus displayed in the dashboard header provide easy access to other dashboards. Menus are organised hierarchically; only those menus appropriate for the object currently highlighted in the tree control are available:

- Select **Global Overview** from any dashboard to return to the Global Overview dashboard.
- Select the name of an agent from the **Agents** menu to navigate to the Operating System Analysis dashboard for that agent.
- Select a server name from the **Servers** menu to navigate to the Server Analysis dashboard for that server.
- Select a database name from the **Databases** menu to navigate to the Database Analysis dashboard for that database.
- Use the **Dashboards** menu to navigate to informational dashboards at the global level, or for the selected agent, server or database.

Dashboards display statistical information in the form of:

- Tables - Tables provide statistical information collected by a PEM probe.
- Pie charts - Pie charts display information collected by the most recent execution of a probe.
- Bar graphs - Bar graphs display comparative statistics collected by the most recent execution of a probe.
- Line graphs - Line graphs display statistical data collected by PEM probes.

Options on the **Dashboard Configuration** dialog allow you to link the time lines of all of the line graphs on the dashboard. To open the **Dashboard Configuration** dialog, click the wrench icon displayed in the dashboard header.



- Set the **Link timelines of all the line charts** slider to **Enable** to indicate that the specified timeline should be applied to line graphs displayed on the dashboard; if set to **Disable**, your preferences will be preserved for later use, but will not modify the amount of data displayed.
- Use the **Days** selector to specify the number of days of gathered data that should be displayed on line graphs.
- Use the **Hour(s)** selector to specify the number of hours of gathered data that should be displayed on line graphs.
- Check the box next to **Remember configuration for this dashboard** to indicate that the customized time span should be applied to the current dashboard only; if left unchecked, the time span will be applied globally to line graphs on all dashboards.

Please note that settings specified on the **Dashboard Configuration** dialog are applied only to the current user's session.

When you've specified your preferences, click **Save** to preserve your changes and exit the dialog; click **Cancel** to exit

the dialog without preserving your changes.

To sort statistics that are provided in table form, click on a column heading; click again to reverse the sort order. Each table offers a stable sort feature - For example, to sort a table by ascending **Session ID** within each user name group, sort first by the **Session ID** column, then sort by the **User Name** column.

Hover your mouse over the upper-right corner of each graph, chart or table to reveal the PEM client toolbar icons. Hover over an icon to display a tooltip that briefly explains the icon's functionality:

- Use the **Refresh** icon to update the information displayed on a dashboard.
- Use the **Save Chart as Image** icon to save the selected chart as a .jpeg image.
- Use the **Full Screen** icon to enlarge the chart to reveal granular details about the charted data.
- Click the **Personalize the chart configuration** icon to access a control panel that allows you to select chart-specific display details.
- Hover over the **Explain** icon to review a description of the information shared in the graph or chart.

In the lower-right corner of each graph or chart is a legend that identifies each item plotted in the graph or chart.

If displayed, click the information icon in the upper-left hand corner of a chart to display a note about the chart content, and if applicable, a link that will allow you to enable one or more probes that retrieve content for the chart.

Accessing Dashboards

Navigation menus in the dashboard header provide easy access to other dashboards. The menus are organized hierarchically, allowing you to jump from object to object at any level:

- The **Global Overview** option opens the **Global Overview** dashboard.
- The **Agents** menu expands to display a list of agents. Select an agent from the list to access the **Operating System Analysis** dashboard for that agent.
- The **Servers** menu expands to display a list of monitored servers. Select a server from the list to access the **Server Analysis** dashboard for that server.
- The **Remote Servers** menu expands to display a list of servers that are monitored by a remote agent. Select a server from the list to access the **Server Analysis** dashboard for the server.
- The **Databases** menu expands to display a list of databases. Select a database from the list to access the **Database Analysis** dashboard for the database.
- The **Dashboards** menu expands to display a list of the dashboards that are available at the global level, or for the current agent, server or database. Select a dashboard from the list to navigate to that dashboard.

Creating custom charts and dashboards

PEM (version 4.0 and above) allows you to create your own **Charts** and **Dashboards**, allowing you to tailor the interface to the requirements of your organization or individual responsibility.

Available Dashboards

PEM offers the following dashboards:

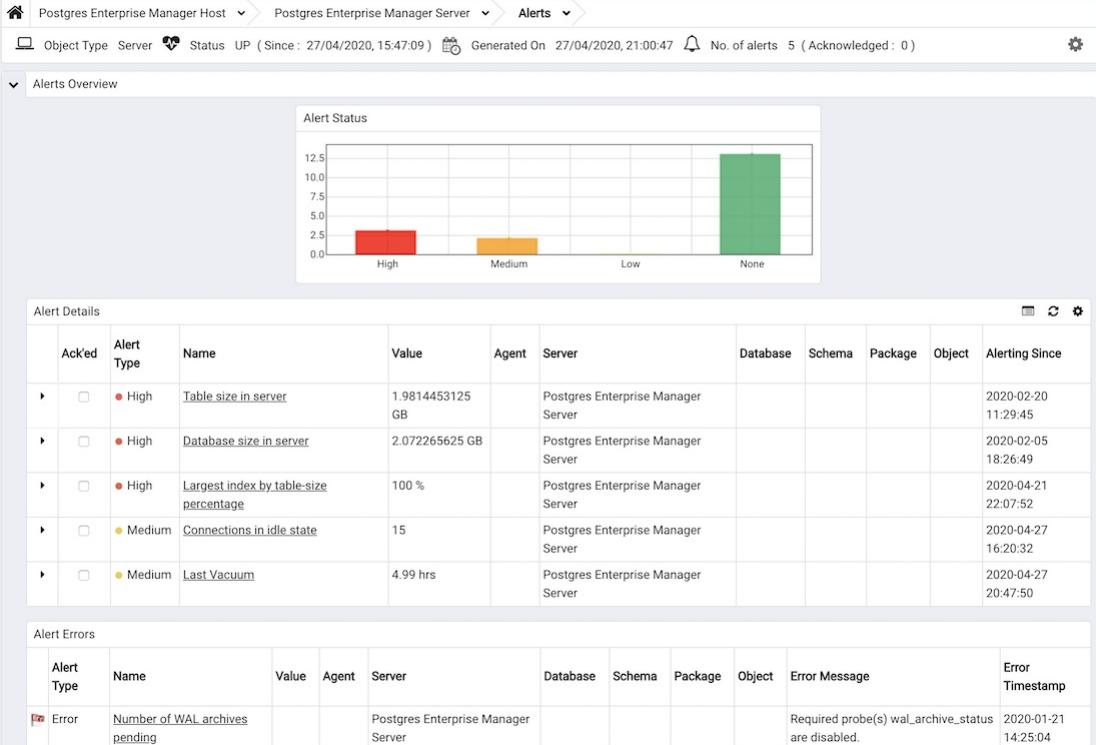
- [Alerts Dashboard](#)
- [Audit Log Dashboard](#)
- [Database Analysis Dashboard](#)
- [Global Overview Dashboard](#)

- I/O Analysis Dashboard <io_analysis_dashboard>
- Memory Analysis Dashboard
- Object Activity Analysis Dashboard
- Operating System Analysis Dashboard
- Probe Log Analysis Dashboard
- Server Analysis Dashboard
- Server Log Analysis Dashboard
- Session Activity Analysis Dashboard
- Session Wait Analysis Dashboard
- Storage Analysis Dashboard
- System Wait Analysis Dashboard
- Streaming Replication Analysis Dashboard

Contents:

7.4.1.1 The Alerts Dashboard

The Alerts Dashboard displays the currently triggered alerts; if opened from the Global Overview, the dashboard displays the current alerts for all monitored nodes on the system. If the Alerts Dashboard is opened from a node within a monitored hierarchy, the report will reflect alerts related to that node, and all monitored objects that reside below that object in the tree control.



The screenshot shows the 'Alerts' dashboard interface. At the top, there's a header bar with navigation links for 'Postgres Enterprise Manager Host', 'Postgres Enterprise Manager Server', and 'Alerts'. Below the header, it displays 'Object Type: Server', 'Status: UP (Since: 27/04/2020, 15:47:09)', 'Generated On: 27/04/2020, 21:00:47', 'No. of alerts: 5 (Acknowledged: 0)', and a gear icon for settings.

The main content area is divided into sections:

- Alerts Overview:** Contains a chart titled 'Alert Status' showing the count of alerts by severity: High (red), Medium (orange), Low (yellow), and None (green). The 'None' category has a value of approximately 12.5.
- Alert Details:** A table listing triggered alerts. The columns include: Acked (checkbox), Alert Type (color-coded: red for High, yellow for Medium), Name, Value, Agent, Server, Database, Schema, Package, Object, and Alerting Since. The table lists five alerts:

Acked	Alert Type	Name	Value	Agent	Server	Database	Schema	Package	Object	Alerting Since
▶	High	Table size in server	1.9814453125 GB	Postgres Enterprise Manager Server						2020-02-20 11:29:45
▶	High	Database size in server	2.072265625 GB	Postgres Enterprise Manager Server						2020-02-05 18:26:49
▶	High	Largest index by table-size percentage	100 %	Postgres Enterprise Manager Server						2020-04-21 22:07:52
▶	Medium	Connections in idle state	15	Postgres Enterprise Manager Server						2020-04-27 16:20:32
▶	Medium	Last Vacuum	4.99 hrs	Postgres Enterprise Manager Server						2020-04-27 20:47:50
- Alert Errors:** A table showing errors. The columns include: Alert Type (Error icon), Name, Value, Agent, Server, Database, Schema, Package, Object, Error Message, and Error Timestamp. One error is listed:

Alert Type	Name	Value	Agent	Server	Database	Schema	Package	Object	Error Message	Error Timestamp
Error	Number of WAL archives pending		Postgres Enterprise Manager Server						Required probe(s) wal_archive_status are disabled.	2020-01-21 14:25:04

Use parameters on the [PEM Server Configurations](#) dialog to specify the auto-refresh rate for the [Alerts](#) dashboard. To access the [Server Configuration](#) dialog, select [Server Configuration...](#) from the PEM web interface [Management](#) menu.

The [Alerts Dashboard](#) header includes the date and time that the page was last updated and a current count of triggered alerts.

The **Alerts Overview** provides an overview of triggered alerts. The right-most bar indicates the total number of configured alerts that are not currently in an alert state; the three left-most bars indicate the number of Low, Medium and High alerts for the selected object. The vertical key on the left side of the graph provides an alert count.

The **Alert Details** table lists the currently triggered alerts for the selected object; if opened from the global overview, the Alert Details table lists all of the currently triggered alerts for all monitored objects. Click a column heading to sort the table by the contents of a selected column; click a second time to reverse the sort order. The table contains detailed information about each alert:

- An alert level icon displays in red for a **High** severity alert, in orange for a **Medium** severity alert, and in yellow for a **Low** severity alert.
- Use the arrow to the right of the alert level icon to access a dialog with detailed information about the alert. Within the dialog, the **Details** tab displays detailed information about the condition that triggered the alert; the **Parameters** tab displays the values of parameters used in the alert definition. Not all alerts return data that can be viewed on the **Details** dialog; for information about which templates display detailed metrics, please see the [alert templates list](#)

Alert Details (Auto-refresh paused whilst rows are expanded. 										
Acked	Alert Type	Name	Value	Agent	Server	Database	Schema	Package	Object	Alerting Since
 	High	Table size in server	1.9814453125 GB	Postgres Enterprise Manager Server						2020-02-20 11:29:45
 										
Table name Schema name Database name Total table size(MB)										
table_statistics	pemhistory	pem	1087							
server_logs	pemdata	pem	263							
index_statistics	pemhistory	pem	237							
session_info	pemhistory	pem	137							
lock_info	pemhistory	pem	88							

- The **Ack'ed** column provides a checkbox to allow you to acknowledge an alert to prevent additional notifications being sent. This flag is cleared automatically if the alert condition clears and is then detected again.
- The **Alert Type** column indicates the severity of the alert.
- The **Name** column displays the names of the currently triggered alerts. Click the name of an alert to open the **Alerting** configuration dialogue that defines the alert.
- The **Value** column displays the value of the metric that triggered the alert.
- If applicable, the **Agent** column indicates the name of the agent on which the alert is defined.
- If applicable, the **Server** column indicates the name of the server triggering the error message.
- If applicable, the **Database** column indicates the name of the database on which the alert is defined.
- If applicable, the **Schema** column indicates the name of the schema on which the alert is defined.
- If applicable, the **Package** column indicates the name of the package on which the alert is defined.
- If applicable, the **Object** column indicates the name of the monitored object on which the alert is defined.
- If the alert definition includes specified parameters, the parameter values are displayed in the **Additional Params** column.
- If the alert definition includes additional specified parameters, the additional parameter values are displayed in the **Additional Params Value** column.
- The **Alerting Since** column displays the date and time that the alert triggered.

The **Alert Errors** table displays configuration-related errors (eg.accidentally disabling a required probe, or improperly configuring an alert parameter):

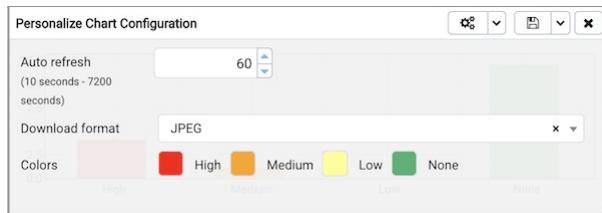
Alert Errors										
Alert Type	Name	Value	Agent	Server	Database	Schema	Package	Object	Error Message	Error Timestamp
 Error	Number of WAL_archives pending		Postgres Enterprise Manager Server						Required probe(s) wal_archive_status are disabled.	2020-01-21 14:25:04

- An alert indicator in the left-most column indicates that the alert was triggered by an Error.

- The **Alert Type** column indicates the severity of the alert.
- The **Name** column displays the name of the alert. Click an alert name to open the configuration dialogue for the alert.
- The **Value** column displays the value of the metric that triggered the alert, if applicable.
- If applicable, the **Agent** column displays the name of the agent triggering the alert.
- If applicable, the **Server** column displays the name of the server triggering the alert.
- If applicable, the **Database** column indicates the name of the database on which the alert is defined.
- If applicable, the **Schema** column indicates the name of the schema on which the alert is defined.
- If applicable, the **Package** column indicates the name of the package on which the alert is defined.
- If applicable, the **Object** column indicates the name of the monitored object on which the alert is defined.
- The **Error Message** column describes the condition that triggered the alert.
- The **Error Timestamp** column displays the date and time that the alert was triggered.

Customizing the Alerts Dashboard

You can customize tables and charts that appear on the Alerts dashboard. To open the **Personalize chart configuration** dialog, click the wrench icon in the upper-right corner.



The fields displayed on the dialog will vary based on the table or chart from which the dialog is opened.

Personalize Chart Configuration						
Auto refresh	Name	Value	Agent	Server	Database	Schema
(10 seconds - 7200 seconds)	Table size in bytes	1,684,570,9125	Postgres Enterprise Manager	Server	2020-02-30	
		GB			11:29:49	
	High	Database size in bytes	1,756,601,5625	Postgres Enterprise Manager	2020-03-05	
		GB			18:26:49	
	High	Connections in idle state	21	Postgres Enterprise Manager	2020-04-27	
			Server		13:58:27	
	High	Last VACUUM	Never run	Postgres Enterprise Manager	2020-01-21	
			Server		14:26:04	
	High	Largest index by table size percentage	100 %	Postgres Enterprise Manager	2020-04-21	
			Server		22:07:32	

Use fields on the **Personalize chart configuration** dialog to provide your display preferences:

- Use the **Auto Refresh** field to specify the number of seconds between updates of the data displayed in the table or chart.
- If applicable, use the **Download as** field to indicate if you would like a chart to be downloaded as a JPEG image or a PNG image.
- If applicable, use the **Colours** selectors to specify the display colors that will be used on a chart.
- If applicable, set the **Show Acknowledged Alerts** switch to **Yes** indicate that you would like the table to display alerts that you have acknowledged with a checkbox in the **Ack'ed** column. Set the field to **No** to indicate that the table should hide any acknowledged alerts. The switch acts as a toggle; acknowledged alerts are not purged from the table content until the time specified in the alert definition passes.

To save your customizations, click the save icon (a check mark) in the upper-right corner; to delete any previous changes and revert to the default values, click the delete icon. Save and Delete drop-down menus allow you to specify if your preferences should be applied to **All Dashboards**, or to a selected server or database. Use the close icon to close the **Personalize chart configuration** dialog without preserving your changes.

7.4.1.2 The Audit Log Analysis Dashboard

The Audit Log Dashboard allows you to browse the audit logs that have been collected from Advanced Server instances which have enabled audit logging and collection with the [Audit Manager](#). If the Audit Log Dashboard is opened from the Global level, it will display logs from all servers. If opened from the Agent level, it will show logs from all servers monitored by that Agent. If opened from the Server level, it will show logs from that server only.



The screenshot shows the Audit Log Analysis Dashboard interface. At the top, there is a navigation bar with tabs: Properties, SQL, Statistics, Dependencies, Dependents, Monitoring, and Audit Log. The Audit Log tab is selected. Below the navigation bar, there is a header row with columns: Object Type, Host Agent, Status, Generated On, and No. of alerts. The status is shown as UP (Since: 15/04/2020, 19:35:06). The generated on date is 29/04/2020, 10:08:32. There are no alerts. The main area is titled "Audit Logs" and contains a table with the following columns: Id, Server, Timestamp, User Name, Database Name, Process ID, Session ID, Transaction ID, Connection From, Command, and Message. The table lists 18 rows of audit log entries, each with a timestamp between 29/04/2020, 10:00:07 and 29/04/2020, 10:08:32. The "Message" column contains various audit log messages such as disconnection, authentication, and SELECT statements.

Id	Server	Timestamp	User Name	Database Name	Process ID	Session ID	Transaction ID	Connection From	Command	Message
7879	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26391	Sea902cf.6717	0	127.0.0.1:46780	idle	disconnection: session time: 0:00:00.014 user=enterprisedb database=postgres host=127.0.0.1 port=46780
7878	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26391	Sea902cf.6717	0	127.0.0.1:46780	idle	statement: SELECT setting FROM pg_settings WHERE name = 'edb_audit_rotation_seconds'
7877	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26391	Sea902cf.6717	0	127.0.0.1:46780	idle	statement: SELECT version();
7876	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26391	Sea902cf.6717	0	127.0.0.1:46780	authentication	connection authorized: user=enterprisedb database=postgres
7875	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26388	Sea902cf.6714	0	127.0.0.1:46774	idle	disconnection: session time: 0:00:00.007 user=enterprisedb database=postgres host=127.0.0.1 port=46774
7874	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26388	Sea902cf.6714	0	127.0.0.1:46774	authentication	connection authorized: user=enterprisedb database=postgres
7873	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	Sea902cf.6710	0	127.0.0.1:46766	idle	disconnection: session time: 0:00:00.029 user=enterprisedb database=postgres host=127.0.0.1 port=46766
7872	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	Sea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT setting FROM pg_settings WHERE name='log_temp_files'
7871	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	Sea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT setting FROM pg_settings WHERE name='log_autovacuum_min_duration'
7870	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	Sea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT setting FROM pg_settings WHERE name='log_min_duration_statement'
7869	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	Sea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT (setting::int/(24*60))::int FROM pg_settings WHERE name = 'log_rotation_age'
7868	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	Sea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT (setting::int/1024)::int FROM pg_settings WHERE name = 'log_rotation_size'
7867	EPAS_12	29/04/2020, 10:00:07	enterprisedb	postgres	26384	Sea902cf.6710	0	127.0.0.1:46766	idle	statement: SELECT upper(setting) FROM pg_settings WHERE name='syslog_facility';

The **Audit Log Dashboard** header includes the date and time that the page was last updated, and a current count of triggered alerts.

Audit Log table entries are loaded on demand in batches; to load additional entries, scroll to the end of the log and the additional rows will be automatically loaded from the database and added to the table. Log entries are shown in chronological order, most recent first.

- The **Id** column identifies the PEM agent that monitors the server that initiated the recorded transaction.
- The **Server** column identifies the server that initiated the recorded transaction.
- The **Timestamp** column shows the date and time that the log entry was made.
- The **User Name** column shows the user which executed the statement in the audit log entry.
- The **Database Name** column shows the database on which the statement in the audit log entry was executed.
- The **Process ID** column shows the ID of the process which executed the statement in the audit log entry.
- The **Session ID** column shows the ID of the session in which the statement in the audit log entry was executed.
- The **Transaction ID** column shows the ID of the transaction in which the statement in the audit log entry was executed.
- The **Connection From** column shows the client's address from where the session was connected.
- The **Command** column shows the type of command executed.
- The **Message** column shows the message associated with the audit log entry.

Click **Show Filters** to display a panel that you can use to filter the audit log entries that are shown in the table below; click on **Hide Filters** to close the panel.

Use the fields within the filter definition box to describe a selection criteria that PEM will use to select a subset of a report for display:

- Use the date and time selectors in the **From** field to specify a starting date and time for the displayed log entries.
- Use the date and time selectors in the **To** field to specify an ending date and time for the displayed log entries.
- Enter a username in the **Username** field to show log entries for the specified user only.
- Enter a database name in the **Database** field to show log entries for the specified database only.
- Enter a command type (for example; 'SELECT', 'authentication' or 'idle') in the **Command Type** field to show log entries of that type only.

7.4.1.3 The Database Analysis Dashboard

The Database Analysis dashboard provides a high-level overview of database activity for the selected database, including a comparative storage analysis of the 5 largest tables/indexes, user activity analysis, weekly I/O analysis, and an activity analysis of the tables that reside in the selected database.



Use parameters on the [PEM Server Configurations](#) dialog to specify the auto-refresh rate for the dashboard. To access the [Server Configuration](#) dialog, select [Server Configuration...](#) from the PEM web interface [Management](#) menu.

The Database Analysis dashboard header displays the date and time that the server started, the date and time that the Database Analysis dashboard was last updated, and the number of alerts currently triggered for the specified database.

(and monitored objects that reside within that database).

The **Storage** bar graph plots the relative size of the 5 largest tables and indexes that reside within the selected database. The vertical key on the left side of the graph indicates each table or index in megabytes; the key on the right side of the chart identifies the tables and indexes by name.

The **Users** section of the Database Analysis dashboard displays information about user connections:

- The **User Activity** graph plots the active and idle connections over the previous week. The vertical key on the left side of the chart indicates the connection count.
- The **Connection Overview** chart provides a comparative display of the active and idle connections currently established with the server (when the most recent probe executed).

The graphs in the **I/O** section present an analysis of I/O activity over the previous week.



- The **Database I/O** graph plots the number of blocks found in cached memory and the number of blocks read from disk over the previous week. The vertical key on the left side of the graph indicates number of blocks hit.
- The **Row Activity** graph displays the row activity for tables residing within the database over the previous week.

The vertical key on the left side of the graph indicates the number of rows.

- The **Commits/Rollbacks** graph displays the number of transactions committed and rolled back within the selected database over the previous week.

The vertical key on the left side of the graph indicates the transaction count.

Top Tables												
Schema	Table Name	Scans	Rows Read	Index Scans	Index Rows Read	Rows Inserted	Rows Updated	Rows Deleted	Hot Rows Updated	Total Rows	Dead Rows	
pemdata	cp_81	1	0	2561	0	0	0	0	0	0	0	
pem	bart_host_public_key	2	0			0	0	0	0	0	0	
pem	bart_log	2	0			0	0	0	0	0	0	
pemhistory	cp_81	2	0	0	0	0	0	0	0	0	0	
pem	ark_api_version	3	8	0	0	12	0	0	0	4	8	
pemdata	mvview_frozenxid	3	0	622	12	1	6	0	6	1	6	
pemdata	slony_replication	3	0	25643	0	0	0	0	0	0	0	
pem	bart_default_config	3	56	0	0	84	0	0	0	28	0	
pem	ark_server_option	3	0	0	0	0	0	0	0	0	0	
pemdata	oc_tablespace	3	20	8922	17832	32	0	0	0	12	20	
pem	agent_runtime	3	4	152803	152801	28	0	22	0	2	26	
pemdata	auto_discover_servers	3	4	129	153	8	0	2	0	2	6	
pemdata	mvview_bloat	3	0	5700	0	0	0	0	0	0	0	
pemdata	efm_cluster_info	3	0	757	0	0	0	0	0	0	0	
pemdata	cp_79	3	0	0	0	0	0	0	0	0	0	
pem	tbl_chart	3	26	108318	278	40	0	1	0	13	27	
pem	server_version	3	50	574	830	75	0	0	0	25	50	
pem	dashboard_settings	3	0	173990	0	0	0	0	0	0	0	
pemdata	streaming_replication_db_conflicts	3	0	0	0	0	0	0	0	0	0	
pemdata	number_of_prepared_transactions	3	10	56941	56482	16	0	0	0	6	10	
pemdata	function_statistics	3	0	5585	0	0	0	0	0	0	0	
pem	chart_metric	3	46	66752	89143	38	0	4	0	32	6	
pem	bart_server_config	3	0	0	0	0	0	0	0	0	0	
pemdata	cp_80	3	0	0	0	0	0	0	0	0	0	
pem	probe_config_view	3	0	54	0	0	0	0	0	0	0	

The **Hot Tables** table provides a detailed analysis of the activity for each table that resides within the selected database. Click a column heading to sort the table by the values within the column; click again to reverse the sort order.

- The **Schema** column identifies the schema in which the table resides.
- The **Table Name** column identifies the name of the table.
- The **Scans** column displays the number of scans performed on the table.
- The **Rows Read** column displays the number of rows read from the specified table.
- The **Index Scans** column displays the number of index scans performed on the specified table.
- The **Index Rows Read** column displays the number of rows read during index scans on the specified table.
- The **Rows Inserted** column displays the number of rows inserted into the specified table.
- The **Rows Updated** column displays the number of rows updated in the specified table.
- The **Rows Deleted** column displays the number of rows deleted from the specified table.
- The **Hot Rows Updated** column displays the number of hot row updates into the table; when a hot row update occurs, the new row occupies the same page as the previous row.
- The **Total Rows** column displays the number of total rows in the table.
- The **Dead Rows** column displays the number of rows that have been deleted, but have not been reclaimed via a VACUUM command or the AUTOVACUUM process.

7.4.1.4 The Global Overview Dashboard

Upon connecting to Postgres Enterprise Manager, the web interface displays the **Global Overview** dashboard. The Global Overview dashboard displays the status of each PEM server and agent, and calls your attention to any triggered alerts on monitored objects.

Use parameters on the [PEM Server Configurations](#) dialog to specify the auto-refresh rate for the dashboard. To access the [Server Configuration](#) dialog, select [Server Configuration...](#) from the PEM web interface [Management](#) menu.

The [Global Overview](#) header displays the date and time that the overview was last updated and the current number of triggered alerts.

The [Enterprise Dashboard](#) bar graph provides an at-a-glance overview of the status of your PEM agents and servers.

The [Agent Status](#) table provides detailed information about the status of each individual PEM agent:

- Check the box in the [Blackout](#) column to disable alert processing for the agent and all servers monitored by the agent. This is useful when undertaking maintenance on the agent or the host on which the agent runs.
- The [Status](#) column reports the current state of the agent; [UP](#), [DOWN](#) or [UNKNOWN](#). A healthy agent displays a green 'check' circle icon; an agent that is down displays a red 'info' circle icon; an agent registered with PEM, but never sent an heartbeat, displays a gray 'question' circle icon. If user changes the colour for UP, DOWN or UNKNOWN status of agents in **Enterprise Dashboard** bar chart, then that color will be reflected for the respective status icon.
- The [Name](#) column displays the name of the agent. Click the name to navigate to the [Operating System Analysis](#) dashboard for the selected host.

- The **Alerts** column displays the number of current alerts triggered on the server.
- The **Version** column displays the agent's version.
- The **Processes** column lists the number of processes running on the agent's host.
- The **Threads** column lists the number of threads running on the agent's host.
- The **CPU Utilisation (%)** column shows the average utilisation of all CPU cores on the host.
- The **Memory Utilisation (%)** column shows the percentage of available RAM memory used on the host.
- The **Swap Utilisation (%)** column shows the percentage of available swap memory used on the host.
- The **Disk Utilisation** column shows the total percentage of disk space used, for all disks on the host.

The **Postgres Server Status** table provides detailed information about the status of each individual server:

- Check the box in the **Blackout** column provides a checkbox to disable alert processing for the server. This is useful when performing maintenance on the server.
- The **Status** column reports the current state of the server; UP, DOWN, UNKNOWN or UNMANAGED.
- A healthy server displays a green 'check' circle icon; a disabled server displays a red 'info' circle icon; an unknown server displays a grey 'question' circle icon; an unmanged server displays a light gray 'user' circle icon. If user changes the colour for UP, DOWN, UNKNOWN or UNMANAGED servers in Enterprise Dashboard bar chart, then that color will be reflected for the respective status icon.
- The **Name** column displays the name of the agent. Click the name to navigate to the **Operating System Analysis** dashboard for the host.
- The **Connections** column reports the current number of connections to the server.
- The **Alerts** column displays the number of current alerts triggered on the server.
- The **Version** column lists the PostgreSQL version and build signature.
- The **Remotely Monitored** column displays a **Yes** if the PEM agent that is bound to the monitored server does not reside on the same host as the server, and a **No** if the agent resides on the same host as the server.

Triggered alerts displayed in the **Alert Status** table include both PEM-defined alerts and user-defined alerts for all PEM-monitored hosts, servers, agents and database objects. The **Alert Status** table will also display an alert if an [agent or server is down](#).

- The **Alarm Type** column reports the alert severity. An icon displays in red for a **High** severity alert, in yellow for a **Medium** severity alert, and in grey for a **Low** severity alert.
- The **Object Description** column displays a description of the object that triggered the alert.
- The **Alert Name** column displays the name of the triggered alert. When viewing the dashboard in the PEM client, you can click the Alert Name to open the configuration dialogue for the alert.
- The **Value** column displays the current value of the object that triggered the alert.
- The **Database** column displays the name of the database with which the alert is associated (if applicable).
- The **Schema** column displays the name of the schema with which the alert is associated (if applicable).
- The **Package** column displays the name of the package with which the alert is associated (if applicable).
- The **Object** column displays the name of the object with which the alert is associated (if applicable).
- The **Additional Params** column displays any additional parameters specified for the alert.
- The **Additional Param Values** column displays any additional parameter values specified for the alert.
- The **Alerting Since** column displays the date and time at which the alert triggered.

7.4.1.5 The I/O Analysis Dashboard

The I/O Analysis dashboard displays usage statistics for a specific database.





pem	probe_column	probe_parameter_unique	512246	2214060	148323	81	1024971
-----	--------------	------------------------	--------	---------	--------	----	---------

Use parameters on the [PEM Server Configurations](#) dialog to specify the auto-refresh rate for the dashboard. To access the [Server Configuration](#) dialog, select [Server Configuration...](#) from the PEM web interface [Management](#) menu.

The I/O Analysis dashboard header displays the date and time that the server started, the date and time that the I/O Analysis dashboard was last updated, and the number of alerts currently triggered for the specified database (and any monitored object that resides within that database).

The graphs in the [I/O Overview](#) provide information about the week's activity for the specified database:

- The [Database I/O](#) graph displays the number of blocks read to and written from disk and memory buffers for the specified database over the course of the previous week.

The vertical key on the left side of the graph charts the block count.

- The [Row Activity](#) graph displays tuple activity for tables residing within the database over the last week.

The vertical key on the left side of the graph charts the row count.

- The [Checkpoints](#) graph displays the number of timed and untimed (requested) checkpoints written for the database over the last week.

The vertical key on the left side of the graph displays the checkpoint count.

A checkpoint is a point in the transaction logging sequence at which all data files have been updated to reflect the information in the log, and data files are flushed to disk. Checkpoints can be automatically generated, or forced by use of the CHECKPOINT command. A timed checkpoint occurs when the checkpoints_timeout parameter time limit is met. An untimed (requested) checkpoint occurs when the checkpoint_segments parameter is met, or when a superuser issues the CHECKPOINT command. Frequent checkpointing can impose extra load on the server, but can reduce recovery time in the event of a crash or hardware failure.

The [Hot Tables/Indexes](#) section of the I/O Analysis dashboard provides an overview of the 5 most scanned tables and indexes that reside within the database.

- The [Hot Tables](#) bar graph represents the comparative usage of the 5 most scanned tables that reside in the database; a vertical key displays the number of table scans.
- The [Hot Indexes](#) bar graph represents the comparative usage of the 5 most scanned indexes that reside in the database; a vertical key displays the number of index scans.

The [Object I/O Details](#) section of the I/O Analysis dashboard provides tables that display the table and index activity for the selected database.

The [Tables Activity](#) table provides a detailed analysis of the activity for the 20 most active tables that reside within the database. Click a column heading to sort the table by the values within the column; click again to reverse the sort order.

Object I/O Details												
Objects Activity												
Schema	Table Name	Scans	Rows Read	Index Scans	Index Rows Read	Rows Inserted	Rows Updated	Rows Deleted	Hot Rows Updated	Total Rows	Dead Rows	
pemdata	cp_81	1	0	2561	0	0	0	0	0	0	0	
pemhistory	cp_81	2	0	0	0	0	0	0	0	0	0	
pem	bart_log	2	0			0	0	0	0	0	0	
pem	bart_host_public_key	2	0			0	0	0	0	0	0	
pem	data_chart	3	6	267	250	10	0	1	0	3	7	
pem	bart_config	3	0	0	0	0	0	0	0	0	0	
pemdata	cp_80	3	0	0	0	0	0	0	0	0	0	
pemdata	function_statistics	3	0	5600	0	0	0	0	0	0	0	
pem	bart_server_config	3	0	0	0	0	0	0	0	0	0	
pem	bart_default_config	3	56	0	0	84	0	0	0	28	0	
pemdata	number_of_prepared_transactions	3	10	57076	56617	16	0	0	0	6	10	
pem	tbl_chart	3	26	108500	284	40	0	1	0	13	27	
pemdata	oc_tablespace	3	20	8937	17862	32	0	0	0	12	20	
pem	ark_server_option	3	0	0	0	0	0	0	0	0	0	
pem	server_version	3	50	575	855	75	0	0	0	25	50	
pem	probe_config_view	3	0	54	0	0	0	0	0	0	0	
pemdata	cp_79	3	0	0	0	0	0	0	0	0	0	
pem	dashboard_settings	3	0	174298	0	0	0	0	0	0	0	
pemdata	streaming_replication_db_conflicts	3	0	0	0	0	0	0	0	0	0	
pemdata	efm_cluster_info	3	0	803	0	0	0	0	0	0	0	

- The **Schema** column identifies the schema in which the table resides.
- The **Table Name** column identifies the name of the table.
- The **Scans** column displays the number of scans performed on the table.
- The **Rows Read** column displays the number of rows read from the specified table.
- The **Index Scans** column displays the number of index scans performed on the specified table.
- The **Index Rows Read** column displays the number of rows read during index scans on the specified table.
- The **Rows Inserted** column displays the number of rows inserted into the specified table.
- The **Rows Updated** column displays the number of rows updated in the specified table.
- The **Rows Deleted** column displays the number of rows deleted from the specified table.
- The **Hot Rows Updated** column displays the number of hot row updates for the table; when a hot row update occurs, the new row occupies the same page as the previous row.
- The **Total Rows** column displays the number of total rows in the table.
- The **Dead Rows** column displays the number of rows that have been deleted, but have not been reclaimed via a VACUUM command or the AUTOVACUUM process.

The **Indexes Activity** table provides a detailed analysis of the activity for the 20 most active indexes. Click a column heading to sort the table by the values within the column; click again to reverse the sort order.

- The **Schema** column identifies the schema in which the index resides.
- The **Table Name** column identifies the name of the table on which the index is defined.
- The **Index Name** column displays the name of the index.
- The **Scans** column displays the number of index scans performed on the specified table.
- The **Rows Read** column displays the number of tuples read during index scans on the specified table.
- The **Rows Fetched** column displays the number of tuples fetched by index scans.
- The **Blocks Read** column displays the number of index blocks read.
- The **Blocks Hit** column displays the number of index blocks hit.

7.4.1.6 The Memory Analysis Dashboard

The **Memory Analysis** dashboard provides an overview of the memory usage for the selected server and server host

for the previous week:



Use parameters on the [PEM Server Configurations](#) dialog to specify the auto-refresh rate for the dashboard. To access the [Server Configuration](#) dialog, select [Server Configuration...](#) from the PEM web interface [Management](#) menu.

The Memory Analysis dashboard header displays the date and time that the server was started, the date and time that the dashboard was last updated and the number of current alerts for objects monitored by the PEM server.

The [Database Server](#) section displays memory usage trends for the selected server.

- The [Server Memory Activity](#) graph displays the previous week's activity on the server; the [Legend](#) at the bottom of the graph provide a key to the colors used to chart information for each database. A vertical key on the left side of the graph indicates the actual block count for each value.
- The [Server Memory Configuration](#) pie chart displays the current memory usage (in megabytes).

The [Host](#) section displays the free and used memory on the host system:

- The [Host Memory Activity](#) chart plots the free and used memory on the host system over the last week.
- Sections of the [Host Memory Configuration](#) pie chart represent the free and available memory on the host system when the last probe executed.

7.4.1.7 The Objects Activity Analysis Dashboard

The Objects Activity Analysis dashboard provides an overview of the size and activity of the objects that reside within the selected database.



		Performance Metrics						Object Activity						Object Storage					
Schema	Table Name	Scans	Rows Read	Index Scans	Index Rows Read	Rows Inserted	Rows Updated	Rows Deleted	Hot Rows Updated	Total Rows	Dead Rows	Object Type	Table Size (MB)	Index Size (MB)	Total(MB)				
pemdata	cp_81	1	0	2561	0	0	0	0	0	0	0	Index	11	11	11				
pemhistory	cp_81	2	0	0	0	0	0	0	0	0	0	Index	10	10	10				
pem	bart_log	2	0				0	0	0	0	0	Index	36	36	36				
pem	bart_host_public_key	2	0			0	0	0	0	0	0	Index	36	36	36				
pem	data_chart	3	6	267	250	10	0	1	0	3	7	Table	16	0	16				
pem	bart_config	3	0	0	0	0	0	0	0	0	0	Index	0	0	0				
pemdata	cp_80	3	0	0	0	0	0	0	0	0	0	Index	0	0	0				
pemdata	function_statistics	3	0	5600	0	0	0	0	0	0	0	Table	0	0	0				
pem	bart_server_config	3	0	0	0	0	0	0	0	0	0	Index	0	0	0				
pem	bart_default_config	3	56	0	0	84	0	0	0	28	0	Table	0	0	0				
pemdata	number_of_prepared_transactions	3	10	57076	56617	16	0	0	0	6	10	Table	16	0	16				
pem	tbl_chart	3	26	108500	284	40	0	1	0	13	27	Table	0	0	0				
pemdata	oc_tablespace	3	20	8937	17862	32	0	0	0	12	20	Table	32	0	32				
pem	ark_server_option	3	0	0	0	0	0	0	0	0	0	Table	0	0	0				
pem	server_version	3	50	575	855	75	0	0	0	25	50	Table	0	0	0				
pem	probe_config_view	3	0	54	0	0	0	0	0	0	0	Table	0	0	0				
pemdata	cp_79	3	0	0	0	0	0	0	0	0	0	Table	0	0	0				
pem	dashboard_settings	3	0	174298	0	0	0	0	0	0	0	Table	0	0	0				
pemdata	streaming_replication_db_conflicts	3	0	0	0	0	0	0	0	0	0	Table	0	0	0				
pemdata	efm_cluster_info	3	0	803	0	0	0	0	0	0	0	Table	0	0	0				
Object Storage																			
Schema	Object			Object Type		Table Size (MB)		Index Size (MB)		Total(MB)									
pemhistory	session_info_keyidx	Index						11		11									
pemdata	oc_function_pkey	Index						10		10									
pemdata	server_logs_command_tag_idx	Index						36		36									
pemdata	server_logs_error_severity_idx	Index						36		36									
pemdata	server_logs_logtime_by_minutes_idx	Index						38		38									
pemdata	server_logs_log_time_idx	Index						38		38									
pemdata	server_logs_pkey	Index						38		38									
pemdata	server_logs_server_id_idx	Index						36		36									
pemhistory	index_size_keyidx	Index						1		1									
pemhistory	index_statistics_keyidx	Index						65		65									
pemhistory	index_statistics_timeidx	Index						26		26									
pemhistory	lock_info_keyidx	Index						26		26									
pemhistory	lock_info_timeidx	Index						14		14									
pemhistory	oc_function_keyidx	Index						21		21									
pemhistory	oc_function_timeidx	Index						4		4									
pemhistory	session_info_timeidx	Index						8		8									
pemhistory	session_waits_keyidx	Index						1		1									
pemhistory	table_frozenxid_keyidx	Index						8		8									
pemhistory	table_frozenxid_timeidx	Index						3		3									
pemhistory	table_statistics_keyidx	Index						136		136									
pemhistory	table_statistics_timeidx	Index						71		71									
pemhistory	table_statistics	Table						904		208									
pemdata	server_logs	Table						727		221									
pemhistory	index_statistics	Table						151		92									
pemhistory	session_info	Table						118		20									
pemhistory	lock_info	Table						49		40									
pem	probe_log	Table						47		47									
pemhistory	oc_function	Table						22		25									
pemdata	oc_function	Table						10		10									
pemhistory	table_frozenxid	Table						10		11									
pemhistory	oc_views	Table						8		1									
pemdata	oc_views	Table						4		0									
pemhistory	session_waits	Table						3		2									
pemhistory	background_writer_statistics	Table						3		1									
pemhistory	database_statistics	Table						3		1									
pemhistory	server_info	Table						2		0									
pemhistory	log_configuration	Table						2		0									
pemhistory	index_size	Table						1		1									
pemhistory	cpu_usage	Table						1		1									
pemhistory	network_statistics	Table						1		0									
pemhistory	oc_index	Table						1		1									
pemhistory	memory_usage	Table						1		0									
pemdata	table_statistics	Table						1		0									
pemhistory	table_bloat	Table						1		0									
pem	jobsteplog	Table						0		0									
pemhistory	table_size	Table						0		0									
pemdata	index_size	Table						0		0									

pemdata	index_statistics	Table	0	0	1
pemhistory	user_info	Table	0	0	1
pemhistory	oc_table	Table	0	0	1

Use parameters on the [PEM Server Configurations](#) dialog to specify the auto-refresh rate for the dashboard. To access the [Server Configuration](#) dialog, select [Server Configuration...](#) from the PEM web interface [Management](#) menu.

The Objects Activity Analysis dashboard header displays the date and time that the server started, the date and time that the Object Activity Analysis dashboard was last updated, and the number of alerts currently triggered for the specified database (and monitored objects that reside within that database).

The bar graphs in the [Size Overview](#) section plot the comparative sizes of the 5 largest tables and indexes that reside within the selected database:

- The [Top 5 Largest Tables](#) bar graph represents the comparative sizes of the 5 largest tables that reside in the database; a vertical key displays the table size in megabytes.
- The [Top 5 Largest Indexes](#) bar graph represents the comparative sizes of the 5 largest indexes that reside in the database; a vertical key displays the index size in megabytes.

The [Objects Activity](#) table provides a detailed analysis of the activity for each table that resides within the database. Click a column heading to sort the table by the values within the column; click again to reverse the sort order.

- The [Schema](#) column identifies the schema in which the specified table resides.
- The [Table Name](#) column identifies the name of the table.
- The [Scans](#) column displays the number of scans performed on the table.
- The [Rows Read](#) column displays the number of rows read from the specified table.
- The [Index Scans](#) column displays the number of index scans performed on the specified table.
- The [Index Rows Read](#) column displays the number of rows read during index scans on the specified table.
- The [Rows Inserted](#) column displays the number of rows inserted into the specified table.
- The [Rows Updated](#) column displays the number of rows updated in the specified table.
- The [Rows Deleted](#) column displays the number of rows deleted from the specified table.
- The [Hot Rows Updated](#) column displays the number of hot row updates into the table; when a hot row update occurs, the new row occupies the same page as the previous row.
- The [Total Rows](#) column displays the number of total rows in the table.
- The [Dead Rows](#) column displays the number of rows that have been deleted, but have not been reclaimed via a VACUUM command or the AUTOVACUUM process.

The [Objects Storage](#) table displays the schema objects that reside in the selected database. Click a column heading to sort the table data by the values within that column; click again to reverse the sort order.

Object Storage					
Schema	Object	Object Type	Table Size (MB)	Index Size (MB)	Total(MB)
pemhistory	session_info_keyidx	Index		11	11
pemdata	oc_function_pkey	Index		10	10
pemdata	server_logs_command_tag_idx	Index		36	36
pemdata	server_logs_error_severity_idx	Index		36	36
pemdata	server_logs_logtime_by_minutes_idx	Index		38	38
pemdata	server_logs_log_time_idx	Index		38	38
pemdata	server_logs_pkey	Index		38	38
pemdata	server_logs_server_id_idx	Index		36	36
pemhistory	index_size_keyidx	Index		1	1
pemhistory	index_statistics_keyidx	Index		65	65
pemhistory	index_statistics_timeidx	Index		26	26
pemhistory	lock_info_keyidx	Index		26	26
pemhistory	lock_info_timeidx	Index		14	14
pemhistory	oc_function_keyidx	Index		21	21
pemhistory	oc_function_timeidx	Index		4	4
pemhistory	session_info_timeidx	Index		8	8
pemhistory	session_waits_keyidx	Index		1	1
pemhistory	table_frozenxid_keyidx	Index		8	8
pemhistory	table_frozenxid_timeidx	Index		3	3
pemhistory	table_statistics_keyidx	Index		136	136
pemhistory	table_statistics_timeidx	Index		71	71
pemhistory	table_statistics	Table	904	208	1113
pemdata	server_logs	Table	727	221	949
pemhistory	index_statistics	Table	151	92	243
pemhistory	session_info	Table	118	20	139
pemhistory	lock_info	Table	49	40	90
pem	probe_log	Table	47	0	47
pemhistory	oc_function	Table	22	25	47
pemdata	oc_function	Table	10	10	21
pemhistory	table_frozenxid	Table	10	11	22
pemhistory	oc_views	Table	8	1	10
pemdata	oc_views	Table	4	0	4
pemhistory	session_waits	Table	3	2	6
pemhistory	background_writer_statistics	Table	3	1	4
pemhistory	database_statistics	Table	3	1	5
pemhistory	server_info	Table	2	0	3
pemhistory	log_configuration	Table	2	0	3
pemhistory	index_size	Table	1	1	2
pemhistory	cpu_usage	Table	1	1	3
pemhistory	network_statistics	Table	1	0	2
pemhistory	oc_index	Table	1	1	2
pemhistory	memory_usage	Table	1	0	1
pemdata	table_statistics	Table	1	0	2
pemhistory	table_bloat	Table	1	0	2
pem	jobsteplog	Table	0	0	1
pemhistory	table_size	Table	0	0	1
pemdata	index_size	Table	0	0	1
pemdata	index_statistics	Table	0	0	1
pemhistory	user_info	Table	0	0	1
pemhistory	oc_table	Table	0	0	1

- The **Schema** column identifies the schema in which the object resides.
- The **Object** column identifies the name of the schema object.
- The **Object Type** column identifies the type of schema object (Table or Index).
- The **Table Size** column lists the size of the table in megabytes (if applicable).
- The **Index Size** column lists the size of the index (or associated index) in megabytes (if applicable).
- The **Total (MB)** column lists the cumulative size (in megabytes) of the specified table and/or indexes and associated TOAST tables.

7.4.1.8 The Operating System Analysis Dashboard

The **Operating System Analysis** dashboard provides a graphical analysis of the resource usage on the system hosting the selected agent.



Use parameters on the [PEM Server Configurations](#) dialog to specify the auto-refresh rate for the dashboard. To access the [Server Configuration](#) dialog, select [Server Configuration...](#) from the PEM web interface [Management](#) menu.

The [Operating System Analysis](#) dashboard header displays the date and time that the server was last booted, the date and time that the display was last updated, and the number of triggered alerts on the system.

The Operating System Analysis dashboard provides an overview of system resources. Within the [OS Overview](#) section:

- The [CPU](#) graph represents the percentage of the CPU used at a given point in time. The vertical key on the left side of the graph indicates the percentage.
- Segments of the [Storage](#) pie chart represent the free and used storage on the host.
- The [Memory](#) graph displays the memory usage on the PEM server.
- The [Process](#) graph plots the number of processes on the system. A vertical key on the left side of the graph displays the process count.

The [Disk](#) section of the [Operating System Analysis](#) dashboard displays charts and information about operating system disk usage.



- The [Disk](#) graph displays the amount of disk space used. The vertical key on the left side of the chart displays the amount of disk space used (in Megabytes). Each horizontal line on the graph represents a different mounted file system; a file system key is provided in the [Legend](#).
- The [I/O](#) graph displays the blocks read from and written to disk. A vertical key on the left side of the graph provides a block count.
- The [Host File System Details](#) table provides information about the host file system:
 - The [File System](#) column displays the name of the file system.
 - The [Size \(GB\)](#) column displays the size of the file system in Gigabytes.
 - The [Used \(GB\)](#) column displays the amount of the file system that is currently storing information.
 - The [Available \(GB\)](#) column displays the amount of space still available on the file system.
 - The [% Used](#) column displays the percentage of the total storage space in use.
 - The [Mounted On](#) column displays the directory or drove on which the file system is mounted.

Graphs in the **Network** section of the **Operating System Analysis** dashboard plot the network and packet traffic:



- The **Packets** graph displays the number of packets sent and received across the network. The **Legend** provides a key to the color charted for each network interface. The vertical key on the left side of the graph indicates the packet count.
- The **Traffic** graph displays the amount of data transferred across the network. The **Legend** provides a key to the color charted for each network interface. The vertical key on the left side of the graph displays the traffic, in KB.

Please note: The network bandwidth may not display if the monitored server is a Linux platform that resides in a virtual machine. This is expected behavior.

7.4.1.9 The Probe Log Analysis Dashboard

The Probe Log Analysis dashboard displays error messages from the PEM agent.

Probe Logs				
ID	Timestamp	Probe Name	Server	Error Message
116854	29/04/2020, 10:05:59	Object Catalog: View	Postgres Enterprise Manager Server	ERROR: canceling statement due to statement timeout
116849	29/04/2020, 10:05:38	Object Catalog: View	Postgres Enterprise Manager Server	ERROR: canceling statement due to statement timeout
116765	29/04/2020, 10:03:39	Object Catalog: Function	Postgres Enterprise Manager Server	ERROR: canceling statement due to statement timeout
116764	29/04/2020, 10:03:37	Object Catalog: View	Postgres Enterprise Manager Server	ERROR: canceling statement due to statement timeout
116757	29/04/2020, 10:00:40	Object Catalog: View	Postgres Enterprise Manager Server	ERROR: canceling statement due to statement timeout
116677	29/04/2020, 09:56:04	Object Catalog: View	Postgres Enterprise Manager Server	ERROR: canceling statement due to statement timeout
116669	29/04/2020, 09:55:34	Object Catalog: View	Postgres Enterprise Manager Server	ERROR: canceling statement due to statement timeout
116637	29/04/2020, 09:54:03	Table Bloat	Postgres Enterprise Manager Server	ERROR: canceling statement due to statement timeout
116585	29/04/2020, 09:51:01	Object Catalog: Function	Postgres Enterprise Manager Server	ERROR: canceling statement due to statement timeout
116584	29/04/2020, 09:51:01	Object Catalog: View	Postgres Enterprise Manager Server	ERROR: canceling statement due to statement timeout
116578	29/04/2020, 09:50:40	Object Catalog: View	Postgres Enterprise Manager Server	ERROR: canceling statement due to statement timeout
116503	29/04/2020, 09:46:08	Table Statistics	Postgres Enterprise Manager Server	ERROR: canceling statement due to statement timeout
116497	29/04/2020, 09:45:57	Object Catalog: Function	Postgres Enterprise Manager Server	ERROR: canceling statement due to statement timeout
116496	29/04/2020,	Object Catalog: View	Postgres Enterprise	ERROR: canceling statement due to statement timeout

The header information includes the date and time that the server was first started, the date and time that the page was last updated, and the current number of triggered alerts.

Use parameters on the [PEM Server Configurations](#) dialog to specify the auto-refresh rate for the dashboard. To access the [Server Configuration](#) dialog, select [Server Configuration...](#) from the PEM web interface [Management](#) menu.

The [Probe Log](#) table displays error messages returned by the PEM Agent. Entries in the Probe Log table may reflect incorrect agent binding information or authentication errors between the PEM agent and the server.

- The [Id](#) column displays a unique identifier for each entry in the table.
- The [Timestamp](#) column displays the date and time that the log entry was made.
- The [Probe Name](#) column displays the name of the probe that recorded the log entry.
- The [Server Name](#) column displays the name of the server on which the error occurred.
- The [Error Message](#) column displays the error message returned by the probe.

7.4.1.10 The Server Analysis Dashboard

The Server Analysis dashboard provides a graphical analysis of a monitored server's usage statistics.



The Server Analysis dashboard header displays the date and time that the server was started, the date and time that the display was last updated, and the number of current alerts for items monitored by the PEM server.

Use parameters on the [PEM Server Configurations](#) dialog to specify the auto-refresh rate for the dashboard. To access the [Server Configuration](#) dialog, select [Server Configuration...](#) from the PEM web interface [Management](#) menu.

Graphs within the [Storage](#) section of the dashboard provide an analysis of the space consumed by databases and tablespaces on the server:

- The [Database Size](#) graph displays the size (in Megabytes) of the 5 largest databases that reside on the PEM

- server. The **Legend** at the bottom of the graph associates each database name with a color in the graph.
- The **Tablespace Size** graph displays the size (in Megabytes) of the 5 largest tablespaces that reside on the PEM server. The **Legend** at the bottom of the graph associates each tablespace name with a color in the graph.

The **Memory** section of the dashboard provides an overview of the efficiency of the buffer cache over the previous week, and an analysis of the current swap memory usage:

- The **Shared Buffers** chart compares the number of data blocks found in the shared memory cache with the number of blocks read from disk. A high hit-to-miss ratio indicates an efficiently configured memory cache.
- The **Host Memory** pie chart displays the current swap memory usage.

The **Users** section of the **Server Analysis** dashboard provides an overview of the user activity on the server:

- The **User Activity** chart displays connection statistics gathered over the last week. The **Legend** at the bottom of the chart provides a key to the data displayed.
- The **Connection Overview** pie chart compares the currently active connections to the currently idle connections.

The **I/O** section of the **Server Analysis** dashboard provides an overview of the transactions processed by the server over the last week:



- The **Disk** chart displays the number of 8KB blocks read from disk, and the number of 8KB blocks written to disk over the last week.
- The **Row Activity** chart plots row activity on tables stored on the server over the past week. The **Legend** at the bottom of the chart provides a key to the data displayed.
- The **Commits/Rollbacks** chart displays the number of transactions committed and rolled back on the selected server within the last week. A vertical count on the left side of the graph indicates the aborted transaction count, while the **Legend** at the bottom of the chart provides a key to the commits and rollbacks charted.

The **Database Analysis** table displays a list of the monitored databases that reside on the server, and the statistics gathered for each database over the last week. Click a column heading to sort the table by the data displayed in the column; click again to reverse the sort order.

Databases											
Databases Statistics											
Database	Connections	TX Committed	TX Rolled Back	Blocks Hit	Blocks Read	Tuples Fetched	Tuples Returned	Tuples Inserted	Tuples Updated	Tuples Deleted	
postgres	3	1148192	22	139399784	5982	61900078	61900078	67	15	0	
pem	16	13239743	5840	7607117142	12978313	1475568227	1475568227	7506885	7929549	460295	
hr	0	67207	1621	13155426	1632	5505911	5505911	0	0	0	
edbstore_temp	0	69973	0	14834311	1453	6529192	6529192	32	1	0	
db01	0	115198	29	15536425	2361	6817777	6817777	374	28	12	
testdb	0	160130	0	11070985	6764	2573029	2573029	238866	416394	0	

- The **Database** column displays the database name.
- The **Connections** column displays the number of current connections to the database.
- The **TX Committed** column displays the number of transactions committed to the database within the last week.
- The **TX Rolled Back** column displays the number of transactions rolled back within the last week.
- The **Blocks Hit** column displays the number of blocks hit in the cache (in megabytes) within the last week.
- The **Blocks Read** column displays the number of blocks read from memory (in megabytes) within the last week.
- The **Tuples Fetched** column displays the number of tuples fetched within the last week.
- The **Tuples Returned** column displays the number of tuples returned within the last week.
- The **Tuples Inserted** column displays the number of tuples inserted into the database within the last week.
- The **Tuples Updated** column displays the number of tuples updated in the database within the last week.
- The **Tuples Deleted** column displays the number of tuples deleted from the database within the last week.

7.4.1.11 The Server Log Analysis Dashboard

The **Server Log Analysis** dashboard displays the log files for the selected server. To view the **Server Log Analysis** dashboard, right-click on the name of a monitored server in the PEM client tree control, and navigate through the **Dashboards** menu, selecting **Server Log Analysis**.



The screenshot shows the 'Server Log' dashboard for a PostgreSQL server. The header includes navigation tabs: Properties, SQL, Statistics, Dependencies, Dependents, Monitoring, and Server Log. Below the header, there's a status bar with 'Object Type: Server', 'Status: UP (Since: 27/04/2020, 15:47:09)', 'Generated On: 29/04/2020, 10:20:00', and 'No. of alerts: 6 (Acknowledged: 0)'. The main area is titled 'Server Logs' and contains a table with columns: id, Timestamp, User Name, Database Name, Process ID, Session ID, Transaction ID, Connection From, Command, and Message. The table lists several log entries, each with a timestamp, user, database, process ID, session ID, transaction ID, connection details, command type (e.g., COPY, COMMIT, UPDATE), and a detailed message about the log entry.

id	Timestamp	User Name	Database Name	Process ID	Session ID	Transaction ID	Connection From	Command	Message
1870601	29/04/2020, 10:15:06	agent1	pem	64373	5ea8fb6.fb75	5002111	127.0.0.1:55512	COPY	duration: 0.187 ms statement: BEGIN;COPY pemdata.server_logs(server_id, log_time, user_name, database_name, process_id, connection_from, session_id, session_line_num, command_tag, session_start_time, virtual_transaction_id, transaction_id, error_severity, sql_state_code, message, detail, hint, internal_query, internal_query_pos, context, query, query_pos, location, application_name) FROM STDIN WITH NULL AS 'NULL' QUOTE " CSV;
1870600	29/04/2020, 10:15:06	agent1	pem	64373	5ea8fb6.fb75	0	127.0.0.1:55512	COMMIT	duration: 0.572 ms statement: END;
1870599	29/04/2020, 10:15:06	agent1	pem	64373	5ea8fb6.fb75	5002110	127.0.0.1:55512	UPDATE	duration: 0.129 ms statement: UPDATE pem.log_configuration SET (last_read_filename, file_offset) = ('/var/lib/pgsql/12/data/log/postgresql-2020-04-29_004252.csv', 3135795) WHERE server_id = 1;
1870598	29/04/2020, 10:15:06	agent1	pem	64373	5ea8fb6.fb75	5002110	127.0.0.1:55512	COPY	duration: 0.289 ms statement: BEGIN;COPY pemdata.server_logs(server_id, log_time, user_name, database_name, process_id, connection_from, session_id, session_line_num, command_tag, session_start_time, virtual_transaction_id, transaction_id, error_severity, sql_state_code, message, detail, hint, internal_query, internal_query_pos, context, query, query_pos, location, application_name) FROM STDIN WITH NULL AS 'NULL' QUOTE " CSV;
1870597	29/04/2020, 10:15:06	agent1	pem	64373	5ea8fb6.fb75	0	127.0.0.1:55512	COMMIT	duration: 0.521 ms statement: END;
1870596	29/04/2020, 10:15:06	agent1	pem	64373	5ea8fb6.fb75	5002109	127.0.0.1:55512	UPDATE	duration: 0.132 ms statement: UPDATE pem.log_configuration SET (last_read_filename, file_offset) = ('/var/lib/pgsql/12/data/log/postgresql-2020-04-29_004252.csv', 3134305) WHERE server_id = 1;
1870595	29/04/2020, 10:15:06	agent1	pem	64373	5ea8fb6.fb75	5002109	127.0.0.1:55512	COPY	duration: 0.200 ms statement: BEGIN;COPY pemdata.server_logs(server_id, log_time, user_name, database_name, process_id, connection_from, session_id, session_line_num, command_tag, session_start_time, virtual_transaction_id, transaction_id, error_severity, sql_state_code, message, detail, hint, internal_query, internal_query_pos, context, query, query_pos, location, application_name) FROM STDIN WITH NULL AS 'NULL' QUOTE " CSV;
1870594	29/04/2020, 10:15:06	agent1	pem	64373	5ea8fb6.fb75	0	127.0.0.1:55512	COMMIT	duration: 0.322 ms statement: END;

The header information on the **Server Log Analysis** dashboard displays the date and time that the server was started, the date and time that the page was last updated, and the current number of triggered alerts.

The **Server Log** table displays the contents of the log files that are stored on the PEM server. For content to be displayed, you must check the box next to **Import logs to PEM** when using Log Manager to configure logging for the server.

Entries are displayed in chronological order, most-recent log entries first. Use the scroll bars to navigate through the log entries, or to view columns that are off of the display.

Headings at the top of the server log table identify the information stored in each column:

- The **Id** column identifies the PEM agent that monitors the server that initiated the recorded transaction.
- The **Server** column identifies the server that initiated the recorded transaction.
- The **Timestamp** column displays the date and time that the log entry was made.
- The **User Name** column displays the name of the user that executed the recorded transaction.
- The **Database Name** column displays the name of the database on which the recorded transaction was executed.
- The **Process ID** column displays the identifier of the process that executed the recorded transaction.
- The **Session ID** column displays the identifier of the session in which the transaction was executed.
- The **Transaction ID** column displays the transaction identifier.
- The **Connection From** column displays the host name or IP address from which the client session connected.
- The **Command** column displays the type of command executed.
- The **Message** column displays the transaction message.

Click **Show Filters** to display a panel that you can use to filter the audit log entries that are shown in the table below; click on **Hide Filters** to close the panel.



Use the fields within the filter definition box to describe a selection criteria that PEM will use to select a subset of a report for display:

- Use the date and time selectors in the **From** field to specify a starting date and time for the displayed log entries.
- Use the date and time selectors in the **To** field to specify an ending date and time for the displayed log entries.
- Enter a username in the **Username** field to show log entries for the specified user only.
- Enter a database name in the **Database** field to show log entries for the specified database only.
- Enter a command type (for example; 'SELECT', 'authentication' or 'idle') in the **Command type** field to show log entries of that type only. Commands that will be displayed in the filtered report.

When you've described the criteria by which you wish to filter the audit logs, click **Filter** to display the filtered server log in the lower portion of the **Server Log Analysis** dashboard. Click the **Hide Filters** label to close the filter definition box.

7.4.1.12 The Session Activity Analysis Dashboard

The Session Activity Analysis dashboard provides information about the session workload and lock activity for the selected server:



The screenshot shows the 'Session Activity' section of the EDB Postgres Enterprise Manager interface. At the top, there are tabs for Properties, SQL, Statistics, Dependencies, Dependents, Monitoring, and Session Activity. Below the tabs, the title 'Postgres Enterprise Manager Host > Postgres Enterprise Manager Server > Session Activity' is displayed, along with a status message 'Status UP (Since: 27/04/2020, 15:47:09) Generated On 29/04/2020, 10:32:26' and 'No. of alerts 6 (Acknowledged: 0)'.

The main area is titled 'Work Load' and contains a table with the following columns: Session ID, User Name, Source, Database Name, Waiting?, Backend Start, Transaction Start, Query Start, Memory Usage, Swap Usage, CPU Usage, IO Reads (# bytes), and IO Writes (# bytes). The table lists 15 rows of session activity data.

Session ID	User Name	Source	Database Name	Waiting?	Backend Start	Transaction Start	Query Start	Memory Usage	Swap Usage	CPU Usage	IO Reads (# bytes)	IO Writes (# bytes)
72556	postgres	127.0.0.1:38696	postgres	Yes	2020-04-29 10:17:30.901977+05:30		2020-04-29 10:24:02.37053+05:30	5.277	2.816	0.000	171464.000	
93857	postgres	127.0.0.1:44690	postgres	No	2020-04-29 10:24:21.36282+05:30	2020-04-29 10:24:21.373882+05:30	2020-04-29 10:24:21.373882+05:30	7.637	2.816	0.000	171463.000	
37325	postgres	127.0.0.1:51258	pem	Yes	2020-04-27 16:17:42.962509+05:30		2020-04-28 16:46:59.926278+05:30	27.359	2.832	0.000	669633.000	
80541	postgres	127.0.0.1:51876	pem	Yes	2020-04-27 15:47:11.693572+05:30		2020-04-29 10:24:08.186452+05:30	27.484	3.254	0.000	964515.000	
80909	agent1	127.0.0.1:51886	pem	Yes	2020-04-27 15:47:19.122199+05:30		2020-04-29 10:24:11.98416+05:30	36.637	3.605	0.000	8018643.000	61
81830	agent1	127.0.0.1:51918	pem	Yes	2020-04-27 15:47:39.499742+05:30		2020-04-29 10:24:17.65002+05:30	5.402	3.480	0.000	275595.000	
81831	agent1	127.0.0.1:51920	pem	Yes	2020-04-27 15:47:39.515256+05:30		2020-04-29 10:24:17.649713+05:30	6.051	2.863	0.000	275595.000	
17375	postgres	127.0.0.1:55154	pem	Yes	2020-04-28 16:45:57.218333+05:30		2020-04-29 08:19:05.921713+05:30	28.441	2.805	0.000	546717.000	
64373	agent1	127.0.0.1:55512	pem	Yes	2020-04-29 09:30:06.515126+05:30		2020-04-29 10:24:21.372505+05:30	142.891	2.785	20.500	43399177.000	4
58913	postgres	192.168.1.13:54699	pem	Yes	2020-04-28 13:10:48.241955+05:30		2020-04-28 19:47:55.297426+05:30	43.063	5.043	0.000	5986609.000	
58914	postgres	192.168.1.13:54710	pem	Yes	2020-04-28 13:10:48.392445+05:30		2020-04-28 19:47:55.714573+05:30	34.492	5.387	0.000	792881.000	
115301	postgres	192.168.1.13:62921	pem	Yes	2020-04-29 09:02:44.994761+05:30		2020-04-29 09:02:45.575466+05:30	13.211	2.801	0.000	2996804.000	

The Session Activity Analysis dashboard header displays the date and time that the server was started, the date and time that the dashboard was last updated and the number of current alerts for the server.

Use parameters on the [PEM Server Configurations](#) dialog to specify the auto-refresh rate for the [Session Activity Analysis](#) dashboard. To access the [Server Configuration](#) dialog, select [Server Configuration...](#) from the PEM client [Management](#) menu.

The [Session Workload](#) table provides information about the current session workload for the server. Click a column heading to sort the table data by the selected column; click the heading a second time to reverse the sort order. The Session Workload table displays the following information:

- The [Session ID](#) column displays the process identifier for the session.
- The [User Name](#) column displays the (role) name of the user that established the client connection to the server.
- The [Source](#) column displays the IP address and port number of the client.
- The [Database Name](#) column displays the name of the database to which the client is connected.
- The [Waiting](#) column displays [Yes](#) if the session is waiting for a lock; [No](#) if the session is not waiting for a lock.
- The [Backend Start](#) column displays the date and time that the client established a connection to the server.
- The [Transaction Start](#) column displays the date and time that the current transaction started, if applicable.
- The [Query Start](#) column displays the date and time that the current query started, if applicable.
- The [Memory Usage](#) column displays the amount of memory used by the session; this column is not displayed if the server is remotely monitored.
- The [Swap Usage](#) column displays the amount of swap space used by the session; this column is not displayed if the server is remotely monitored.
- The [CPU Usage](#) column displays the amount of CPU resources used by the session; this column is not displayed if the server is remotely monitored.
- The [IO Reads \(#bytes\)](#) column displays the number of bytes used during read transactions the session; this column is not displayed if the server is remotely monitored.
- The [IO Writes \(#bytes\)](#) column displays the number of bytes used during write transactions the session; this column is not displayed if the server is remotely monitored.

The [Session Lock Activity](#) table displays a list of locks held by processes on the server. Click a column heading to sort the table data by the selected column; click the heading a second time to reverse the sort order. The Session Lock Activity table displays the following information:

- The **Session ID** column displays the process ID for the session.
- The **User Name** column displays the name of the user holding (or waiting for) the lock.
- The **Source** column displays the IP address and port number of the client.
- The **Database Name** column displays the name of the database to which the client is connected.
- The **Blocked** column indicates if the lock request is blocked by another lock.
- The **Blocked By** column specifies the session ID of the session that is holding the lock.
- The **Lock Type** column displays the type of lock that is held by the client. Lock Type may be:
 - advisory** - a user-defined lock created by pg_advisory_lock() or pg_advisory_lock_shared()
 - extend** - a lock held while extending a table or index
 - object** - a lock held on a database object
 - page** - a lock held on a page (within the shared buffer cache)
 - relation** - a lock held on the metadata describing a table, view, or sequence (to prevent another session from altering the table, view, or sequence)
 - transactionid** - a lock held on a transaction ID (one session typically waits for another transaction to complete by waiting on the other session's transaction ID)
 - tuple** - lock held on a tuple (typically, a tuple which has been inserted, updated, or deleted, but not yet committed)
 - userlock** - a user-defined lock created with the LOCK statement
 - virtualxid** - a lock identified by a virtual transaction ID.
- The **Object ID** column displays the OID of the relation, or NULL if the object is not a relation (of part of a relation).
- The **Mode** column displays the name of the lock mode held (or sought) by the process.
- The **Transaction Start** column displays the date and time that the transaction started.

7.4.1.13 The Session Waits Analysis Dashboard

The **Session Wait Analysis** dashboard provides an overview of the current DRITA wait events for an Advanced Server session. For more information about DRITA wait events, please see the EDB Postgres Advanced Server Guide.



Use parameters on the [PEM Server Configurations](#) dialog to specify the auto-refresh rate for the **Alerts** dashboard. To access the **Server Configuration** dialog, select **Server Configuration...** from the PEM web interface **Management** menu.

The Session Wait Analysis dashboard header displays the date and time that the server started, the date and time that the dashboard was last updated, and the number of alerts currently triggered for the specified database (and monitored objects that reside within that database).

The [Session Waits Overview](#) displays statistics gathered by the most recent execution of the PEM probe:

- The [Session Waits By Number Of Waits](#) pie chart displays the 5 most frequently encountered wait events, per Advanced Server session. For more information about the events that can cause a wait event, see the EDB Postgres Advanced Server Guide.
- The [Session Waits By Time Waited](#) pie chart displays the 5 wait events that consume the most time, per Advanced Server session. To gather and display data in the [Session Time Waits by Time Waited](#) pie chart, you must modify the `postgresql.conf` file for the monitored server, setting `timed_statistics = on`, and restart the server. Please note that this will cause server performance to degrade. For more information about using Advanced Server DRITA timers and the events that can cause a wait event, please see the EDB Postgres Advanced Server Guide.

The [Session Waits Details](#) table lists the current system wait events for the selected database. Click a column heading to sort the table by the column data; click again to reverse the sort order. The table displays:

- The [User](#) column displays the name of the user that encountered the wait.
- The [Wait Name](#) column displays the name of the wait event.
- The [Wait Count](#) column displays the total number of waits encountered by the user.
- The [Time \(ms\)](#) displays the number of milliseconds that the user waited for the specified event.
- The [Wait Time \(%\)](#) column displays the percentage of the total wait time consumed by the specified wait event.

To gather and display data in the Time (ms) and Wait Time (%) columns, you must modify the `postgresql.conf` file for the monitored server, setting `timed_statistics = on`, and restart the server. Please note that this will cause server performance to degrade. For more information about using Advanced Server DRITA timers, please see the EDB Postgres Advanced Server Guide.

7.4.1.14 The Storage Analysis Dashboard

The [Storage Analysis](#) dashboard provides information about the size of objects stored on the server and about available storage space on the server.

The screenshot displays the Storage Analysis dashboard in the EDB Postgres Enterprise Manager interface. The top navigation bar includes tabs for Properties, SQL, Statistics, Dependencies, Dependents, Monitoring, Storage, and Storage (selected). The header also shows the server name (Postgres Enterprise Manager Host - EPAS_12), status (UP), and alert count (4 acknowledged: 0).

Storage Overview:

- Database Overview:** A pie chart showing database sizes: 83.59% (red), 2.66% (blue), 6.65% (green), and 7.10% (yellow).
- Tablespace Overview:** A pie chart showing tablespace sizes: 100.00% (red) and 0.00% (blue).
- Host Overview:** A pie chart showing host storage usage: 49.28% (red) and 50.72% (green).

Database Details:

Database Name	Database Size (MB)	Tablespace Name
postgres	12	pg_default
edbstore	30	pg_default
testdb	32	pg_default
edb	377	pg_default

Tablespace Details:

Tablespace Name	Tablespace Size (MB)
pg_global	0
pg_default	478

Host File System Details:

File System	Size (GB)	Used (GB)	Available (GB)	% Used	Mounted On
/dev/sda1	0.46	0.13	0.31	29.711751662971174	/boot
hugetlbfs	0.00	0.00	0.00	0	/dev/hugepages
mqueue	0.00	0.00	0.00	0	/dev/mqueue
biffmt_misc	0.00	0.00	0.00	0	/proc/sys/fs/biffmt_misc
gvfsd-fuse	0.00	0.00	0.00	0	/run/user/1000/gvfs
cgroup	0.00	0.00	0.00	0	/sys/fs/cgroup/bikic
cgroup	0.00	0.00	0.00	0	/sys/fs/cgroup/cpuacct
cgroup	0.00	0.00	0.00	0	/sys/fs/cgroup/cpuset
cgroup	0.00	0.00	0.00	0	/sys/fs/cgroup/devices
cgroup	0.00	0.00	0.00	0	/sys/fs/cgroup/freezer
cgroup	0.00	0.00	0.00	0	/sys/fs/cgroup/hugetlb
cgroup	0.00	0.00	0.00	0	/sys/fs/cgroup/memory
cgroup	0.00	0.00	0.00	0	/sys/fs/cgroup/net_cls.net_prio
cgroup	0.00	0.00	0.00	0	/sys/fs/cgroup/perf_event
cgroup	0.00	0.00	0.00	0	/sys/fs/cgroup/pids
cgroup	0.00	0.00	0.00	0	/sys/fs/cgroup/systemd
fusectl	0.00	0.00	0.00	0	/sys/fs/fuse/connections
pstore	0.00	0.00	0.00	0	/sys/fs/pstore
selinuxfs	0.00	0.00	0.00	0	/sys/fs/selinux
configfs	0.00	0.00	0.00	0	/sys/kernel/config
debugfs	0.00	0.00	0.00	0	/sys/kernel/debug
securityfs	0.00	0.00	0.00	0	/sys/kernel/security
sunrpc	0.00	0.00	0.00	0	/var/lib/nfs/rpc_pipefs
/dev/sda3	30.88	14.53	14.78	49.57179512812823	/
/dev/fuse	0.00	0.00	0.00	0	/root/.cache/doc

Use parameters on the [PEM Server Configurations](#) dialog to specify the auto-refresh rate for the dashboard. To access the [Server Configuration](#) dialog, select [Server Configuration...](#) from the PEM web interface [Management](#) menu.

The Storage Analysis dashboard header displays the date and time that the PEM server started, the date and time that the dashboard was most recently updated, and the number of triggered alerts on objects monitored by the PEM server.

The Storage Overview section displays information about the size of databases, tablespaces and the host:

- The [Database Overview](#) pie chart shows the relative size of monitored databases stored on the server. The key (located below the chart) matches the database name to the respective color on the chart.
- The [Tablespace Overview](#) pie chart shows the relative size of tablespaces on the server. The key (located below the chart) matches the tablespace name to the respective color on the chart.

- The **Host Overview** pie chart represents the amount of used and free storage space on the server as of the last probe execution.

The **Database Details** table displays the size of each database stored on the server. Click a column heading to sort the table by the specified column; click again to reverse the sort order.

- The **Database Name** column displays the name of the database.
- The **Database Size (MB)** column displays the size of the database in megabytes.
- The **Tablespace Name** column displays the name of the default tablespace assigned to the database.

The **Tablespace Details** table lists the name and size (in megabytes) of each tablespace defined for the server. Click a column heading to sort the table by the specified column; click again to reverse the sort order.

The **Host File System Details** table displays information about the file systems that reside on the system that hosts the PEM server:

- The **File System** column displays the name of the file system.
- The **Size (GB)** column displays the size of the file system in megabytes.
- The **Used (GB)** column displays the amount of the file system that is currently storing information.
- The **Available (GB)** column displays the amount of space available on the file system.
- The **% Used** column displays the percentage of the total storage space in use.
- The **Mounted On** column displays the directory on which the file system is mounted.

7.4.1.15 The System Wait Analysis Dashboard

The System Wait Analysis dashboard provides an overview of the current DRITA wait events for an Advanced Server database. For more information about DRITA wait events, please see the EDB Postgres Advanced Server Guide.



Use parameters on the [PEM Server Configurations](#) dialog to specify the auto-refresh rate for the **Alerts** dashboard. To access the **Server Configuration** dialog, select [Server Configuration...](#) from the PEM web interface **Management** menu.

The System Waits Analysis dashboard header displays the date and time that the server started, the date and time that the System Waits Analysis dashboard was last updated, and the number of alerts currently triggered for the specified database (and monitored objects that reside within that database).

The **System Waits Overview** displays statistics gathered by the most recent execution of the PEM probe:

- The **System Waits by Number of Waits** pie chart displays the 5 most frequently encountered wait events for the selected Advanced Server server. For more information about the events that can cause a wait event, see the EDB Postgres Advanced Server Guide.
- The **System Waits by Time Waited** pie chart displays the 5 wait events that consume the most time for the selected Advanced Server server. To gather and display data in the **System Waits by Time Waited** pie chart, you must modify the `postgresql.conf` file for the monitored server, setting `timed_statistics = on`, and restart the server. Please note that this will cause server performance to degrade. For more information about using Advanced Server DRITA timers, please see the EDB Postgres Advanced Server Guide.

The **System Waits Details** table lists the current system wait events for the selected server. Click a column heading to sort the table by the column data; click again to reverse the sort order. The table displays:

- The **Event** column displays the name of the wait event.
- The **Wait Count** column contains the number of times that the wait event occurred.
- The **Percent of Total** column displays the percentage of the total wait count consumed by this event.
- The **Time Waited (ms)** displays the number of milliseconds that the server waited for the event.
- The **Percent of Time Waited** displays the percentage of the total wait time consumed by this event.
- The **Average Wait Time (ms)** column displays the average wait time for this event.

To gather and display data in the **Time Waited (ms)** and **Percent of Time Waited** columns, you must modify the `postgresql.conf` file for the monitored server, setting `timed_statistics = on`, and restart the server. Please note that this will cause server performance to degrade. For more information about using Advanced Server DRITA timers, please see the EDB Postgres Advanced Server Guide.

7.4.1.16 The Streaming Replication Analysis Dashboard

The **Streaming Replication Analysis** Dashboard displays statistical information about WAL activity for a monitored server. By default, replication probes are disabled; to view the **Streaming Replication Analysis** dashboard, you must enable probes on the primary and replica nodes. To enable the probes on the primary node, highlight the name of the primary server in the PEM client **Browser** tree control, and select **Manage Probes...** from the **Management** menu. Use the **Manage Probes** tab to enable the following probes:

- Streaming Replication
- WAL Archive Status

To enable the probes on the replica node, highlight the name of the replica server in the PEM client **Browser** tree control, and select **Manage Probes...** from the **Management** menu. Use the **Manage Probes** tab to enable the following probe:

- Streaming Replication Lag Time

Then, to open the **Streaming Replication Analysis** dashboard, navigate to the **Monitoring** tab, and:

1. Select the name of the agent that monitors the node from the **Agents** drop-down menu.
2. Select the name of the monitored server from the **Servers** drop-down menu.
3. Select **Streaming Replication Analysis** from the **Dashboards** drop-down menu.

The **Streaming Replication Analysis** dashboard header includes the date and time that the server was last started, the date and time that the page was last updated, and a current count of triggered alerts.

When accessing the **Streaming Replication Analysis** dashboard for the primary node of a replication scenario, the dashboard displays information about the write-ahead log activity for the server.



The **WAL Archive Status** graph displays WAL activity; the vertical key on the left side of the graph indicates the archive count; times are displayed across the bottom of the graph.

The **WAL Segment Lag** graph displays the segment lag for the replica nodes that are associated with the selected server. The vertical key on the left side of the graph indicates the archive count. Each node is displayed in a different color on the graph. The **Legend** provides a key to the identity (hostname and port) of each graphed replica node.

The **WAL Page Lag** graph displays the page lag activity for each replica node associated with the selected server. The vertical key on the left side of the graph indicates the page count. Each node is displayed in a different color on the graph. The **Legend** provides a key to the identity (hostname and port) of each graphed replica node.

Monitoring a Replica Node

When accessing the **Streaming Replication Analysis** dashboard for the replica node of a replication scenario, the dashboard displays information about the write-ahead log activity for the server.



The **WAL Archive Status** graph displays WAL activity; the vertical key on the left side of the graph indicates the archive count; times are displayed across the bottom of the graph.

The **WAL Segment Lag** graph displays the segment lag for the replica nodes that are associated with the selected server. The vertical key on the left side of the graph indicates the archive count. Each replica node is displayed in a different color on the graph. The **Legend** provides a key to the identity (hostname and port) of each graphed slave node.

The **WAL Page Lag** graph displays the page lag activity for each replica node associated with the selected server. The vertical key on the left side of the graph indicates the page count. Each node is displayed in a different color on the graph. The **Legend** provides a key to the identity (hostname and port) of each graphed slave node.

The **Replication Time Lag** graph displays the delay between the time that an operation is performed on the primary node of the replication scenario and the time that the operation is written to the replica node. The vertical key on the left side of the graph indicates the replication delay in minutes. Hover your mouse over a point on the graph to display the date and time that corresponds to that coordinate.

A label at the bottom of the dashboard confirms the status of the replication replica.

Monitoring a Failover Manager Cluster

If you have configured PEM to monitor a **Failover Manager** cluster, the Streaming Replication Analysis dashboard will display tables that provide an overview of the clusters status and configuration, and information about each cluster member. To display cluster information on the Streaming Replication dashboard, you must provide the following information on the **Advanced** tab of the server **Properties** dialog for each node of the cluster:

- Use the **EFM Cluster Name** field to specify the name of the Failover Manager cluster. The cluster name is the prefix of the name of the cluster properties file. For example, if your cluster properties file is named **efm.properties**, your cluster name is **efm**.
- Use the **EFM Installation Path** field to specify the location of the Failover Manager binary file. By default, the Failover Manager binary file is installed in **/usr/edb/efm-3.1/bin**.

The **Failover Manager Cluster Status** section of the Streaming Replication Analysis dashboard displays information about the monitored cluster:

The screenshot shows the Failover Manager Cluster Status section of the dashboard. It contains two tables:

Properties		Values	
Cluster Name	efm		
Failover Manager Agent Running Status	UP		
Allowed Node List	172.16.177.194, 172.16.23.156		
Standby Priority List	172.16.23.156		
Missing Nodes	0		
Minimum Standbys	0		
Membership Coordinator	172.16.177.194		
Cluster Status Message			

Agent Type	Address	Agent	DB	XLog Location	XLog Receive	Status Information	XLog Information	VIP	VIP Status
Primary	172.16.177.194	UP	UP	0/8000140					False
Standby	172.16.23.156	UP	UP	0/8000140	0/8000140				False

The **Failover Manager Cluster Information** table provides information about the Failover Manager cluster:

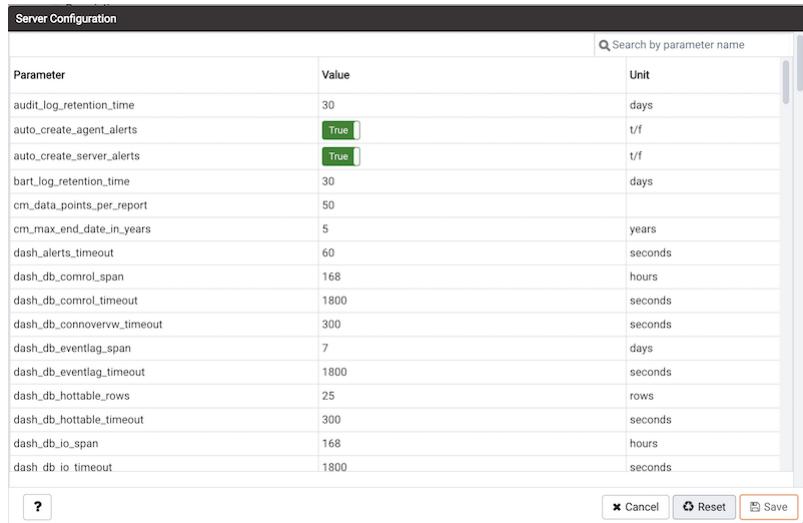
- The **Properties** column displays the name of the cluster property.
- The **Values** column displays the current value of the property.

The **Failover Manager Node Status** table displays information about each node of the Failover Manager cluster:

- The **Agent Type** column displays the type of agent that resides on the node; the possible values are Primary, Replica, Witness, Idle, and Promoting.
- The **Address** column displays the IP address of the node.
- The **Agent** column displays the status of the agent that resides on the node.
- The **DB** column displays the status of the database that resides on the node.
- The **XLog Location** column displays the transaction log location of the database.
- The **Status Information** column displays any error-related information about the node.
- The **XLog Information** column displays any error-related information about the transaction log.
- The **VIP** column displays the VIP address that is associated with the node.
- The **VIP Status** column displays **True** if the VIP is active for the node, **False** if the VIP is not.

7.4.2 Server Configuration

You can use the **Server Configuration** dialogue to modify values of user-configurable parameters that control the behavior of Postgres Enterprise Manager. To access the **Server Configuration** dialog, connect to the PEM server, and select **Server Configuration...** from the **File** menu.



Enter a parameter name in the search box in the upper-right corner of the dialog to locate a specific parameter in the list.

To modify a parameter value, edit the content displayed in the **Value** field to the right of a parameter name. Click the **Save** icon in the upper-right corner of the dialog to save your changes, or click the **Close** button to exit the dialog without applying the changes.

A list of configuration options may be found [here](#).

Contents:

7.4.2.1 Configuration Options

A number of aspects of PEM's behaviour can be controlled using global configuration options. Use the [Server Configuration dialogue](#) to manage Server Options. The configuration parameters used are listed below.

Please note that this list is subject to change.

Parameter name	Value/Unit	Description
audit_log_retention_time	30 days	Specifies the number of days that an audit log will be retained on the PEM server.
auto_create_agent_alerts	true	Specifies whether to create default agent level alerts automatically when an agent is registered.
auto_create_server_alerts	true	Specifies whether to create default server level alerts automatically when a server is bound to an agent.
chart_disable_bullets	false	Enable/disable bullets on line charts on dashboards and Capacity Manager reports.
cm_data_points_per_report	50	Specifies the number of data points to plot on charts on Capacity Manager reports.
cm_max_end_date_in_years	5 years	Specifies the maximum amount of time that the Capacity Manager will extrapolate data for. Ensures that threshold-based end dates of on reports do not get extrapolated indefinitely.

Parameter name	Value/Unit	Description
dash_alerts_timeout	60 seconds	Specifies the number of seconds after which the components of the Alerts dashboard are auto-refreshed.
dash_db_comrol_span	7 days	Specifies the number of days worth of data to plot on the Commit/Rollback Analysis chart on the Database Analysis dashboard and Server Analysis dashboard.
dash_db_comrol_timeout	1800 seconds	Specifies the number of seconds after which the Commits/Rollbacks line chart is auto-refreshed on the Database Analysis dashboard and Server Analysis dashboard.
dash_db_connovervw_timeout	300 seconds	Specifies the number of seconds after which the Connection Overview pie chart is auto-refreshed in the Database Analysis dashboard.
dash_db_eventlag_span	7 days	Specifies the number of days worth of data to plot on the Number of Events Lag chart for slony replication on the Database Analysis dashboard.
dash_db_eventlag_timeout	1800 seconds	Specifies the number of seconds after which the Number of Events Lag line chart for slony replication is auto-refreshed on the Database Analysis dashboard.
dash_db_hottable_rows	25 rows	Specifies the number of rows to show on the HOT Table Analysis table on the Database Analysis dashboard.
dash_db_hottable_timeout	300 seconds	Specifies the number of seconds after which the Hot Tables table is auto-refreshed in the Database Analysis dashboard.
dash_db_io_span	7 days	Specifies the number of days worth of data to plot on the Database I/O Analysis chart on the Database Analysis dashboard and I/O Analysis dashboard.
dash_db_io_timeout	1800 seconds	Specifies the number of seconds after which the Database I/O line chart is auto-refreshed on the Database Analysis dashboard and I/O Analysis dashboard.
dash_db_rowact_span	7 days	Specifies the number of days worth of data to plot on the Row Activity Analysis chart on the Database Analysis dashboard, the I/O Analysis dashboard, and the Server Analysis dashboard.
dash_db_rowact_timeout	1800 seconds	Specifies the number of seconds after which the Row Activity line chart is auto-refreshed on the Database Analysis dashboard, the I/O Analysis dashboard, and the Server Analysis dashboard.
dash_db_storage_timeout	300 seconds	Specifies the number of seconds after which the Storage bar chart is auto-refreshed in the Database Analysis dashboard.
dash_db_timelag_span	7 days	Specifies the number of days worth of data to plot on the Time Lag chart for Slony replication on the Database Analysis dashboard.
dash_db_timelag_timeout	1800 seconds	Specifies the number of seconds after which the Time Lag line chart for slony replication is auto-refreshed on the Database Analysis dashboard.

Parameter name	Value/Unit	Description
dash_db_useract_span	7 days	Specifies the number of days worth of data to plot on the User Activity Analysis chart on the Database Analysis dashboard.
dash_db_useract_timeout	1800 seconds	Specifies the number of seconds after which the User Activity line chart is auto-refreshed in the Database Analysis dashboard.
dash_efm_timeout	300 seconds	Specifies the number of seconds after which the Failover Manager Node Status and Failover Manager Cluster Info line chart is auto-refreshed on the Streaming Replication dashboard.
dash_global_overview_timeout	30 seconds	Specifies the number of seconds after which the components of the Global Overview dashboard are auto-refreshed.
dash_header_timeout	60 seconds	Specifies the number of seconds after which the information on the header of all the dashboards are auto-refreshed.
dash_io_chkpt_span	7 days	Specifies the number of days worth of data to plot on the Checkpoints chart on the I/O Analysis dashboard.
dash_io_chkpt_timeout	1800 seconds	Specifies the number of seconds after which the Checkpoints line chart is auto-refreshed on the I/O Analysis dashboard.
dash_io_hotindx_timeout	300 seconds	Specifies the number of seconds after which the Hot Indexes bar chart is auto-refreshed on the I/O Analysis dashboard.
dash_io_hottbl_timeout	300 seconds	Specifies the number of seconds after which the Hot Tables bar chart is auto-refreshed on the I/O Analysis dashboard.
dash_io_index_objectio_rows	25 rows	Specifies the number of rows displayed on the Index Activity table on the I/O Analysis dashboard and the Object Activity Analysis dashboard.
dash_io_index_objectio_timeout	60 seconds	Specifies the number of seconds after which the Index Activity table is auto-refreshed on the I/O Analysis dashboard and the Object Activity Analysis dashboard.
dash_io_objectio_rows	25 rows	Specifies the number of rows displayed in the Object I/O Details table on the I/O Analysis dashboard and Object Activity Analysis dashboard.
dash_io_objectio_timeout	300 seconds	Specifies the number of seconds after which the Object I/O Details table is auto-refreshed on the I/O Analysis dashboard and Object Activity Analysis dashboard.
dash_memory_hostmemact_span	7 days	Specifies the number of days worth of data to plot on the Host Memory Activity Analysis chart on the Memory Analysis dashboard.
dash_memory_hostmemact_timeout	1800 seconds	Specifies the number of seconds after which the Host Memory Activity line chart is auto-refreshed on the Memory Analysis dashboard.
dash_memory_hostmemconf_timeout	300 seconds	Specifies the number of seconds after which the Host Memory Configuration pie chart is auto-refreshed on the Memory Analysis dashboard and Server Analysis dashboard.

Parameter name	Value/Unit	Description
dash_memory_servmemact_span	7 days	Specifies the number of days worth of data to plot on the server Memory Activity Analysis chart on the Memory Analysis dashboard.
dash_memory_servmemact_timeout	1800 seconds	Specifies the number of seconds after which the Server Memory Activity line chart is auto-refreshed on the Memory Analysis dashboard.
dash_memory_servmemconf_timeout	300 seconds	Specifies the number of seconds after which the Server Memory Configuration pie chart is auto-refreshed on the Memory Analysis dashboard.
dash_objectact_objstorage_rows	15 rows	Specifies the number of rows to show on the Object Storage table on the Object Activity Analysis dashboard.
dash_objectact_objstorage_timeout	300 seconds	Specifies the number of seconds after which the Object Storage table is auto-refreshed in the Object Activity Analysis dashboard.
dash_objectact_objtopindexes_timeout	300 seconds	Specifies the number of seconds after which the Top 5 Largest Indexes bar chart is auto-refreshed in the Object Activity Analysis dashboard.
dash_objectact_objtoptables_timeout	300 seconds	Specifies the number of seconds after which the Top 5 Largest Tables bar chart is auto-refreshed in the Object Activity Analysis dashboard.
dash_os_cpu_span	7 days	Specifies the number of days worth of data to plot on the CPU chart on the Operating System Analysis dashboard.
dash_os_cpu_timeout	1800 seconds	Specifies the number of seconds after which the CPU line chart is auto-refreshed on the Operating System Analysis dashboard.
dash_os_data_span	7 days	Specifies the number of days worth of data to plot on the I/O line chart on the Operating System Analysis dashboard.
dash_os_disk_span	7 days	Specifies the number of days worth of data to plot on the Utilisation chart on the Operating System Analysis dashboard.
dash_os_hostfs_timeout	1800 seconds	Specifies the number of seconds after which the Host File System Details table is auto-refreshed on the Operating System Analysis dashboard.
dash_os_io_timeout	1800 seconds	Specifies the number of seconds after which the I/O line chart is auto-refreshed on the Operating System Analysis dashboard.
dash_os_memory_span	7 days	Specifies the number of days worth of data to plot on the Memory chart on the Operating System Analysis dashboard.
dash_os_memory_timeout	1800 seconds	Specifies the number of seconds after which the Memory line chart is auto-refreshed on the Operating System Analysis dashboard.
dash_os_packet_span	7 days	Specifies the number of days worth of data to plot on the Packet chart on the Operating System Analysis dashboard.

Parameter name	Value/Unit	Description
dash_os_packet_timeout	1800 seconds	Specifies the number of seconds after which the Network Packets line chart is auto-refreshed on the Operating System Analysis dashboard.
dash_os_process_span	7 days	Specifies the number of days worth of data to plot on the Process chart on the Operating System Analysis dashboard.
dash_os_process_timeout	1800 seconds	Specifies the number of seconds after which the Process line chart is auto-refreshed on the Operating System Analysis dashboard.
dash_os_storage_timeout	1800 seconds	Specifies the number of seconds after which the Storage pie chart is auto-refreshed on the Operating System Analysis dashboard.
dash_os_traffic_span	7 days	Specifies the number of days worth of data to plot on the Traffic chart on the Operating System Analysis dashboard.
dash_os_traffic_timeout	1800 seconds	Specifies the number of seconds after which the Traffic line chart is auto-refreshed on the Operating System Analysis dashboard.
dash_os_util_timeout	1800 seconds	Specifies the number of seconds after which the Utilisation line chart is auto-refreshed on the Operating System Analysis dashboard.
dash_probe_log_timeout	300 seconds	Specifies the number of seconds after which the Probe Log table is auto-refreshed on the Probe Log Analysis dashboard.
dash_replication_archivestat_span	7 days	Specifies the number of days worth of data to plot on the WAL Archive Status chart on the Streaming Replication Analysis dashboard.
dash_replication_archivestat_timeout	1800 seconds	Specifies the number of seconds after which the WAL Archive Status line chart is auto-refreshed on the Streaming Replication dashboard.
dash_replication_pagelag_span	7 days	Specifies the number of days worth of data to plot on the WAL Lag Pages chart on the Streaming Replication dashboard.
dash_replication_pagelag_timeout	1800 seconds	Specifies the number of seconds after which the WAL Lag Pages line chart is auto-refreshed on the Streaming Replication dashboard.
dash_replication_segmentlag_span	7 days	Specifies the number of days worth of data to plot on the WAL Lag Segments chart on the Streaming Replication dashboard.
dash_replication_segmentlag_timeout	1800 seconds	Specifies the number of seconds after which the WAL Lag Segments line chart is auto-refreshed on the Streaming Replication dashboard.
dash_replication_timelag_span	7 days	Specifies the number of days worth of data to plot on the Replication Lag Time chart on the Streaming Replication dashboard.
dash_replication_timelag_timeout	1800 seconds	Specifies the number of seconds after which the Replication Lag Time line chart is auto-refreshed on the Streaming Replication dashboard.

Parameter name	Value/Unit	Description
dash_server_buffers_written	168 hours	Specifies the number of days worth of data to plot on the Background Writer Statistics chart on the Server Analysis dashboard.
dash_server_buffers_written_timeout	300 seconds	Specifies the number of seconds after which the Background Writer Statistics line chart is auto-refreshed on the Server Analysis dashboard.
dash_server_connovervw_timeout	300 seconds	Specifies the number of seconds after which the Connection Overview pie chart is auto-refreshed in the Server Analysis dashboard.
dash_server_database_timeout	300 seconds	Specifies the number of seconds after which the Databases table is auto-refreshed in the Server Analysis dashboard.
dash_server_dbsize_span	7 days	Specifies the number of days worth of data to plot on the Database Size Analysis chart on the Server Analysis dashboard.
dash_server_dbsize_timeout	1800 seconds	Specifies the number of seconds after which the Database Size line chart is auto-refreshed in the Server Analysis dashboard.
dash_server_disk_timeout	1800 seconds	Specifies the number of seconds after which the Disk line chart is auto-refreshed in the Server Analysis dashboard.
dash_server_global_span	7 days	Specifies the number of days worth of data to plot on the Disk line chart on the Server Analysis dashboard.
dash_server_sharedbuff_span	7 days	Specifies the number of days worth of data to plot on the Shared Buffer chart on the Server Analysis dashboard.
dash_server_sharedbuff_timeout	1800 seconds	Specifies the number of seconds after which the Shared Buffers line chart is auto-refreshed in the Server Analysis dashboard.
dash_server_tabspaceysize_span	7 days	Specifies the number of days worth of data to plot on the Tablespace Size chart on the Server Analysis dashboard.
dash_server_tabspaceysize_timeout	1800 seconds	Specifies the number of seconds after which the Tablespace Size line chart is auto-refreshed in the Server Analysis dashboard.
dash_server_useract_span	7 days	Specifies the number of days worth of data to plot on the User Activity chart on the Server Analysis dashboard.
dash_server_useract_timeout	1800 seconds	Specifies the number of seconds after which the User Activity line chart is auto-refreshed in the Server Analysis dashboard.
dash_sessact_lockact_timeout	300 seconds	Specifies the number of seconds after which the Session Lock Activity table is auto-refreshed in the Session Activity Analysis dashboard.
dash_sessact_workload_timeout	300 seconds	Specifies the number of seconds after which the Session Workload table is auto-refreshed in the Session Activity Analysis dashboard.
dash_sess_waits_nowaits_timeout	300 seconds	Specifies the number of seconds after which the Session Waits By Number Of Waits pie chart is auto-refreshed in the Session Waits Analysis dashboard.

Parameter name	Value/Unit	Description
dash_sess_waits_timewait_timeout	300 seconds	Specifies the number of seconds after which the Session Waits By Time Waited pie chart is auto-refreshed in the Session Waits Analysis dashboard.
dash_sess_waits_waitdtl_timeout	300 seconds	Specifies the number of seconds after which the Session Waits Details table is auto-refreshed in the Session Waits Analysis dashboard.
dash_storage_dbdtls_timeout	300 seconds	Specifies the number of seconds after which the Database Details table is auto-refreshed in the Storage Analysis dashboard.
dash_storage_dbovervw_timeout	300 seconds	Specifies the number of seconds after which the Database Overview pie chart is auto-refreshed in the Storage Analysis dashboard.
dash_storage_hostdtls_timeout	300 seconds	Specifies the number of seconds after which the Host Details table is auto-refreshed in the Storage Analysis dashboard.
dash_storage_hostovervw_timeout	300 seconds	Specifies the number of seconds after which the Host Overview pie chart is auto-refreshed in the Storage Analysis dashboard.
dash_storage_tblspcdtls_timeout	300 seconds	Specifies the number of seconds after which the Tablespace Details table is auto-refreshed in the Storage Analysis dashboard.
dash_storage_tblspcovervw_timeout	300 seconds	Specifies the number of seconds after which the Tablespace Overview pie chart is auto-refreshed in the Storage Analysis dashboard.
dash_sys_waits_nowaits_timeout	300 seconds	Specifies the number of seconds after which the System Waits By Number Of Waits pie chart is auto-refreshed in the System Waits Analysis dashboard.
dash_sys_waits_timewait_timeout	300 seconds	Specifies the number of seconds after which the System Waits By Time Waited pie chart is auto-refreshed in the System Waits Analysis dashboard.
dash_sys_waits_waitdtl_timeout	300 seconds	Specifies the number of seconds after which the System Waits Details table is auto-refreshed in the System Waits Analysis dashboard.
deleted_charts_retention_time	7 days	Specifies the number of days that a custom chart (displayed on a user-defined dashboard) is stored.
deleted_probes_retention_time	7 days	Specifies the number of days that a custom probe (displayed on a user-defined dashboard) is stored.
download_chart_format	jpeg	Specifies the format in which a downloaded chart will be stored. May be jpeg or png.
flapping_detection_state_change	3	Specifies the number of state changes detected within a specified interval to define a given alert as flapping.- Flapping starts when more than N state changes have occurred over $[N + 1 * \text{min(probe_interval)} * 2]$ minutes and the fine state is not None. Where the default value of N is 2 or 3, and $\text{min(probe_interval)}$ is the smallest interval for all the probes used by the alert.- Flapping ends when ZERO state changes have occurred over $[2 * N * \text{min(probe_interval)}]$ minutes.

Parameter name	Value/Unit	Description
job_retention_time	30 days	Specifies the number of days that non-recurring scheduled tasks and their associated logs are retained, after their execution time.
long_running_transaction_minutes	5 minutes	Specifies the number of minutes a query executes for before being considered long running.
nagios_cmd_file_name	<file_name>	Specifies nagios command file to which passive service check result will be sent.
nagios_enabled	t	Specifies whether alert notification will be submitted to nagios or not.
nagios_medium_alert_as_critical	f	Specifies whether medium level PEM alert will be considered as critical in nagios.
nagios_spool_retention_time	7 days	Specifies the number of days to retain nagios messages in the spool table before they are discarded.
reminder_notification_interval	24 hours	Specifies the number of hours after which a reminder email is sent in case an alert has not been cleared.
server_log_retention_time	30 days	Specifies the number of days that the server log is retained on the PEM server.
show_data_tab_on_graph	false	If 'true', a Data tab is added to each graph. Select the Data tab to review the data that is plotted on the graph.
smtp_authentication	false	Specifies whether to enable/disable authentication over SMTP.
smtp_enabled	true	Specifies whether to enable/disable sending of emails.
smtp_encryption	false	Specifies whether to send SMTP email using an encrypted connection.
smtp_password		Specifies the password to be used to connect to the SMTP server.
smtp_port	25	Specifies the SMTP server port to be used for sending email.
smtp_server	127.0.0.1	Specifies the SMTP server host address to be used for sending email.
smtp_spool_retention_time	7 days	Specifies the number of days to retain sent email messages in the spool table before they are discarded.
smtp_username		Specifies the username to be used to connect to SMTP server.
snmp_community	public	Specifies the SNMP community used when sending traps. Used only with SNMPv1 and SNMPv2.
snmp_enabled	true	Specifies whether to enable/disable sending SNMP traps.
snmp_port	162	Specifies the SNMP server port to be used for sending SNMP traps.
snmp_server	127.0.0.1	Specifies the SNMP server host address to be used for sending SNMP traps.
snmp_spool_retention_time	7 days	Specifies the number of days to retain sent traps in the spool table before they are discarded.
snmp_security_name		Specifies the user name or security name for sending SNMP traps. Used only with SNMPv3.
snmp_security_engine_id		Specifies the Engine id of the SNMP Agent on the SNMP Server. Used only with SNMPv3.

Parameter name	Value/Unit	Description
snmp_security_level	NOAUTH_NOPRIV	Specifies Security level and its possible values can be: AUTH_NOPRIV - Authentication, No Privacy AUTH_PRIV - Authentication, Privacy NOAUTH_NOPRIV - no Authentication, no Privacy. Used only with SNMPv3.
snmp_context_name		Specifies the Context name, the identifier for MIB objects when sending SNMP traps. Used only with SNMPv3.
snmp_context_engine_id		Specifies the Context engine id, the identifier for MIB objects when sending SNMP traps. If not specified, snmp_security_engine_id will be used. Used only with SNMPv3.
snmp_authentication_protocol	NONE	Specifies the authentication type for SNMP traps. Its possible values can be NONE, HMACMD5 or HMACSHA. Used only with SNMPv3.
snmp_privacy_protocol	NONE	Specifies the privacy protocol for SNMP traps. Its possible values can be NONE, DES, AES128, IDEA, AES192, or AES256. Used only with SNMPv3.
snmp_authentication_password		Specifies the authentication password associated with security name mentioned in snmp_security_name. Used only for SNMPv3.
snmp_privacy_password		Specifies the privacy password associated with security name mentioned in snmp_security_name. Used only for SNMPv3.
webclient_help_pg	EnterpriseDB hosted documentation	Specifies the location of the online PostgreSQL core documentation.

7.4.3 Log Manager

Use the Log Manager wizard to specify logging preferences for a Postgres database server. Log Manager supports Advanced Server and PostgreSQL versions 9.0 (and later). The Log Manager wizard assists in modifying configuration parameters that control:

- Where log files are written.
- How often log files are written.
- The type of information written to log files.
- The format of log file entries.

Before using Log Manager to define logging properties for a server, you must specify the name of the associated Advanced Server or PostgreSQL database server in the **Service ID** field on the **Advanced** tab of the **New Server Registration** (or **Properties**) dialog. If you do not specify the name of the service in the **Service ID** field, the server will not be made available for configuration on the **Server Selection** dialog.

For example, if you are setting logging preferences for an Advanced Server 9.4 instance that resides on a Linux host, set the **Service ID** field on the **Advanced** tab of the **Properties** dialog for the monitored server to **ppas-9.4**.

To run the Log Manager, select the **Log Manager** menu option from the **Management** menu of the PEM client. The

wizard opens, welcoming you to the Log Manager:



Click **Next** to continue to the **Server selection** dialog:



The **Server selection** dialog displays a list of the server connections monitored by PEM. Check the box next to the name of a server (or servers) to which the Log Manager wizard will apply the specified configuration. Log Manager is disabled for any server displaying a red exclamation mark to the left of its name in the Server selection tree control; there are several reasons that a server may not be enabled:

- Only a server that specifies a **Service ID** on the **Advanced** tab of the **Properties** dialog can be configured by Log Manager.

To provide a service ID, right click on the server name in the tree control, and select **Disconnect Server** from the context menu; if prompted, provide a password. Then, open the context menu for the server, and select **Properties**. Navigate to the **Advanced** tab, and provide the name of the service in the **Service ID** field; click **Save** to save your change and exit the dialog.

- If the PEM agent bound to the server does not have sufficient privileges to restart the server, the server will be disabled.
- If the PEM agent bound to the server is an older version than the associated PEM server, the server will be disabled.

Click **Next** to continue:



Use options within the **Import logs** box to specify how often log files will be imported to PEM.

- Set the **Import logs to PEM** switch to **Yes** to specify that log files will be imported to PEM, and displayed on the Server Log Analysis dashboard.
- Use the **Import Frequency** drop-down list box to specify how often log files are imported to PEM. This option is only enabled when the **Import logs to PEM** option is enabled. The default value is 5 minutes.

Use the options in the **Log rotation configuration** box to specify the maximum length (lifespan or size) of a log file.

- Use the **Rotation size** field to specify the maximum size in megabytes of an individual log file. The default value is 10 MB; when set to 0, no limit is placed on the maximum size of a log file.
- Use the **Rotation time** field to specify the number of whole days that should be stored in each log file. The default value is 1 day.

Use the **Truncate on Rotation** switch to specify server behavior for time-based log file rotation:

- Select **ON** to specify that the server should overwrite any existing log file that has the same name that a new file would take.
- Select **OFF** to specify that the server should append any new log file entries to an existing log file with the same name that a new log file would take. This is the default behavior.

Click **Next** to continue to the **Where to Log** dialog:



Use the fields on the **Where to Log** dialog to specify where log files should be written. Select an option from the **Log destination** box to specify a destination for the server log output:

- Set the **stderr** switch to **Yes** to specify that log files should be written to **stderr**. By default, server log entries are written to **stderr**.
- Set the **csvlog** switch to **Yes** to specify that log files should be written to file in a comma-separated value format. This option is automatically enabled (and no longer editable) if you have selected **Import logs to PEM** on the **Schedule** dialog; if you are not importing server log files to PEM, this option is editable.
- Set the **syslog** switch to **Yes** to specify that log files should be written to the system log files.
- On Windows, set the **eventlog** switch to **Yes** to specify that log files should be written to the event log.

Use options in the **Log collection** box to specify collection preferences. Use the **Log Collector** switch to instruct the server to re-direct captured log messages (directed to STDERR) into log files:

- Specify **Enable** to instruct the server to re-direct captured error messages to a log file. By default, Log Collector is enabled.
- Specify **Disable** to instruct the server that it should not re-direct error messages to a log file.

Use the **Log Silent Mode** switch to instruct the server to run silently in the background, disassociated from the controlling terminal:

- Select **Enable** to instruct the server to run silently in the background.
- Select **Disable** to instruct the server to display log file entries on the controlling terminal as each log entry is written.

Use options in the Log Directory box to specify log file location preferences:

- Set the **Change log directory for selected servers?** switch to **Yes** to specify that each set of log files should be maintained in a separate directory.
- When **Log Collector** is enabled, you can use the **Directory name** field to specify the directory to which the log file will be written. By default, logs are written to the **pg_log** directory under the installation directory of the monitored server.

When **Import logs to PEM** is disabled, you can use the **Log file name** field to specify the filename to which the logs will be written. The 'DEFAULT' value in the **Log File Name** field represents 'postgresql-%Y-%m-%d_%H%M%S.log' for all the PostgreSQL servers and 'enterprisedb-%Y-%m-%d_%H%M%S.log' for all the Postgres Plus Advanced Servers.

When logging to **syslog** is enabled, you can use the **Syslog facility** drop-down list box to specify which syslog facility should be used.

When logging to **syslog** is enabled, you can use the **Syslog ident** field to specify the program name that will identify Advanced Server entries in system logs.

Click **Next** to continue:



Use the fields on the **When to Log** dialog to specify which events will initiate a log file entry. The severity levels (in order of severity, from most severe to least severe) are:

Severity	Description
panic	Errors that cause all database sessions to abort.
fatal	Errors that cause a session to abort.
log	Information messages of interest to administrators.
error	Errors that cause a command to abort.
warning	Error conditions in which a command will complete but may not perform as expected.
notice	Items of interest to users. This is the default.
info	Information implicitly requested by the user.
debug5 through debug1	Detailed debugging information useful to developers

- Use the **Client min messages** drop-down list box to specify the minimum severity level that will be sent to the client application. The default value is **notice**.
- Use the **Log min messages** drop-down list box to specify the minimum severity level that will be written to the server log. The default value is **warning**.
- By default, when an error message is written to the server log, the text of the SQL statement that initiated the log entry is not included. Use the **Log min error statement** drop-down list box to specify a severity level that will trigger SQL statement logging. If a message is of the specified severity or higher, the SQL statement that produced the message will be written to the server log. The default value is **error**.
- Use the **Log min duration statement** field to specify a statement duration (in milliseconds); any statements that exceed the specified number of milliseconds will be written to the server log. The length of time that it took for the statement to execute will be included in the log entry. A value of -1 disables all duration-based logging; a value of 0 logs all statements and their duration. The default value is **-1**.
- Use the **Log temp files** field to specify a file size in kilobytes; when a temporary file reaches the specified size, it will be logged. The default value is **-1**.

- Use the `Log autoVacuum min duration` field to specify a time length in milliseconds; if auto-vacuuming exceeds the length of time specified, the activity will be logged. The default value is `-1`.

Click **Next** to continue to the **What to log** dialog:



Use the fields on the **What to log** dialog to specify log entry options that are useful for debugging and auditing.

The switches in the **Debug options** box instruct the server to include information in the log files related to query execution that may be of interest to a developer:

- Set the **Parse tree** switch to **Yes** to instruct the server to include the parse tree in the log file. The default value is **No**.
- Set the **Rewriter output** switch to **Yes** to instruct the server to include query rewriter output in the log file. The default value is **No**.
- Set the **Execution plan** switch to **Yes** to instruct the server to include the execution plan for each executed query in the log file. The default value is **No**.

By default, **Indent Debug Options Output in Log** option is set to **No**. When this option is enabled, the server indents each line that contains a parse tree entry, a query rewriter entry or query execution plan entry. While indentation makes the resulting log file more readable, it results in a longer log file. To enable indentation of log file entries related to debugging, move the switch to **Yes**.

Use the switches in the **General options** box to instruct the server to include auditing information in the log file:

- Set the **Checkpoints** switch to **Yes** to include checkpoints and restartpoints in the server log. By default, this is set to **No**.
- Set the **Connections** switch to **Yes** to include each attempted connection to the server (as well as successfully authenticated connections) in the server log. By default, this is set to **No**.
- Set the **Disconnections** switch to **Yes** to include a server log entry for each terminated session that provides the session information and session duration. By default, this is set to **No**.
- Set the **Duration** switch to **Yes** to include the amount of time required to execute each logged statement in the server log. By default, this is set to **No**.
- Set the **Hostname** switch to **Yes** to include both the IP address and host name in each server log entry (by default, only the IP address is logged). Please note that this may cause a performance penalty. By default, this is set to **No**.

- Set the **Lock Waits** switch to **Yes** to instruct the server to write a log entry for any session that waits longer than the time specified in the **deadlock_timeout** parameter to acquire a lock. This is useful when trying to determine if lock waits are the cause of poor performance. By default, this is set to **No**.
- Use the **Error verbosity** drop-down list box to specify the detail written to each entry in the server log.
 - Select **default** to include the error message, DETAIL, HINT, QUERY and CONTEXT in each server log entry.
 - Select **terse** to log only the error message, excluding the DETAIL, HINT, QUERY and CONTEXT information from each server log entry.
 - Select **verbose** to include the error message, the DETAIL, HINT, QUERY and CONTEXT error information, SQLSTATE error code and source code file name, the function name, and the line number that generated the error.
- Use the **Prefix string** field to specify a printf-style string that is written at the beginning of each log file entry. The **Escape** characters in the following table represent the information described in the **Information** column. Some information is available to **Session** processes only; **Helper** processes can provide all of the information specified in the **Prefix String**. The default value is %t (timestamp without milliseconds).

You can include:

Escape	Information	Session/Helper
%a	Application Name	Session
%u	User Name	Session
%d	Database Name	Session
%r	Remote host name or IP address, and remote port	Session
%h	Remote host name or IP address	Session
%p	Process ID	Helper

%t	Time stamp without milliseconds	Helper
%m	Time stamp with milliseconds	Helper
%i	Command tag: type of statement that generated the log entry	Session
%e	SQLSTATE error code	Helper
%c	Session identifier	Helper
%l	Line number of the log entry	Helper
%s	Process start time stamp	Helper
%v	Virtual transaction ID (backendID/localXID)	Helper
%x	Transaction ID (0 if not assigned)	Helper
%q	Produces no output, but instructs non-session processes to stop at this point in the string; will be ignored by session processes	Helper
%%	Literal %	Helper

- Use the **Statements** drop-down list box to specify which SQL statements will be included in the server log. The default is **none**; valid options are:

- **none** - Specify **none** to disable logging of SQL statements.
- **ddl** - Specify **ddl** to instruct the server to log ddl (data definition language) statements, such as CREATE, ALTER, and DROP.
- **mod** - Specify **mod** to instruct the server to log all **ddl** statements, as well as all **dml** (data modification language) statements, such as INSERT, UPDATE, DELETE, TRUNCATE and COPY FROM.
- **all** - Specify **all** to instruct the server to log all SQL statements.

Click **Next** to continue:



Use the options on the **Schedule Logging Changes** dialog to select a time that logging configuration changes will be applied. Note that when you apply the configuration changes specified with the Log Manager wizard, the server will be restarted, temporarily interrupting use of the database server for users.

- Set the **Configure Logging Now** switch to **Yes** to specify that PEM will configure logging and restart the server when you have completed the Log Manager wizard.
- Set the **Configure Logging Now** switch to **No** and use the **Schedule it for some other time** date selector to specify a convenient time for the server to restart.

Click **Finish** to complete the wizard, and either restart the server, or schedule the server restart for the time specified on the scheduling dialog.

When you have completed the Log Manager wizard, you can use the **Scheduled Tasks** dialog to confirm that the configuration file update and server restart have been scheduled.

7.4.4 Audit Manager

You can use the PEM Audit manager to configure, enable, and disable audit logging of EDB Postgres Advanced Server instances. The Audit manager also enables audit log collection, allowing you to view log data on the [Audit Log Dashboard](#).

To run the Audit manager wizard, select **Audit manager...** from the PEM client **Management** menu. Audit manager opens, displaying the **Welcome** dialog:



Click **Next** to continue:



Use the **Select servers** tree control to specify the servers to which the auditing configuration will be applied. To make a server available in the tree control, you must provide the **Service ID** on the PEM **Server** dialog. Note that only EDB Postgres Advanced Server supports auditing; PostgreSQL servers will not be included in the tree control.

Click **Next** to continue:



Use the controls on the **Audit parameters configuration** dialog to specify configuration details that will be applied to each server:

- Use the **Auditing** switch to **Enable** or **Disable** auditing on the specified servers.
- Use the **Audit destination** drop-down to select a destination for the audit logs; select **File** or **Syslog**. Please note this feature is supported on Advanced Server 10 and newer releases only.
- Use the **Import logs to PEM** switch to instruct PEM to periodically import log records from each server to the PEM Server. Set the switch to **Yes** to import log files; the default is **No**.
- Use the **Import frequency** drop-down listbox to specify how often PEM will collect log records from monitored servers when log collection is enabled.
- Use the **Log format** drop-down listbox to select the raw log format that will be written on each server. If log collection is enabled, the PEM server will use CSV format.
- Use the **File name** field to specify the format used when generating log file names. By default, the format is set to **audit-%Y-%m-%d_%H%M%S** if log collection is enabled.

Use fields in the **Log directory** box to specify information about the directory in which the log files will be saved:

- Move the **Change log directory for selected servers?** switch to **Yes** to enable the **Directory name** field.
- Use the **Directory name** field to specify the name of the directory on each server into which audit logs will be written. The directory specified will be created as a sub-directory of the **data** directory on the server.

Click **Next** to continue:



The **Audit log configuration** dialog is only available if you have specified a value of **Enable** in the **Auditing** field. Use the controls on the **Audit log configuration** dialog to specify log configuration details that will be applied to each server:

- Use the **Connection attempts** switch to specify if connection attempts should be logged. Specify: **None** to disable connection logging, **All** to indicate that all connection attempts will be logged, or **Failed** to log any connection attempts that fail.
- Use the **Disconnection attempts** switch to specify if disconnections should be logged. Specify **None** to specify that disconnections should not be logged, or **All** to enable disconnection logging.
- Use the **Log statements** field to specify the statement types that will be logged. Click within the field, and select from:
 - Select - All statements that include the SELECT keyword will be logged
 - Error - All statements that result in an error will be logged.
 - DML - All DML (Data Modification Language) SQL statements will be logged.
 - DDL - All DDL (Data Definition Language) SQL statements (those that add, delete or alter data) will be logged.
 - Check the box next to **Select All** to select all statement types.
 - Check the box next to **Unselect All** to deselect all statement types.
- Use the **Audit tag** field to specify a tracking tag for the collected logs. Please note that audit tagging functionality is available only for Advanced Server versions 9.5 and later. If you are defining auditing functionality for multiple servers, and one or more of the servers are version 9.5 or later, this field will be enabled, but if selected, tagging functionality will only apply to those servers that are version 9.5 or later.

Use the fields in the **Log rotation** box to specify how the log files are managed on each server:

- Use the **Enable?** switch to specify that logfiles should be rotated. Please note that a new log file should be used periodically to prevent a single file becoming unmanageably large.
- Use the **Day** drop-down listbox to select a day or days on which the log file will be rotated.
- Use the **Size (MB)** field to specify a size in megabytes at which the log file will be rotated.
- Use the **Time (seconds)** field to specify the number of seconds between log file rotations.

Click **Next** to continue:



Use the **Schedule auditing changes** dialog to specify when the new configuration will be applied to the servers:

- Set the **Configure logging now?** switch to **Yes** to apply the configuration immediately.
- Use the **Time?** selector to schedule the audit configuration for a later time; use the date and time selectors to specify the date and time at which the PEM server will apply the configuration.

Click the **Finish** button to schedule a job to apply the configuration to each server. The job will consist of two tasks. One task will update the audit logging configuration on the server, and one task will restart the server with the new configuration.

The scheduled jobs can be viewed in the [Task Viewer](#), and the results in the [Log Viewer](#) when opened from the appropriate server or agent.

7.4.5 Postgres Log Analysis Expert

The Postgres Log Analysis Expert analyzes the log files of servers that are registered with PEM, and produces a report that provides an overview of your Postgres cluster's usage based on log file entries. You can use information on the Log Analysis Expert reports to make decisions about optimizing your cluster usage and configuration to improve performance.

Before invoking the Postgres Log Analysis Expert, you must specify the **Service ID** on the **Advanced** tab of the server's properties dialog, and use the Log Manager wizard to enable log collection by the PEM server. To invoke the Log Manager wizard, select the **Log Manager...** option from the **Management** menu; check the box next to **Import logs to PEM** in the **Import Logs** panel of the wizard to enable log collection.

To open the **Postgres Log Analysis Expert** wizard, select the **Postgres Log Analysis Expert...** option from the **Management** menu of the PEM client. When the wizard's **Welcome** dialog opens, click **Next** to continue.



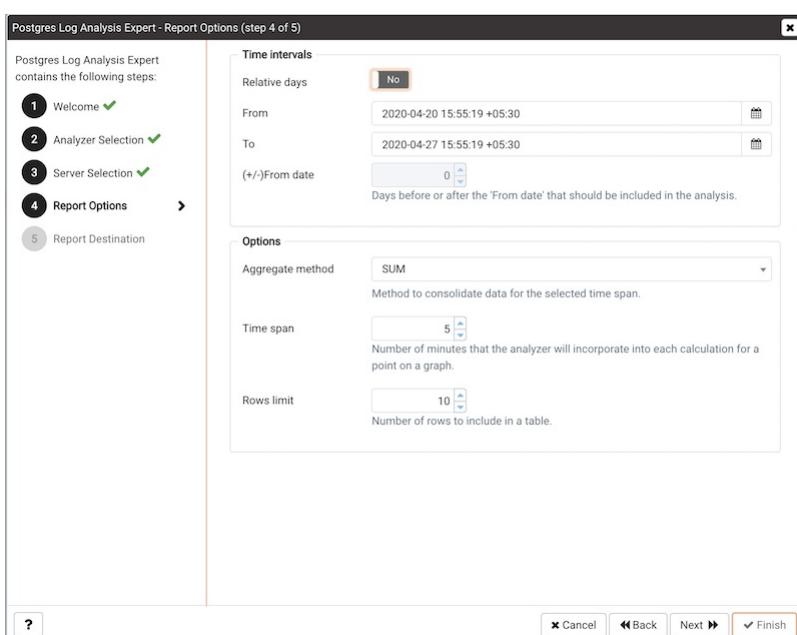
The wizard's **Analyzer selection** dialog displays a list of **Analyzers** from which you can select. Each Analyzer generates a corresponding table, chart, or graph that contains information gleaned from the log files.



Check the box to the left of an Analyzer to indicate that the Log Analysis Expert should prepare the corresponding table, chart or graph. After making your selections, click **Next** to continue to the **Server selection** tree control.



Use the tree control to specify which servers you would like the Postgres Log Analysis Expert to analyze. If you select multiple servers, the resulting report will contain the corresponding result set for each server in a separate (but continuous) list. Click **Next** to continue.



Use the fields in the **Options** section to specify the analysis method and the maximum length of any resulting tables:

- Use the **Aggregate method** drop-down to select the method used by the Log Analysis Expert to consolidate data for the selected time span - select from:
 - **SUM** - **SUM** instructs the analyzer to calculate a value that is the sum of the collected values for the specified time span.
 - **AVG** - **AVG** instructs the analyzer to calculate a value that is the average of the collected values for the specified time span.
 - **MAX** - **MAX** instructs the analyzer to use the maximum value that occurs within a specified time span.
 - **MIN** - **MIN** instructs the analyzer to use the minimum value that occurs within a specified time span.
- Use the **Time span** field to specify the number of minutes that the analyzer will incorporate into each calculation for a point on a graph. For example, if the **Time span** is '5 minutes', and the **Aggregate method** is 'AVG', each point on the given graph will contain the average value of the activity that occurred within a five minute time span.

- Use the **Rows limit** field to specify the maximum number of rows to include in a table.

Use the fields in the **Time Intervals** section to specify the time range that the Log Analysis Expert will analyze:

- Set **Relative days** to **Yes** to enable the **(+/-) From date** field and specify the number of days before or after the date and time selected in the **From** field.
- Use the **From** field to specify the starting date and time for the analysis.
- Use the **To** field to specify the ending date and time for the analysis.
- Use the **(+/-) From date** selector to specify the number of days before or after the **From** date that should be included in the analysis.

When you've specified the report options, click **Next** to continue.



You can select the default option (**Finish**) to view the Log Analysis Expert report in the PEM client's tabbed browser, or click the radio button next to **Download the report** to save a copy of the report to an HTML file for later use.

Reviewing the Postgres Log Analysis Expert Report

If you've elected to review the report immediately, the Postgres Log Analysis Expert report will be displayed in the PEM Client window. If the report contains an analysis of more than one monitored server, the graphs will be displayed in sets; first the graphs, tables and charts that display statistics for one server, then the graphs for the next server in the report.

The Postgres Log Analysis Expert Report header displays the date and time that the report was generated, the time period that the report spans, and the Aggregation method specified when defining the report. The name of the server for which information is displayed is noted at the start of each section of the report.

The report displays the tables, graphs and charts that were selected in the Log Analysis Expert wizard. Use the **Jump To** button (located in the lower-right hand corner of the screen) to navigate to a specific graphic.

The screenshot shows the 'Postgres Log Analysis Expert' interface. At the top, it displays the analysis period (Interval: 2020-04-20 15:55:19 - 2020-04-27 15:55:19), generation time (Generated: 2020-04-27 16:00:51), span (Span: 5 Minutes), aggregate (Aggregate: SUM), and a 'Go to:' dropdown set to 'Postgres Enterprise Manager Server'. Below this, a tree view node 'Postgres Enterprise Manager Server(192.168.1.18:5432)' is expanded, showing two main sections: 'Summary Statistics' and 'Hourly DML Statistics'.

Summary Statistics

Settings	Values
Number of unique queries	151649
Total queries	155045
Total queries duration	
First query	27/04/2020 15:47:09.86 IST
Last query	27/04/2020 15:53:59.611 IST
Queries peak time	27/04/2020 15:49:38 IST queries 2821
Number of events	155045
Number of unique events	1
Total number of sessions	348
Total duration of sessions	
Average sessions duration	
Total number of connections	0
Total number of databases	0

Hourly DML Statistics

Time	Database name	Statement	Count	Min duration	Max duration	Avg duration
27/04/2020 15:00	db01	SELECT	80	0.05	105.40	4.90
27/04/2020 15:00	edbstore_temp	SELECT	58	0.02	66.58	4.64
27/04/2020 15:00	hr	SELECT	48	0.01	29.26	1.87
27/04/2020 15:00	pem	COPY	1641	0.65	43.81	2.08
27/04/2020 15:00	pem	DELETE	73	0.44	8.74	1.04
27/04/2020 15:00	pem	INSERT	190	0.06	9.50	2.19

The report may include one or more of the following:

- The **Summary Statistics** table displays a summary of server activity for the selected server.
 - The **Number of unique queries** row displays the count of unique queries made against the selected server in the specified time period.
 - The **Total queries** row displays the count of queries made against the selected server in the specified time period.
 - The **Total queries duration** row displays the amount of time used to execute queries against the server.
 - The **First query** row displays the time (within the specified time period) that the first query executed against the server.
 - The **Last query** row displays the time (within the specified time period) that the last query executed against the server.
 - The **Queries peak time** row displays the point in time (within the specified time period) that query activity reached its highest level.
 - The **Number of events** row displays the count of log events within the specified time period.
 - The **Number of unique events** row displays the count of unique server events.
 - The **Total number of sessions** row displays a count of the number of sessions recorded within the time period.
 - The **Total duration of sessions** row displays the amount of time that sessions were connected (during the specified time period).
 - The **Average sessions duration** row displays the average length of each session.
 - The **Total number of connections** row displays the number of user connections made to the server.
 - The **Total number of databases** row displays the number of databases on the selected server.
- The **Hourly DML Statistics** table displays the statistics related to the use of various DML commands (SELECT, INSERT, UPDATE, DELETE, COPY and FETCH) within a one-hour period. To generate values in the **Min Duration(sec)**, **Max Duration(sec)**, and **Avg Duration(sec)** columns of this table, you must specify a value greater than or equal to **0** in the `log_min_duration_statement` configuration parameter. You can set the parameter by either modifying the `postgresql.conf` file with your editor of choice, or by specifying a value of **0** or greater in the **Log Min Duration Statement** field of the **Log Manager** wizard.
 - The **Time** column displays the start of the one-hour period for which data was analyzed.
 - The **Database** column displays the name of the database in which the specified DML command executed.
 - The **Command Type** column displays the DML command type.

- The **Total Count** column displays the number of times that a command of the specified command type executed during the one-hour period analyzed by the report.
 - The **Min Duration(sec)** column displays the shortest amount of time (in seconds) used by the server to respond to the specified command type.
 - The **Max Duration(sec)** column displays the longest amount of time (in seconds) used by the server to respond to the specified command type.
 - The **Avg Duration(sec)** column displays the average length of time (in seconds) used by the server when responding to the specified command type.
- The **DML Statistics Timeline** section of the Log Analysis Expert report displays information about DML statement usage:
 - The line graph displays an analysis of statement usage during the selected time period. Hover over a specific point to view detailed information about that point on the graph.
 - The pie chart displays the percent of statement usage of each respective DML statement type during the selected time period.
 - The **DDL Statistics Timeline** section of the Log Analysis Expert report displays information about DDL statement usage:
 - The line graph displays an analysis of statement usage during the selected time period. Hover over a specific point to view detailed information about that point on the graph.
 - The pie chart displays the percent of statement usage of each respective DDL statement type during the selected time period.
 - The **Commit and Rollback Statistics Timeline** section of the Log Analysis Expert report displays information about the COMMIT, ROLLBACK, and SAVEPOINT statements logged during the specified time period:
 - The line graph displays an analysis of the commit and rollback activity during the specified time period. Hover over a specific point to view detailed information about that point on the graph.
 - The pie chart displays the comparative percent of COMMIT, SAVEPOINT, or ROLLBACK statements executed during the specified time period.
 - The **Checkpoint Statistics Timeline** section of the Log Analysis Expert report displays information about the checkpoint operations logged during the specified time period:
 - The line graph displays an analysis of the checkpoint operation activity during the specified time period. Hover over a specific point to view detailed information about that point on the graph.
 - The pie chart displays the comparative percent of different types of checkpoint activity logged during the specified time period.
 - The **Log Event Statistics** table lists log entries with a severity level of WARNING, ERROR, FATAL, PANIC, HINT or CONTEXT. The level of logging detail for error messages is controlled by the **log_min_error_statement** parameter. You can set the parameter by either modifying the **postgresql.conf** file with your editor of choice, or by specifying a value in the **Log Min Error Statement** field of the **Log Manager** wizard.
 - The **Error Severity** column lists the severity level of the log entry.
 - The **Message** column lists the log message.
 - The **Total Count** column lists the number of times that the log entry has occurred.
 - The **Log Statistics** table lists log entries that indicate an operational severity level of LOG, DETAIL, DEBUG, NOTICE, INFO or STATEMENT. The level of logging detail for informational messages is controlled by the **log_min_messages** parameter. You can set the parameter by either modifying the **postgresql.conf** file with your editor of choice, or by specifying a value in the **Log Min Messages** field of the **Log Manager** wizard.
 - The **Error Severity** column lists the severity level of the log entry.
 - The **Total Count** column lists the number of times that the log entry has occurred.
 - The **Temp Generated Queries** table displays a list of queries that have created temporary files.
 - The **Log Time** column displays the time that the log entry was generated.
 - The **TempFile Size(Bytes)** column displays the size of the temporary file in bytes.

- The **Query** column displays the text of the query that created the temporary file.
- The **Temp File Statistics Timeline** graph displays the size of temporary files over the specified time period. Hover over a specific point to view detailed information about that point on the graph.
- The **Lock Statistics Timeline** section of the Log Analysis Expert report displays information about the locks held during the specified time period:
 - The graph displays the number of locks held at any given point during the time period. Hover over a specific point to view detailed information about that point on the graph.
 - The pie chart displays the relative percentage of each type of lock used during the selected time period.
- The **Waiting Statistics Timeline** section of the Log Analysis Expert report displays information about DML statements that are waiting for a lock during the specified time period:
 - The graph displays the number of DML statements that are waiting at any given point during the time period; each colored line represents a statement type. Hover over a specific point to view detailed information about that point on the graph.
 - The pie chart displays the relative percentage of each type of DML statement that waited for a lock during the selected time period.
- The **Idle Statistics Timeline** section of the Log Analysis Expert report displays information about the amount of time that a connection to the server is idle. An **IDLE** server is waiting for a connection from a client. A connection that is **IDLE in transaction** has started a transaction, but has not yet committed or rolled back the transaction and is waiting for a command from the client. A session that is **IDLE in transaction (aborted)*** has started a transaction, but has not yet committed or rolled back the transaction and is waiting for a command from the client; an error has occurred within the transaction and the transaction can only be rolled-back.
 - The graph displays the times at which the server is **IDLE**, **IDLE in transaction**, and **IDLE in transaction (aborted)**. Hover over a specific point to view detailed information about that point on the graph.
 - The pie chart displays the relative percentage of each type of lock used during the selected time period.
- The **Autovacuum Statistics** table displays statistics about autovacuum activity on monitored servers.
 - The **Log Time** column displays the time that the autovacuum activity was written to the log.
 - The **Relation** column displays the name of the table on which the autovacuum was performed.
 - The **Index Details** column displays the number of index scans that were performed.
 - The **Page Details** column displays the number of pages that were removed, and the number of pages that remain.
 - The **Tuple Details** column displays the number of tuples that were removed, and the number of tuples that remain.
 - The **Buffer Usage** column displays the number of buffers hit, missed, or dirty.
 - The **Read Rate** column displays the average read rate in MB's per second.
 - The **System Usage** column displays the percent of CPU time used performing autovacuum activities.
- The **Autoanalyze Statistics** table displays logged autoanalyze activity.
 - The **Log Time** column displays the time that the autoanalyze activity was written to the log.
 - The **Relation** column displays the name of the table on which the autoanalyze was performed.
 - The **System Usage** column displays the percent of CPU time used performing autoanalyze activities.
- The **Slow Query Statistics** table displays the slowest queries executed on monitored servers. The table will include the number of entries specified in the **Rows Limit** field of the Log Analysis Expert.
 - The **Log Time** column displays the time that the query activity was written to the log.
 - The **Tag** column displays the command type.
 - The **Query** column displays the text of the performed query.
 - The **Parameters** column displays the parameters (if the query is a parameterized query).
 - The **Duration** column displays the length of time that it took the server to execute the query.
 - The **Host** column displays name of the host on which the query executed.
 - The **Database** column displays the name of the database on which the query executed.

- The **Frequently Executed Query Statistics** table displays the most frequently executed query statements. The table will include the number of entries specified in the **Rows Limit** field of the Log Analysis Expert.
 - The **Query** column displays the text of the performed query.
 - The **Parameters** column displays the parameters (if the query is a parameterized query).
 - The **No. of Times Executed** column displays the number of times that the query executed.
 - The **Total Duration** column displays the length of time that it took the server to execute the query.
 - The **Most Time Executed Query Statistics** table displays the queries that took the most execution time on the server. The table will include the number of entries specified in the **Rows Limit** field of the Log Analysis Expert.
 - The **Query** column displays the text of the performed query.
 - The **Parameters** column displays the parameters (if the query is a parameterized query).
 - The **No. of Times Executed** column displays the number of times that the query executed.
 - The **Total Duration** column displays the length of time that it took the server to execute the query.
 - The **Connections Overview Timeline** section of the Log Analysis Expert report displays information about successful and unsuccessful connection attempts during the specified time period:
 - The **Timestamp** graph displays the number of server connections attempted and connections authenticated at any given point during the specified time period. Hover over a specific point to view detailed information about that point on the graph.
 - The **Summary** pie chart displays the relative percentage of connections attempted and connections authenticated during the specified time period.
-

7.4.6 Tuning Wizard

The Tuning Wizard reviews your PostgreSQL or Advanced Server installation, and recommends a set of configuration options that will help tune the installation to best suit its anticipated workload. Please note that benchmarking systems or systems with a high work load may require additional manual tuning to reach optimum performance.

Before using the Tuning Wizard, you must specify the name of the service in the **Service ID** field on the **Advanced** tab of the server's ``Properties` <pem_define_connection>`` dialog. PEM will use the service name when restarting the service after tuning.

The Tuning Wizard can only make recommendations for those servers that reside on the same server as their bound PEM agent. If you have specified a value of **Yes** in the **Remote monitoring** field when defining your server, the server will not be displayed in the Tuning Wizard tree control.

To open the Tuning Wizard, select **Tuning Wizard...** from the **Management** menu of the PEM client. The Tuning Wizard opens, welcoming you:



Click **Next** to continue to the server selection dialog:



Expand the **Servers** node of the tree control to view a list of the servers that are currently monitored by PEM that are available for tuning.

Check a box to the left of a server name to select the server for tuning. Please note: the Tuning Wizard displays a red warning symbol to the left of a server name in the tree control if the service name for that server is not provided on the server's Properties dialog.

Click **Next** to continue to the **Configuration** dialog:



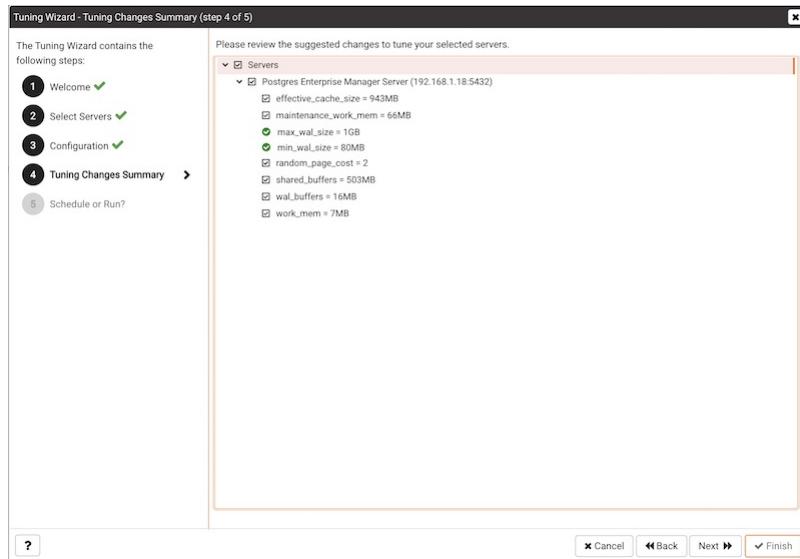
Select an option in the **Machine utilization** field to specify the type of work performed by the selected servers. The type of work performed by the server determines how the tuning wizard will allocate system resources:

- Select **Dedicated** to dedicate the majority of the system resources to the database server.
- Select **Mixed use** to dedicate a moderate amount of system resources to the database server.
- Select **Developer workstation** to dedicate a relatively small amount of system resources to the database server.

Select an option in the **Workload Selection** field to specify the type of workload typically performed on the selected server:

- Select **OLTP** if the selected server is used primarily to process online transaction workloads.
- Select **Mixed** if the selected server provides a mix of transaction processing and data reporting.
- Select **Data warehouse** if the server is used for heavy data reporting.

Click **Next** to continue to the **Tuning Changes Summary** dialog:



The tree control on the **Tuning Changes Summary** dialog displays the parameter setting modifications recommended for each server analyzed by the Tuning Wizard. Use the checkboxes next to a server or parameter name to select the recommendations that tuning wizard will either include in a preview report or apply:

- A checked box to the left of a parameter name specifies that the Tuning Wizard will include the parameter setting.
- A checked box to the left of a server name specifies that the Tuning Wizard will include all parameter setting

recommendations for the specified server.

Specify which Tuning Wizard recommendations you wish to include in a report or apply, and click **Next** to continue.

Use the **Schedule or Run?** dialog to either:

- Specify a time that PEM will apply the changes.
- Generate a report that details the recommended changes.

The selected actions will apply to all of the changes noted on the **Tuning Changes Summary**. If you opt to generate a report, PEM will create a report that contains a list of the current values and recommended modifications to the configuration parameters selected on the **Tuning Changes Summary** dialog. Note that to implement changes, you will need to invoke the Tuning Wizard a second time, specifying the parameters you wish to modify on the **Tuning Changes Summary** dialog.

Select **Schedule changes** to view your scheduling options.



You can:

- Set the **Configuration now?** slider to **Yes** to apply the tuning wizard's recommendations and restart the server now.
- Set the **Configuration now?** slider to **No** to enable the **Time?** field and use the calendar selector to specify a time for PEM to apply the tuning wizard's recommendations and restart the server. Note that if you schedule a time for the changes to be applied, you will not be provided with a preview of the change recommendations.

Select **Generate report** to view your report options.



You can:

- Set the **View report now?** slider to **Yes** to display the Tuning Wizard report onscreen.
- Set the **View report now?** slider to **No** to enable the **Save the report to file** field and use the calendar selector to specify a file name and location to which PEM will write the Tuning Wizard report.

Click the **Finish** button to either apply the Tuning Wizard's modifications or generate a report and exit the Tuning Wizard.

Tuning Wizard Report		
Generated On: 2020-04-27 16:19:52 Go to: Postgres Enterprise Manager Server		
Summary		
Number of servers selected: 1 Machine utilization: Dedicated Workload profile: OLTP		
Server: Postgres Enterprise Manager Server (192.168.1.18:5432)		
GUC Parameter	Original Value	Recommended Value
effective_cache_size	4096MB	943MB
maintenance_work_mem	64MB	66MB
random_page_cost	4	2
shared_buffers	128MB	503MB
wal_buffers	4MB	16MB
work_mem	4MB	7MB

You can confirm that Tuning Wizard has implemented the recommended changes by reviewing the postgresql.conf file for the modified server. The Tuning Wizard adds a comment above each modified parameter in the postgresql.conf file when the change is applied:

```
root@localhost:/opt/PostgresPlus/9.5AS/data
File Edit View Terminal Help
# The value for shared_buffers was changed by the Postgres Enterprise Manager Tuning Wizard at 2017-03-06 09:15:32
#shared_buffers = 196MB
# The value for shared_buffers was changed by the Postgres Enterprise Manager Tuning Wizard at 2017-03-06 09:18:23
shared_buffers = 196MB          # (change requires restart)
#huge_pages = try
#temp_buffers = 8MB
#max_prepared_transactions = 0      # zero disables the feature
# Note: Increasing max_prepared_transactions costs ~600 bytes of shared memory
# per transaction slot, plus lock space (see max_locks_per_transaction).
# It is not advisable to set max_prepared_transactions nonzero unless you
# actively intend to use prepared transactions.
#work_mem = 4MB          # min 64kB
# The value for work_mem was changed by the Postgres Enterprise Manager Tuning Wizard at 2017-03-06 09:15:32
#work_mem = 3MB
# The value for work_mem was changed by the Postgres Enterprise Manager Tuning Wizard at 2017-03-06 09:18:23
work_mem = 3MB
#maintenance_work_mem = 64MB      # min 1MB
# The value for maintenance_work_mem was changed by the Postgres Enterprise Manager Tuning Wizard at 2017-03-06 09:15:32
#maintenance_work_mem = 25MB
--More--(215)
```

You can also confirm a parameter value by querying the server. For example, to confirm the value of the shared_buffers parameter, open a SQL command line using either the **Query Tool** (accessed through the **Tools** menu) or the psql

client, and issue the command:

`SHOW shared_buffers;`

The value returned by the server will confirm that the parameter has been modified.

7.4.7 Postgres Expert

Postgres Expert analyzes the configuration of servers that are registered with the Enterprise Manager, and provides advice about:

- [Server Performance](#)
- [Server Security](#)
- [Server Configuration](#)

Postgres Expert is an advisory utility; after analyzing the selected servers, Postgres Expert produces a report containing analysis of potential performance and security issues, along with suggestions for addressing each such issue.

To use the Postgres Expert wizard select the [Postgres Expert](#) option from the [Management](#) menu in the PEM client. When the wizard's [Welcome](#) window opens; click [Next](#) to continue:



The wizard displays a tree control that allows you to choose the [Experts](#) and [Rules](#) with which Postgres Expert will evaluate the specified server or database.



The tree control categorizes the Rules under three Expert headings:

- Select from the **Configuration Expert** rules to analyze the parameter settings of the server or operating system to find any adjustments that might improve system performance.
- Select from the **Schema Expert** rules to analyze schema objects (locating missing primary keys, foreign keys without indexes, etc).
- Select from the **Security Expert** rules to review the system to find security vulnerabilities.

Use the checkbox to the left of an expert or rule to indicate that the Postgres Expert should analyze the configuration of the selected servers for any best practice deviations related to the selected item.

- Use the checkbox next to **Experts/Rules** to select or deselect all of the items listed in the tree control.
- Use the checkbox next to the name of an expert to select or deselect all of the configuration items listed under that node of the tree control.
- Use the checkbox next to a rule to select or deselect the rule for inclusion in the Postgres Expert report.

After making your selections, click **Next** to continue to the **Server/Databases** tree control.



If you select multiple servers or databases, the resulting report will contain a separate analysis of each target. Select or de-select the servers and databases that you would like Postgres Expert to analyze, and select **Next** to continue.



You can select the default option and click **Finish** to immediately view an onscreen report from Postgres Expert, or check the box next to **Download the report** to save a copy of the report to an HTML file for later use. If you choose to download the report, the report will be saved in your default downloads directory.

Reviewing the Postgres Expert Report

If you've elected to review the report immediately, the PEM client will display the report on the **Postgres Expert** tab.

The screenshot shows the 'Postgres Expert Report' interface. At the top left, it says 'Generated On: 2020-04-27 16:57:47'. At the top right, there's a 'Go to:' dropdown set to 'Postgres Enterprise Manager Server'. Below the header, a 'Summary' section displays statistics: 'Servers Tested: 1', 'Rules Checked: 31', 'High Alerts: 1', 'Medium Alerts: 3', and 'Low Alerts: 2'. A section titled 'Server: Postgres Enterprise Manager Server (192.168.1.18:5432)' is expanded, showing two tables of recommendations:

Rule	Database	Severity
> Check checkpoint_completion_target	-	Medium
> Check effective_cache_size	-	Medium
> Check effective_io_concurrency	-	Low
> Check reducing random_page_cost	-	Low

Rule	Database	Severity
> Check data and transaction log on same drive	-	High
> Check for missing foreign key indexes	db01	Medium

A report summary in the upper-left corner of the Postgres Expert Report lists statistics about the analysis, including the number of servers analyzed, the number of rules tested, and the number of alerts raised in each severity category.

If your report contains recommendations for more than one server, you can use the **Jump to** selector in the upper-right corner of the report as a navigation tool; select a server from the list to move to the portion of the report containing information for the selected server.

For each server analyzed, the Postgres Expert returns recommendations from the **Configuration Expert**, the **Schema Expert**, and the **Security Expert**. Each expert returns a list of rules that raised an alert, the database that the rule pertains to, and the severity level of the alert. Click on a rule name to view detailed information about the selected rule:

Section Heading

Contains

Trigger

A description of the rule that raised the alert.

Recommended Value

The value to which Postgres Expert recommends setting the selected parameter.

Description

Information and advice about the parameter that caused the alert.

Current Values

The current value(s) of the parameter(s).

The screenshot shows the 'Postgres Expert Report' interface. At the top, it displays 'Generated On: 2020-04-27 16:57:47' and a 'Go to:' dropdown set to 'Postgres Enterprise Manager Server'. Below this is a 'Summary' section with statistics: 'Servers Tested: 1', 'Rules Checked: 31', 'High Alerts: 1', 'Medium Alerts: 3', and 'Low Alerts: 2'. A tree view shows 'Server: Postgres Enterprise Manager Server (192.168.1.18:5432)'. Under 'Advisor: Configuration Expert', there's a table for 'Rule' with one entry: 'Check checkpoint_completion_target' (Severity: Medium). A detailed description explains that checkpoints write dirty buffers to disk to ensure reliable recovery. Under 'Recommended Value', it suggests adjusting 'checkpoint_completion_target' to 0.9. Current values show 'checkpoint_completion_target' at 0.5. Other rules listed include 'Check effective_cache_size' (Medium), 'Check effective_ioConcurrency' (Low), and 'Check reducing random_page_cost' (Low). Under 'Advisor: Schema Expert', there's a table for 'Rule' with two entries: 'Check data and transaction log on same drive' (High) and 'Check for missing foreign key indexes' (Medium). The 'Check for missing foreign key indexes' rule is expanded to show a table with columns 'Rule', 'Database', and 'Severity'. It lists a single entry for database 'db01' with severity 'Medium'.

For more information about each rule checked by the Postgres Expert, see:

7.4.7.1 Schema Expert Recommendations

Rule	Check for missing primary keys
Recommendation	Ensure tables have a primary key
Trigger	Postgres Expert detected a table with no defined primary key.
Severity	Low

Description: Primary keys are used to define the set of columns that make up the unique key to each row in the table. Whilst they are similar to unique indexes, primary keys cannot contain NULL values, thus are always able to identify a single row. Tools such as Postgres Enterprise Manager and other pieces of software such as ORM will automatically detect primary keys on tables and use their definition to identify individual rows.

Rule	Check for missing foreign key indexes
Recommendation	Ensure columns of child tables in foreign key relationships are indexed.
Trigger	Postgres Expert detected a child table with no index on referencing column(s).
Severity	Medium

Description: Foreign keys are used to define and enforce relationships between child and parent tables. The foreign key specifies that values in one or more columns of the child table must exist (in the same combination, if more than one column) in the referenced column(s) of the parent table. A unique index is required to be present on the referenced columns in the parent table, however an index is not required, but is generally advisable, on the referencing columns of the child table to allow cascading updates to the parent to be executed efficiently.

Rule	Check Database Encoding
------	-------------------------

Recommendation	Avoid encoding as SQL_ASCII for databases
Trigger	encoding = SQL_ASCII
Severity	Medium

Description: The database is created to store data using the SQL_ASCII encoding. This encoding is defined for 7 bit characters only; the meaning of characters with the 8th bit set (non-ASCII characters 127-255) is not defined. Consequently, it is not possible for the server to convert the data to other encodings. If you're storing non-ASCII data in the database, you're strongly encouraged to use a proper database encoding representing your locale character set to take benefit from the automatic conversion to different client encodings when needed. If you store non-ASCII data in an SQL_ASCII database, strange characters may be written to or read from the database, caused by code conversion problems. This may cause problems when accessing the database using different client programs and drivers. For most installations, Unicode (UTF8) encoding will provide the most versatility.

Rule	Check for too many indexes
Recommendation	Don't overload a table with too many indexes.
Trigger	Postgres Expert has detected that a table has more than 10 indexes.
Severity	Low, Medium or High (based on number of indexes)

Description: Whilst indexes can speed up SELECT queries by allowing Postgres to quickly locate records, it is important to choose which indexes are required carefully to ensure they are used. Maintaining an index has a cost, and the more indexes there are to update, the slower INSERT, UPDATE or DELETE queries can become. There are no hard and fast rules to tell you how many indexes are required on a particular table -the DBA must balance the need for indexes for different types of SELECT queries and constraints against the cost of maintaining them.

Configuration Item	Check data and transaction log on same drive
Recommendation	Avoid using the same storage device for the data directory and transaction logs.
Trigger	Postgres Expert has detected that a data directory and transaction log directory share a device.
Severity	High

Description: Postgres' performance can be adversely affected on medium to heavily loaded systems if both the data and the transaction logs (WAL) are stored on the same device. It is considered good practice to store them on separate physical devices if performance is an issue. On busy servers, significant performance gains may be seen when separating the data directory and transaction log directory onto different physical storage devices.

Rule	Check tablespace and transaction log on same drive
Recommendation	Avoid using the same storage device for the transaction logs and a tablespace.
Trigger	Postgres Expert has detected that transaction log directory and a tablespace other than pg_default share a device.
Severity	Medium

Description: Before updating database files to reflect data modifications, the server writes the change to the transaction log. The database files may be separated onto different devices using tablespaces (defined storage areas used by the database server). On busy servers, significant performance gains may be seen when separating tablespace directories and the transaction log directory onto different physical storage devices.

Rule	Check multiple tablespace on same drive
Recommendation	Avoid using the same storage device for multiple tablespaces.

Trigger	Postgres Expert has detected that multiple tablespaces share a device.
Severity	Low
<p>Description: Multiple tablespaces may be defined in the database to allow tables and indexes to be distributed into different storage areas, usually for performance reasons for example, tables with high performance requirements may be stored on expensive , high speed disks, while archive data may be stored on much larger, but slower devices. There is usually little to be gained from having more than one tablespace on a single device (because the cost and access characteristics will be identical), except in very unusual situations where it may be desirable to configure them with different planner cost parameters.</p>	

7.4.7.2 Security Expert Recommendations

Rule	Check SSL for improved performance
Recommendation	Consider disabling SSL for improved performance.
Trigger	ssl = on and listen_addresses in ('localhost', '127.0.0.1', '::1')
Severity	Low

Description: SSL authentication is invaluable for protecting against connection-spoofing and eavesdropping attacks, but it is not always necessary for adequate security. When PostgreSQL accepts only local connections, or when it accepts only connections from a trusted network where malicious network traffic is not a concern, SSL encryption may not be necessary. Consider changing this setting if the current value is not appropriate for your environment.

Note: Even when SSL encryption is enabled, PostgreSQL servers should be further protected using an appropriate firewall configuration.

Rule	Check SSL for improved connection security
Recommendation	Consider using SSL for improved connection security.
Trigger	ssl = off and listen_addresses not in ('localhost', '127.0.0.1', '::1')
Severity	Medium

Description: The configuration variable listen_addresses indicates that your system may accept non-local connection requests, but SSL is not enabled. If PostgreSQL is exposed only to a secure, trusted internal network, this configuration is appropriate for maximum performance. Otherwise, you should consider enabling SSL. SSL offers two main advantages. First, it provides a more secure mechanism for authorizing connections to the database, helping to prevent unauthorized access. Second, SSL prevents eavesdropping attacks, where data sent from the database to clients, or from clients to the database, is viewed by an attacker while in transit. Consider changing this setting if the current value is not appropriate for your environment.

Note: Even when SSL encryption is enabled, PostgreSQL servers should be further protected using an appropriate firewall configuration.

Rule	Check TRUST authentication is disabled
Recommendation	Avoid trust and ident authentication on unsecured networks.
Trigger	trust or ident authentication allowed to any host other than 127.0.0.1 or ::1

Severity	High
----------	------

Description: An attacker with access to your network can easily use the trust and ident authentication methods to subvert your network. If PostgreSQL is not running on a secure network, with firewalls in place to prevent malicious traffic, the use of these authentication methods should be avoided.

Rule	Check Password authentication on unsecured networks
Recommendation	Avoid password authentication on unsecured networks.
Trigger	(connection_type = 'host' or connection_type = 'hostnoss') and method = 'password'
Severity	High

Description: Passwords should not be transmitted in plaintext over unsecured networks. The use of md5 authentication provides slightly better security, but can still allow accounts to be compromised by a determined attacker. SSL encryption is a superior alternative. To require the use of SSL, set the connection type to hostssl in the pg_hba.conf file.

Rule	Check SSL for increased security
Recommendation	Consider requiring SSL.
Trigger	ssl = on in postgresql.conf, but no hostssl lines in pg_hba.conf
Severity	Medium

Description: SSL encrypts passwords and all data transmitted over the connection, providing increased security. To require the use of SSL, set the connection type to hostssl in the pg_hba.conf file.

7.4.7.3 Configuration Expert Recommendations

Rule	Check shared_buffers
Recommendation	Consider adjusting shared_buffers
Trigger	shared_buffers < (OS == Windows ? 64MB : MIN(0.20 * (system_memory - 256MB), 6GB)) or shared_buffers > (OS == Windows ? 512MB : MAX(0.35 * system_memory, 8GB))
Recommended Value	system_memory < 1GB ? MAX((system_memory - 256MB) / (OS == Windows ? 6 : 3), 64MB), OS == Windows ? MAX(system_memory / 8, 256MB) : MAX(system_memory / 4, 8GB)
Severity	Medium

Description: The configuration variable shared_buffers controls the amount of memory reserved by PostgreSQL for its internal buffer cache. Setting this value too low may result in "thrashing" the buffer cache, resulting in excessive disk activity and degraded performance. However, setting it too high may also cause performance problems. PostgreSQL relies on operating system caching to a significant degree, and setting this value too high may result in excessive "double buffering" that can degrade performance. It also increases the internal costs of managing the buffer pool. On UNIX-like systems, a good starting value is approximately 25% of system memory, but not more than 8GB. On Windows systems, values between 64MB and 512MB typically perform best. The optimal value is workload-dependent, so it may be worthwhile to try several different values and benchmark your system to determine which one delivers best performance.

Note: PostgreSQL will fail to start if the necessary amount of shared_memory cannot be located. This is usually due to an operating system limitation which can be raised by changing a system configuration setting, often called shmall. See the

documentation for more details. You must set this limit to a value somewhat higher than the amount of memory required for shared_buffers, because PostgreSQL's shared memory allocation also includes amounts required for other purposes.

Rule	Check work_mem
Recommendation	Consider adjusting work_mem
Trigger	given spare_mem = system_memory - (OS == Windows ? 256MB : MAX(0.25 * system_memory, 8GB)) then work_mem < MAX(1MB, spare_mem / 512) or work_mem > (spare_mem / 128)
Recommended Value	given spare_mem defined as on the previous line then MAX(1MB, spare_mem / 256)
Severity	Medium

Description: The configuration variable `work_mem` controls the amount of memory PostgreSQL will use for each individual hash or sort operation. When a sort would use more than this amount of memory, the planner will arrange to perform an external sort using disk files. While this algorithm is memory efficient, it is much slower than an in-memory quick sort. Similarly, when a hash join would use more than this amount of memory, the planner will arrange to perform it in multiple batches, which saves memory but is likewise much slower. In either case, the planner may in the alternative choose some other plan that does not require the sort or hash operation, but this too is often less efficient. Therefore, for good performance it is important to set this parameter high enough to allow the planner to choose good plans. However, each concurrently executing query can potentially involve several sorts or hashes, and the number of queries on the system can vary greatly. Therefore, a value for this setting that works well when the system is lightly loaded may result in swapping when the system becomes more heavily loaded. Swapping has very negative effects on database performance and should be avoided, so it is usually wise to set this value somewhat conservatively.

Note: `work_mem` can be adjusted for particular databases, users, or user-and -database combinations by using the commands `ALTER ROLE` and `ALTER DATABASE`. It can also be changed for a single session using the `SET` command. This can be helpful when particular queries can be shown to run much faster with a value of `work_mem` that is too high to be applied to the system as a whole.

Rule	Check max_connections
Recommendation	Consider using a connection pooler
Trigger	max_connections > 100
Severity	Medium

Description: The configuration variable `max_connections` is set to a value greater than 100. PostgreSQL performs best when the number of simultaneous connections is low. Peak throughput is typically achieved when the connection count is limited to approximately twice the number of system CPU cores plus the number of spindles available for disk I/O (in the case of an SSD or other non-rotating media, some experimentation may be needed to determine the "effective spindle count"). Installing a connection pooler, such as pgpool-II or pgbouncer, can allow many clients to be multiplexed onto a smaller number of server connections, sometimes resulting in dramatic performance gains.

Rule	Check maintenance_work_mem
Recommendation	Consider adjusting maintenance_work_mem
Trigger	spare_mem = system_memory - (OS == Windows ? 256MB : MAX(0.25 * system_memory, 8GB)) then maintenance_work_mem < MAX(16MB, spare_mem / 32) or maintenance_work_mem > MIN(spare_mem / 8, 256MB)
Recommended Value	spare_mem as defined on the previous line then MIN(spare_mem/16, 256MB)
Severity	Low

Description: The configuration variable `maintenance_work_mem` controls the amount of memory PostgreSQL will use

for maintenance operations such as CREATE INDEX and VACUUM. Increasing this setting from the default of 16MB to 256MB can make these operations run much faster. Higher settings typically do not produce a significant further improvement. On PostgreSQL 8.3 and higher, multiple autovacuum processes may be running at one time (up to autovacuum_max_workers, which defaults to 3), and each such process will use the amount of dedicated memory dictated by this parameter. This should be kept in mind when setting this parameter, especially on systems with relatively modest amounts of physical memory, so as to avoid swapping. Swapping has very negative effects on database performance and should be avoided. If the value recommended above is less than 256MB, it is chosen with this consideration in mind. However, the optimal value is workload-dependent, so it may be worthwhile to experiment with higher or lower settings.

Rule	Check effective_io_concurrency
Recommendation	Consider adjusting effective_io_concurrency
Trigger	effective_io_concurrency < 2
Severity	Low

Description: If the PostgreSQL data files are located on a RAID array or SSD, effective_io_concurrency should be set to the approximate number of I/O requests that the system can service simultaneously. For RAID arrays, this is typically equal to the number of drives in the array. For SSDs, some experimentation may be needed to determine the most effective value. Setting this parameter to an appropriate value improves the performance of bitmap index scans. The default value of 1 is appropriate for cases where all PostgreSQL data files are located on a single spinning medium.

Rule	Check fsync is enabled
Recommendation	Consider configuring fsync = on.
Trigger	fsync = off
Severity	High

Description: When fsync is set to off, a system crash can result in unrecoverable data loss or non-obvious corruption. fsync = off is an appropriate setting only if you are prepared to erase and recreate all of your databases in the event of a system crash or unexpected power outage.

Note: Much of the performance benefit obtained by configuring fsync = off can also be obtained by configuring synchronous_commit = off. However, the latter setting is far safer: in the event of a crash, the last few transactions committed might be lost if they have not yet made it to disk, but the database will not be corrupted.

Rule	Check wal_sync_method
Recommendation	On Windows, consider configuring wal_sync_method = fsync or wal_sync_method = fsync_writethrough.
Trigger	OS = Windows and wal_sync_method not in ('fsync', 'fsync_writethrough')
Severity	High

Description: In order to guarantee reliable crash recovery, PostgreSQL must ensure that the operating system flushes the write-ahead log to disk when asked to do so. On Windows, this can be achieved by setting wal_sync_method to fsync or fsync_writethrough, or by disabling the disk cache on the drive where the write-ahead log is written. (It is safe to leave the disk cache enable if a battery-back disk cache is in use.)

Note: In cases where the loss of a very recently committed transaction is acceptable, the performance impact of flushing the write ahead log to disk can be mitigated by setting synchronous_commit = off. In other situations, the use of a battery-backed RAID controller is recommended.

Rule	Check wal_sync_method
------	-----------------------

Recommendation	On Mac OS X, consider configuring wal_sync_method = fsync_writethrough.
Trigger	OS == MacOS X and wal_sync_method != fsync_writethrough
Severity	High

Description: In order to guarantee reliable crash recovery, PostgreSQL must ensure that the operating system flushes the write-ahead log to disk when asked to do so. On MacOS X, this can be achieved by setting wal_sync_method to fsync_writethrough or by disabling the disk cache on the drive where the write-ahead log is written. It is safe to leave the disk cache enable if a battery-back disk cache is in use.

Note: In cases where the loss of a very recently committed transaction is acceptable, the performance impact of flushing the write ahead log to disk can be mitigated by setting synchronous_commit = off. In other situations, the use of a battery-backed RAID controller is recommended.

Rule	Check wal_buffers
Recommendation	Consider adjusting wal_buffers
Trigger	wal_buffers < 1MB or wal_buffers > 16MB
Severity	Medium

Description: Increasing the configuration parameter wal_buffers from the default value of 64kB to 1MB or more can reduced the number of times the database must flush the write-ahead log, leading to improved performance under some workloads. There is no benefit to setting this parameter to a value greater than the size of a WAL segment (16MB).

Rule	Check commit_delay
Recommendation	Consider setting commit_delay = 0.
Trigger	commit_delay != 0
Severity	Low

Description: Setting the commit_delay configuration parameter to a non-zero value causes the system to wait for the specified number of microseconds before flushing the write-ahead log to disk at commit time, potentially allowing several concurrent transactions to commit with a single log flush. In most cases, this does not produce a performance benefit, and in some cases, it can produce a performance regression. Unless you have confirmed through benchmarking that a non-default value for this parameter produces a performance benefit, the default value of 0 is recommended.

Rule	Check checkpoint_segments
Recommendation	Consider adjusting checkpoint_segments.
Trigger	checkpoint_segments < 10 or checkpoint_segments > 300
Severity	Medium

Description: In order to ensure reliable and efficient crash recovery, PostgreSQL periodically writes all dirty buffers to disk. This process is called a checkpoint. Checkpoints occur when (1) the number of write-ahead log segments written since the last checkpoint exceeds checkpoint_segments, (2) the amount of time since the last checkpoint exceeds checkpoint_timeout, (3) the SQL command CHECKPOINT is issued, or (4) the system completes either shutdown or crash recovery. Increasing the value of checkpoint_segments will reduce the frequency of checkpoints and will therefore improve performance, especially during bulk loading. The main downside of increasing checkpoint_segments is that, in the event of a crash, recovery will require a longer period of time to return the database to a consistent state. In addition, increasing checkpoint_segments will increase disk space consumption during periods of heavy system activity. However, because the theoretical limit on the amount of additional disk space that will be consumed for this reason is less than 32MB per additional checkpoint segment, this is often a small price to pay for improved performance.

Values between 30 and 100 are often suitable for modern systems. However, on smaller systems, a value as low as 10 may be appropriate, and on larger systems, a value as 300 may be useful. Values outside this range are generally not worthwhile.

Rule	Check checkpoint_completion_target
Recommendation	Consider adjusting checkpoint_completion_target.
Trigger	checkpoint_completion_target != 0.9
Severity	Medium

Description: In order to ensure reliable and efficient crash recovery, PostgreSQL periodically writes all dirty buffers to disk. This process is called a checkpoint. Beginning in PostgreSQL 8.3, checkpoints take place over an extended period of time in order to avoid swamping the I/O system. `checkpoint_completion_target` controls the rate at which the checkpoint is performed, as a function of the time remaining before the next checkpoint is due to start. A value of 0 indicates that the checkpoint should be performed as quickly as possible, whereas a value of 1 indicates that the checkpoint should complete just as the next checkpoint is scheduled to start. It is usually beneficial to spread the checkpoint out as much as possible; however, if `checkpoint_completion_target` is set to a value greater than 0.9, unexpected delays near the end of the checkpoint process can cause the checkpoint to fail to complete before the next one needs to start. Because of this, the recommended setting is 0.9.

Rule	Check effective_cache_size
Recommendation	Consider adjusting effective_cache_size.
Trigger	effective_cache_size < 0.5 * system_memory or effective_cache_size > MAX(0.9 * system_memory, system_memory - 1GB)
Recommended value	0.75 * system_memory
Severity	Medium

Description: When estimating the cost of a nested loop with an inner index-scan, PostgreSQL uses this parameter to estimate the chances that rows from the inner relation which are fetched multiple times will still be in cache when the second fetch occurs. Changing this parameter does not allocate any memory, but an excessively small value may discourage the planner from using indexes that would in fact speed up the query. The recommended value is 75% of system memory.

Rule	Check default_statistics_target
Recommendation	Consider adjusting default_statistics_target.
Trigger	default_statistics_target < 25 or default_statistics_target > 400
Recommended value	100
Severity	Medium

Description: PostgreSQL uses statistics to generate good query plans. These statistics are gathered either by a manual `ANALYZE` command or by an automatic analyze launched by the autovacuum daemon, and they include the most common values in each column of each database table, the approximate distribution of the remaining values, the fraction of rows which are `NULL`, and several other pieces of statistical information.

`default_statistics_target` indicates the level of detail that should be used in gathering and recording these statistics. A value of 100, which is the default beginning in PostgreSQL 8.4, is reasonable for most workloads. For very simple queries, a smaller value may be useful, while for complex queries especially against large tables, a higher value may work better. In some case, it can be helpful to override the default statistics target for specific table columns using `ALTER TABLE .. ALTER COLUMN .. SET STATISTICS`.

Rule	Check planner methods is enabled
Recommendation	Avoid disabling planner methods.
Trigger	any <code>enable*</code> GUC is off
Severity	High

Description: The `enable_bitmapscan`, `enable_hashagg`, `enable_hashjoin`, `enable_indexscan`, `enable_material`, `enable_mergejoin`, `enable_nestloop`, `enable_seqscan`, `enable_sort`, and `enable_tidscan` parameters are intended primarily for debugging and should not be turned off. It can sometimes be helpful to disable one or more of these parameters for a particular query, when there is no other way to obtain the desired plan. However, none of these parameters should ever be turned off on a system-wide basis.

Rule	Check <code>track_counts</code> is enabled
Recommendation	Consider configuring <code>track_counts = on</code> .
Trigger	<code>track_counts = off</code>
Severity	High

Description: Autovacuum will not function properly if `track_counts` is disabled. Regular vacuuming is crucial to system stability and performance.

Rule	Check autovacuum is enabled
Recommendation	Consider configuring <code>autovacuum = on</code> .
Trigger	<code>autovacuum = off</code>
Severity	High

Description: Enabling autovacuum is an important part of maintaining system stability and performance. Although disabling autovacuum may be useful during bulk loading, it should always be promptly reenabled when bulk loading is completed. Leaving autovacuum disabled for extended periods of time will result in table and index "bloat", where available free space is not reused, resulting in uncontrolled table and index growth. Reversing such bloat requires invasive maintenance using `CLUSTER`, `REINDEX`, and/or `VACUUM FULL`. Allowing autovacuum to work normally is usually sufficient to avoid the need for such maintenance.

Rule	Check configuring <code>seq_page_cost</code>
Recommendation	Consider configuring <code>seq_page_cost <= random_page_cost</code> .
Trigger	<code>seq_page_cost > random_page_cost</code>
Severity	Medium

Description: `seq_page_cost` and `random_page_cost` are parameters used by the query parameter to determine the optimal plan for each query. `seq_page_cost` represents the cost of a sequential page read, while `random_page_cost` represents the cost of a random page read. While these costs might be equal, if, for example, the database is fully cached in RAM, the sequential cost can never be higher. The PostgreSQL query planner will produce poor plans if `seq_page_cost` is set higher than `random_page_cost`.

Rule	Check reducing <code>random_page_cost</code>
Recommendation	Consider reducing <code>random_page_cost</code> to no more than twice <code>seq_page_cost</code> .
Trigger	<code>random_page_cost > 2 * seq_page_cost</code>
Severity	Low

Description: seq_page_cost and random_page_cost are parameters used by the query parameter to determine the optimal plan for each query. seq_page_cost represents the cost of a sequential page read, while random_page_cost represents the cost of a random page read. random_page_cost should always be greater than or equal to seq_page_cost, but it is rarely beneficial to set random_page_cost to a value more than twice seq_page_cost. However, the correct values for these variables are workload-dependent. If the database's working set is much larger than physical memory and the blocks needed to execute a query will rarely be in cache, setting random_page_cost to a value greater than twice seq_page_cost may maximize performance.

Rule	Check increasing seq_page_cost
Recommendation	Consider increasing seq_page_cost.
Trigger	seq_page_cost < cpu_tuple_cost, seq_page_cost < cpu_index_tuple_cost, or seq_page_cost < cpu_operator_cost
Severity	Medium

Description: The cost of reading a page into the buffer cache, even if it is already resident in the operating system buffer cache, is rarely less than the cost of a CPU operation. Thus, the value of the configuration parameter seq_page_cost should usually be greater than the values of the configuration parameters cpu_tuple_cost ,cpu_index_tuple_cost, and cpu_operator_cost.

7.4.8 Capacity Manager

PEM's Capacity Manager analyzes collected statistics (metrics) to generate a graph or table that displays the historical usage statistics of an object, and can project the anticipated usage statistics for an object. You can configure Capacity Manager to collect and analyze metrics for a specific:

- Host/operating system
- EDB Postgres Advanced Server or PostgreSQL server
- Database
- Database object (table, index, function etc).

You can tailor the content of the Capacity Manager report by choosing a specific metric (or metrics) to include in the report, the time range over which the metrics were gathered, and a high or low threshold for the metrics analyzed. You can also specify a start and end date for the Capacity Manager report. If the end date of the report specifies a time in the future, Capacity Manager will analyze the **historical** usage of the selected object to extrapolate the **projected** object usage in the future.

To open Capacity Manager, select the **Capacity Manager...** option from the **Management** menu in the PEM client window; the **Capacity Manager** wizard opens, displaying a tree control on the **Metrics** tab.



Expand the tree control on the **Metrics** tab to select the metrics that will be included in the Capacity Manager report.

When defining report options, you can specify an **aggregation** method for each selected metric. The aggregation method determines how Capacity Manager will analyze the data points within the sampling period to reduce the data to a more visually meaningful quantity within a report (if required). The aggregation method can instruct Capacity Manager to compute an average of the data within a time period, the high or low value, or the first sampled value.

Use the **Options** tab to specify additional report details.

When defining the boundaries of a Capacity Manager report, specify the starting date and time, and an end boundary. The end boundary can be a point in time or a threshold boundary (when the data meets a specified criteria). If the sample contains more data points than the number of points specified by the `cm_data_points_per_report <pem_config_options>` configuration parameter, Capacity Manager applies the aggregation method to calculate a reduced number of graph points for the report.

Report Templates

You can save a report definition as a template for future reports. Capacity Manager report templates may be accessed by all PEM users. To save a report definition as a template:

1. Use the **Metrics** and **Options** tabs to define your report.
2. Click the **Save** button to open the **Save Template** dialog.
3. Provide a report name in the Title field, select a location to store the template in the tree control.
4. Click **OK**.

When creating a report, you can use the **Load Template** button to browse and open an existing template. Once opened, the report definition may be modified if required, and optionally saved again, either as a new template, or overwriting the original template. Use the **Manage Templates** button open a dialog that allows you to rename or remove unwanted templates.

Available Metrics

Please Note that the available metrics will vary by platform, and are subject to change. The available metrics may include the metrics described in the table below:

Metric Name	Description
# Dead Tuples	The number of dead tuples in the selected table.
# Dead Tuples+	The cumulative number of dead tuples in the selected table.
# Heap Tuples Fetched by Index Scans	The number of heap tuples fetched by index scans.
# Heap Tuples Fetched by Index Scans+	The cumulative number of heap tuples fetched by index scans.
# Idle Backends+	The cumulative number of currently idle backend clients.
# Index Scans	The number of index scans performed on the specified object.
# Index Scans+	The cumulative number of index scans performed on the specified object.
# Index Tuples Read	The number of index tuples read.
# Index Tuples Read+	The cumulative number of index tuples read.
# Live Tuples	The number of tuples visible to transactions.
# Live Tuples+	The cumulative number of tuples visible to transactions.
# Pages Estimated by ANALYZE	The number of pages estimated by ANALYZE.
# Pages Estimated by ANALYZE+	The cumulative number of pages estimated by ANALYZE.
# Sequential Scans	The number of sequential scans performed on the specific table.
# Sequential Scans+	The cumulative number of sequential scans performed on the specific table.
# Sequential Scan Tuples	The number of tuples sequentially scanned in the specific table.
# Sequential Scan Tuples+	The cumulative number of tuples sequentially scanned in the specific table.
# Tuples Deleted	The number of tuples deleted.
# Tuples Deleted+	The cumulative number of tuples deleted.
# Tuples Estimated by ANALYZE	The number of live (visible) tuples estimated by ANALYZE.
# Tuples Estimated by ANALYZE+	The cumulative number of live tuples estimated by ANALYZE.
# Tuples HOT Updated	The number of tuples HOT updated. In a HOT update, the new tuple resides in the same block as the original tuple and the tuples share an index entry.
# Tuples HOT Updated+	The cumulative number of tuples HOT updated.
# Tuples Inserted	The number of tuples inserted into the specified table.
# Tuples Inserted+	The cumulative number of tuples inserted into the specified table.
# Tuples Updated	The number of tuples updated in the selected table.
# Tuples Updated+	The cumulative number of tuples updated in the selected table.
Blocks Hit	The number of blocks found in the cache.
Blocks Hit+	The cumulative number of blocks found in the cache.
Blocks Read	The number of blocks read.
Blocks Read+	The cumulative number of blocks read.
Blocks Read from InfiniteCache	The number of blocks read from InfiniteCache.

Metric Name	Description
Blocks Read from InfiniteCache+	The cumulative number of blocks read from InfiniteCache.
Blocks Written	The number of blocks written.
Blocks Written+	The cumulative number of blocks written.
Buffers Allocated	The number of buffers allocated.
Buffers Allocated+	The cumulative number of buffers allocated.
Buffers Written - Backends	The number of buffer blocks written to disk by server processes (processes connected to a client application).
Buffers Written - Backends+	The cumulative number of buffer blocks written to disk by server processes.
Buffers Written - Checkpoint	The number of blocks written to disk by the checkpoint process.
Buffers Written - Checkpoint+	The cumulative number of blocks written to disk by the checkpoint process.
Buffers Written - Cleaning Scan	The number of blocks written to disk by the autovacuum process.
Buffers Written - Cleaning Scan+	The cumulative number of blocks written to disk by the autovacuum process.
Bytes Received (KB)	The number of bytes received from the client (in kilobytes).
Bytes Received (KB)+	The cumulative number of bytes received (in kilobytes).
Bytes Sent (KB)	The number of bytes sent to the client (in kilobytes).
Bytes Sent (KB)+	The cumulative number of bytes sent (in kilobytes).
Checkpoints - Timed	The number of checkpoint operations triggered by the checkpoint interval.
Checkpoints - Timed+	The cumulative number of checkpoint operations triggered by the checkpoint interval.
Checkpoints - Untimed	The number of checkpoint operations triggered by checkpoint size.
Checkpoints - Untimed+	The cumulative number of checkpoint operations triggered by checkpoint size.
Database Size (MB)	The size of the specified database (in megabytes).
Free RAM Memory	The amount of free RAM memory (in megabytes).
Free Swap Memory	The amount of free swap space on disk (in megabytes).
Heap Blocks Hit	The number of heap blocks found in the cache.
Heap Blocks Hit+	The cumulative number of heap blocks found in the cache.
Heap Blocks Read	The number of heap blocks read.
Heap Blocks Read+	The cumulative number of heap blocks read.
Index Blocks Hit	The number of index blocks found in the cache.
Index Blocks Hit+	The cumulative number of index blocks found in the cache.
Index Blocks Read	The number of index blocks read.
Index Blocks Read+	The cumulative number of index blocks read.
Index Size (MB)	The size of the specified index (in megabytes).
In Packets Discards	The number of inbound packets discarded.
In Packets Discards+	The cumulative number of inbound packets discarded.
In Packets Errors	The number of inbound packets that contain errors.
In Packets Errors+	The cumulative number of inbound packets that contain errors.

Metric Name	Description
Link Bandwidth (Mbit/s)	The speed of the network adapter (in megabits per second).
Load Average - 15 Minute	CPU saturation (in percent) - 15 minute sampling average.
Load Average - 1 Minute	CPU saturation (in percent) - 1 minute sampling average.
Load Average - 5 Minute	CPU saturation (in percent) - 5 minute sampling average.
Load Percentage	CPU saturation in percent.
Number of Prepared Transactions+	The cumulative number of prepared transactions.
Number of WAL Files+	The cumulative number of write-ahead log files.
Out Packets Discards	The number of outbound packets discarded.
Out Packets Discards+	The cumulative number of outbound packets discarded.
Out Packets Errors	The number of outbound packets that contain errors.
Out Packets Errors+	The cumulative number of outbound packets that contain errors.
Packets Received	The number of packets received.
Packets Received+	The cumulative number of packets received.
Packets Sent	The number of packets sent.
Packets Sent+	The cumulative number of packets sent.
Size (MB)	The total size of the disk (in megabytes).
Size of Indexes (MB)	The size of indexes on the specified table (in megabytes).
Space Available (MB)	The current disk space available (in megabytes).
Space Used (MB)	The current disk space used (in megabytes).
Table Size (MB)	The size of the specified table (in megabytes).
Tablespace Size (MB)	The size of the specified tablespace (in megabytes).
Temp Buffers (MB)	The size of temporary buffers (in megabytes).
Toast Blocks Hit	The number of TOAST blocks found in the cache.
Toast Blocks Hit+	The cumulative number of TOAST blocks found in the cache.
Toast Blocks Read	The number of TOAST blocks read.
Toast Blocks Read+	The cumulative number of TOAST blocks read.
Total RAM Memory	The total amount of RAM memory on the system (in megabytes).
Total Swap Memory	The total amount of swap space on the system (in megabytes).
Total Table Size w/Indexes and Toast	The total size of the specified table (including indexes and associated oversized attributes).
Transactions Aborted	The number of aborted transactions.
Transactions Aborted+	The cumulative number of aborted transactions.
Transactions Committed	The number of committed transactions.
Transactions Committed+	The cumulative number of committed transactions.
Tuples Deleted	The number of tuples deleted from the specified table.
Tuples Deleted+	The cumulative number of tuples deleted from the specified table.

Metric Name	Description
Tuples Estimated by ANALYZE	The number of visible tuples in the specified table.
Tuples Estimated by ANALYZE+	The cumulative number of visible tuples in the specified table.
Tuples Fetched	The number of tuples fetched from the specified table.
Tuples Fetched+	The cumulative number of tuples fetched from the specified table.
Tuples HOT Updated	The number of tuples HOT updated. In a HOT update, the new tuple resides in the same block as the original tuple and the tuples share an index entry.
Tuples HOT Updated+	The cumulative number of tuples HOT updated. In a HOT update, the new tuple resides in the same block as the original tuple and the tuples share an index entry.
Tuples Inserted	The number of tuples inserted into the specified table.
Tuples Inserted+	The cumulative number of tuples inserted into the specified table.
Tuples Returned	The number of tuples returned in result sets.
Tuples Returned+	The cumulative number of tuples returned in result sets.
Tuples Updated	The number of tuples updated in the specified table.
Tuples Updated+	The cumulative number of tuples updated in the specified table.
WAL Segment Size (MB)	The segment size of the write-ahead log (in megabytes).

!!! Note The '+' following the name of a metric signifies that the data for the metric is gathered cumulatively; those metrics that are not followed by the '+' sign are collected as a 'point-in-time' value.

Contents:

7.4.8.1 Tab 1 (Metrics)

To create a Capacity Manager report, expand the tree control on the **Metrics** tab to locate the metrics that are available for the node that you wish to analyze.



To include a metric in the Capacity Manager report, check the box to the left of the name of the metric on the **Metrics** tab.



Capacity Manager will use the aggregation method specified by the **Aggregation** drop-down listbox (located at the bottom of the **Metrics** tab). The aggregation method instructs Capacity Manager how to evaluate and plot the metric values. Select from:

- Average: Use the average of the values recorded during the time period.
- Maximum: Use the maximum value recorded during the time period.
- Minimum: Use the minimum value recorded during the time period.
- First: Use the first value recorded during the time period.

To remove a metric from the Capacity Manager report, uncheck the box to the left of the name of a metric.

Move the slider next to **Graph/chart metrics individually?** to **Yes** to instruct Capacity Manager to produce a separate report for each metric selected on the **Metrics** tab. If the option is set to **No**, all selected metrics will be merged into a single graph or table.

Click the **Generate** button to display the report onscreen (accepting the default configuration options), or continue to the **Options** tab to specify sampling boundaries, report type and report destination.

7.4.8.2 Tab 2 (Options)

Use the fields on the **Options** tab to specify the starting and ending boundaries of the Capacity Manager report, the type of report generated, and the location to which the report will be displayed or written.



Use the fields within the **Time Period** box to define the boundaries of the Capacity Manager report:

- Use the **Period** drop-down listbox to select the type of time period you wish to use for the report. You can select:

Value	Description
Start time and end time	Specify a start and end date and time for the report.
Start time and threshold	Specify a start date and time, and a threshold to determine the end time and date for the report.
Historical days and extrapolated days	Specify a start date for the report that is a number of days in the past, and an end date that is a number of days in the future. This option is useful for report templates that do not specify fixed dates.
Historical days and threshold	Specify a start date that is a number of days in the past, and end it when a threshold value is reached.

After specifying the type of time period for the report, select from other options in the **Time Period** box to refine the time period:

- Use the date and time selectors next to the **Start time** field to specify the starting date and time of the sampling period, or select the number of **Historical day(s)** of data to include in the report. By default, Capacity Manager will select a start time that is one week prior to the current date and time. The date and time specified in the **Start time** field must not be later than the current date/time.
- Use the date and time selectors next to the **End time** field to specify an end boundary for the report, or select the number of **Extrapolated day(s)** of data to include in the report. The end boundary can be either a time, a number of days in the future, or the point at which a selected metric reaches a user-specified threshold value. The time specified in the **End time** field must be later than the time specified in the **Start time** field.

Note that if you select an end date and time in the future, Capacity Manager will use historical usage information to extrapolate anticipated future usage. Since the projected usage is based on the sampling of historical data, the accuracy of the future usage trend will improve with a longer sampling period.

To specify a threshold value, use the drop-down listbox in the **Threshold** field to select a metric (from the metrics specified on the **Metrics** tab), an operator (**Exceeds** or **Falls below**), and to enter a target value for the metric. If you choose to define the end of the report using a threshold, the Capacity Manager report will terminate when the value for the selected metric exceeds or falls below the specified value.

Please Note: If you specify a starting boundary that is later than the ending boundary for the report, the status bar will display an error informing you that you must enter a valid time.

The `cm_max_end_date_in_years` configuration parameter defines a default time value for the end boundary of Capacity Manager reports. If you specify a threshold value as the end boundary of a report, and the anticipated usage of the boundary is not met before the maximum time has passed (as specified in the `cm_max_date_in_years` parameter), the report will terminate at the time specified by the `cm_max_date_in_years` parameter. By default, `cm_max_end_date_in_years` is 5; use the `'Server Configuration` dialog <pem_server_config> to modify the value of `cm_max_end_date_in_years`.

Please Note: The PEM client will display time in the PEM client's timezone, rather than the timezone in which the PEM server resides.

Use the fields in the `Report` box to specify the report type and destination.

The radio buttons next to `Include on report` specify the type of report produced by Capacity Manager. Choose from:

- Select `Graph` to instruct Capacity Manager to display the report in the form of a line graph in the PEM client window.
- Select `Table of data` to instruct Capacity Manager to display a table containing the report data in the PEM client window.
- Select `Graph and table of data` to instruct Capacity Manager to display both a line graph and a data table in the PEM client window.

Use the `Report destination` radio buttons to instruct Capacity Manager where to display or save the report:

- Select `New tab` to instruct Capacity Manager to display the report on a new tab in the PEM client. You must select `New tab` to display the first generation of a Capacity Manager report; for subsequent reports, you may select `Previous` tab.
- Select `Previous` tab to instruct Capacity Manager to re-use a previously opened tab when displaying the report.
- Select `Download the report as a file` and specify a file name to instruct Capacity Manager to write the report to the specified file.

Reports saved to file are stored in HTML format. You can review Capacity Manager reports with any web browser that supports Scalable Vector Graphics (SVG). Browsers that do not support SVG will be unable to display Capacity Manager graphs and may include unwanted characters.

When you have specified the report boundaries and selected the type and destination of the Capacity Manager report, click the `Generate` button to create the report.

7.4.9 Alerting

Postgres Enterprise Manager monitors a system for conditions that require user attention. An alert definition contains a system-defined or user-defined set of conditions that PEM compares to the system statistics; if the statistics deviate from the boundaries specified for that statistic, the alert triggers, displaying a `High` (red), `Low` (yellow) or `Medium` (orange) severity warning in the left-most column of the `Alerts Status` table on the `Global Overview` dashboard, and optionally sends a notification via email to `Email Groups` or SNMP trap/notification receivers <snmp_mib_generation>.

Alerts Status							
	Alarm Type	Object Description	Alert Name	Value	Database	Schema	Package
▶	● High	EDB Postgres Advanced Server 11	Database size in server	113 MB			
▶	● High	EDB Postgres Advanced Server 11	Last Vacuum	Never ran			
▶	● High	EDB Postgres Advanced Server 11	Last AutoVacuum	140.21 hrs			
▶	● High	EPAS_12	Table size in server	410 MB			
▶	● Medium	EPAS_12	Last Vacuum	5.18 hrs			
▶	● High	EPAS_12	Database size in server	455 MB			
▶	● Medium	EPAS_12	Last AutoVacuum	5.16 hrs			
▶	● High	N/A	Alert Errors	3			
▶	● High	PGSQL12_Centos7_1	Server Down	1			
▶	● High	PGSQL12_Centos7_1	Last Vacuum	Never ran			
▶	● High	PGSQL12_Centos7_1	Last AutoVacuum	Never ran			
▶	● High	Postgres Enterprise Manager Server	Largest index by table-size percentage	100 %			
▶	● High	Postgres Enterprise Manager Server	Database size in server	2.072265625 GB			
▶	● High	Postgres Enterprise Manager Server	Table size in server	1.9814453125 GB			
▶	● Medium	Postgres Enterprise Manager Server	Connections in idle state	12			
▶	● Medium	Postgres Enterprise Manager Server	Last Vacuum	4.99 hrs			

The PEM server includes a number of pre-defined alerts that are actively monitoring your servers. If the alert definition makes details available about the cause of the alert, you can click the down arrow to the right of the severity warning to access a dialog with detailed information about the condition that triggered the alert. Please note that Alert Details section lists top 10 entries only in the general tab.

Alert Details (Auto-refresh paused whilst rows are expanded. ⏪)

Acked	Alert Type	Name	Value	Agent	Server	Database	Schema	Package	Object	Alerting Since
▶	<input type="checkbox"/> ● High	Table size in server	1.9814453125 GB		Postgres Enterprise Manager Server					2020-02-20 11:29:45

General Parameters

Table name	Schema name	Database name	Total table size(MB)
table_statistics	pemhistory	pem	1087
server_logs	pemdata	pem	263
index_statistics	pemhistory	pem	237
session_info	pemhistory	pem	137
lock_info	pemhistory	pem	88

PEM also provides an interface that allows you to create customized alerts. Each alert uses metrics defined on an alert template. An alert template defines how the server will evaluate the statistics for a resource or metric. The PEM server includes a number of pre-defined alert templates, or you can create custom alert templates.

Using the Alerts Dashboard

Use the **Dashboards** menu (at the top of the **Global Overview** dashboard) to access the **Alerts Dashboard**. The Alerts Dashboard displays a summary of the active alerts and the status of each alert:

The screenshot shows the 'Alerts Overview' section of the EDB Postgres Enterprise Manager. At the top, there's a header with navigation items like 'Postgres Enterprise Manager Host', 'Postgres Enterprise Manager Server', and 'Alerts'. Below the header, a status bar displays 'Object Type: Server', 'Status: UP (Since: 27/04/2020, 15:47:09)', 'Generated On: 27/04/2020, 21:00:47', 'No. of alerts: 5 (Acknowledged: 0)', and a gear icon.

The main area contains three sections:

- Alert Status:** A bar chart titled 'Alert Status' showing the count of alerts for different severities. The y-axis ranges from 0.0 to 12.5. The bars show approximately 3 for High, 2 for Medium, 0 for Low, and 12 for None.
- Alert Details:** A table listing triggered alerts. The columns include Acked, Alert Type, Name, Value, Agent, Server, Database, Schema, Package, Object, and Alerting Since. The data shows five alerts: 'Table size in server' (High, 1.9814453125 GB), 'Database size in server' (High, 2.072265625 GB), 'Largest index by table-size percentage' (High, 100 %), 'Connections in idle state' (Medium, 15), and 'Last Vacuum' (Medium, 4.99 hrs).
- Alert Errors:** A table listing configuration-related errors. The columns include Alert Type, Name, Value, Agent, Server, Database, Schema, Package, Object, Error Message, and Error Timestamp. One error is listed: 'Number of WAL archives pending' (Error) with the message 'Required probe(s) wal_archive_status are disabled.' at 2020-01-21 14:25:04.

The **Alerts Overview** section displays a graphic representation of the active alerts, as well as a count of the current High, Low and Medium alerts. The vertical bar on the left of the graph provides the count of the alerts displayed in each column. Hover over a bar to display the alert count for the selected alert severity in the upper-right hand corner of the **Alerts Status** graph.

The **Alert Details** table provides a list of the alerts that are currently triggered. The entries are prioritized from high-severity to lower-severity; each entry includes information that will allow you to identify the alert and recognize the condition that triggered the alert. Click the name of an alert to review the alert definition, or the down arrow next to the alert icon to review the metrics that triggered the alert.

The **Alert Errors** table displays configuration-related errors (eg. accidentally disabling a required probe, or improperly configuring an alert parameter). You can use the information provided in the **Error Message** column to identify and resolve the conflict that is causing the error; for additional assistance, contact [EnterpriseDB Support](#).

Managing Alerts

PEM's **Manage Alerts** tab allows you to define custom alerts or modify existing alerts. To open the **Manage Alerts tab**, select **Manage Alerts...** from the **Management** menu. The Manage Alerts tab provides an easy way to review the alerts that are currently defined for the object that is highlighted in the PEM client tree control; simply select an object to see the alerts that are defined for that object.

Manage Alerts							
Name	Auto created?	Template	Enable?	Interval		History retention	
				Default?	Minutes	Default?	Days
Average table bloat in server	Yes	Average table bloat in server	Yes	Yes	1	Yes	30
Connections in idle-in-transaction state	Yes	Connections in idle-in-transaction state	Yes	Yes	1	Yes	30
Connections in idle-in-transaction state, a...	Yes	Connections in idle-in-transaction state, a...	Yes	Yes	1	Yes	30
Connections in idle state	Yes	Connections in idle state	Yes	Yes	1	Yes	30
Database size in server	Yes	Database size in server	Yes	Yes	1	Yes	30
Highest table bloat in server	Yes	Highest table bloat in server	Yes	Yes	1	Yes	30
Largest index by table-size percentage	Yes	Largest index by table-size percentage	Yes	Yes	1	Yes	30
Last AutoVacuum	Yes	Last AutoVacuum	Yes	Yes	1	Yes	30
Last Vacuum	Yes	Last Vacuum	Yes	Yes	1	Yes	30

The **Manage Alerts** tab also provides **Quick Links** that provide quick access to dialogs that allow you to:

- [Copy an alert](#) from one object to one or more objects.
- [Create or modify an alert template](#).
- [Create or Modify an email group](#).
- [Manage PEM Server configuration](#) details.
- Access the PEM online help.

You can configure an alert to notify Nagios network-alerting software when that alert is triggered. For more information, see [Using PEM with Nagios](#).

To [create a new alert](#), click the add icon in the upper-right corner of the **Alerts** table.

Contents:

7.4.9.1 Creating and Managing Alerts

Use options accessed through the **Manage Alerts** tab to create, copy, or modify an alert. To open the **Manage Alerts** tab, select **Manage Alerts...** from the PEM client's **Management** menu.

Manage Alerts							
Name	Auto created?	Template	Enable?	Interval		History retention	
				Default?	Minutes	Default?	Days
Average table bloat in server	Yes	Average table bloat in server	Yes	Yes	1	Yes	30
Connections in idle-in-transaction state	Yes	Connections in idle-in-transaction state	Yes	Yes	1	Yes	30
Connections in idle-in-transaction state, a...	Yes	Connections in idle-in-transaction state, a...	Yes	Yes	1	Yes	30
Connections in idle state	Yes	Connections in idle state	Yes	Yes	1	Yes	30
Database size in server	Yes	Database size in server	Yes	Yes	1	Yes	30
Highest table bloat in server	Yes	Highest table bloat in server	Yes	Yes	1	Yes	30
Largest index by table-size percentage	Yes	Largest index by table-size percentage	Yes	Yes	1	Yes	30
Last AutoVacuum	Yes	Last AutoVacuum	Yes	Yes	1	Yes	30
Last Vacuum	Yes	Last Vacuum	Yes	Yes	1	Yes	30

Use the **Quick Links** toolbar to open dialogs and tabs that you can use to manage alerts and alerting behavior:

- Select **Copy Alerts** to open the [Copy Alert Configuration](#) dialog and copy an alert definition.

- Select **Alert Templates** to open the **Alert Template** tab, and create or modify an alert template.
- Select **Email Groups** to open the **Email Groups** tab, and manage or create an email group.
- Select **Webhooks** to open the **Webhooks** tab, and manage or create a webhook endpoint.
- Select **Server Configuration** to open the **server configuration** dialog and review or modify server configuration settings.
- Select **Help** to open the PEM online help.

The **Alerts** table displays the alerts that are defined for the item currently highlighted in the PEM client tree control. You can use the **Alerts** table to modify an existing alert, or to create a new alert.

Creating a New Alert

To open the alert definition dialog and create a new alert, click the **Add** icon (+) in the upper-right corner of the table.

The screenshot shows the 'General' tab of the Alert Definition dialog. The 'General' tab is selected and highlighted in red. Other tabs like 'Notification' and 'Script execution' are visible but not selected. The 'Name' field is empty and highlighted with a red border. The 'Description' field is also empty. The 'Template' section includes a dropdown menu labeled 'Select from the list' with a note about templates generating values for comparison. The 'Enable?' switch is set to 'Yes'. The 'Interval' section has a 'Default?' switch set to 'Yes' and a 'Minutes' selector set to '1'. Below it is a note about confirming alert conditions. The 'History retention' section has a 'Default?' switch set to 'Yes' and a 'Days' selector set to '30'. Below it is a note about storing data for history. The 'Threshold values' section contains an 'Operator' dropdown set to '>', 'Low' and 'High' input fields, and a 'Unit' dropdown. Below it is a note about triggering criteria. The 'Parameter Options' section shows a table with a single row for 'server' with value 'Postgres Enterprise Manager Server'. There is also a 'No' button for 'Auto created'.

Use the fields on the **General** tab to provide information about the alert:

- Enter the name of the alert in the **Name** field.
- Use the drop-down listbox in the **Template** field to select a template for the alert. An alert template is a function that uses one (or more) metrics or parameters to generate a value to which PEM compares user-specified alert boundaries. If the value returned by the template function evaluates to a value that is within the boundary of a user-defined alert (as specified by the **Operator** and **Threshold values** fields), PEM raises an alert, adds a notice to the **Alerts overview** display, and performs any actions specified on the template.
- Use the **Enable?** switch to specify if the alert is enabled (**Yes**) or disabled (**No**).
- Use the controls in the **Interval** box to specify how often the alert should confirm if the alert conditions are satisfied. Use the **Minutes** selector to specify an interval value. Use the **Default** switch to set or reset the **Minutes** value to the default (recommended) value for the selected template.
- Use controls in the **History retention** box to specify the number of days that PEM will store data collected by the alert. Use the **Days** selector to specify the number of days that the data will be stored. Use the **Default** switch to set or reset the **Days** value to the default value (30 days).
- Use controls in the **Threshold values** box to define the triggering criteria for the alert. When the value specified

in the **Threshold Values** fields evaluates to greater-than or less-than the system value (as specified with the **Operator**), PEM will raise a **Low**, **Medium** or **High** level alert:

- Use the **Operator** drop-down listbox to select the operator that PEM will use when evaluating the current system values.
 - Select a greater-than sign (>) to indicate that the alert should be triggered when the system values are greater than the values entered in the **Threshold values** fields.
 - Select a less-than sign (<) to indicate that the alert should be triggered when the system values are less than the values entered in the **Threshold values** fields.
- Use the threshold fields to specify the values that PEM will compare to the system values to determine if an alert should be raised. Please note that you must specify values for all three thresholds (**Low**, **Medium**, and **High**):
 - Enter a value that will trigger a low-severity alert in the **Low** field.
 - Enter a value that will trigger a medium-severity alert in the **Medium** field.
 - Enter a value that will trigger a high-severity alert in the **High** field.

The **Parameter Options** table contains a list of parameters that are required by the selected template; the table displays both pre-defined parameters, and parameters for which you must specify a value. Please note that you must specify a value for any parameter that displays a prompt in the **Value** column.

Use the **Notification** tab to specify how PEM will behave if an alert is raised.

PEM can send a notification or execute a script if an alert is triggered, or if an alert is cleared.

Use the fields in the **Email** tab to specify the email group that will receive an email notification if the alert is triggered at the specified level. Use the **Email Groups** tab to create an email group that contains the address of the user or users that will be notified when an alert is triggered. To access the **Email Groups** tab, click the **Email Groups** icon located in the **Quick Links** menu of the **Manage Alerts** tab.

To instruct PEM to send an email when a specific alert level is reached, set the slider next to an alert level to **Yes**, and use the drop-down listbox to select the pre-defined user or group that will be notified.

Please note that you must [configure the PEM Server](#) to use an SMTP server to deliver email before PEM can send email notifications.

The screenshot shows the 'Webhook' tab selected in the top navigation bar. Under 'Enable?' is a 'Yes' button. Under 'Override default configuration?' is a 'No' button. The 'Alert types' section contains four rows: 'Low alerts?' with 'Jira', 'Medium alerts?' with 'GitHub Jira', 'High alerts?' with 'Slack GitHub Jira', and 'Cleared alerts?' with 'GitHub Jira'. Each row has an 'x' button to remove the endpoint.

Use the fields in the **Webhook** tab to specify the webhook endpoints that will receive a notification if the alert is triggered at the specified level. Use the **Webhooks** tab to create an endpoint that contains the details of URL that will be notified when an alert is triggered along with other details like payload. To access the **Webhooks** tab, click the **Webhooks** icon located in **Quick Links** menu of the **Manage Alerts** tab.

By default **Webhook** notifications will be sent to created endpoints according to their default settings. To disable the **Webhook** set the slider next to **Enable** field to **No**.

Also to override default settings set the slider next to **Override default configuration?** to **Yes**, and use the drop-down listbox to select the pre-defined endpoints.

The screenshot shows the 'SNMP' tab selected in the top navigation bar. Under 'Trap notification': 'Send trap?' is set to 'No', 'SNMP version' is set to 'v3', and 'Low alert?' is set to 'No'. Below these are 'Medium alert?' and 'High alert?' both set to 'No'.

Use the **Trap notification** options to configure trap notifications for this alert:

- Set the **Send trap** slider to **Yes** to send SNMP trap notifications when the state of this alert changes.
- Set the **SNMP Ver** to **v1**, **v2**, or **v3** to identify the SNMP version.
- Use the **Low alert**, **Med alert** and **High alert** sliders to select the level(s) of alert that will trigger the trap. For example, if you set the slider next to **High alert** to **Yes**, PEM will send a notification when an alert with a high severity level is triggered.

Please note that you must [configure the PEM Server](#) to send notifications to an SNMP trap/notification receiver before notifications can be sent. For sending SNMP v3 traps, pemAgent will use 'User Security Model(USM)' which is in charge of authenticating, encrypting, and decrypting SNMP packets.

Also note while sending SNMP v3 traps, agent will create `snmp_boot_counter` file. This file will get created in location mentioned by `batch_script_dir` parameter in `agent.cfg`, if this parameter is not configured or if directory is not accessible due to authentication restrictions then in operating systems temporary directory, if that is also not possible then in user's home directory.

Please see How SNMP traps are formed? <[snmp_trap_details](#)>

General Notification Script execution

Email Webhook SNMP Nagios

Set "Submit passive service check result to Nagios" to "Yes" to instruct the PEM server to notify Nagios when the alert is triggered or cleared.

Nagios notification

Submit passive service check result to Nagios? No

Use the field in the **Nagios notification** box to instruct the PEM server to notify Nagios network-alerting software when the alert is triggered or cleared. For detailed information about configuring and using Nagios with PEM, please see [Using PEM with Nagios](#).

- Set the **Submit passive service check result to Nagios** switch to **Yes** to instruct the PEM server to notify Nagios when the alert is triggered or cleared.

General Notification **Script execution**

Script execution

Execute script? No Execute on alert cleared? No

PEM Server Monitored Server

Code

Use the fields in the Script execution box to (optionally) define a script that will be executed if an alert is triggered, and to specify details about the script execution.

- Set the Execute script slider to Yes to instruct PEM to execute the provided script if an alert is triggered.
- Set the Execute on alert cleared slider to Yes to instruct PEM to execute the provided script when the situation that triggered the alert has been resolved.
- Use the selector to indicate if the script should execute on the PEM Server or the Monitored Server.
- Provide the script that PEM should execute in the Code field. You can provide a batch/shell script, or SQL code. Within the script you can use the placeholders to replace the following:
 - %AlertID% - the id of the triggered alert.
 - %AlertName% - the name of the triggered alert.
 - %ObjectName% - the name of the server or agent on which the alert was triggered.
 - %ObjectType% - the type on which alert was generated.
 - %ThresholdValue% - the threshold value reached by the metric when the alert triggered.
 - %CurrentValue% - the current value of the metric that triggered the alert.
 - %CurrentState% - the current state of the alert.
 - %OldState% - the previous state of the alert.
 - %AlertRaisedTime% - the time that the alert was raised, or the most recent time that the alert state was changed.
 - %AgentID% - the id of the agent by which alert was generated.
 - %AgentName% - the name of the agent by which alert was generated.
 - %ServerID% - the id of the server on which alert was generated.
 - %ServerName% - the name of the server on which alert was generated.
 - %ServerIP% - the ip or address of the server on which alert was generated.
 - %ServerPort% - the port of the server on which alert was generated.
 - %DatabaseName% - the name of the database on which alert was generated.
 - %SchemaName% - the name of the schema on which alert was generated.
 - %PackageName% - the name of the package on which alert was generated.
 - %DatabaseObjectName% - the name of the database object name like table name, function name etc on which alert was generated.
 - %Parameters% - the list of custom parameters used to generate the alert.
 - %AlertInfo% - the detailed database object level information of the alert.

Use the fields in the **Script execution** tab to (optionally) define a script that will be executed if an alert is triggered, and to specify details about the script execution.

- Set the **Execute script** slider to **Yes** to instruct PEM to execute the provided script if an alert is triggered.
- Set the **Execute on alert cleared** slider to **Yes** to instruct PEM to execute the provided script when the situation that triggered the alert has been resolved.
- Use the radio buttons next to **Execute script on** to indicate that the script should execute on the **PEM Server** or the **Monitored Server**.
- Provide the script that PEM should execute in the **Code** field. You can provide a batch/shell script, or SQL code. Within the script, you can use placeholders for the following:
 - %AlertID%** - this placeholder will be replaced with the id of the triggered alert.
 - %AlertName%** - this placeholder will be replaced with the name of the triggered alert.
 - %ObjectName%** - this placeholder will be replaced with the name of the server or agent on which the alert was triggered.
 - %ObjectType%** - this placeholder will be replaced with the type of the object on which the alert was triggered.

- `%ThresholdValue%` - this placeholder will be replaced with the threshold value reached by the metric when the alert triggered.
- `%CurrentValue%` - this placeholder will be replaced with the current value of the metric that triggered the alert.
- `%CurrentState%` - this placeholder will be replaced with the current state of the alert.
- `%OldState%` - this placeholder will be replaced with the previous state of the alert.
- `%AlertRaisedTime%` - this placeholder will be replaced with the time that the alert was raised, or the most recent time that the alert state was changed.
- `%AgentID%` - this placeholder will be replaced with the id of the agent by which alert was generated.
- `%AgentName%` - this placeholder will be replaced with the name of the agent by which alert was generated.
- `%ServerID%` - this placeholder will be replaced with the id of the server on which alert was generated.
- `%ServerName%` - this placeholder will be replaced with the name of the server on which alert was generated.
- `%ServerIP%` - this placeholder will be replaced with the IP or address of the server on which alert was generated.
- `%ServerPort%` - this placeholder will be replaced with the port of the server on which alert was generated.
- `%DatabaseName%` - this placeholder will be replaced with the name of the database on which alert was generated.
- `%SchemaName%` - this placeholder will be replaced with the name of the schema on which alert was generated.
- `%PackageName%` - this placeholder will be replaced with the name of the package on which alert was generated.
- `%DatabaseObjectName%` - this placeholder will be replaced with the name of the database object on which alert was generated.
- `%Parameters%` - this placeholder will be replaced with the list of custom parameters used to generate the alert.
- `%AlertInfo%` - this placeholder will be replaced with the detailed database object level information of the alert.

When you have defined the alert attributes, click the edit icon to close the alert definition editor, and then the save icon (in the upper-right corner of the `Alerts` table). To discard your changes, click the refresh icon; a popup will ask you to confirm that you wish to discard the changes.

Note

Suppose you need to use the alert configuration placeholder values in an external script. You can do it either by passing them as the command-line arguments or exporting them as environment variables. Please note that the external script must have proper execution permissions.

- You can run the script with any of the placeholders as command-line argument.

For eg:

```
#!/bin/bash

bash <path_to_script>/script.sh "%AlertName% %AlertLevel% %AlertDetails%"
```

- You can define the environment variables for any of the placeholders and then use those environment variables in the script.

For eg:

```
#!/bin/bash

export AlertName=%AlertName%
export AlertState=%AlertState%

bash <path_to_script>/script.sh
```

Modifying an Existing Alert

Use the **Alerts** table to manage an existing alert or create a new alert. Highlight an object in the PEM client tree control to view the alerts that monitor that object.

	Name	Auto created?	Template	Enable?	Interval		History retention	
					Default?	Minutes	Default?	Days
<input checked="" type="checkbox"/>	Audit config mismatch	<input checked="" type="checkbox"/>	Audit config mismatch	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Average table bloat in server	<input checked="" type="checkbox"/>	Average table bloat in server	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Connections in idle-in-transaction state	<input checked="" type="checkbox"/>	Connections in idle-in-transaction state	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Connections in idle-in-transaction state, as a percentage	<input checked="" type="checkbox"/>	Connections in idle-in-transaction state, as a percentage	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Connections in idle state	<input checked="" type="checkbox"/>	Connections in idle state	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Database size in server	<input checked="" type="checkbox"/>	Database size in server	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Highest table bloat in server	<input checked="" type="checkbox"/>	Highest table bloat in server	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Largest index by table-size percentage	<input checked="" type="checkbox"/>	Largest index by table-size percentage	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Last AutoVacuum	<input checked="" type="checkbox"/>	Last AutoVacuum	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Last Vacuum	<input checked="" type="checkbox"/>	Last Vacuum	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Log config mismatch	<input checked="" type="checkbox"/>	Log config mismatch	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Number of prepared transactions	<input checked="" type="checkbox"/>	Number of prepared transactions	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Number of WAL archives pending	<input checked="" type="checkbox"/>	Number of WAL archives pending	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Number of WAL files	<input checked="" type="checkbox"/>	Number of WAL files	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Server Down	<input checked="" type="checkbox"/>	Server Down	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Table_Size_Alerts	<input type="checkbox"/>	Table size in server	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Table size in server	<input checked="" type="checkbox"/>	Table size in server	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Total connections	<input checked="" type="checkbox"/>	Total connections	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Total connections as percentage of max_connections	<input checked="" type="checkbox"/>	Total connections as percentage of max_connections	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30
<input checked="" type="checkbox"/>	Total table bloat in server	<input checked="" type="checkbox"/>	Total table bloat in server	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	1	<input checked="" type="checkbox"/>	30

You can modify some properties of an existing alert in the **Alerts** table:

- The **Name** column displays the name of the alert; to change the alert name, simply replace the name in the table, and click the save icon.
- The **Auto created?** column indicates if the alert definition was automatically created; **Yes** indicates that the alert was created by PEM, and **No** indicates that the alert was manually created.
- The **Template** column displays the name of the alert template that specifies properties used by the alert. You can use the drop-down listbox to change the alert template associated with an alert.
- Use the **Enable?** switch to specify if an alert is enabled (Yes) or disabled (No).
- Use the **Interval** column to specify how often PEM should check to see if the alert conditions are satisfied. Set the **Default** switch to **No** and specify an alternate value (in **Minutes**), or return the **Default** switch to **Yes** to reset the value to its default setting. By default, PEM will check the status of each alert once every minute.
- Use the **History retention** field to specify the number of days that PEM will store data collected by the alert. Set the **Default** switch to **No** and specify an alternate value (in **Days**), or return the **Default** switch to **Yes** to reset the value to its default setting. By default, PEM will recommend storing historical data for 30 days.

Click the **Edit** icon to the left of an alert name to open the **Alert details** editor and access the complete alert definition. After modifying an alert in the editor, click the **Save** button to make your changes persistent.

Deleting an Alert

To mark an alert for deletion, highlight the alert name in the **Alerts** table and click the delete icon to the left of the name; the alert will remain in the list, but in red strike-through font.

Alerts

Name	Auto created?	Template	Enable?	Interval		History retention	
				Default?	Minutes	Default?	Days
Audit config mismatch	Yes	Audit config mismatch	Yes	Yes	1	Yes	30
Average table bloat in server	Yes	Average table bloat in server	Yes	Yes	1	Yes	30
Connections in idle-in-transaction state	Yes	Connections in idle-in-transaction state	Yes	Yes	1	Yes	30
Connections in idle-in-transaction state, as a percentage of max_connections	Yes	Connections in idle-in-transaction state, as a percentage of max_connections	Yes	Yes	1	Yes	30
Connections in idle state	Yes	Connections in idle state	Yes	Yes	1	Yes	30
Database size in server	Yes	Database size in server	Yes	Yes	1	Yes	30
Highest table bloat in server	Yes	Highest table bloat in server	Yes	Yes	1	Yes	30
Largest index by table-size percentage	Yes	Largest index by table-size percentage	Yes	Yes	1	Yes	30
Last AutoVacuum	Yes	Last AutoVacuum	Yes	Yes	1	Yes	30
Last Vacuum	Yes	Last Vacuum	Yes	Yes	1	Yes	30
Log config mismatch	Yes	Log config mismatch	Yes	Yes	1	Yes	30
Number of prepared transactions	Yes	Number of prepared transactions	Yes	Yes	1	Yes	30
Number of WAL archives pending	Yes	Number of WAL archives pending	Yes	Yes	1	Yes	30
Number of WAL files	Yes	Number of WAL files	Yes	Yes	1	Yes	30
Server Down	Yes	Server Down	Yes	Yes	1	Yes	30
Table_Size_Alerts	No	Table size in server	Yes	Yes	1	Yes	30
Table size in server	Yes	Table size in server	Yes	Yes	1	Yes	30
Total connections	Yes	Total connections	Yes	Yes	1	Yes	30
Total connections as percentage of max_connections	Yes	Total connections as percentage of max_connections	Yes	Yes	1	Yes	30
Total table bloat in server	Yes	Total table bloat in server	Yes	Yes	1	Yes	30

The delete icon acts as a toggle; you can undo the deletion by clicking the delete icon a second time; when you click the Save icon, the alert definition will be permanently deleted.

Example

The screen shown below defines an alert (named System Usage High) that monitors the committed transactions on the system:

General	Notification	Script execution
Name	System Usage High	
Description	The average space wasted by tables on host, in MB.	
Template	Average table bloat on host	
Enable?	Yes	
Interval	Default? Yes Minutes 1	
Use fields in the Interval box to specify how often the alert should confirm that alert conditions are satisfied.		
History retention	Default? Yes Days 30	
Use fields in the History retention box to specify the number of days that PEM will store data collected by the alert.		
Threshold values	Operator < Low 20 Medium 10 High 5 Unit MB	

To re-create this example, highlight the name of a PEM Agent in the tree-control, and select **Manage Alerts...** from the PEM client **Management** menu. When the **Manage Alerts** tab opens, click the add icon (+) in the upper-right hand corner of the **Alerts** table to open the alert editor.

Fields on the **General** tab instruct PEM to use the Disk busy percentage template to create the alert. The PEM server will check the free memory available once every minute, and:

- Trigger a low-severity alert if the free memory available drops below 20%
- Trigger a medium-severity alert if the free memory available drops below 10%
- Trigger a high-severity alert if the free memory available drops below 5%

General Notification Script execution

Email Webhook SNMP Nagios

To configure notification for an alert use the below fields to specify one or multiple user group that will receive an email notification if the alert is triggered at the specific level. Use the dropdown list box to select pre-defined group that will be sent an alert of the selected level is triggered. Please note that you must configure the PEM Server to use an SMTP server to deliver email before PEM can send email notifications.

Alert types

All alerts?	<input checked="" type="checkbox"/> No	<input type="checkbox"/> <Default>
Low alerts?	<input checked="" type="checkbox"/> No	<input type="checkbox"/> <Default>
Medium alerts?	<input checked="" type="checkbox"/> No	<input type="checkbox"/> <Default>
High alerts?	<input checked="" type="checkbox"/> No	<input type="checkbox"/> <Default>

General Notification **Script execution**

Script execution

Execute script? Yes Execute on alert cleared? No

Execute script on PEM Server Monitored Server

Code

```
#!/bin/sh
echo "%AlertID% - %AlertName%"
export agentID="%AlertID%"
sh /tmp/notify_third_party.sh "%CurrentValue%" "%AlertRaisedTime%"
```

Use the fields in the Script execution box to (optionally) define a script that will be executed if an alert is triggered, and to specify details about the script execution.

- Set the Execute script slider to Yes to instruct PEM to execute the provided script if an alert is triggered.
- Set the Execute on alert cleared slider to Yes to instruct PEM to execute the provided script when the situation that triggered the the alert has been resolved.
- Use the selector to indicate if the script should execute on the PEM Server or the Monitored Server.
- Provide the script that PEM should execute in the Code field. You can provide a batch/shell script, or SQL code. Within the script you can use the placeholders to replace the following:
 - %AlertID% - the id of the triggered alert.
 - %AlertName% - the name of the triggered alert.
 - %ObjectName% - the name of the server or agent on which the alert was triggered.
 - %ObjectType% - the type on which alert was generated.
 - %ThresholdValue% - the threshold value reached by the metric when the alert triggered.
 - %CurrentValue% - the current value of the metric that triggered the alert.
 - %CurrentState% - the current state of the alert.
 - %OldState% - the previous state of the alert.
 - %AlertRaisedTime% - the time that the alert was raised, or the most recent time that the alert state was changed.
 - %AgentID% - the id of the agent by which alert was generated.
 - %AgentName% - the name of the agent by which alert was generated.
 - %ServerID% - the id of the server on which alert was generated.
 - %ServerName% - the name of the server on which alert was generated.
 - %ServerIP% - the ip or address of the server on which alert was generated.
 - %ServerPort% - the port of the server on which alert was generated.
 - %DatabaseName% - the name of the database on which alert was generated.
 - %SchemaName% - the name of the schema on which alert was generated.
 - %PackageName% - the name of the package on which alert was generated.
 - %DatabaseObjectName% - the name of the database object name like table name, function name etc on which alert was generated.
 - %Parameters% - the list of custom parameters used to generate the alert.
 - %AlertInfo% - the detailed database object level information of the alert.

Fields on the **Notifications** tab instruct PEM to:

- Send an email notification to the **administrator** email group.
- Submit a passive service check result to Nagios.
- Execute the script shown in the **Code** field when the alert is triggered.
 - To invoke a script on a Linux system, you must modify the entry for **batch_script_user** parameter of agent.cfg file and specify the user that should be used to run the script. You can either specify a non-root user or root for this parameter. If you do not specify a user, or the specified user does not exist, then the script will not be executed. Restart the agent after modifying the file. If pemagent is being run by a non-root user then the value of **batch_script_user** will be ignored and the script will be executed by the same non-root user that is being used for running the pemagent.
 - To invoke a script on a Windows system, set the registry entry for **AllowBatchJobSteps** to true and restart

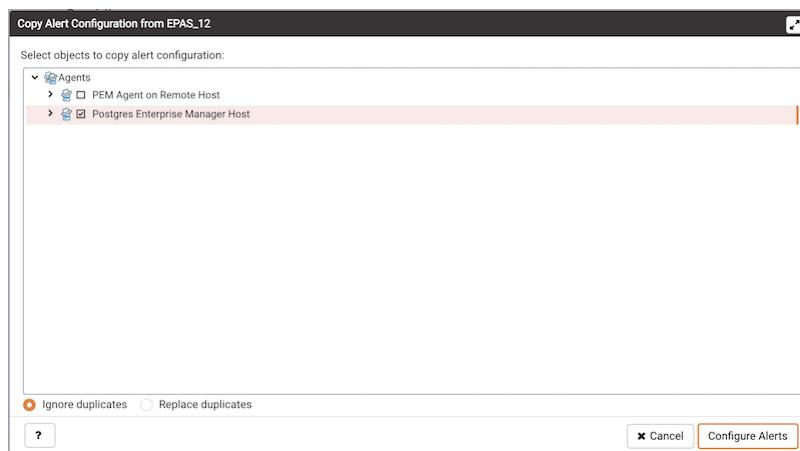
the PEM agent. PEM registry entries are located in
HKEY_LOCAL_MACHINE\Software\Wow6432Node\EnterpriseDB\PEM\agent.

Click the edit icon to close the editor and add the example to the **Alert List**; click the save icon before closing the **Manage Alerts** tab to save your work.

7.4.9.2 Copy Alerts

To speed up the deployment of alerts in the PEM system, you can copy alert definitions from one object to one or more target objects.

To copy alerts from an object, highlight the object name (from which you will copy alerts) in the PEM client tree control, and select the **Manage Alerts...** option from the **Management** menu. When the **Manage Alerts** tab opens, click the **Copy Alerts** icon (located on the **Quick Links** toolbar) to open the **Copy Alert Configuration** dialog.



The **Copy Alert Configuration** dialog copies all alerts from the object highlighted in the PEM client tree control to the object or objects selected on the dialog. Expand the tree control to select a node or nodes to specify the target object(s). Please note that the tree control displays a red warning indicator next to the source object.

To copy alerts to multiple objects at once, select a parent node of the targets. For example, to copy the alerts from one table to all tables in a schema, you can simply select the checkbox next to the schema name. PEM will only copy alerts to targets that are of the same type as the source object.

Check the **Ignore duplicates** radio button to prevent PEM from updating any existing alerts on the target objects with the same name as those being copied. Check the **Replace duplicates** radio button to replace existing alerts with alerts of the same name from the source object.

Click the **Configure Alerts** button to proceed to copy the alerts from the source object to all objects of the same type in, or below those objects selected on the **Copy Alert Configuration** dialog. When the copy is complete, a popup will notify you that the alerts have been copied to the selected target(s).

7.4.9.3 Alert Templates

An alert definition contains a system-defined or user-defined set of conditions that PEM compares to the system statistics; if the statistics deviate from the boundaries specified for that statistic, the alert triggers, and the PEM client displays a warning on the [Alerts Overview](#) page, and optionally sends a notification to a monitoring user.

The table below lists the system-defined alert templates that you can use to create an alert; please note that this list is subject to change, and may vary by system.

Within the table, the alerts are sorted by the target of the alert. The **Template Name** and **Description** columns identify and describe the behavior of the template. If the template **Details** column specifies **Yes**, metrics returned by the alert and alert parameters (if applicable) are accessible in the [Alerts table](#) on the [Global Overview](#) or [Alerts](#) dashboards:

Templates applicable on Agent

Template Name	Description	Details
Load Average (1 minute)	1-minute system load average.	
Load Average (5 minutes)	5-minute system load average.	
Load Average (15 minutes)	15-minute system load average.	
Load Average per CPU Core (1 minutes)	1-minute system load average per CPU core.	
Load Average per CPU Core (5 minutes)	5-minute system load average per CPU core.	
Load Average per CPU Core (15 minutes)	15-minute system load average per CPU core.	
CPU utilization	Average CPU consumption.	
Number of CPUs running higher than a threshold	Number of CPUs running at greater than K% utilization.	Yes
Free memory percentage	Free memory as a percent of total system memory.	
Memory used percentage	Percentage of memory used.	
Swap consumption	Swap space consumed (in megabytes).	
Swap consumption percentage	Percentage of swap area consumed.	
Disk Consumption	Disk space consumed (in megabytes).	
Disk consumption percentage	Percentage of disk consumed.	
Disk Available	Disk space available (in megabytes).	
Disk busy percentage	Percentage of disk busy.	
Most used disk percentage	Percentage used of the most utilized disk on the system.	Yes
Total table bloat on host	The total space wasted by tables on a host, in MB.	
Highest table bloat on host	The most space wasted by a table on a host, in MB.	
Average table bloat on host	The average space wasted by tables on host, in MB.	
Table size on host	The size of tables on host, in MB.	
Database size on host	The size of databases on host, in MB.	
Number of ERRORS in the logfile on agent N in last X hours	The number of ERRORS in the logfile on agent N in last X hours.	
Number of WARNINGS in the logfile on agent N in last X hours	The number of WARNINGS in the logfile on agent N in last X hours.	
Number of WARNINGS or ERRORS in the logfile on agent N in last X hours	The number of WARNINGS or ERRORS in the logfile on agent N in last X hours.	
Package version mismatch	Check for package version mismatch as per catalog.	Yes

Template Name	Description	Details
Total materialized view bloat on host	The total space wasted by materialized views on a host, in MB.	
Highest materialized view bloat on host	The most space wasted by a materialized view on a host, in MB.	
Average materialized view bloat on host	The average space wasted by materialized views on host, in MB.	
Materialized view size on host	The size of materialized views on host, in MB.	
Agent Down	Specified agent is currently down.	

Templates applicable on Server

Template Name	Description	Details
Total table bloat in server	The total space wasted by tables in server, in MB.	
Largest table (by multiple of unbloated size)	Largest table in server, calculated as a multiple of its own estimated unbloated size; exclude tables smaller than N MB.	
Highest table bloat in server	The most space wasted by a table in server, in MB.	Yes
Average table bloat in server	The average space wasted by tables in server, in MB.	
Table size in server	The size of tables in server, in MB.	Yes
Database size in server	The size of databases in server, in MB.	Yes
Number of WAL files	Total number of Write Ahead Log files.	
Number of prepared transactions	Number of transactions in prepared state.	
Total connections	Total number of connections in the server.	Yes
Total connections as percentage of max_connections	Total number of connections in the server as a percentage of maximum connections allowed on server, settings.	
Unused, non-superuser connections	Number of unused, non-superuser connections on the server, user_info, settings.	
Unused, non-superuser connections as percentage of max_connections	Number of unused, non-superuser connections on the server as a percentage of max_connections, user_info, settings.	
Ungranted locks	Number of ungranted locks in server.	
Percentage of buffers written by backends	The percentage of buffers written by backends vs. the total buffers written.	
Percentage of buffers written by checkpoint	The percentage of buffers written by the checkpoints vs. the total buffers written.	
Buffers written per second	Number of buffers written per second, over the last two probe cycles.	
Buffers allocated per second	Number of buffers allocated per second, over the last two probe cycles.	
Connections in idle state	Number of connections in server that are in idle state.	Yes
Connections in idle-in-transaction state	Number of connections in server that are in idle-in-transaction state.	Yes
Connections in idle-in-transaction state,as percentage of max_connections	Number of connections in server that are in idle-in-transaction state, as a percentage of maximum connections allowed on server, settings	
Long-running idle connections	Number of connections in the server that have been idle for more than N seconds.	Yes

Template Name	Description	Details
Long-running idle connections and idle transactions	Number of connections in the server that have been idle or idle-in-transaction for more than N seconds.	Yes
Long-running idle transactions	Number of connections in the server that have been idle in transaction for more than N seconds.	Yes
Long-running transactions	Number of transactions in server that have been running for more than N seconds.	Yes
Long-running queries	Number of queries in server that have been running for more than N seconds. It does not include the long running vacuum or auto vacuum queries.	Yes
Long-running vacuums	Number of vacuum operations in server that have been running for more than N seconds.	Yes
Long-running autovacuums	Number of autovacuum operations in server that have been running for more than N seconds.	Yes
Committed transactions percentage	Percentage of transactions in the server that committed vs. that rolled-back over last N minutes.	
Shared buffers hit percentage	Percentage of block read requests in the server that were satisfied by shared buffers, over last N minutes.	
Tuples inserted	Tuples inserted into server over last N minutes.	
InfiniteCache buffers hit percentage	Percentage of block read requests in the server that were satisfied by InfiniteCache, over last N minutes.	
Tuples fetched	Tuples fetched from server over last N minutes.	
Tuples returned	Tuples returned from server over last N minutes.	
Dead Tuples	Number of estimated dead tuples in server.	
Tuples updated	Tuples updated in server over last N minutes.	
Tuples deleted	Tuples deleted from server over last N minutes.	
Tuples hot updated	Tuples hot updated in server, over last N minutes.	
Sequential Scans	Number of full table scans in server, over last N minutes.	
Index Scans	Number of index scans in server, over last N minutes.	
Hot update percentage	Percentage of hot updates in the server over last N minutes.	
Live Tuples	Number of estimated live tuples in server.	
Dead tuples percentage	Percentage of estimated dead tuples in server.	
Last Vacuum	Hours since last vacuum on the server.	
Last AutoVacuum	Hours since last autovacuum on the server.	
Last Analyze	Hours since last analyze on the server.	
Last AutoAnalyze	Hours since last autoanalyze on the server.	
Percentage of buffers written by backends over last N minutes	The percentage of buffers written by backends vs. the total buffers written over last N minutes.	
Table Count	Total number of tables in server.	
Function Count	Total number of functions in server.	
Sequence Count	Total number of sequences in server.	
A user expires in N days	Number of days before a user's validity expires.	
Index size as a percentage of table size	Size of the indexes in server, as a percentage of their tables' size.	
Largest index by table-size percentage	Largest index in server, calculated as percentage of its table's size, oc_index, table_size.	

Template Name	Description	Details
Number of ERRORS in the logfile on server M in the last X hours	The number of ERRORS in the logfile on server M in last X hours.	
Number of WARNINGS in the logfile on server M in the last X hours	The number of WARNINGS in logfile on server M in the last X hours.	
Number of WARNINGS or ERRORS in the logfile on server M in the last X hours	The number of WARNINGS or ERRORS in the logfile on server M in the last X hours.	
Number of attacks detected in the last N minutes	The number of SQL injection attacks occurred in the last N minutes.	
Number of attacks detected in the last N minutes by username	The number of SQL injection attacks occurred in the last N minutes by username.	
Number of replica servers lag behind the primary by write location	Streaming Replication: number of replica servers lag behind the primary by write location.	Yes
Number of replica servers lag behind the primary by flush location	Streaming Replication: number of replica servers lag behind the primary by flush location.	Yes
Number of replica servers lag behind the primary by replay location	Streaming Replication: number of replica servers lag behind the primary by replay location.	Yes
Replica server lag behind the primary by write location	Streaming Replication: replica server lag behind the primary by write location in MB.	Yes
Replica server lag behind the primary by flush location	Streaming Replication: replica server lag behind the primary by flush location in MB.	Yes
Replica server lag behind the primary by WAL pages	Streaming Replication: replica server lag behind the primary by WAL pages.	
Replica server lag behind the primary by WAL segments	Streaming Replication: replica server lag behind the primary by WAL segments.	
Replica server lag behind the primary by replay location	Streaming Replication: replica server lag behind the primary by replay location in MB.	Yes
Replica server lag behind the primary by size (MB)	Streaming Replication: replica server lag behind the primary by size in MB.	Yes
Total materialized view bloat in server	The total space wasted by materialized views in server, in MB.	
Largest materialized view (by multiple of unbloated size)	Largest materialized view in server, calculated as a multiple of its own estimated unbloated size; exclude materialized views smaller than N MB.	
Highest materialized view bloat in server	The most space wasted by a materialized view in server, in MB.	
Average materialized view bloat in server	The average space wasted by materialized views in server, in MB.	
Materialized view size in server	The size of materialized view in server, in MB.	
View Count	Total number of views in server.	
Materialized View Count	Total number of materialized views in server.	
Audit config mismatch	Check for audit config parameter mismatch	Yes
Server Down	Specified server is currently inaccessible.	
Number of WAL archives pending	Streaming Replication: number of WAL files pending to be replayed at replica.	
Number of minutes lag of replica server from primary server	Streaming Replication: number of minutes replica node is lagging behind the primary node.	

Template Name	Description	Details
Log config mismatch	Check for log config parameter mismatch.	Yes

Templates applicable on Database

Template Name	Description	Details
Total table bloat in database	The total space wasted by tables in database, in MB.	
Largest table (by multiple of unbloated size)	Largest table in database, calculated as a multiple of its own estimated unbloated size; exclude tables smaller than N MB.	
Highest table bloat in database	The most space wasted by a table in database, in MB.	
Average table bloat in database	The average space wasted by tables in database, in MB.	
Table size in database	The size of tables in database, in MB.	Yes
Database size	The size of the database, in MB.	
Total connections	Total number of connections in the database.	Yes
Total connections as percentage of max_connections	Total number of connections in the database as a percentage of maximum connections allowed on server, settings.	
Ungranted locks	Number of ungranted locks in database.	
Connections in idle state	Number of connections in database that are in idle state.	Yes
Connections in idle-in-transaction state	Number of connections in database that are in idle-in-transaction state.	Yes
Connections in idle-in-transaction state,as percentage of max_connections	Number of connections in database that are in idle-in-transaction state, as a percentage of maximum connections allowed on server, settings.	
Long-running idle connections	Number of connections in the database that have been idle for more than N seconds.	Yes
Long-running idle connections and idle transactions	Number of connections in the database that have been idle or idle-in-transaction for more than N seconds.	Yes
Long-running idle transactions	Number of connections in the database that have been idle in transaction for more than N seconds.	Yes
Long-running transactions	Number of transactions in database that have been running for more than N seconds.	Yes
Long-running queries	Number of queries in database that have been running for more than N seconds. It does not include the long running vacuum or auto vacuum queries.	Yes
Long-running vacuums	Number of vacuum operations in database that have been running for more than N seconds.	Yes
Long-running autovacuums	Number of autovacuum operations in database that have been running for more than N seconds.	Yes
Committed transactions percentage	Percentage of transactions in the database that committed vs. that rolled-back over last N minutes.	
Shared buffers hit percentage	Percentage of block read requests in the database that were satisfied by shared buffers, over last N minutes.	
InfiniteCache buffers hit percentage	Percentage of block read requests in the database that were satisfied by InfiniteCache, over last N minutes.	
Tuples fetched	Tuples fetched from database over last N minutes.	
Tuples returned	Tuples returned from database over last N minutes.	
Tuples inserted	Tuples inserted into database over last N minutes.	

Template Name	Description	Details
Tuples updated	Tuples updated in database over last N minutes.	
Tuples deleted	Tuples deleted from database over last N minutes.	
Tuples hot updated	Tuples hot updated in database, over last N minutes.	
Sequential Scans	Number of full table scans in database, over last N minutes.	
Index Scans	Number of index scans in database, over last N minutes.	
Hot update percentage	Percentage of hot updates in the database over last N minutes.	
Live Tuples	Number of estimated live tuples in database.	
Dead Tuples	Number of estimated dead tuples in database.	
Dead tuples percentage	Percentage of estimated dead tuples in database.	
Last Vacuum	Hours since last vacuum on the database.	
Last AutoVacuum	Hours since last autovacuum on the database.	
Last Analyze	Hours since last analyze on the database.	
Last AutoAnalyze	Hours since last autoanalyze on the database.	
Table Count	Total number of tables in database.	
Function Count	Total number of functions in database.	
Sequence Count	Total number of sequences in database.	
Index size as a percentage of table size	Size of the indexes in database, as a percentage of their tables' size.	
Largest index by table-size percentage	Largest index in database, calculated as percentage of its table's size, oc_index, table_size	
Database Frozen XID	The age (in transactions before the current transaction) of the database's frozen transaction ID.	
Number of attacks detected in the last N minutes	The number of SQL injection attacks occurred in the last N minutes.	
Number of attacks detected in the last N minutes by username	The number of SQL injection attacks occurred in the last N minutes by username.	
Queries that have been cancelled due to dropped tablespaces	Streaming Replication: number of queries that have been cancelled due to dropped tablespaces.	
Queries that have been cancelled due to lock timeouts	Streaming Replication: number of queries that have been cancelled due to lock timeouts.	
Queries that have been cancelled due to old snapshots	Streaming Replication: number of queries that have been cancelled due to old snapshots.	
Queries that have been cancelled due to pinned buffers	Streaming Replication: number of queries that have been cancelled due to pinned buffers.	
Queries that have been cancelled due to deadlocks	Streaming Replication: number of queries that have been cancelled due to deadlocks.	
Total events lagging in all slony clusters	Slony Replication: total events lagging in all slony clusters.	Yes
Events lagging in one slony cluster	Slony Replication: events lagging in one slony cluster.	
Lag time (minutes) in one slony cluster	Slony Replication: lag time (minutes) in one slony cluster.	
Total rows lagging in xdb single primary replication	xDB Replication: Total rows lagging in xdb single primary replication	Yes
Total rows lagging in xdb multi primary replication	xDB Replication: Total rows lagging in xdb multi primary replication	Yes

Template Name	Description	Details
Total materialized view bloat in database	The total space wasted by materialized views in database, in MB.	
Largest materialized view (by multiple of unbloated size)	Largest materialized view in database, calculated as a multiple of its own estimated unbloated size; exclude materialized views smaller than N MB.	
Highest materialized view bloat in database	The most space wasted by a materialized view in database, in MB.	
Average materialized view bloat in database	The average space wasted by materialized views in database, in MB.	
Materialized view size in database	The size of materialized view in database, in MB.	
View Count	Total number of views in database.	
Materialized View Count	Total number of materialized views in database.	

Templates applicable on Schema

Template Name	Description	Details
Total table bloat in schema	The total space wasted by tables in schema, in MB.	Yes
Largest table (by multiple of unbloated size)	Largest table in schema, calculated as a multiple of its own estimated unbloated size; exclude tables smaller than N MB.	
Highest table bloat in schema	The most space wasted by a table in schema, in MB.	
Average table bloat in schema	The average space wasted by tables in schema, in MB.	
Table size in schema	The size of tables in schema, in MB.	Yes
Tuples inserted	Tuples inserted in schema over last N minutes.	
Tuples updated	Tuples updated in schema over last N minutes.	
Tuples deleted	Tuples deleted from schema over last N minutes.	
Tuples hot updated	Tuples hot updated in schema, over last N minutes.	
Sequential Scans	Number of full table scans in schema, over last N minutes.	
Index Scans	Number of index scans in schema, over last N minutes.	
Hot update percentage	Percentage of hot updates in the schema over last N minutes.	
Live Tuples	Number of estimated live tuples in schema.	
Dead Tuples	Number of estimated dead tuples in schema.	
Dead tuples percentage	Percentage of estimated dead tuples in schema.	
Last Vacuum	Hours since last vacuum on the schema.	
Last AutoVacuum	Hours since last autovacuum on the schema.	
Last Analyze	Hours since last analyze on the schema.	
Last AutoAnalyze	Hours since last autoanalyze on the schema.	
Table Count	Total number of tables in schema.	
Function Count	Total number of functions in schema.	
Sequence Count	Total number of sequences in schema.	
Index size as a percentage of table size	Size of the indexes in schema, as a percentage of their table's size.	
Largest index by table-size percentage	Largest index in schema, calculated as percentage of its table's size, oc_index, table_size	

Template Name	Description	Details
Materialized View bloat	Space wasted by the materialized view, in MB.	
Total materialized view bloat in schema	The total space wasted by materialized views in schema, in MB.	
Materialized view size as a multiple of unbloated size	Size of the materialized view as a multiple of estimated unbloated size.	
Largest materialized view (by multiple of unbloated size)	Largest materialized view in schema, calculated as a multiple of its own estimated unbloated size; exclude materialized view smaller than N MB.	
Highest materialized view bloat in schema	The most space wasted by a materialized view in schema, in MB.	
Average materialized view bloat in schema	The average space wasted by materialized views in schema, in MB.	
Materialized view size	The size of materialized view, in MB.	
Materialized view size in schema	The size of materialized views in schema, in MB.	
View Count	Total number of views in schema.	
Materialized View Count	Total number of materialized views in schema.	
Materialized View Frozen XID	The age (in transactions before the current transaction) of the materialized view's frozen transaction ID.	

Templates applicable on Table

Template Name	Description	Details
Table bloat	Space wasted by the table, in MB.	
Table size	The size of table, in MB.	
Table size as a multiple of unbloated size	Size of the table as a multiple of estimated unbloated size.	
Tuples inserted	Tuples inserted in table over last N minutes.	
Tuples updated	Tuples updated in table over last N minutes.	
Tuples deleted	Tuples deleted from table over last N minutes.	
Tuples hot updated	Tuples hot updated in table, over last N minutes.	
Sequential Scans	Number of full table scans on table, over last N minutes.	
Index Scans	Number of index scans on table, over last N minutes.	
Hot update percentage	Percentage of hot updates in the table over last N minutes.	
Live Tuples	Number of estimated live tuples in table.	
Dead Tuples	Number of estimated dead tuples in table.	
Dead tuples percentage	Percentage of estimated dead tuples in table.	
Last Vacuum	Hours since last vacuum on the table.	
Last AutoVacuum	Hours since last autovacuum on the table.	
Last Analyze	Hours since last analyze on the table.	
Last AutoAnalyze	Hours since last autoanalyze on the table.	
Row Count	Estimated number of rows in a table.	
Index size as a percentage of table size	Size of the indexes on table, as a percentage of table's size.	

Template Name	Description	Details
Table Frozen XID	The age (in transactions before the current transaction) of the table's frozen transaction ID.	

Global Templates

Template Name	Description	Details
Agents Down	Number of agents that haven't reported in recently.	
Servers Down	Number of servers that are currently inaccessible.	
Alert Errors	Number of alerts in an error state.	

7.4.9.4 Custom Alert Templates

An alert template is a prototype that defines the properties of an [alert](#). An alert instructs the server to compare the current state of the monitored object to a threshold (specified in the alert template) to determine if a situation exists that requires administrative attention.

You can use the [Alert Templates](#) tab to define a custom alert template or view the definitions of existing alert templates. To open the [Alert Template](#) tab, select the [Manage Alerts...](#) menu option from the [Management](#) menu; when the [Manage Alerts](#) tab opens, select [Alert Templates](#) from the [Quick Links](#) menu.

Template name	Description	Target type	Applies to server	Check frequency (minutes)
<small>No data is currently available for custom alert template.</small>				

Use the [Show System Template](#) drop-down listbox to filter the alert templates; select a type from the listbox to view all of the templates for that level of the PEM hierarchy.

Reviewing an Existing Alert Template

To view the definition of an existing template (including PEM pre-defined alert templates), use the [Show System Template](#) drop-down listbox to select the type of object monitored. When you select the object type, the [Alert Templates](#) table will display the currently defined alert templates that correspond with that object type. Highlight a template name and click the edit icon (at the left end of the row) to review the template definition.

Alert Templates						Show System Template: Global
	Template name	Description	Target type	Applies to server	Check frequency (minutes)	
<input checked="" type="checkbox"/>	Agents Down	Number of agents that haven't reported in re...	Global	ALL	1	
<input checked="" type="checkbox"/>	Alert Errors	Number of alerts in an error state.	Global	ALL	1	
<input checked="" type="checkbox"/>	Servers Down	Number of servers that are currently inacces...	Global	ALL	1	

Use the edit button to the left of a template name to view detailed information about the template:

- General information is displayed on the **General** tab.
- The names of probes that provide data for the template are listed on the **Probe Dependency** tab.
- The names of any parameters referred to in the SQL code are listed on the **Parameters** tab.
- The SQL code that defines the behavior of the alert is displayed on the **SQL** tab.

Defining a New Alert Template

To define a new alert template, use the **Show System Template** drop-down listbox to select **None**, and click the **Add** icon (+) located in the upper-right corner of the alert template table.

The screenshot shows the 'General' tab of the alert template configuration interface. It includes the following fields:

- Template name:** A field containing a red warning icon and a placeholder text area.
- Description:** A field containing a red warning icon and a placeholder text area.
- Target type:** A dropdown menu set to 'Server' with a help text: 'Use the Target type field to select the type of object that will be the focus of the alert.'
- Applies to server:** A dropdown menu set to 'ALL' with a help text: 'Use the Applies to server field to specify the server to which the alert will be applied; specify a single server type or ALL.'
- History retention:** A dropdown menu set to '30' with a help text: 'Use the History retention field to specify the number of days that the result of the alert execution will be stored on the PEM server.'
- Threshold unit:** An empty input field with a help text: 'Use the Threshold unit field to specify the unit type of the threshold value.'
- Auto create:** A dropdown menu set to 'No' with a help text: 'Use the Auto create field to automatically create alerts using this template. Please provide default threshold values. If the value is "Yes" then all the added parameters in Parameters tab will be removed as we do not support parametrised auto alerts.'
- Operator:** A dropdown menu set to '>' with three adjacent input fields labeled 'Low', 'Med', and 'High'.
- Check frequency (minutes):** A dropdown menu set to '1' with a help text: 'Use the Check frequency field to specify the number of minutes between alert executions. This value specifies how often the server will invoke the SQL code specified on the SQL tab and compare the result to the threshold value specified in the alert definition.'

Use fields on the **General** tab to specify general information about the template:

- Use the **Template name** field to specify a name for the new alert template; this field is required.
- Use the **Description** field to provide a description of the alert template; this field is required.
- Use the **Target type** drop-down listbox to select the type of object that will be the focus of the alert.
- Use the **Applies to server** drop-down listbox to specify the server type (EDB Postgres Advanced Server or PostgreSQL) to which the alert will be applied; you can specify a single server type, or **ALL**.
- Use the **History retention** field to specify the number of days that the result of the alert execution will be stored on the PEM server.
- Use the **Threshold unit** field to specify the unit type of the threshold value.

- Use fields in the **Auto create** box to indicate if PEM should use the template to generate an automatic alert. If enabled, PEM will automatically create an alert when a new server or agent (as specified by the **Target type** drop-down listbox) is added, and delete that alert when the target object is dropped.
 - Move the **Auto create?** slider to **Yes** to indicate that PEM should automatically create alerts based on the template. If you modify an existing alert template, changing the **Auto create?** slider from **No** to **Yes**, PEM will create alerts on the existing agents and servers. Please note that if you change the slider from **Yes** to **No**, the default threshold values in existing alerts will be erased, and cannot be recovered.
 - Use the **Operator** drop-down listbox to select the operator that PEM will use when evaluating the current system values.
 - Select a greater-than sign (>) to indicate that the alert should be triggered when the system values are greater than the values entered in the **Threshold values** fields.
 - Select a less-than sign (<) to indicate that the alert should be triggered when the system values are less than the values entered in the **Threshold values** fields.

- Use the threshold fields to specify the values that PEM will compare to the system values to determine if an alert should be raised. Please note that you must specify values for all three thresholds (**Low**, **Medium**, and **High**):
 - Enter a value that will trigger a low-severity alert in the **Low** field.
 - Enter a value that will trigger a medium-severity alert in the **Medium** field.
 - Enter a value that will trigger a high-severity alert in the **High** field.

- Use the **Check frequency** field to specify the default number of minutes between alert executions. This value specifies how often the server will invoke the SQL code specified in the definition and compare the result to the threshold value specified in the template.

The screenshot shows the 'Probe Dependency' tab of a configuration interface. At the top, there are tabs for General, Probe Dependency, Parameters, and SQL. The Probe Dependency tab is active. Below the tabs, there is a section titled 'Probes' with a dropdown menu labeled 'Select probe from list'. There are also two input fields: 'Display name' and 'Internal name'.

Use the fields on the **Probe Dependency** tab to specify the names of probes referred to in the SQL query specified on the **SQL** tab:

- Use the **Probes** drop-down listbox to select from a list of the available probes; highlight a probe name, and click the **Add** button to add the probe to the list of probes used by the alert template. To remove a probe from the selected probes list, highlight the probe name, and click the **Delete** icon.

The screenshot shows the 'Parameters' tab of a configuration interface. At the top, there are tabs for General, Probe Dependency, Parameters, and SQL. The Parameters tab is active. Below the tabs, there is a table with columns 'Name', 'Data type', and 'Unit'. A note at the bottom states: 'Add (*) button is disabled in case of system template or value of "Auto create" is Yes in General tab.'

Use fields on the **Parameters** tab to define the parameters that will be used in the SQL code specified on the **SQL** tab. Click the **Add** icon, and:

- Use the **Name** field to specify the parameter name.
- Use the **Data type** drop-down listbox to select the type of parameter.
- Use the **Unit** field to specify the type of unit specified by the parameter.

When you've defined a new parameter, click the **Add/Change** button to save the definition and add the parameter to the parameter list.

To modify an existing parameter definition, highlight a parameter name in the list, modify the parameter values in the fields at the bottom of the tab, and click **Add/Change** to preserve the changes. To remove one or more parameter definitions, highlight the parameter name(s) and click the **Remove** button.



Use the **Code** field on the **SQL** tab to provide the text of the SQL query that the server will invoke when executing the alert. The SQL query will provide the result against which the threshold value is compared; if the alert result deviates from the specified threshold value, an alert will be raised.

Within the query, parameters defined on the **Parameters** tab should be referenced (sequentially) by the variable **param_x**, where **x** indicates the position of the parameter definition within the parameter list. For example, **param_1** refers to the first parameter in the parameter list, **param_2** refers to the second parameter in the parameter list, and so on.

The query can also include the following pre-defined variables:

Variable Description	Variable Name
agent identifier	'\${agent_id}'
server identifier	'\${server_id}'
database name	'\${database_name}'
schema name	'\${schema_name}'
table, index, sequence or function name	'\${object_name}'

Please Note: If the specified query is dependent on one or more probes from different levels within the PEM hierarchy (server, database, schema, etc.), and a probe becomes disabled, any resulting alerts will be displayed as follows:

- If the alert definition and the probe referenced by the query are from the same level within the PEM hierarchy, the server will display any alerts that reference the alert template on the Alert Error table of the Global Alert Dashboard.
- If the alert definition and the probe referenced by the query are from different levels of the PEM hierarchy, the server will display any triggered alerts that reference the alert template on the Alert Details table of the hierarchy on which the Alert was defined.

Use the **Detailed Information SQL** field to provide a SQL query that will be invoked if the alert is triggered. The result set of the query may be displayed as part of the detailed alert information on the **Alerts** dashboard or **Global Overview** dashboard.

After defining a new alert template, click the **Add/Change** button to save the definition and add the template to the **Alert Templates list**. Click **Cancel** to exit the **Alert Templates** dialog without saving changes.

After defining a template, you can use the **Manage Alerts** tab to create and enable an alert based on the template.

Deleting an Alert Template

To delete an alert template, highlight the template name in the alert templates table, and click the **Delete** icon (located to the left of the alert template name). The alert history will persist for the length of time specified on the **History Retention** field in the template definition.

7.4.9.5 Email Groups

Postgres Enterprise Manager monitors your system for conditions that require user attention. You can use an email group to specify the email addresses of users that the server will notify if current values deviate from threshold values specified in an alert definition. An email group has the flexibility to notify multiple users, or target specific users during user-defined time periods.

Please note that you must configure the PEM Server to use an SMTP server to deliver email before PEM can send email notifications.

Use the **Email Groups** tab to configure groups of SMTP email recipients. To access the **Email Groups** tab, select **Manage Alerts...** from the PEM client's **Management** menu; when the **Manage Alerts** tab opens, select **Email Groups** from the **Quick Links** toolbar.

Email Groups	
	Group name
<input checked="" type="checkbox"/>	acctg_admin
<input checked="" type="checkbox"/>	hr_resources
<input checked="" type="checkbox"/>	sales
<input checked="" type="checkbox"/>	<Default>

The **Email Groups** tab displays a list of the currently defined email groups. Highlight a group name and click the edit icon (at the far left end of the row) to modify an existing group.

To define a new email group, click the Add icon (+) in the upper-right corner of the table.

The screenshot shows the 'Email Groups' configuration screen. At the top, there's a table with two rows: '<Default>' and 'sales'. A modal dialog is open for the 'sales' row. Inside the modal, under the 'Email Group' tab, the 'Group Name' is set to 'sales'. Below this is a detailed description of email group options. The 'Options' tab is also visible, showing fields for 'Reply to addresses', 'CC addresses', 'BCC addresses', 'Subject prefix', 'From time', and 'To time'.

Use the **Email Group** tab to define an email group and its members:

- Provide a name for the email group in the **Group name** field.

Each row within the email group definition will associate a unique set of email addresses with a specific time period. When an alert is triggered, the server will evaluate the times specified in each row and send the message to those group members whose definitions are associated with the time that the alert triggered.

Click the Add icon (+) in the group members table to open the **Options** tab, and add the member addresses that will receive notifications for the time period specified:

- Enter a comma-delimited list of recipient addresses in the **Reply to addresses** field.
- Enter a comma-delimited list of return addresses in the **Reply to addresses** field.
- Enter a comma-delimited list of addresses that will receive a copy of the email in the **Cc addresses** field.
- Enter a comma-delimited list of addresses that will receive a copy of the email (without the knowledge of other recipients) in the **Bcc addresses** field.
- Enter the email address that messages to this group should be sent from in the **From address** field.
- Provide a comment that will be used as a subject line prefix for any emails sent as part of a notification in the **Subject prefix** field.
- Use the **From time** and **To time** time selectors to specify the time range for notifications to the group member(s) that are identified on this row of the email group dialog. When an alert is triggered, the server will evaluate the times specified in each row and send the message to those group members whose definitions include the current time. Provide the **From time** and **To time** values in the locale of the PEM client host, and the PEM server will translate the time into other time zones as required.

When you've identified the member or members that will receive an email during a specific time period, click the add icon to specify another time period and the email addresses that will be notified during those hours. When you've

finished defining the email group, click the save icon.

Deleting an Email Group

To mark an email group for deletion, highlight the group name in the **Email Groups** table and click the delete icon to the left of the name; the alert will remain in the list, but in red strike-through font.

Email Groups	
	Group Name
<input checked="" type="checkbox"/>	sales
<input checked="" type="checkbox"/>	<Default>

The delete icon acts as a toggle; you can undo the deletion by clicking the delete icon a second time; when you click the save icon, the email group definition will be permanently deleted.

7.4.9.6 Webhooks

Postgres Enterprise Manager monitors your system for conditions that require user attention. You can use a webhook to create the endpoints that will receive a notification if current values deviate from threshold values specified in an alert definition. PEM sends a notification to multiple webhook endpoints, or to specific target webhook endpoints based on the events triggered.

Please note that you must configure the PEM Server to use webhooks to receive notification of alert events on threshold value violations in your configured applications.

Use the **Webhooks** tab to configure endpoint recipients. To access the **Webhooks** tab, select **Manage Alerts...** from the PEM client's **Management** menu; when the **Manage Alerts** tab opens, select **Webhooks** from the **Quick Links** toolbar.

Dashboard	Properties	SQL	Statistics	Dependencies	Dependents	Monitoring	Manage Alerts	Webhooks
Description								
Webhooks: Webhooks are user defined HTTP callbacks which are triggered by specific events. You can use webhooks to receive notifications of alert events on threshold values violations, in your configured applications.								
Webhooks								
Name	URL	Enable?	Alert notifications	High alerts?	Medium alerts?	Low alerts?	Cleared alerts?	
			No data					

The **Webhooks** tab displays a list of the currently defined recipient applications as endpoints. Highlight an endpoint and click the edit icon (at the far left end of the row) to modify an existing endpoint.

Creating a Webhook

To define a new webhook, click the **Add** icon (+) in the upper-right corner of the table.

Use the **General** tab to define the basic details of the webhook:

- Provide a name for the webhook in the **Name** field.
- Specify a webhook URL where all the notifications will be delivered in the **URL** field.
- Set the request method type used to make the call in the **Request Method** field i.e. **POST** or **PUT**.
- By default **webhooks** will be enabled; to disable a webhook set **Enable?** to **No**.

Note

The above **Enable?** setting will work only if **enable_webhook** parameter is set to true in **agent.cfg** file. By default, **enable_webhook** parameter is set to true only for the Agent running on the PEM Server Host. For all other Agents running on other hosts, it needs to be set to true manually.

Defining a Webhook SSL configurations

You can define the Webhook SSL parameters in the respective agent configuration file or registry in windows. You can find the list of Webhook SSL parameters in PEM Agent Configuration Parameters <pem_agent_config_params> section. If you add or remove any of the agent configuration parameters, you must restart the agent to apply them.

- On 32 bit Windows systems, PEM registry entries for Webhooks are located in **HKEY_LOCAL_MACHINE\Software\EnterpriseDB\PEM\agent\WEBHOOK**
- On 64 bit Windows systems, PEM registry entries for Webhooks are located in **HKEY_LOCAL_MACHINE\Software\Wow6432Node\EnterpriseDB\PEM\agent\WEBHOOK**
- On Linux systems, PEM configuration options for Webhooks are stored in the **agent.cfg** file, located (by default) in **/usr/edb/pem/agent/etc**

```
[WEBHOOK/Django]
webhook_ssl_key=<webhook_client_ssl_key_path>
webhook_ssl_crt=<webhook_client_ssl_certificate_path>
webhook_ssl_ca_crt=<webhook_server_ca_certificate_path>
webhook_ssl_crl=<crl_file_path_to_validate_webhook_server>
allow_insecure_webhooks=<true|false>
```

Name	Type	Data
(Default)	REG_SZ	(value not set)
AllowInsecureWebhooks	REG_SZ	true
WebhookSSLCaCrt	REG_SZ	server.ca.pem
WebhookSSLCrt	REG_SZ	client.crt
WebhookSSLKey	REG_SZ	client.key

Use the **HTTP Headers** tab to define the header parameters to pass while calling the webhook endpoints:

- All the values will be specified as a key and value pair.
- Specify a key parameter in the **Key** field and a value in the **Value** field.
- To add multiple **HTTP Headers**, click the **Add** icon (+) in the upper-right corner of the **HTTP Headers** table.
- To delete the **HTTP Headers**, click on **Delete** icon to the left of the **Key**; the alert will remain in the list, but in strike-through font. Click the **Save** button to reflect the changes.
- To edit the **HTTP Headers**, click on the **Edit** icon to the left of **Key**.

```

1   "payload": {
2     "version": "1.0",
3     "type": "alert",
4     "name": "%AlertName%",
5     "alert": {
6       "id": "%AlertID%",
7       "AlertName": "%AlertName%",
8       "Server/Agent": "%ObjectName%",
9       "Server": "%ServerName%",
10      "ServerID": "%ServerID%",
11      "ServerPort": "%ServerPort%",
12      "Database": "%DatabaseName%",
13      "DatabaseID": "%DatabaseID%",
14      "ObjectType": "%ObjectType%",
15      "Object": "%ObjectName%",
16      "PackageName": "%PackageName%",
17      "DatabaseObjectName": "%DatabaseObjectName%",
18      "Metric": "%Metric%",
19      "AlertID": "%AlertID%",
20      "ObjectType": "%ObjectType%",
21      "ThresholdValue": "%ThresholdValue%",
22      "CurrentValue": "%CurrentValue%",
23      "OldState": "%OldState%",
24      "AlertRaisedTime": "%AlertRaisedTime%",
25      "State": "%currentState%", "%AlertDetectedAt"
26    }
27  }
28 }
29

```

Placeholders:

- %AlertID% - the id of the triggered alert.
- %AlertName% - the name of the triggered alert.
- %ObjectName% - the name of the server or agent on which the alert was triggered.
- %ObjectType% - the type on which alert was generated.
- %ThresholdValue% - the threshold value reached by the metric when the alert triggered.
- %CurrentValue% - the current value of the metric that triggered the alert.
- %.currentState% - the current state of the alert.
- %OldState% - the previous state of the alert.
- %AlertRaisedTime% - the time that the alert was raised, or the most recent time that the alert state was changed.
- %AgentID% - the id of the agent by which alert was generated.
- %AgentName% - the name of the agent by which alert was generated.
- %ServerID% - the id of the server on which alert was generated.
- %ServerName% - the name of the server on which alert was generated.
- %ServerPort% - the port of the server on which alert was generated.
- %DatabaseID% - the id of the database on which alert was generated.
- %DatabaseName% - the name of the schema on which alert was generated.
- %PackageName% - the name of the package on which alert was generated.
- %Metric% - the metric on which alert was generated.
- %Placeholder% - the list of custom parameters used to generate the alert.
- %AlertInfo% - the detailed database object level information of the alert.

Use the **Payload** tab to define the JSON data to be sent to the endpoint when an alert is triggered:

- **Type** specifies data to be sent in format type (i.e. JSON).
- Use **Template** to configure JSON data sent to endpoints. Within the **Template**, you can use placeholders for the following:
 - %AlertID% - the id of the triggered alert.
 - %AlertName% - the name of the triggered alert.
 - %ObjectName% - the name of the server or agent on which the alert was triggered.
 - %ObjectType% - the type on which alert was generated.
 - %ThresholdValue% - the threshold value reached by the metric when the alert triggered.
 - %CurrentValue% - the current value of the metric that triggered the alert.
 - %.currentState% - the current state of the alert.
 - %OldState% - the previous state of the alert.
 - %AlertRaisedTime% - the time that the alert was raised, or the most recent time that the alert state was changed.
 - %AgentID% - the id of the agent by which alert was generated.
 - %AgentName% - the name of the agent by which alert was generated.

- `%ServerID%` - the id of the server on which alert was generated.
- `%ServerName%` - the name of the server on which alert was generated.
- `%ServerIP%` - the ip or address of the server on which alert was generated.
- `%ServerPort%` - the port of the server on which alert was generated.
- `%DatabaseName%` - the name of the database on which alert was generated.
- `%SchemaName%` - the name of the schema on which alert was generated.
- `%PackageName%` - the name of the package on which alert was generated.
- `%DatabaseObjectName%` - the name of the database object name like table name, function name etc on which alert was generated.
- `%Parameters%` - the list of custom parameters used to generate the alert.
- `%AlertInfo%` - the detailed database object level information of the alert.

- Click on the **Test Connection** button, to test notification delivery to the mentioned endpoint.

The screenshot shows the 'Webhooks' configuration page. It lists a single webhook entry for 'Django' with the URL `http://192.168.1.2:8000/public-w...`. The 'Enable?' field is set to 'Yes'. Under the 'Alert notifications' section, all alert levels (High, Medium, Low, Cleared) are enabled ('Yes'). The 'Notifications' tab is selected, showing that 'All alerts?' is also enabled ('Yes').

Name	URL	Enable?	High alerts?	Medium alerts?	Low alerts?	Cleared alerts?
Django	<code>http://192.168.1.2:8000/public-w...</code>	Yes	Yes	Yes	Yes	Yes

General **HTTP Headers** **Payload** **Notifications**

Select Yes to enable the alert notifications based on threshold value violations. You can override these notification setting from manage alert tab specific to alert.

All alerts? **Yes** Select Yes to enable all the alert notifications.

Alert notifications

High alerts?	Yes
Medium alerts?	Yes
Low alerts?	Yes
Cleared alerts?	Yes

Use the **Notifications** tab to specify an alert level for webhook endpoints:

- Set **All alerts** to **Yes** to enable all alert levels to send notifications.
- To instruct PEM to send a notification when a specific alert level is reached, set the slider next to an alert level to **Yes**. Please note that you must set **All alerts** to **No** to configure an individual alert level.

Deleting a Webhook

To mark a webhook for deletion, highlight the webhook name in the **Webhooks** table and click the delete icon to the left of the name; the alert will remain in the list, but in strike-through font.

The screenshot shows the 'Webhooks' configuration page with two entries: 'Django' and 'Slack'. Both entries have their 'Enable?' field set to 'Yes'. The 'Django' entry has a delete icon to its left, indicating it is marked for deletion. The 'Notifications' tab is selected, showing that 'All alerts?' is enabled ('Yes').

Name	URL	Enable?	High alerts?	Medium alerts?	Low alerts?	Cleared alerts?
Django	<code>http://192.168.1.2:8000/public-w...</code>	Yes	Yes	Yes	Yes	Yes
Slack	<code>https://hooks.slack.com/services/TQ...</code>	Yes	Yes	Yes	Yes	Yes

The delete icon acts as a toggle; you can undo the deletion by clicking the delete icon a second time; when you save your work (by clicking the save icon), the webhook definition will be permanently deleted.

7.4.9.7 SNMP MIB Generation

PEM allows alerts to be sent as SNMP traps or notifications to receivers such as network monitoring tools. To enable such tools to understand these notifications, a MIB file may be generated that describes the different alerts and accompanying information that PEM may send. The `pem.generate_alert_mib()` SQL function in the PEM database may be used to generate the MIB file from the alert templates defined in the database. For example:

```
psql.exe -U postgres -d pem -A -t -c "SELECT pem.generate_alert_mib();" > PEM-ALERTING-MIB
```

7.4.9.8 SNMP Trap Details

Every SNMP trap send by PEM starts with oid `.1.3.6.1.4.1.27645.5444`, Significance of each identifier in oid is as follow's.

Identifier	Meaning
1	ISO, ISO is the group that established the OID standard
3	org, Organization identification schemes registered according to ISO/IEC 6523-2
6	dod, United States Department of Defense (DoD)
1	internet, Communication will be via Internet/network
4	private, This is a device manufactured by a private entity (not government)
1	enterprise, The device manufacturer is classified as an enterprise
27645	PostgreSQL global development group
5444	pem

How OID's are formed?

PEM's SNMP trap has following oid format `1.3.6.1.4.1.27645.5444.<alert_target_level_identifier>.<alert_identifier>`

Following table lists down possible values for `<alert_target_level_identifier>`.

Identifier	Alert Target Level
1	Agent
2	Server
3	Database
4	Schema
5	Table
6	Index
7	Sequence
8	Function
9	Global

`<alert_identifier>` is unique identifier for each alert, which you can find `insnmp_oid` column of `pem.alert_template` table.

For example, `snmp_oid` for Agent Down alert template is 34, hence trapOID for agent down alert will be

1.3.6.1.4.1.27645.5444.1.34

How OID's for binding variables are formed?

Every binding variable oid has following format 1.3.6.1.4.1.27645.5444.10.<binding_variable_identifier>, where 10 is identifier for binding variable Following table lists down possible values for <binding_variable_identifier>

Identifier	Variable Name
1	alertName
2	agentID
3	serverID
4	agentName
5	serverName
6	databaseName
7	schemaName
8	objectName
9	thresholdvalue
10	previousValue
11	value
12	previousStatus
13	status
14	recordedTime
15	downObjects
16	detailedInformation

For example, 1.3.6.1.4.1.27645.5444.10.1 is oid for binding variable alertName.

Details of each snmp traps in pem.snmp_spool table. For example,

```
pem=# select * from pem.snmp_spool;
-[ RECORD 1 ]-----+
-----+
-----+
id          | 1
trap_oid    | .1.3.6.1.4.1.27645.5444.1.34
enterprise_oid | .1.3.6.1.4.1.27645.5444
trap_version | 2
varbinding_oid |
.1.3.6.1.4.1.27645.5444.10.1|.1.3.6.1.4.1.27645.5444.10.2|.1.3.6.1.4.1.27645.5444.10.
4
varbinding_value | Agent Down|Postgres Enterprise Manager Host|
{0.1,0.2,0.3}|0|1|CLEAR|HIGH|2020-06-22 15:51:03.266437+10
sent_status   | s
recorded_time | 22-JUN-20 15:51:03.266437 +10:00
```

7.4.9.9 Using PEM with Nagios

The PEM server can send a passive alert result to Nagios network-alerting software when an alert is triggered. To instruct the PEM server to notify Nagios of a triggered alert, you must:

- Enable Nagios notification for each alert that will trigger a notification from the PEM server to Nagios. Please note that PEM alerting must be configured before you create the host.cfg file and services.cfg file.
- Configure Nagios-related behaviors of the PEM server.
- Create the host.cfg and services.cfg configuration files.
- If necessary, modify the Nagios configuration file and restart the Nagios server.
- Install the PEM Agent on the system where Nagios server is installed and register it with the PEM Server. Set `enable_nagios` configuration to `true` in the agent.cfg for that agent, and restart the agent service.

Detailed information about each configuration step is listed below.

After configuring the server to enable Nagios alerting, any triggered alerts will send a passive check result to the Nagios service. The syntax of a passive alert is:

```
[timestamp] PROCESS_SERVICE_CHECK_RESULT; host_name ; service_name ; service_status ;
```

Where:

- `timestamp` is the date and time that the alert was triggered.
- `host_name` is the name of the server or agent.
- `service_name` is the name of the alert.
- `service_status` is the numeric service status value:
 - 0 if the service status is *OK*
 - 1 if the service status is *WARNING*
 - 2 if the service status is *CRITICAL*
 - 3 if the service status is *UNKNOWN*

The PEM server uses the following rules to evaluate the service status:

- If the PEM alert level is `CLEARED`, the warning message will read *OK*
- If the PEM alert level is `LOW`, the warning message will read *WARNING*
- If the `is_nagios_medium_alert_as_critical` flag (specified in the PEM server configuration dialog) is set to `FALSE` and the alert level is `MEDIUM`, the warning message will read *WARNING*
- If the `is_nagios_medium_alert_as_critical` flag (specified in the PEM server configuration dialog) is set to `TRUE` and the alert level is `MEDIUM`, the warning message will read *CRITICAL*
- If the PEM alert level is `HIGH`, the warning message will read *CRITICAL*

Enabling Nagios Notification for an Alert

The PEM server maintains a unique set of notification properties for each enabled alert. Use the `Notification` tab of the `Manage Alerts` <pem_alerting_dialog> tab to specify that (when triggered), a given alert will send an alert notice to Nagios. To modify the notification properties of an alert, right-click on the name of the object monitored by the alert, and select `Manage Alerts...` from the Management menu. When the `Manage Alerts` tab opens, locate the alert, and then click the edit button to the left of the alert name in the `Alerts` list. When the `Manage Alerts` tab opens, select the `Notification`` tab.

General **Notification**

Email notification

All alerts?	<input type="checkbox"/> No	<Default>
Low alerts?	<input type="checkbox"/> No	<Default>
Medium alerts?	<input type="checkbox"/> No	<Default>
High alerts?	<input type="checkbox"/> No	<Default>

To configure notifications for an alert, use the fields in the Email notification box to specify the user or user group that will receive an email notification if the alert is triggered at the specified level. Use the drop-down listbox to select a pre-defined group that will be sent a notification if an alert of the selected level is triggered. Please note that you must configure the PEM Server to use an SMTP server to deliver email before PEM can send email notifications.

Trap notification

Send trap?	<input type="checkbox"/> No	SNMP version	v3	Low alert?	<input type="checkbox"/> No
Medium alert?	<input type="checkbox"/> No			High alert?	<input type="checkbox"/> No

Use the Trap notification options to configure trap notifications for this alert. Note that you must configure the PEM Server to send notifications to an SNMP trap/notification receiver before notifications can be sent.

Nagios notification

Submit passive service check result to Nagios?	<input checked="" type="checkbox"/> Yes
--	---

Set "Submit passive service check result to Nagios" to "Yes" to instruct the PEM server to notify Nagios when the alert is triggered or cleared.

Script execution

Execute script?	<input type="checkbox"/> No	Execute on alert cleared?	<input type="checkbox"/> No
Execute script on	<input type="radio"/> PEM Server <input checked="" type="radio"/> Monitored Server		
Code	<pre> </pre>		

Use the fields in the Script execution box to (optionally) define a script that will be executed if an alert is triggered, and to specify details about the script execution.

- Set the Execute script slider to Yes to instruct PEM to execute the provided script if an alert is triggered.
- Set the Execute on alert cleared slider to Yes to instruct PEM to execute the provided script when the situation that triggered the the alert has been resolved.
- Use the selector to indicate if the script should execute on the PEM Server or the Monitored Server.
- Provide the script that PEM should execute in the Code field. You can provide a batch/shell script, or SQL code. Within the script you can use the placeholders to replace the following:
 - %AlertName% - the name of the triggered alert.
 - %ObjectName% - the name of the server or agent on which the alert was triggered.
 - %ThresholdValue% - the threshold value reached by the metric when the alert triggered.
 - %CurrentValue% - the current value of the metric that triggered the alert.
 - %CurrentState% - the current state of the alert.
 - %OldState% - the previous state of the alert.
 - %AlertRaisedTime% - the time that the alert was raised, or the most recent time that the alert state was changed.

To enable Nagios notification, move the slider next to **Submit passive service check result to Nagios** to **Yes**; before exiting the **Manage Alerts** tab, click the save icon to preserve your changes.

Configuring Nagios-related behavior of the PEM Server

You can use the **Server Configuration** dialog to provide information about your Nagios configuration to the PEM server. To open the **Server Configuration** dialog, select **Server Configuration...** from the PEM client's **Management** menu.

Server Configuration		
		Search by parameter name
flapping_detection_state_change	3	
job_failure_notification	<input type="checkbox"/> False	t/f
job_notification_email_group	default	
job_retention_time	30	days
job_status_change_notification	<input type="checkbox"/> False	t/f
long_running_transaction_minutes	5	minutes
max_metrics_per_group_chart	16	
nagios_cmd_file_name	/usr/local/nagios/var/rw/nagios.cmd	
nagios_enabled	<input checked="" type="checkbox"/> True	t/f
nagios_medium_alert_as_critical	<input type="checkbox"/> False	t/f
nagios_spool_retention_time	7	days
probe_log_retention_time	30	days
reminder_notification_interval	24	hours
server_log_retention_time	30	days
show_data_points_on_graph	<input type="checkbox"/> False	t/f
show_data_tab_on_graph	<input type="checkbox"/> False	t/f
show_unmanaged_servers	<input checked="" type="checkbox"/> True	t/f

?
Cancel
Reset
Save

Four server configuration parameters specify information about your Nagios installation and PEM server behavior related to Nagios:

- Use the `nagios_cmd_file_name` parameter to specify the location of the Nagios pipeline file that will receive passive check alerts from PEM. The default value of this parameter is `/usr/local/nagios/var/rw/nagios.cmd`. The parameter specifies the default file location; if your `nagios.cmd` file resides in an alternate location, specify the file location in the `Value` field.
- Move the slider in the `nagios_enabled` parameter to `Yes` to instruct the PEM server to send passive check alerts to Nagios.
- Use the `nagios_medium_alert_as_critical` slider to specify the warning severity that the PEM server will pass to Nagios if a medium alert is triggered:
 - If the `is_nagios_medium_alert_as_critical` flag is set to `FALSE` and the alert level `MEDIUM`, the warning message will read `WARNING`
 - If the `is_nagios_medium_alert_as_critical` flag is set to `TRUE` and the alert level `MEDIUM`, the warning message will read `CRITICAL`
- Use the `nagios_spool_retention_time` parameter to specify the number of days of notification history that will be stored on the PEM server. The default value is 7 days.

After modifying parameter values, click the save icon to preserve your changes.

Creating the hosts.cfg and services.cfg File

The templates.cfg file (by default, located in `/usr/local/nagios/etc/objects`) specifies the properties of a generic-host and generic-service. The properties specify the parameters used in the hosts.cfg and services.cfg files.

In most cases (when PEM is installed in a default configuration), you will not be required to modify the templates.cfg file before creating the hosts.cfg and services.cfg files. If necessary, you can modify the templates.cfg file to specify alternate values for parameters or to create new templates.

Before modifying the Nagios configuration file, use the following command to create a hosts.cfg file that contains information about the PEM hosts that reside on the local system:

```
./psql -U postgres -p 5433 -d pem -A -t -c "select pem.create_nagios_host_config('generic-host')">>
/usr/local/nagios/etc/objects/hosts.cfg
```

Then, use the following command to create a `services.cfg` file that contains information about the PEM services that reside on the local system:

```
./psql -U postgres -p 5433 -d pem -A -t -c "select pem.create_nagios_service_config('generic-service')">>
/usr/local/nagios/etc/objects/services.cfg
```

If you wish to use a custom template.cfg file entry, specify the entry name in place of generic-host or generic-service in the above commands.

Modifying the Nagios Configuration File

After creating the host.cfg and services.cfg files, you must specify their location in the Nagios configuration file (by default, /usr/local/nagios/etc/nagios.cfg). Modify the configuration file, adding entries that specify the location of the files:

```
cfg_file=/usr/local/etc/objects/hosts.cfg cfg_file=/usr/local/etc/objects/services.cfg
```

You can use the following command to confirm that Nagios is properly configured:

```
/usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
```

After confirming that Nagios is configured correctly, restart the Nagios service:

```
/usr/local/nagios/bin/nagios -d /usr/local/nagios/etc/nagios.cfg
```

7.4.10 Using the Manage Charts Tab

You can use the `Manage Charts` tab to access dialogs that allow you to create or modify a custom line chart or table, or import a Capacity Manager template for use in a custom chart. After defining a chart, you can display the chart on a custom dashboard. To open the `Manage Charts` tab, select `Manage Charts...` from the PEM client `Management` menu.

	Name	Type	Level	Metrics Category
<input checked="" type="checkbox"/>	test1	Table	Database	Database Object Activity
<input checked="" type="checkbox"/>	Test1	Line Chart	Agent	Alerts

The `Manage Charts` tab provides a `Quick Links` menu that allows you to access dialogs to:

- [Create a New Chart](#) for use on a custom dashboard.

- Import a Capacity Manager template to use as a template for creating a custom chart.

The **Custom Charts** table displays a list of user-defined charts; when a chart is newly added, the font displays in green. When you add an additional chart or refresh the screen, the name of the chart is displayed in black.

Custom Charts				
	Name	Type	Level	Metrics Category
<input checked="" type="checkbox"/>	test1	Table	Database	Database Object Activity
<input checked="" type="checkbox"/>	Test1	Line Chart	Agent	Alerts

Use the search box in the upper-right hand corner of the **Custom Charts** section to search through your custom charts. Specify a:

- Chart name
- Type
- Level
- Metrics Category

Use icons to the left of a charts name in the **Custom Charts** table to manage a chart:

- Click the edit icon to open the **Chart Configuration** wizard and modify aspects of the chart or table.
- Click the delete icon to delete the selected chart.

Contents:

7.4.10.1 Creating a New Chart

Click the **Create New Chart** icon in the **Quick Links** section of the **Manage Charts** tab to open the **Create Chart** wizard. The **Create Chart** wizard will walk you through the steps required to define a new chart.

The screenshot shows the 'Create Chart - Chart Configuration (step 1 of 4)' dialog. On the left, a sidebar lists steps: 1. Configure Chart (selected), 2. Select Metrics, 3. Set Options, 4. Set Permissions. The main area contains fields for Name (Top_Five_Table_Chart), Category (Database Object Activity), Type (Line chart), and Description. At the bottom are buttons for Cancel, Back, Next, and Finish.

Use the fields on the **Configure Chart** dialog to specify general information about the chart:

- Specify the name of the chart in the **Name** field.

- Use the drop-down listbox in the **Category** field to specify the category in which this chart will be displayed; when adding a custom chart to a custom dashboard, the chart will be displayed for selection in the **Category** specified.
- Use the radio buttons in the **Type** field to specify if the chart will be a **Line chart** or a **Table**.
- Provide a description of the chart in the **Description** field. The description will be displayed to the user viewing the chart (on a custom dashboard) when they click the information icon.

When you've completed the fields on the **Configure Chart** dialog, click **Next** to continue.

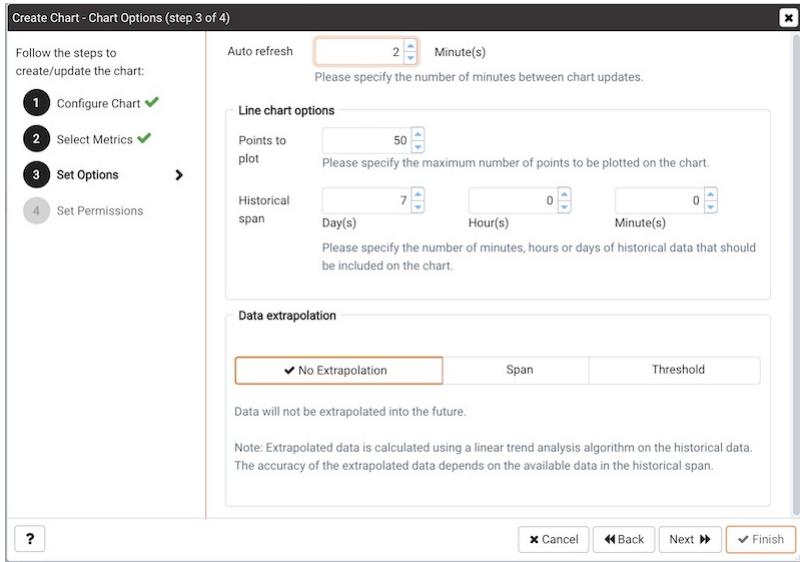


Use the fields on the **Select Metrics** dialog to select the metrics that will be displayed on the chart:

- Use the **Metric level** drop-down listbox to specify the level of the PEM hierarchy from which you wish to select metrics. You can specify **Agent**, **Database**, or **Server**. Each level offers access to a unique set of probes and metrics.
- Use the tree control in the **Available metrics** box to select the metrics that will be displayed on the chart.
 - If you are creating a table, you may only select metrics from one probe; each node of the tree control lists the metrics returned by a single probe. Expand a node of the tree control, and check the boxes to the left of a metric name to include that metric data in the table.
 - If you are creating a line chart, expand the nodes of the tree control and double-click each metric that you would like to include in the chart.
- Use the fields in the **Selected metrics** panel to specify how the metric data will be displayed in your chart. The selection panel displays the name of the metric in the (non-modifiable) **Metric [Probe]** column. You can:
 - Click the garbage can icon to delete a metric from the list of selected metrics.
 - Use the drop-down listboxes in the **Selection Criteria** column to specify the order of the data displayed.
 - Use the **Limit** field to specify the number of rows in a table or lines in a chart:
 - The maximum number of lines allowed in a chart is 32.
 - The maximum number of rows allowed in a table is 100.
- If you are creating a line chart, PEM supports comparisons of cross-hierarchy metrics.
 - Click the compare icon to open a selection box that allows you to select one or more probe-specific attributes (i.e. CPUs, interfaces, databases, etc.) to compare in the chart.
 - Click the copy icon to apply your selections to all of the metrics for the same probe. When the popup opens,

click **Yes** to confirm that other selections for the same probe will be overwritten, or **No** to exit the popup without copying the attributes.

When you've completed the fields on the **Select Metrics** dialog, click **Next** to continue.



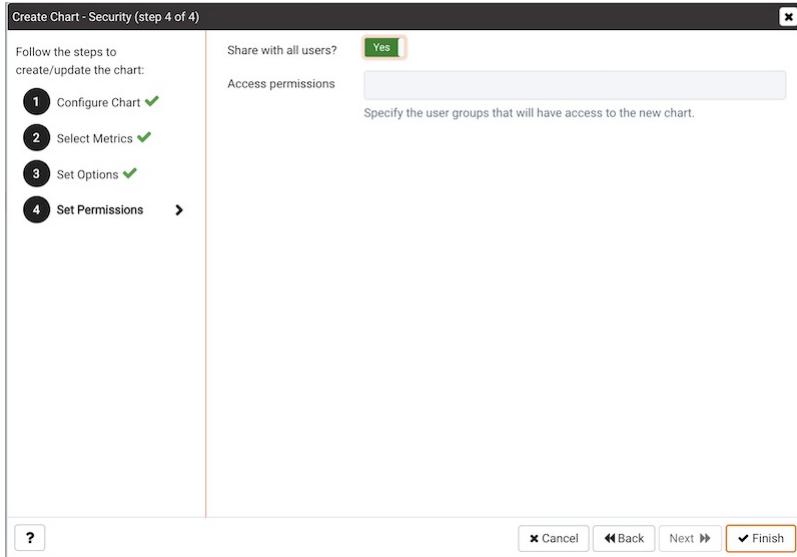
Use the fields on the **Set Options** dialog to specify display options for your chart:

- Use the **Auto Refresh** field to specify the number of minutes between chart updates - choose a value from 1 to 120. The default auto refresh rate is 2 minutes.

Use fields under the **Line chart options** heading to specify display preferences for a line chart:

- Use the **Points to plot** field to specify the maximum number of points that will be plotted on the chart.
- Use the fields to the right of the **Historical span** label to specify how much historical data should be displayed on the chart:
 - Use the **Day(s)** field to specify the number of days of historical data that should be included on the chart.
 - Use the **Hour(s)** field to specify the number of hours of historical data that should be included on the chart.
 - Use the **Minute(s)** field to specify the number of minutes of historical data that should be included on the chart.
- Use the fields in the **Data extrapolation** box to specify if PEM should generate extrapolated data based on historical data.
 - Click the **No Extrapolation** label to omit extrapolated data from the chart.
 - Click the **Span** label to use the **Days** and **Hours** selectors to specify the period of time spanned by the metrics on the chart.
 - Click the **Threshold** label to use threshold selectors to specify a maximum or minimum value for the chart.

When you've completed the fields on the **Set Options** dialog, click **Next** to continue.



Use the fields on the **Set Permissions** dialog to specify display options for your chart:

- Set the **Share with all** slider to **Yes** to indicate that the chart will be available to all authorized users, or **No** to restrict access to the users or groups specified in the **Access permissions** field.
- Use the **Access permissions** field to select the group or groups that will have access to the chart.

When you've finished defining the chart, click **Finish** to save your edits and add your chart to the list on the **Manage Charts** tab:

Name	Type	Level	Metrics Category
<input checked="" type="checkbox"/> test1	Table	Database	Database Object Activity
<input checked="" type="checkbox"/> Test1	Line Chart	Agent	Alerts
<input checked="" type="checkbox"/> Top_Five_Table_Chart	Line Chart	Database	Database Object Activity

7.4.10.2 Importing a Capacity Manager Template

Selecting the **Import Capacity Manager Template** from the **Manage Charts** tab's **Quick Links** section opens the **Create Chart** dialog, allowing you to select from your saved Capacity Manager templates. When the dialog opens, use the **Import capacity template** drop-down listbox to select the template you would like to use for your chart.



Use the fields on the **Create Chart** dialog to provide information about the chart:

- Specify the name of the chart in the **Name** field.
- Use the drop-down listbox in the **Category** field to specify the category in which this chart will be displayed. When adding a custom chart to a custom dashboard, the chart will be displayed for selection in the **Category** specified.
- Use the radio buttons in the **Type** field to specify if the chart will be a **Line chart** or a **Table**.
- Provide a description of the chart in the **Description** field. The description will be displayed to the user viewing the chart (on a custom dashboard) when they click the information icon.

Click **Next** to continue to the **Select Metrics** window.



The **Select Metrics** window displays details about the metrics that are used by the template. When you've reviewed the metrics, click **Next** to continue to the **Set Options** window.



Use the fields on the **Set Options** window to specify display options for your chart:

- Use the **Auto Refresh** field to specify the number of minutes between chart updates - choose a value from 1 to 999. The default auto refresh rate is 2 minutes.
- Use the fields in the **Data extrapolation** box to specify the time period covered by the chart. You can either:
 - click the **Historical days and extrapolated days** label and:
 - specify the number of days of historical data that should be charted in the **Historical** field.
 - specify the number of projected days that should be charted in the **Extrapolated** field.
 - or, click the **Historical days and threshold** label and:
 - provide the number of days of historical data that should be charted in the **Historical** field.
 - use the threshold selection fields to specify the threshold value at which the chart will end.

When you've completed the **Set Options** window, click **Next** to continue.



After making any required modifications to the chart definition, click **Finish** to save your edits. PEM will open a popup, confirming that the edits have been saved:

Custom Charts				
	Name	Type	Level	Metrics Category
<input checked="" type="checkbox"/>	Cap_man_bkts_read_write	Capacity Chart	Database	Database Object Activity
<input checked="" type="checkbox"/>	test1	Table	Database	Database Object Activity
<input checked="" type="checkbox"/>	Test1	Line Chart	Agent	Alerts
<input checked="" type="checkbox"/>	Top_Five_Table.Chart	Line Chart	Database	Database Object Activity

7.4.11 The PEM Manage Dashboards Tab

PEM displays performance statistics through a number of system-defined dashboards; each dashboard contains a series of summary views that contain charts, graphs and tables that display statistics related to the selected object. You can use the Manage Dashboards tab to create and manage custom dashboards that display the information that is most relevant to your system.

Name	Level	Description
test1	Database	test1
Test1	Agent	

To create a custom dashboard, click the **Create New Dashboard** link (located in the **Quick Links** section of the **Manage Dashboards** tab). To modify an existing dashboard, click the edit icon to the left of a dashboard name. The dashboard editor will open, displaying the definition of the dashboard. When you've finished modifying the dashboard's definition, click the **Save** button to preserve your changes; click **Cancel** to exit without saving your changes. To delete a dashboard, click the delete icon to the left of a dashboard name. A popup will ask you to confirm that you wish to delete the dashboard; click **OK** to delete the selected dashboard.

Contents:

7.4.11.1 Creating a Custom Dashboard

You can use the PEM dashboard editor to create or modify a user-defined dashboard. The custom dashboard may include pre-defined charts, user-defined charts or a mix of pre-defined and user-defined charts. To create a new dashboard, select **Create New Dashboard...** from the **Quick Links** section of the **Manage Dashboards** tab.



Use the fields in the **Configure** section to specify general information about the dashboard:

- Specify a name for the dashboard in the **Name** field. The name specified will also be the title of the dashboard if the title is displayed.
- Use the **Level** drop-down listbox to specify the level of the PEM hierarchy within the PEM client on which the dashboard will be displayed. A dashboard may be accessed via the **Dashboards** menu on a **Global** level, an **Agent** level, the **Server** level or the **Database** level. Each selected level within the list will expose a different set of metrics on which the custom dashboard's charts may be based.
- Provide a description of the dashboard in the **Description** field.

Provide information in the fields in the **Ops dashboard options** box if the dashboard will be used as an Ops dashboard:

- Set the **Ops Dashboard?** field to **Yes** to instruct the server to create a dashboard that is formatted for display on an **Ops monitor**.
- Set the **Show Title?** field to **Yes** to display the dashboard name at the top of the Ops dashboard.
- Use the **Font** drop-down list box to select a custom font style for the title. The selected font style will be displayed in the **Preview** box.
- Use the **Font size** drop-down list box to select a custom font size for the title. The selected font style will be displayed in the **Preview** box.

Use the **Permissions** box to specify the users that will be able to view the new dashboard:

- Set the **Share with all** slider to **Yes** to instruct the server to allow all **Teams** to access the dashboard, or set **Share with all** to **No** to enable the **Access permissions** field.
- Use the **Access permissions** field to specify which roles can view the new dashboard. Click in the field, and select from the list of users to add a role to the list of users with dashboard access.

When you've completed the **Configure Dashboard** section, click the arrow in the upper-right corner to close the section, and access the **Dashboard Layout Design** section.



Click the edit icon in a section header to specify a section name; then, click the add icon (+) to add a chart to the section.



Use the arrows to the right of each chart category to display the charts available and select a chart.



Use the chart detail selectors to specify placement details for the chart:

- * Use the **Chart width** selector to indicate the width of the chart; select 50% to display the chart in half of the dashboard, or 100% to use the whole dashboard width.
- * Use the **Chart alignment** selector to indicate the position of the chart within the section:

- Select **Left** to indicate that the chart should be left-justified.
- Select **Center** to indicate that the chart should be centered.
- Select **Right** to indicate that the chart should be right-justified.

Please note that tables are always displayed centered.

When creating or editing a custom dashboard, you can use drag and drop to re-arrange the charts within a section or to move a chart to a different section.

To add another chart to your dashboard, click the add icon (+) in the section header. When you've finished editing the dashboard, click the **Save** button to save your edits and exit.

To exit without saving your changes, click the **Cancel** button.

7.4.11.2 Creating an Ops Dashboard

You can use the PEM [dashboard editor](#) to create a custom dashboard formatted for display on an Ops monitor. An Ops dashboard displays the specified charts and graphs, while omitting header information and minimizing extra banners, titles, and borders.

To create an Ops dashboard, provide detailed information about the Ops display in the [Ops dashboard options](#) section of the [Create Dashboard](#) dialog:



- Set the **Ops Dashboard?** field to **Yes** to instruct the server to create a dashboard that is formatted for display on an Ops monitor.
- Set the **Show Title?** field to **Yes** to display the dashboard name at the top of the Ops dashboard.
- Use the **Font** drop-down list box to select a custom font style for the title. The selected font style will be displayed in the **Preview** box.
- Use the **Font size** drop-down list box to select a custom font size for the title. The selected font style will be displayed in the **Preview** box.

After adding charts and tables to the Ops dashboard, click the **Save** button to save your work. You can then access the dashboard by navigating through the [Dashboards](#) menu of the hierarchy level specified in the **Level** field on the [New Dashboard](#) dialog.

7.4.12 The Manage Probes Tab

A [probe](#) is a scheduled task that returns a set of performance metrics about a specific monitored server, database, operating system or agent. You can use the Manage Probes tab to override the default configuration and customize the behavior of each probe. To open the Manage Probes tab, select [Manage Probes...](#) from the [Management](#) menu.

Properties SQL Statistics Dependencies Dependents Monitoring **Manage Probes** x

Description

Manage Custom Probes: PEM uses probes to retrieve statistics from a monitored server, database, operating system or agent. You can view, reconfigure, delete, or create your own custom probes.

Copy Probes: PEM allows copying of probes from any chosen object recursively down through the object hierarchy. Click on Copy Probes to quickly copy the displayed probe configuration to a selected target.

Quick Links

 Manage Custom Probes
 Copy Probes
 Help

Probes

Probe name	Execution Frequency		Enabled?	Data Retention			
	Default?	Minutes		Seconds	Default?	Probe Enable?	Default?
Background Writer Statistics	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Blocked Session Information	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Data and Log File Analysis	<input checked="" type="checkbox"/>	0	10	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Database Frozen XID	<input checked="" type="checkbox"/>	720	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Database Size	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Database Statistics	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	90
Failover Manager Cluster Info	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	7
Failover Manager Node Status	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	7
Lock Information	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Number of Prepared Transactions	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Number of WAL Files	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Object Catalog: Database	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Object Catalog: Tablespace	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
PG HBA Conf	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Server Information	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Server log Configuration	<input checked="" type="checkbox"/>	0	10	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Session Information	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Settings	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Streaming Replication	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	180
Streaming Replication Database Conflicts	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	180
Streaming Replication Lag Time	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	180
Tablespace Size	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
User Information	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
WAL Archive Status	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	180

The **Manage Probes** tab provides a set of **Quick Links** that you can use to create and manage probes:

- Click the **Manage Custom Probes** <pem_custom_probes> icon to open the Custom Probes` tab and create or modify a custom probe.
- Click the **Copy Probes** <copy_probe_config> icon to open the Copy Probe` dialog, and copy the probe configurations from the currently selected object to one or more monitored objects.

A probe monitors a unique set of metrics for each specific object type (server, database, database object, or agent); select the name of an object in the tree control to review the probes for that object.

To modify the properties associated with a probe, highlight the name of a probe, and customize the settings that are displayed in the **Probes** table:

- Move the **Default** switch in the **Execution Frequency** columns to **No** to enable the **Minutes** and **Seconds** selectors, and specify a non-default value for the length of time between executions of the probe.
- Move the **Default** switch in the **Enabled?** column to **No** to change the state of the probe, and indicate if the probe is active or not active.
- Move the **Default** switch in the **Data Retention** column to **No** to enable the **Day(s)** field and specify the number of days that information gathered by the probe is stored on the PEM server.

The **Manage Probes** tab may also display information about probes that cannot be modified from the current node; if a probe cannot be modified from the current dialog, the switches for that probe are disabled. Generally, a disabled probe can be modified from a node that is higher in the hierarchy of the PEM client tree control. Select another object in the tree control to change which probes are displayed or enabled on the **Manage Probes** tab.

Contents:

7.4.12.1 Custom Probes

Use the **Custom Probes** tab to create a new probe or modify an existing probe. After creating or modifying a probe, you can incorporate the data gathered by custom probes into existing or new charts or graphs. To open the **Custom Probes** tab, select the **Manage Custom Probes** icon from the **Quick Links** section of the **Manage Probes** tab.

Probe name	Collection method	Target type	Execution frequency	Probe enabled?	Data retention
Top_Five_Large_Tables	SQL	Table	1 0	<input checked="" type="checkbox"/> Yes	1

Use the **Show system probes?** switch to display the system probes on the **Custom Probes** tab.

To modify an existing probe, click the **Edit** icon located to the left of a probe name. To create a new probe, click the **Add** icon in the upper-right corner of the tab.

Probe name	Collection method	Target type	Execution frequency		Probe enabled?	Data retention
	SQL	Server	Minutes 5	Seconds 0	Yes	1

General [Columns](#) [Code](#) [Alternate Code](#)

Probe name ⚠ Please specify Probe name

Collection method SQL

Use the Collection method field to specify the probe type. Use the drop-down to select:

- SQL (the probe will gather information via a SQL statement)
- WMI (the probe will gather information via a Windows Management Instrumentation extension)
- Batch/Shell Script (the probe will use a command script or shell script to gather information). Please note that batch probes are platform specific. If you specify a collection method of Batch, you must specify a platform type in the Platform field.

Target type Server

Use the Target type drop-down to select the object type that the probe will monitor.

Execution frequency

Minutes: 5 Seconds: 0

Probe enabled? Yes

Use the Enabled? switch to specify if the probe is enabled by default. Specify Yes to enable the probe by default, or No to specify that the probe is disabled by default.

Data retention 1

Use the Data retention field to specify the number of days that gathered information will be retained in the probe's history table.

Discard from history? No

Use the Discard from history field to specify if the server should create a history table for the probe. Select Yes to discard probe history, or No to retain the probe history in a table.

Platform *nix

Use the Platform drop-down to specify the type of platform that the probe will monitor. This field is enabled only when the Collection method is Batch/Shell Script.

Use the fields on the **General** tab to modify the definition of an existing probe or to specify the properties of a new probe.

- Use the **Probe name** field to provide a name for a new probe.
- Use the **Collection method** field to specify the probe type. Use the drop-down listbox to select from:
 - SQL** (the probe will gather information via a SQL statement)
 - WMI** (the probe will gather information via a Windows Management Instrumentation extension)
 - Batch/Shell Script** (the probe will use a command-script or shell-script to gather information).

Before creating a batch probe on a Linux system, you must modify the **agent.cfg** file, setting the **allow_batch_probes** parameter equal to **true** and restart the PEM agent. The **agent.cfg** file is located in **/opt/PEM/agent/etc**.

On 64-bit Windows systems, agent settings are stored in the registry. Before creating a batch probe, modify the registry entry for the **AllowBatchProbes** registry entry and restart the PEM agent. PEM registry entries are located in **HKEY_LOCAL_MACHINE\Software\Wow6432Node\EnterpriseDB\PEM\agent**.

Please note that batch probes are platform-specific. If you specify a collection method of **Batch**, you must specify a platform type in the **Platform** field.

To invoke a script on a Linux system, you must modify the entry for **batch_script_user** parameter of **agent.cfg** file and specify the user that should be used to run the script. You can either specify a non-root user or root for this parameter. If you do not specify a user, or the specified user does not exist, then the script will not be executed. Restart the agent after modifying the file. If pemagent is being run by a non-root user then the value of **batch_script_user** will be ignored and the script will be executed by the same non-root user that

is being used for running the pemandent.

- Use the **Target type** drop-down listbox to select the object type that the probe will monitor. **Target type** is disabled if **Collection method** is **WMI**.
- Use the **Minutes** and **Seconds** selectors to specify how often the probe will collect data.
- Use the **Probe enable?** switch to specify if the probe is enabled. Specify **Yes** to enable the probe, or **No** to specify that the probe is disabled.
- Set the **Data retention** switch to **Yes** to specify the number of days that gathered information will be retained in the probe's history table.
- Use the switch next to **Discard from history** to specify if the server should create a history table for the probe. Select **Yes** to discard probe history, or **No** to retain the probe history in a table.
- Use the **Platform** drop-down listbox to specify the type of platform that the probe will monitor. This field is enabled only when the **Collection method** is **Batch**.

Probe name		Collection method	Target type	Execution frequency		Probe enabled?	Data retention
<input checked="" type="checkbox"/>		SQL	Server	5	0	<input checked="" type="checkbox"/> Yes	1

General Columns Code Alternate Code

Name	Internal name	Column type	Data type	Unit	Graphable?	Is PIT?	Calculate PIT?
		Non key	numeric		<input checked="" type="checkbox"/> Yes	<input type="checkbox"/> No	<input type="checkbox"/> No

General

Name ⚠

Internal name

Column type Non key
Use the Column Type drop-down to specify if the column is a Key column (a primary key) or a Non key column. Non-key columns are generally metric items (values that can be graphed).

Data type numeric
Use the Data type drop-down to specify the type of data that will be stored in the column.

Unit
Unit field to specify the unit of measure that applies to the metric stored in the column. This unit is displayed on the Y-Axis of a custom chart or a Capacity Manager chart. This is an optional field.

Graphable? Yes
Set the Graphable switch to Yes to specify that the defined metric may be graphed, and that the probe should be accessible from the Capacity Manager or Manage Charts dialogs.

Is PIT? No
Set the Is PIT switch to Yes to specify that the metric is stored by point-in-time (by default). 'Point-in-time' metrics are those metrics that change (increase or decrease) at any given point of time. For example, database size is a point-in-time metric; at any given point-in-time, the size of the database is fluctuating. Metrics that are not point-in-time (also referred to as cumulative metrics) are metrics whose size always increases over time. For example, Blocks Read and Tuples Read are cumulative metrics; the value stays the same or increases.

Calculate PIT? No
Set the Calculate PIT switch to Yes to specify that the server should calculate a point-in-time value for the metric data. Calculate PIT is disabled if Is PIT is Yes.

⚠ Please specify column name

⚠ Please specify Probe name

Use the **Columns** tab to define the columns in which the probe data will be stored. Navigate to the **Columns** tab, and click the **Add** button (in the upper-right corner) to define a new column.

- Provide a name for the column in the **Name** field.
- The **Internal name** field is not enabled for user-defined probes.
- Use the **Column type** drop-down listbox to specify if the column is a **Key** column (a primary key) or a **Non key** column. Non-key columns are generally metric items (values that can be graphed).

- Use the **Data type** drop-down listbox to specify the type of data that will be stored in the column.
- Use the **Unit** field to specify the unit of measure that applies to the metric stored in the column. This unit is displayed on the Y-Axis of a custom chart or a Capacity Manager chart. This is an optional field.
- Use the **Graphable** switch to specify if the defined metric may be graphed, and that the probe should be accessible from the Capacity Manager or Manage Charts dialogs.
- Use the **Is PIT** switch to specify if the metric is stored by point-in-time (by default).

'Point-in-time' metrics are those metrics that change (increase or decrease) at any given point of time. For example, database size is a point-in-time metric; at any given point-in-time, the size of the database is fluctuating. Metrics that are not point-in-time (also referred to as cumulative metrics) are metrics whose size always increases over time. For example, Blocks Read and Tuples Read are cumulative metrics; the value stays the same or increases.

- Use the **Calculate PIT** switch to specify that the server should calculate a point-in-time value for the metric data. **Calculate PIT** is disabled if **Is PIT** is **Yes**.

PEM allows you to store point-in-time-values of cumulative metrics as well. PEM subtracts the last collected value of a cumulative metric from the current value, and stores the difference as a point-in-time value.



Use the **Code** tab to specify the default code that will be executed by the probe.

- If the probe is a SQL probe, you must specify the SQL SELECT statement invoked by the probe on the **Code** tab. The column names returned by the query must match the **Internal Name** specified on the **Column** tab. The number of columns returned by the query, as well as the column name, datatype, etc. must match the information specified on the **Columns** tab.
- If the probe is a Batch probe, you must specify the shell or .bat script that will be invoked when the probe runs. The output of the script should be as follows:
 - The first line must contain the names of the columns provided on the **Columns** tab. Each column name should be separated by a tab (t) character.
 - From the second line onwards, each line should contain the data for each column, separated by a tab character.
 - If a specified column is defined as key column, make sure the script does not produce duplicate data for that column across lines of output.
 - The number of columns specified in the **Columns** tab and their names, data type, etc. should match with the output of the script output.
- If the probe is a WMI probe, you must specify the WMI query as a SELECT WMI query. The column name referenced in the SELECT statement should be same as the name of the corresponding column specified on the **Column** tab. The column names returned by the query must match the **Internal Name** specified on the **Column** tab. The number of columns returned by the query, as well as the column name, datatype, etc. must match the information specified on the **Columns** tab.

The screenshot shows the 'Probes' configuration page in the EDB Postgres Enterprise Manager. The 'Alternate Code' tab is active. A warning message at the bottom of the form says 'Please specify Probe name'. The 'Applies to all database server versions?' switch is set to 'Yes'. The 'Database version(s)' dropdown is empty. The 'Probe code' column is blank.

Use the **Alternate Code** tab to provide code that will be invoked if the probe fires on a specific version of the server. To provide version-specific code, move the **Applies to all database server versions?** switch to **No**, and click the **Add** button. Then, use the **Database version(s)** drop-down listbox to select the version to which the code will apply. After selecting the version, click the **Edit** button (to the left of the version name) to provide the code that will execute when the probe fires.

If you select a database version, and leave the **Probe code** column blank, PEM will invoke the code specified on the **Code** tab when the probe executes on a server that matches that version.

When you've finished defining the probe, click the **Save** icon (in the corner of the **Custom Probes** tab) to save the definition, and make the probe data available for use on custom charts and graphs.

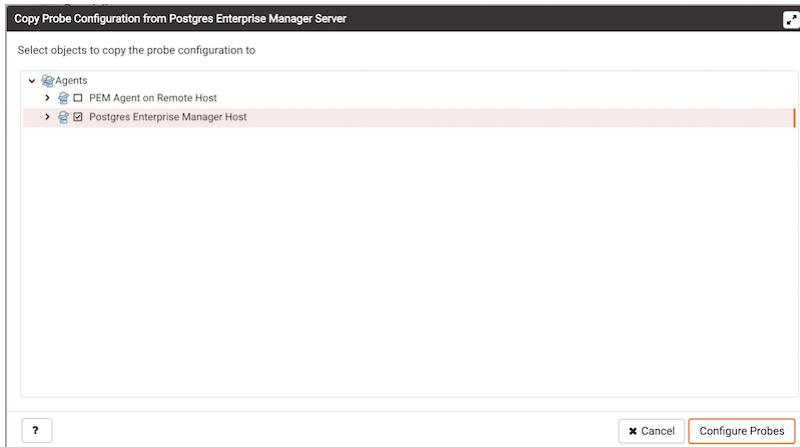
Deleting Probes

You may delete only user-defined probes. When you delete a probe, the probe is marked for deletion, and displayed in red font in the Custom Probes table. Probes marked for deletion will be deleted when you click the Save icon (preserving changes made on the Custom Probes tab). During the deletion the probe definition is deleted and any corresponding tables are dropped from the `pemdata` and `pemhistory` schemas.

System probes are the built-in probes provided by PEM, and are part of the PEM schema. You may only modify system probes; if you attempt to delete a system probe, you will receive an error from PEM.

7.4.12.2 Copy Probe Configuration

You can use the **Copy Probe Configuration...** dialog to copy probe definitions from one monitored object to one or more monitored objects of the same type. To open the **Copy Probe Configuration...** dialog, highlight the object from which you are copying probes in the PEM client tree control, and select **Manage Probes...** from the **Management** menu. When the **Manage Probes** tab opens, click on **Copy Probe** to open the **Copy Probe Configuration** dialog:



The dialog will copy the probe definitions from the object through which the [Copy Probe Configuration](#) dialog was opened, to the location(s) selected on the [Copy Probe Configuration](#) dialog tree control.

Note that if you specify a parent node in the [Copy Probe Configuration](#) tree control, PEM will copy the probe configurations to each object (of the same type) that resides under that node in the tree control. For example, to copy the probe definitions from one schema to all schemas that reside within a database, you only need to select the parent database of the target schemas. Please note that a red warning symbol is displayed to the left of the name of a listed target object if that object is the source of the probe that is being copied.

When you have selected the target object or objects, click the [Configure Probes](#) button to copy the probe definitions to the location selected on the dialog.

7.4.12.3 Probe Configuration

A [probe](#) is a scheduled task that returns a set of performance metrics about a specific monitored server, database, operating system or agent. You can use the [Manage Probes](#) tab to override the default configuration and customize the behavior of each probe. To open the [Manage Probes](#) tab, select [Manage Probes...](#) from the [Management](#) menu.

Properties SQL Statistics Dependencies Dependents Monitoring **Manage Probes** x

Description

Manage Custom Probes: PEM uses probes to retrieve statistics from a monitored server, database, operating system or agent. You can view, reconfigure, delete, or create your own custom probes.

Copy Probes: PEM allows copying of probes from any chosen object recursively down through the object hierarchy. Click on Copy Probes to quickly copy the displayed probe configuration to a selected target.

Quick Links

 Manage Custom Probes
 Copy Probes
 Help

Probes

Probe name	Execution Frequency		Enabled?		Data Retention		Days
	Default?	Minutes	Seconds	Default?	Probe Enable?	Default?	
Background Writer Statistics	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Blocked Session Information	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Data and Log File Analysis	<input checked="" type="checkbox"/>	0	10	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Database Frozen XID	<input checked="" type="checkbox"/>	720	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Database Size	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Database Statistics	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	90
Failover Manager Cluster Info	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	7
Failover Manager Node Status	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	7
Lock Information	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Number of Prepared Transactions	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Number of WAL Files	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Object Catalog: Database	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Object Catalog: Tablespace	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
PG HBA Conf	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Server Information	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Server log Configuration	<input checked="" type="checkbox"/>	0	10	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Session Information	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Settings	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
Streaming Replication	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	180
Streaming Replication Database Conflicts	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	180
Streaming Replication Lag Time	<input checked="" type="checkbox"/>	5	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	180
Tablespace Size	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	180
User Information	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	180
WAL Archive Status	<input checked="" type="checkbox"/>	30	0	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	180

The **Manage Probes** tab provides a set of **Quick Links** that you can use to create and manage probes:

- Click the **Manage Custom Probes** <pem_custom_probes> icon to open the Custom Probes` tab and create or modify a custom probe.
- Click the **Copy Probes** <copy_probe_config> icon to open the Copy Probe` dialog, and copy the probe configurations from the currently selected object to one or more monitored objects.

A probe monitors a unique set of metrics for each specific object type (server, database, database object, or agent); select the name of an object in the tree control to review the probes for that object.

To modify the properties associated with a probe, highlight the name of a probe, and customize the settings that are displayed in the **Probes** table:

- Move the **Default** switch in the **Execution Frequency** columns to **No** to enable the **Minutes** and **Seconds** selectors, and specify a non-default value for the length of time between executions of the probe.
- Move the **Default** switch in the **Enabled?** column to **No** to change the state of the probe, and indicate if the probe is active or not active. If data from a probe that is **Disabled** is used in a chart, the chart will display an information icon in the upper-left corner that allows you to enable the probe by clicking the provided link.
- Move the **Default** switch in the **Data Retention** column to **No** to enable the **Day(s)** field and specify the number of days that information gathered by the probe is stored on the PEM server.

The **Manage Probes** tab may also display information about probes that cannot be modified from the current node; if a probe cannot be modified from the current dialog, the switches for that probe are disabled. Generally, a disabled probe can be modified from a node that is higher in the hierarchy of the PEM client tree control. Select another object in the tree control to change which probes are displayed or enabled on the **Manage Probes** tab.

Contents:

7.4.12.3.1 Probes

A **probe** is a scheduled task that retrieves information about the database objects that are being monitored by the PEM agent. PEM uses the collected information to build the graphs displayed on each homepage. The **Manage Probes** tab (accessed via the **Management** menu) allows you to modify the data collection schedule and the length of time that PEM will retain information returned by a specific probe.

Unless otherwise noted, Postgres Enterprise Manager enables the probes listed in the table below:

Probe Name	Information Monitored by Probe	Probe Configuration Level
Background Writer Statistics	<p>This probe monitors information about the background writer. The information includes:</p> <ul style="list-style-type: none"> - The number of timed checkpoints - The number of requested checkpoints - The number of buffers written (by checkpoint) - The number of buffers written (by background writer) - The number of background writer cycles - The number of background buffers written - The number of buffers allocated 	Server
Blocked Session Information	This probe returns information about blocked sessions.	Server
CPU Usage	This probe monitors CPU Usage information.	Agent
Data and Log File Analysis	<p>This probe monitors information about log files. The information includes:</p> <ul style="list-style-type: none"> - The name of the log file - The directory in which the log file resides 	Server
Database Frozen XID	This probe monitors the frozen XID of each database.	Server
Database Size	<p>This probe monitors information about the size of the monitored databases. The information includes:</p> <ul style="list-style-type: none"> - The time the information was gathered - The database name - The database size (in MB's) 	Server
Database Statistics	<p>This probe monitors database statistics. The information includes:</p> <ul style="list-style-type: none"> - The number of backends - The number of transactions committed - The number of transactions rolled back - The number of blocks read - The number of blocks hit - The number of rows returned - The number of rows fetched - The number of rows inserted - The number of rows updated - The number of rows deleted 	Server

Probe Name	Information Monitored by Probe	Probe Configuration Level
Disk Busy Info	<p>This probe monitors information about disk activity.</p> <ul style="list-style-type: none"> - Note: This probe is not supported on Mac OS X, Solaris or HP-UX 	Agent
Disk Space	<p>This probe monitors information about disk space usage. The information includes:</p> <ul style="list-style-type: none"> - The amount of disk space used - The amount of disk space available 	Agent
EDB Audit Configuration	This probe monitors the audit logging configuration of Postgres Plus Advanced Servers.	Server
Failover Manager Cluster Info	This probe monitors a Failover Manager cluster, returning information about the cluster. This probe is disabled unless a cluster name and path of the Failover Manager binary is provided on the Server Properties dialog.	Server
Failover Manager Node Status	This probe monitors a Failover Manager cluster, returning detailed about each node within the cluster. This probe is disabled unless a cluster name and path of the Failover Manager binary is provided on the Server Properties dialog.	Server
Function Statistics	<p>This probe monitors a database, retrieving information about functions. The information includes:</p> <ul style="list-style-type: none"> - Function names - Argument types - Return values 	Database
Index Size	<p>This probe monitors a database, retrieving information about indexes. The information includes:</p> <ul style="list-style-type: none"> - The name of the index - The time the data was gathered - The size of the index (in MB's) 	Database
Index Statistics	<p>This probe monitors index statistics. The information includes:</p> <ul style="list-style-type: none"> - The number of index scans - The number of rows read - The number of rows fetched - The number of blocks read - The number of blocks hit 	Database
Installed Packages	<p>This probe monitors the packages that are currently installed. The information gathered includes:</p> <ul style="list-style-type: none"> - The name of the installed package - The version of the installed package - The date and time that the probe executed 	Agent
IO Analysis	<p>This probe monitors disk I/O information in. The information includes:</p> <ul style="list-style-type: none"> - The number of blocks read - The number of blocks written - The date and time that the probe executed - Note: This probe is not supported on Mac OS X 	Agent
Load Average	<p>This probe monitors CPU load averages. The information includes:</p> <ul style="list-style-type: none"> - The 1-minute load average - The 5-minute load average - The 15-minute load average - Note: This probe is not supported on Windows 	Agent
Lock Information	<p>This probe monitors lock information. The information includes:</p> <ul style="list-style-type: none"> - The database name - The lock type - The lock mode - The process holding the lock 	Server

Probe Name	Information Monitored by Probe	Probe Configuration Level
Memory Usage	<p>This probe monitors information about system memory usage. The information includes:</p> <ul style="list-style-type: none"> - Total RAM in MB - Free RAM in MB - Total swap memory in MB - Free swap memory in MB - Shared system memory in MB (It is used by tuning wizard to tune the memory parameters for the database server) - On non-windows system, it is <code>shmmax</code> value and read from <code>/proc/sys/kernel/shmmax</code> - On windows, it is same as total memory. 	Agent
Network Statistics	<p>This probe monitors network statistics. The information includes:</p> <ul style="list-style-type: none"> - The interface IP address - The number of packets sent - The number of packets received - The number of bytes sent - The number of bytes received - The link speed (in MB/second) 	Agent
Number of Prepared Transactions	This probe stores the number of prepared transactions.	Server
Number of WAL Files	This probe monitors the number of WAL files.	Server
Object Catalog: Database	<p>This probe monitors a list of databases and their properties. The information includes:</p> <ul style="list-style-type: none"> - The database name - The database encoding type - If the database allows user connections or system connections 	Server
Object Catalog: Foreign Key	<p>This probe monitors a list of foreign keys and their properties. The information includes:</p> <ul style="list-style-type: none"> - The name of the table that contains the foreign key - The name of the table that the foreign key references - The name of the database in which the table resides - The name of the schema in which the table resides 	Schema
Object Catalog: Function	<p>This probe monitors a list of functions and their properties. The information includes:</p> <ul style="list-style-type: none"> - The name of the function - The name of the schema in which the function resides - The name of the database in which the function resides 	Schema
Object Catalog: Index	<p>This probe monitors a list of indexes and their properties. The information includes:</p> <ul style="list-style-type: none"> - The name of the index - The name of the table that the index is associated with - The name of the database in which the indexed table resides 	Schema
Object Catalog: Schema	This probe monitors a list of schemas and their associated databases and servers.	Database
Object Catalog: Sequence	This probe monitors a list of sequences and their properties.	Schema

Probe Name	Information Monitored by Probe	Probe Configuration Level
Object Catalog: Table	<p>This probe monitors a list of table information. The information includes:</p> <ul style="list-style-type: none"> - The table name - The name of the schema in which the table resides - The name of the database in which the schema resides - A Boolean indicator that indicates if the table has a primary key 	Schema
Object Catalog: Tablespace	This probe monitors a list of tablespaces.	Server
Operating System Information	This probe monitors the operating system details and boot time.	Agent
Package Catalog	<p>This probe monitors the packages that are currently available for installation. The information gathered includes:</p> <ul style="list-style-type: none"> - The package name - The package version 	Agent
PG HBA Conf	This probe monitors authentication configuration information from the <code>pg_hba.conf</code> file.	Server
Server Information	This probe monitors information about servers.	Server
Session Information	<p>This probe monitors session information. The information includes:</p> <ul style="list-style-type: none"> - The name of the session user - The date and time that the session connected to the server - The status of the session at the time that the information was gathered (idle, waiting, etc) - The client address and port number 	Server
Settings	This probe monitors the values currently assigned to GUC variables.	Server
SQL Protect	This probe monitors a server, retrieving information about SQL injection attacks.	Server
Slony Replication	This probe monitors lag data for clusters replicated using Slony.	Database
Streaming Replication	<p>This probe monitors a cluster that is using streaming replication, retrieving information about:</p> <ul style="list-style-type: none"> - The sent Xlog location (in bytes) - The write Xlog location (in bytes) - The flush Xlog location (in bytes) - The replay Xlog location (in bytes) - The Xlog lag (in segments) - The Xlog lag (in pages) 	Server
Streaming Replication Lag Time	<p>This probe monitors a cluster that is using streaming replication, retrieving lag information about:</p> <ul style="list-style-type: none"> - Replication lag time (in seconds) - Current status of replication (running/paused) 	Server

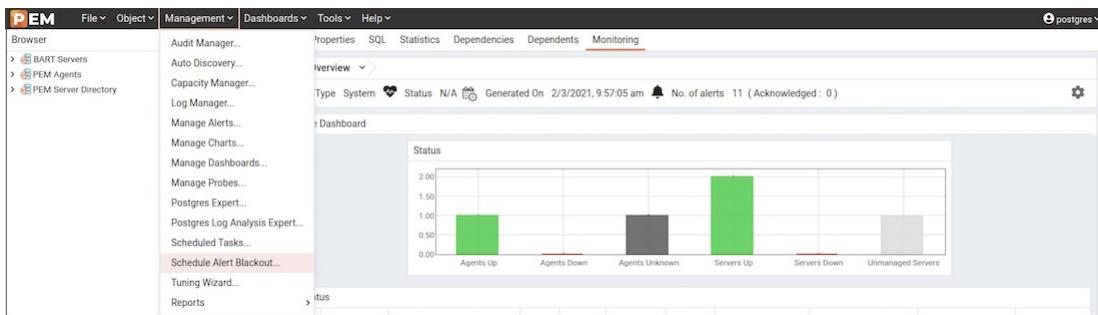
Probe Name	Information Monitored by Probe	Probe Configuration Level
Streaming Replication Database Conflicts	<p>This probe monitors a database that is using streaming replication, retrieving information about any conflicts that arise. This includes information about queries that have been canceled due to:</p> <ul style="list-style-type: none"> - The # of drop tablespace conflicts - The # of lock timeout conflicts - The # of old snapshot conflicts - The # of pinned buffer conflicts - The # of deadlock conflicts 	Server
Table Bloat	<p>This probe monitors information about the current table bloat. The information includes:</p> <ul style="list-style-type: none"> - The name of the table - The name of the schema in which the table resides - The estimated number of pages - The estimated number of wasted pages - The estimated number of bytes per row 	Database
Table Frozen XID	This probe monitors the frozen XID of each table.	Schema
Table Size	<p>This probe monitors information about table size. The information includes:</p> <ul style="list-style-type: none"> - Table size (in MB's) - Total index size (in MB's) - Total table size, with indexes and TOAST (in MB's) 	Database
Table Statistics	<p>This probe monitors table statistics. The information includes:</p> <ul style="list-style-type: none"> - The number of sequential scans - The number of sequential scan rows - The number of index scans - The number of index scan rows - The number of rows inserted - The number of rows updated - The number of rows deleted - The number of live rows - The number of dead rows - The last VACUUM - The last auto-vacuum - The last ANALYZE - The last auto-analyze - The number of pages estimated by ANALYZE - The number of rows estimated by ANALYZE 	Database
Tablespace Size	This probe monitors a list of tablespaces and their sizes.	Server
User Information	<p>This probe monitors a list of the current users. The stored information includes:</p> <ul style="list-style-type: none"> - The user name - The user type (superuser vs. non-superuser) - The server to which the user is connected 	Server
WAL Archive Status	<p>This probe monitors the status of the WAL archive. The stored information includes:</p> <ul style="list-style-type: none"> - The # of WAL archives done - The # of WAL archives pending - The last archive time - The # of WAL archives failed - The time of the last failure 	Server

Probe Name	Information Monitored by Probe	Probe Configuration Level
xDB Replication	This probe monitors lag data for clusters replicated using xDB replication.	Database

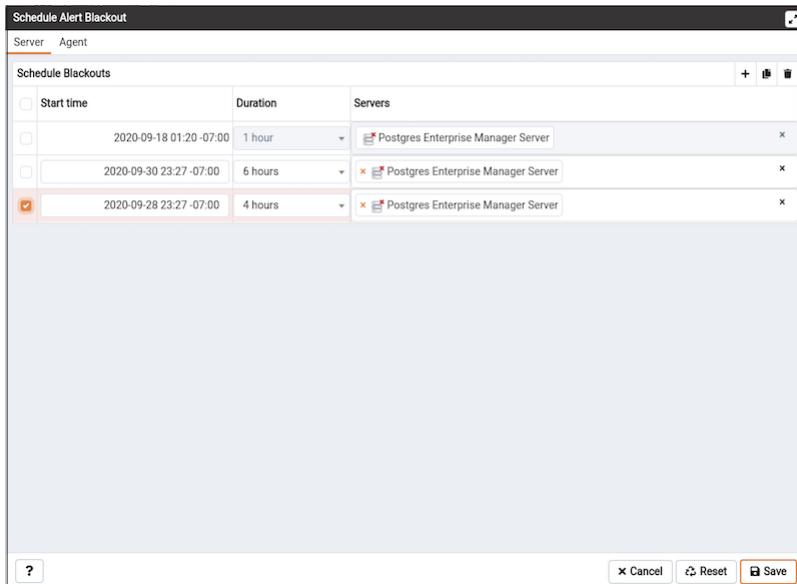
7.4.13 Schedule Alert Blackout

You can use the **Schedule Alert Blackout** option on the **Management** menu to schedule an alert blackout for your Postgres servers and PEM Agents during maintenance. Alerts will not be raised during a defined blackout period.

To schedule an alert blackout, click on the **Management** menu and select **Schedule Alert Blackout**.



When the **Schedule Alert Blackout** dialog opens, use the tabs on the dialog to define the blackout period for servers and agents. Open the **Server** tab and click the Add icon (+) at the top right corner to add new row.



Use the fields on the **Server** tab to provide information about a Server to blackout the alerts:

- Use the **Start time** field to provide the date and time to start the alert blackout.
- Use the **Duration** field to provide the interval for which you want to blackout the alerts.
- Use the **Servers** field to provide the server name for which you want to blackout the alerts. You can also select multiple servers to blackout the alerts at same time.

Once all the details are provided, you can save the details by clicking on **Save** button on the right bottom corner of the dialog. Once saved, it cannot be edited. The alerts will not be displayed on the Alerts dashboard for the scheduled interval of that particular server.

You can also schedule a blackout period for PEM Agents via the **Agent** tab on the dialog. Open the **Agent** tab and click the Add icon (+) at the top right corner to add new row.

Start time	Duration	Agents
2020-09-18 03:33 -07:00	1 hour	Postgres Enterprise Manager Host
2020-09-25 03:33 -07:00	4 hours	Agent2
2020-09-30 03:33 -07:00	23 hours	Postgres Enterprise Manager Host
2020-09-21 23:23 -07:00	6 hours	Agent2

Buttons at the bottom: ? (Help), Cancel, Reset, Save (highlighted).

Use the fields on the **Agent** tab to provide the information about an Agent to blackout the alerts:

- Use the **Start time** field to provide the date and time to start the alert blackout.
- Use the **Duration** field to provide the interval for which you want to blackout the alerts.
- Use the **Agents** field to provide the Agent name for which you want to blackout the alerts. All server level alerts, for the servers bind to that particular agent will blackout.

Once all the details are provided, you can save the details by clicking on **Save** button on the right bottom corner of the dialog. Once saved, it cannot be edited. The alerts will not be displayed on the Alert dashboard for the scheduled interval for that **PEM Agent**.

You can use **Clone** button from the top right corner of dialog, to clone the scheduling of alert blackout. Select the servers or agents you want to clone and then click on **Clone** button to create the cloned copy of all the selected servers or agents. You can edit newly created schedules as needed, and then click **Save**.

You can use **Delete** button from the top right corner of dialog to remove a scheduled alert blackout. Select the servers or agents and then click on highlighted **Delete** button in the right top corner to remove the scheduled alerts associated with that server or agent.



Select a server for which you wish to delete the scheduled alert blackout and then click on the **Delete** button. The server will ask for confirmation before deleting that row.



You can use the **Reset** button to reset the details on the Alert Blackout dialog to the default settings. Please note that all saved blackouts will remain unaffected after resetting the current dialog values.

7.4.14 Scheduled System Jobs

PEM defines system jobs to take care of cleanup activities at scheduled intervals. All of the system jobs are enabled by default, and are scheduled to deploy on a regular interval. You can query the `pem.job` table in the `pem` database to review a list of the system jobs. The current schedule for system jobs is stored in the `pem.schedule` table in the `pem` database.

The system job names, their descriptions, and default deployment intervals are listed in the table below:

7.4.15 Scheduled Task Tab

You can schedule the execution of user-defined tasks on registered servers for a time that is most convenient, and least intrusive to your users. Tasks may be one-off, or recurring and are comprised of one or more steps, which may be a SQL script, a batch/shell script, or an internal function in the PEM agent. You can view pending tasks on the *Scheduled Tasks tab*.

To open the **Scheduled Tasks** tab, select either a PEM Agent or a managed server in the tree control of the PEM client and select **Scheduled Tasks...** from the **Management** menu.

The tab features a legend, displaying the icons that identify the status of each task.

The **Manage Tasks** table displays a list of tasks. Set **Show system tasks?** to **Yes** to display system tasks; if it is set to **No**, only user-defined tasks are displayed. System tasks are displayed with a grey background, and may not be modified.

Use the **Refresh** icon to update the list of tasks displayed in the table. The table displays general information about each task.

- The **Execution** drop-down provides access to detailed information about each step in the task.
- The **Status** field lists the status of the current task.
- The **Enabled?** switch displays **Yes** if the task is enabled; **No** if the task is disabled.
- The **Name** field displays the name of the task.
- The **Agent** or **Server** field displays the name of the agent responsible for executing the task or the server on which the task will execute.

Highlight the name of a user-defined task and click the **Edit** icon (to the left of a task name) to access detailed information about the selected task.

The screenshot shows the 'Tasks' section of the EDB Postgres Enterprise Manager interface. A single task, 'Job_backup_pem_db', is listed in the 'Manage Tasks' table. The task details are shown in the 'General' tab, including its status (Never ran), owner (postgres), name (Job_backup_pem_db), agent (Postgres Enterprise Manager Host), description (Job for taking backup of pem database), and creation date (2020-04-28 15:38:07).

The **General** tab displays information about the scheduled task:

- The **Status** field lists the status of the current task.
- The **Enabled?** switch displays **Yes** if the task is enabled; **No** if the task is disabled.
- The **Name** field displays the name of the task.
- The **Agent or Server** field displays the name of the agent responsible for executing the task or the server on which the task will be performed.
- The **Description** field displays a description of the task.
- The **Last run** field displays the date and time of the last execution of the task.
- The **Next run** field displays the date and time of the next scheduled execution of the task.
- The **Created** field displays the date and time that the task was defined.

Highlight the name of a user-defined task and open the **Steps** arrow to review a list of the steps within the task.

The screenshot shows the 'Tasks' section with multiple tasks listed. The 'Job_backup_pem_db' task is selected and its steps are expanded. The 'Steps' table shows two steps: 'Modify postgresql.conf' and 'Server Restart'. The 'Execution date' for both steps is 2020-04-27. The 'Description' for the first step is 'Server ID: 1, agent ID: 1'. The 'Log details' button is visible for each step.

Step	Type	Status	Result	Start / Next run	Duration	Output	Log details
Modify postgresql.conf	Internal			0 2020-04-27 14:32:49	00:00:00.223211		
Server Restart	Internal			0 2020-04-27 14:32:49	00:00:13.287379		

The list of steps displays general information about each step in the task:

- The **Execution date** field displays the date on which the step will execute. Step history is grouped by execution date; use the arrow to the left of an execution date to expand the node and review the task logs for that date.
- The **Description** field displays a description of the step.

Use the arrow to the left of an execution date (in the **Steps** column) to view detailed information about the step:

- The **Step** field displays a description of the step.
- The **Type** field displays the task type.
- The **Status** field lists the status of the current task.
- If applicable, the **Result** field displays code generated during the execution of the step.
- The **Start/Next run** field displays the date and time at which the task executed or will execute again.

- The **Duration** field displays the length of time that the task required for execution.
- The **Output** field displays the result set returned by the execution of the task. By default, it displays the first 250 characters. You can also change the display characters by changing the **Schedule Tasks** options in the **Preference** dialog.
- The **Log details** field allows you to open the log in the new browser window and also download the complete log.

To delete a user-defined task, highlight the name of the task, and click the **Delete** icon located to the left of a task's name. The task will be marked for deletion, and removed when you click the **Save** icon (located in the upper-right corner of the **Manage Tasks** table).

!!! Note Tasks with no **Next run** date will automatically be removed from the PEM server when the last run date is more than `probe_log_retention_time <pem_config_options>` days ago.

Please note that if any of the scheduled tasks for backup, restore, validate host, validate server or delete obsolete backup for any of the BART Server gets deleted, it will not display under the **BART Tool Activities** graph of BART Server's dashboard. However, it gets listed under the **Initiated Server Backups** list.

7.4.16 Creating a PEM Scheduled Job

You can create a PEM scheduled job to perform a set of custom-defined steps in the specified sequence. These steps may contain SQL code or a batch/shell script that you may run on a server that is bound with the agent. You can schedule these jobs to suit your business requirements. For example, you can create a job for taking a backup of a particular database server and schedule it to run on a specific date and time of every month.

To create or manage a PEM scheduled job, use the PEM tree control to browse to the PEM agent for which you want to create the job. The tree control will display a Jobs node, under which currently defined jobs are displayed. To add a new job, right click on the Jobs node, and select Create Job... from the context menu.

When the Create Agent Job dialog opens, use the tabs on the Create - Agent Job dialog to define the steps and schedule that make up a PEM scheduled job.



Use the fields on the **General** tab to provide general information about a job:

- Provide a name for the job in the **Name** field.
- Move the **Enabled** switch to the **Yes** position to enable a job, or **No** to disable a job.
- Use the **Comment** field to store notes about the job.

Name	Enabled?	Kind	On error
Step1_backup	True	SQL	Success

General **Code**

Name: Step1_backup
Enabled?: Yes
Kind: SQL
On error: Success
Server: Postgres Enterprise Manager Server (192.168.1.19:5432)
Database: pem
Comment:

Save

Use the **Steps** tab to define and manage the steps that the job will perform. Click the Add icon (+) to add a new step; then click the compose icon (located at the left side of the header) to open the step definition dialog:

Name
Job_backup_pem_db

General

Name: Job_backup_pem_db
Enabled?: Yes
Comment: Job for taking backup of pem database

Save

Use fields on the step definition dialog to define the step:

- Provide a name for the step in the **Name** field; please note that steps will be performed in alphanumeric order by name.
- Use the **Enabled** switch to include the step when executing the job (**True**) or to disable the step (**False**).
- Use the **Kind** switch to indicate if the job step invokes SQL code (**SQL**) or a batch script (**Batch**).
- If you select **SQL**, use the **Code** tab to provide SQL code for the step.

- If you select **Batch**, use the **Code** tab to provide the batch script that will be executed during the step.

Use the **On error** drop-down to specify the behavior of pgAgent if it encounters an error while executing the step. Select from:

- Fail** - Stop the job if you encounter an error while processing this step.
- Success** - Mark the step as completing successfully, and continue.
- Ignore** - Ignore the error, and continue.
- If you have selected SQL as your input for **Kind** switch, provide the following additional information:
 - Use the **Server** field to specify the server that is bound with the agent for which you are creating the PEM scheduled job.
 - Use the **Database** field to specify the database that is associated with the server that you have selected.
- Use the **Comment** field to provide a comment about the step.



Use the context-sensitive field on the step definition dialog's **Code** tab to provide the SQL code or batch script that will be executed during the step:

- If the step invokes SQL code, provide one or more SQL statements in the **SQL query** field.
- If the step invokes a batch script, provide the script in the **Code** field. If you are running on a Windows server, standard batch file syntax must be used. When running on a Linux server, any shell script may be used, provided that a suitable interpreter is specified on the first line (e.g. `#!/bin/sh`). Along with the defined inline code, you can also provide the path of any batch script, shell script, or SQL file on the filesystem.

To invoke a script on a Linux system, you must modify the entry for **batch_script_user** parameter of agent.cfg file and specify the user that should be used to run the script. You can either specify a non-root user or root for this parameter. If you do not specify a user, or the specified user does not exist, then the script will not be executed. Restart the agent after modifying the file. If pemagent is being run by a non-root user then the value of **batch_script_user** will be ignored and the script will be executed by the same non-root user that is being used for running the pemagent.

To invoke a script on a Windows system, set the registry entry for **AllowBatchJobSteps** as true and restart the PEM agent. PEM registry entries are located in HKEY_LOCAL_MACHINE\Software\Wow6432Node\EnterpriseDB\PEM\agent.

After providing all the information required by the step, click the **Save** button to save and close the step definition

dialog.

Click the add icon (+) to add each additional step, or select the **Schedules** tab to define the job schedule.

Note

When you create the Job, the time fields under **Schedules** tab takes the timezone of the client machine. While saving the jobs on the PEM Server, the timezone gets converted according to PEM Server's timezone irrespective of PEM Agent's location.

Click the Add icon (+) to add a schedule for the job; then click the compose icon (located at the left side of the header) to open the schedule definition dialog:



Use the fields on the Schedules definition tab to specify the days and times at which the job will execute.

- Provide a name for the schedule in the **Name** field.
- Use the **Enabled** switch to indicate that pgAgent should use the schedule (**Yes**) or to disable the schedule (**No**).
- Use the calendar selector in the **Start** field to specify the starting date and time for the schedule.
- Use the calendar selector in the **End** field to specify the ending date and time for the schedule.
- Use the **Comment** field to provide a comment about the schedule.

Select the **Repeat** tab to define the days on which the schedule will execute.



Use the fields on the **Repeat** tab to specify the details about the schedule in a cron-style format. The job will execute on each date or time element selected on the **Repeat** tab.

Click within a field to open a list of valid values for that field; click on a specific value to add that value to the list of selected values for the field. To clear the values from a field, click the X located at the right-side of the field.

Use the fields within the **Days** box to specify the days on which the job will execute:

- Use the **Week Days** field to select the days on which the job will execute.
- Use the **Month Days** field to select the numeric days on which the job will execute. Specify the **Last Day** to indicate that the job should be performed on the last day of the month, regardless of the date.
- Use the **Months** field to select the months in which the job will execute.

Use the fields within the **Times** box to specify the times at which the job will execute:

- Use the **Hours** field to select the hour at which the job will execute.
- Use the **Minutes** field to select the minute at which the job will execute.

Select the **Exceptions** tab to specify any days on which the schedule will **not** execute.



Use the fields on the **Exceptions** tab to specify days on which you wish the job to not execute; for example, you may wish for jobs to not execute on national holidays.

Click the Add icon (+) to add a row to the exception table, then:

- Click within the **Date** column to open a calendar selector, and select a date on which the job will not execute. Specify **<Any>** in the **Date** column to indicate that the job should not execute on any day at the time selected.
- Click within the **Time** column to open a time selector, and specify a time on which the job will not execute. Specify **<Any>** in the **Time** column to indicate that the job should not execute at any time on the day selected.

Select the **Notifications** tab to configure the email notification settings on job level:



Use the fields on the **Notifications** tab to configure the email notification settings for a job:

- Use the **Send the notifications** field to specify when you want the email notifications to be sent.
- Use the **Email group** field to specify the email group that should receive the email notification.

When you've finished defining the schedule, you can use the **SQL** tab to review the code that will create or modify your job.

```

1 DO $$
2 DECLARE
3     jid integer;
4     scid integer;
5 BEGIN
6     -- Creating a new job
7     INSERT INTO pem.job(
8         agent_id, jobname, jobdesc, jobenabled, notify, email_group_id
9     ) VALUES (
10        1::integer, 'Job_backup_pem_db'::text, 'Job for taking backup of pem database'::text, true, 'ALWAYS'::
11    ) RETURNING jobid INTO jid;
12
13 -- Steps
14
15 -- Inserting a step (jobid: NULL)
16 INSERT INTO pem.jobstep(
17     jstjobid, jstname, jstenabled, jstkind, jstonerror, jstcode, jstdesc,
18     server_id, database_name
19 ) VALUES (
20     jid,
21     'step1_backup'::text,
22     true,
23     's)::character(1),
24     's)::character(1),
25     ''::text, ''::text,
26     '1'::integer,
27     'pem'::text
28 );
29
30
31 -- Schedules
32 -- Inserting a schedule

```

Click the **Save** button to save the job definition, or **Cancel** to exit the job without saving. Use the **Reset** button to remove your unsaved entries from the dialog.

After saving a job, the job will be listed under the **Jobs** node of the PEM tree control of the server on which it was defined. The **Properties** tab in the PEM console will display a high-level overview of the selected job, and the Statistics tab will show the details of each run of the job. To modify an existing job or to review detailed information about a job, right-click on a job name, and select **Properties** from the context menu.

7.4.17 Sending email notifications for a job

You can configure the settings in PEM console for sending the SMTP trap on success or failure of a system-generated job (listed under scheduled tasks) or a custom-defined agent job. For information on custom-defined agent job, see ‘Creating PEM Scheduled Jobs’. These email notification settings can be configured at following three levels (in order of precedence) to send email notifications to the specified user group:

- Job level
- Agent level
- PEM server level (default level)

Configuring job notifications at job level

You can configure email notification settings at job level only for a custom-defined agent job in one of the following

ways:

- For a new agent job, you can configure the email notification settings in the **Notification** tab of **Create-Agent Job** wizard while creating the job itself.
- For an existing custom-defined job, you can edit the properties of the job and configure the notification settings.



Use the fields on the Notifications tab to configure the email notification settings on job level:

- Use the **Send the notifications** field to specify when you want the email notifications to be sent.
- Use the **Email group** field to specify the email group that should receive the email notification.

Configuring job notifications at agent level

Select the agent in the tree view, right click and select **Properties**. In the Properties dialog, select the **Job notifications** tab.



Use the fields on the Job notifications tab to configure the email notification settings on agent level:

- Use the **Override default configuration?** switch to specify if you want the agent level job notification settings to override the default job notification settings. If you select Yes for this switch, you can use the rest of the settings on this dialog to define when and to whom the job notifications should be sent. Please note that the rest of the settings on this dialog work only if you enable the **Override default configuration?** switch.
- Use the **Email on job completion?** switch to specify if the job notification should be sent on the successful job completion.
- Use the **Email on a job failure?** switch to specify if the job notification should be sent on the failure of a job.
- Use the **Email group** field to specify the email group to whom the job notification should be sent.

Configuring job notifications at server level

You can use the **Server Configuration** dialog to provide information about your email notification configuration at PEM server level. To open Server Configuration dialog, select **Server Configuration...** from the PEM client's Management menu.



Four server configuration parameters specify information about your job notification preferences at PEM server level:

- Use the **job_failure_notification** switch to specify if you want to send email notification after each job failure.
- Use the **job_notification_email_group** parameter to specify the email group that should receive the

email notification.

- Use the `job_retention_time parameter` to specify the number of days that non-recurring scheduled tasks should be retained in the system.
- Use the `job_status_change_notification` switch to specify if you want to send email notification after each job status change, irrespective of its status being a failure, success, or interrupted.

7.4.18 Task Viewer

Postgres Enterprise Manager runs tasks on managed servers by scheduling them on the PEM server for the agent on the managed server to execute at the appropriate time. Tasks may be one-off, or recurring and are comprised of one or more steps, which may be a SQL script, a batch/shell script, or an internal function in the PEM agent. Tasks may be viewed using the *Scheduled Tasks dialogue*.

To open the *Scheduled Tasks dialogue*, select either a PEM Agent or a managed server in the tree control of the PEM client and select the `Scheduled Tasks` menu option from the `Server` sub-menu of the `Management` menu, or from the context menu.



The dialogue displays the task data relating to the selected object when it was opened. The following details are shown:

- Status - The status of the task following the last execution.
- Enabled? - An indicator showing whether the task is enabled or not.
- Name - The name of the task.
- Server - The server on which the task will be executed, where applicable.
- Description - A description of the task.
- Last Run - The time the task was last executed.
- Next Run - The time the task is next scheduled to execute, if any.
- Created - The time and date that the task was created.

In order to [view the log records](#) for a task, select it in the list and click the `Log Viewer` button.

In order to remove tasks and their associated log records (if present), click the checkbox to select each task to be removed, and then click the `Remove` button.

!!! Note Tasks with no next run date will automatically be removed from the PEM server when the last run date is more than `probe_log_retention_time <pem_config_options>` days ago.

Contents:

7.4.18.1 Log Viewer

When PEM executes [scheduled tasks](#), log records are created to record the status of each step of the task for diagnostic purposes. Log records can be viewed on the *Log Viewer* dialogue, opened from the [Scheduled Task](#) dialogue.

The screenshot shows the 'Tasks' section of the Log Viewer dialogue. It displays a table of scheduled tasks with columns for Logs, Status, Enabled?, Name, Agent, and Owner. Three tasks are listed: 'PEM Log Manager Log Import - Server 1' (Agent: Postgres Enterprise Manager Host, Owner: postgres), 'Server log configuration request' (Agent: Postgres Enterprise Manager Host, Owner: postgres), and another 'Server log configuration request' (Agent: Postgres Enterprise Manager Host, Owner: postgres). Below this is a table for 'Steps' with columns for Steps, Execution date, and Description. A single step is shown for '2020-04-27' with the description 'Server ID: 1, agent ID: 1'. At the bottom is a detailed table for the first step, with columns for Step, Type, Status, Result, Start / Next run, Duration, and Output. Two rows are listed: 'Modify postgresql.conf' (Internal, Result: 0) and 'Server Restart' (Internal, Result: 0).

The dialogue displays the log data relating to each step of the task:

- Step - The name of the step.
- Type - The type of the step, one of SQL, Batch or Internal.
- Status - The status of the step.
- Result - The numeric result of the step. For a batch step, this will be the return code of the script.
- Start / Next Run - The schedule for the next run.
- Duration - The duration of the step.
- Output - The output text from the step, if any.

7.4.19 Monitoring a Failover Manager Cluster

You can configure PEM to display status information about one or more Failover Manager clusters on the Streaming Replication dashboard. Before configuring PEM to monitor a Failover Manager cluster, you must install and configure Streaming Replication and Failover Manager on the cluster.

Please note that your Streaming Replication `standby.signal` file must include the following parameters:

- `primary_conninfo`
- `promote_trigger_file`

For information about installing and configuring Failover Manager and Streaming Replication, please see the EnterpriseDB Failover Manager Guide, available at www.enterprisedb.com.

To configure PEM to monitor a Failover Manager cluster, use the PEM client to create a server definition for the primary node of the Failover Manager cluster. Use the tabs on the [New Server Registration](#) dialog to specify general connection properties for the primary node; use fields on the [Advanced](#) tab to specify information about the Failover Manager cluster:

- Use the `EFM Cluster Name` field to specify the name of the Failover Manager cluster. The cluster name is the prefix of the name of the cluster properties file. For example, if your cluster properties file is named `efm.properties`, your cluster name is `efm`.
- Use the `EFM Installation Path` field to specify the location of the Failover Manager binary file. By default, the Failover Manager binary file is installed in `/usr/efm-x.x/bin`.

After saving the server definition, the primary node will be included in the list of servers under the **PEM Server Directory** in the PEM client **Object browser** tree, and will be displayed on the **Global Overview** dashboard.

To include Failover Manager information on the Streaming Analysis dashboard, you must enable the following probes for each node in the Failover Manager cluster:

- Failover Manager Cluster Info
- Failover Manager Node Status

To enable a probe, right click on the node name, and select **Manage Probes** from the **Management** menu.

To view the **Streaming Replication Analysis** dashboard and the status of the Failover Manager cluster, right click on the name of the primary node in the **Object browser** tree control and navigate through the **Dashboards** menu to select **Streaming Replication Analysis**.

Promoting a Cluster

Select the **Replace Cluster Primary** from **Server** under the **Tools** menu to start the failover process. When you select **Replace Cluster Primary**, a popup opens, asking you to confirm that you wish to replace the current primary node:



Select **No** to exit the popup without replacing the current primary node.

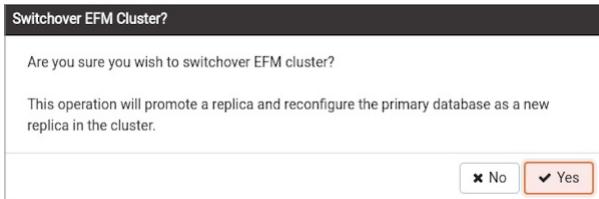
Select **Yes** to remove the current primary node from the Failover Manager cluster and promote a replica node to the role of read/write primary node within a Failover Manager cluster. The node with the highest promotion priority (defined in Failover Manager) will become the new primary node. PEM will display a dialog, reporting the job status.



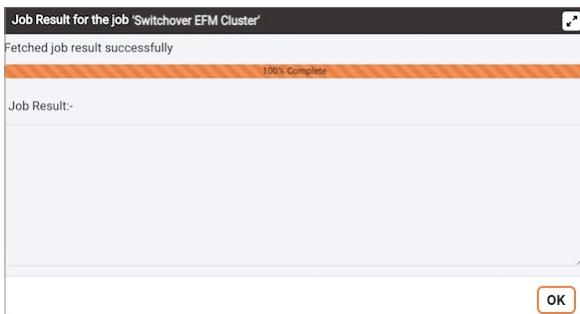
When the job completes and the Streaming Replication Analysis dashboard refreshes, you can review the **Failover Manager Node Status** table to confirm that a replica node has been promoted to the role of primary within the Failover Manager cluster.

Switchover EFM Cluster

You can use the PEM client to switchover the primary node of a Failover Manager cluster with a replica node. To initiate the switchover process, select **Switchover EFM Cluster** from the **Tools** menu. A dialog opens, asking you to confirm that you wish to switchover EFM cluster.



Select **Yes** to switch over EFM cluster from the Failover Manager cluster and promote a replica node to the role of read/write primary node and reconfigure the primary database as a new replica within a Failover Manager cluster. The node with the highest promotion priority (defined in Failover Manager) will become the new primary node. PEM will display a dialog, reporting the job status.



When the job completes and the Streaming Replication Analysis dashboard refreshes, you can review the **Failover Manager Node Status** table to confirm that a switchover within the Failover Manager cluster.

7.4.20 Performance Diagnostic

You can use the Performance Diagnostic dashboard to analyze the database performance for Postgres instances by monitoring the wait events. To display the diagnostic graphs, PEM uses the data collected by EDB Wait States module.

Performance Diagnostic feature is supported for Advanced Server databases from PEM 7.6 version onwards and for PostgreSQL databases it is supported from PEM 8.0 onwards.

Note

For PostgreSQL databases, Performance Diagnostics is supported only for versions 10, 11, and 12 installed on the supported CentOS or RHEL platforms.

For more information on EDB Wait States, see [EDB Postgres Advanced Server Guide](#).

You can analyze the Wait States data on multiple levels by narrowing down your selection of data. Each level of the graph is populated on the basis of your selection of data at the higher level.

Prerequisite:

- For PostgreSQL, you need to install `edb_wait_states_<X>` package from `edb.repo` where `<X>` is the version of PostgreSQL Server. You can refer to [EDB Build Repository](#) for the steps to install this package. For Advanced Server, you need to install `edb-as<X>-server-edb-modules`, Where `<X>` is the version of Advanced Server.
- Once you ensure that EDB Wait States module of EDB Postgres Advanced Server is installed, then configure the list of libraries in the `postgresql.conf` file as below:

```
shared_preload_libraries = '$libdir/edb_wait_states'
```

Restart the database server, and then create the following extension in the maintenance database:

```
CREATE EXTENSION edb_wait_states;
```

- You must have super user privileges to access the Performance Diagnostic dashboard.

You get the following error while accessing the Performance Diagnostic dashboard if the above prerequisites are not met:



To open the Performance Diagnostic dashboard, select **Server** and then **Performance Diagnostic...** from the **Tools** menu of the PEM client.



By default, the top most Performance Diagnostic graph pulls the data of last one hour, starting from current date and time. This graph shows the time series containing the number of active sessions. Each point of this time series represents the active sessions and wait events at a particular time and last 15 seconds. These sessions may or may not be waiting for an wait event, or using the CPU at a particular point in time. This time series is generated based on the wait event samples collected by the **edb_wait_states** extension.

You can also use the **Preferences** dialog to display Performance Diagnostic in a new browser tab. Use **Open in New Browser Tab?** to display the Performance Diagnostics dashboard in a new browser tab.

The range selection in the first graph is 10 minutes. You can use the **Last** drop-down list box to select the duration for which you want to see the graph: select the last 1 hour, last 4 hours, last 12 hours, or last 24 hours. You can also select the date and time through which you want the data to be displayed.



The first graph displays the number of active sessions (and - wait event types) for the selected time interval. You can narrow down the timeline in the first graph to analyze the data for a specific time period.

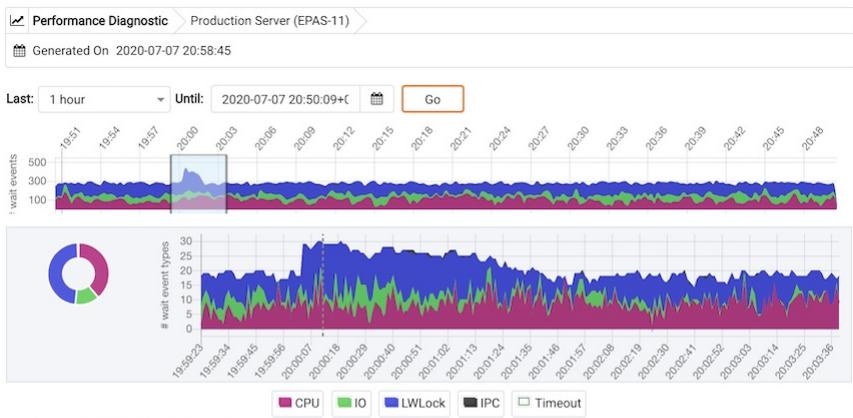
Next section plots the following graphs based on the selected time interval in the first graph:

1. Donut graph - It shows total wait event types according to the time range selection in the first graph. It helps you understand how much time was spent by those session on waiting for an event.
2. Line graph - It plots a time series with each point representing the active sessions for each sample time.

To differentiate each wait event types and the CPU usage clearly, the graph for each wait event type is displayed in a different color.

Select a particular time on the **Line graph** for which you wish to analyze the wait events; the third section displays the wait event details in the Performance Diagnostics dashboard on the basis of your selected particular time in the second graph. The third section displays wait event details on three tabs:

- The **SQL** tab displays the list of SQL queries having wait events for the selected sample time.
- The **Users** tab displays the details of the wait events grouped by users for selected sample time.
- The **Waits** tab displays the number of wait events belonging to each wait event type for the selected sample time.



Wait Event Details		Type to filter	
SQL	Users	Waits	
Load By Waits	SQL	Number of sessions	
	<code>select l_returnflag, l_linenumber, sum(l_quantity) a</code>	4	
	<code>select c_custkey, c_name, sum(l_extendedprice * (\$1</code>	3	
	<code>select s_name, count(*) as numwait from supplier, li</code>	3	
	<code>select s_suppkey, s_name, s_address, s_phone, total_</code>	2	
	<code>select o_orderpriority, count(*) as order_count from</code>	2	

You can click on the graph legends to show or hide a particular wait event type in all the graphs. This will make the analysis of a specific wait event type easier.



You can filter the data displayed in the rows under all the three tabs. You can also sort the data alphabetically by clicking on the column headers.

SQL tab

Wait Event Details			
SQL	Users	Waits	
Load By Waits ▾	SQL		year
	<code>select nation, o_year, sum(amount) as sum_profit from</code>	2	
	<code>select o_year, sum(case when nation = \$1 then volume</code>	1	
	<code>select l_shipmode, sum(case when o_orderpriority = \$1</code>	1	
	<code>select supp_nation, cust_nation, l_year, sum(volume)</code>	1	

Users tab

Wait Event Details			
SQL	Users	Waits	
Load By Waits ▾	Users	Number of Events	Type to filter
	enterprisedb	6	Execution Count
	test2	6	
	test1	5	
	test3	5	
	test4	3	

Waits tab

Wait Event Details			
SQL	Users	Waits	
Load By Wait ▾	Wait Event Type	Wait Event	Number of Events
	LWLock	buffer_mapping	21
	IO	DataFileRead	3
	LWLock	buffer_io	1

Click on the Eye icon in any row of the SQL tab to display a new tab with details of the query of that particular row. This page displays Query ID and its corresponding session IDs in a dropdown list at that particular selected sample time in the Query information section. You can select the session ID for the selected query for which you want to analyze the data. You will see the details corresponding to the selected session ID and query ID. The Query information table also displays the SQL query. If the SQL query is being displayed partially, click the down arrow at the bottom of the section to view the complete SQL query.

The **Wait event types** section displays the total number of wait event types for the selected session ID and query ID. It shows two type of graphs:

1. Donut graph - It shows the proportions of categorical data, with the size of each piece representing the proportion of each wait event type.
2. Timeline bar graph - It can be used to visualize trends in counts of wait event types over time.

To differentiate clearly, each wait event type is represented by a different color in the bar graph.



The **Wait events** section has a table displaying all the wait events occurred during the query execution. It displays data in decreasing order by number of the wait events. Second table displays the wait event with sample time occurred over the period of whole query execution. It allows to analyze the wait events during the query execution over the period of time. It shows the actual samples collected by the EDB Wait States extension for that particular query ID and session ID.

7.4.21 Reports

You can generate the System Configuration report and Core Usage report for all locally and remotely managed servers. To generate this report, select **Reports** from the **Management** Menu.

Reports has following options:

- System Configuration Report (JSON)
- System Configuration Report (HTML)
- Core Usage Report (JSON)
- Core Usage Report (HTML)

Please note that only superusers or the users with the pem_admin role permission can download the System

Configuration or Core Usage reports.

Also note that information in these reports will reflect the latest probe run time.

System Configuration Report

The System Configuration Report provides detailed information about the PEM Agents group, PEM Server directory group and custom groups listed under browser tree. These groups can contain Postgres Enterprise Manager, PEM Agents and Database servers. You can download this report in HTML as well as in JSON format.

The **Postgres Enterprise Manager Summary** provides details about:

- The Postgres Enterprise Manager backend database server version
- Application Version
- User name accessing the application
- Python version
- Flask version
- Platform specific information

The **Summary**, details about the number of agents and servers are provided.

The screenshot shows the 'System Configuration Report' interface. At the top, it displays the generation date: 'Generated On: 2020-04-28 14:30:49'. A dropdown menu 'Go to:' is set to 'PEM Agents'. Below this, there are two main sections: 'Postgres Enterprise Manager Summary' and 'Summary'.

Postgres Enterprise Manager Summary:

Parameter	Value												
Name	Postgres Enterprise Manager												
Backend version	PostgreSQL 12.1 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-39), 64-bit												
App version	7.14.0-dev (schema: 20200303)												
User	postgres												
Python version	3.7.5												
Flask version	1.0.4												
Platform	<table border="1"> <tr> <td>System</td> <td>: Darwin</td> </tr> <tr> <td>Node</td> <td>: Laptop358.bn.in</td> </tr> <tr> <td>Release</td> <td>: 18.7.0</td> </tr> <tr> <td>Version</td> <td>: Darwin Kernel Version 18.7.0: Thu Jan 23 06:52:12 PST 2020; root:xnu-4903.278.25~1/RELEASE_X86_64</td> </tr> <tr> <td>Machine</td> <td>: x86_64</td> </tr> <tr> <td>Processor</td> <td>: i386</td> </tr> </table>	System	: Darwin	Node	: Laptop358.bn.in	Release	: 18.7.0	Version	: Darwin Kernel Version 18.7.0: Thu Jan 23 06:52:12 PST 2020; root:xnu-4903.278.25~1/RELEASE_X86_64	Machine	: x86_64	Processor	: i386
System	: Darwin												
Node	: Laptop358.bn.in												
Release	: 18.7.0												
Version	: Darwin Kernel Version 18.7.0: Thu Jan 23 06:52:12 PST 2020; root:xnu-4903.278.25~1/RELEASE_X86_64												
Machine	: x86_64												
Processor	: i386												

Summary:

Parameter	Value												
Agents	<table border="1"> <tr> <td>Windows</td> <td>: 0</td> </tr> <tr> <td>Linux</td> <td>: 2</td> </tr> </table>	Windows	: 0	Linux	: 2								
Windows	: 0												
Linux	: 2												
Servers	<table border="1"> <tr> <td>PG</td> <td>: 2</td> </tr> <tr> <td>EPAS</td> <td>: 2</td> </tr> <tr> <td>Unknown</td> <td>: 0</td> </tr> <tr> <td>Locally Managed</td> <td>: 3</td> </tr> <tr> <td>Remotely Managed</td> <td>: 1</td> </tr> <tr> <td>Unmanaged</td> <td>: 0</td> </tr> </table>	PG	: 2	EPAS	: 2	Unknown	: 0	Locally Managed	: 3	Remotely Managed	: 1	Unmanaged	: 0
PG	: 2												
EPAS	: 2												
Unknown	: 0												
Locally Managed	: 3												
Remotely Managed	: 1												
Unmanaged	: 0												

At the bottom, there are two navigation links: 'Group: PEM Agents' and 'Group: PEM Server Directory'.

The **Group: PEM Agents**, details about PEM Agent, CPU, Disk Utilization as well as Memory details are provided.

System Configuration Report

Generated On: 2020-04-28 14:30:49

Go to: PEM Agents

> Postgres Enterprise Manager Summary

> Summary

> Group: PEM Agents

Agent: Postgres Enterprise Manager Host

> Agent Details

Parameter	Value
Platform	Linux-x64
OS	CentOS Linux release 7.5.1804 (Core)
Version	7.14.0-dev
Active	True
Hostname	localhost.localdomain
Domain Name	(none)
Bound Local Servers	<ul style="list-style-type: none"> » Postgres Enterprise Manager Server » EDB Postgres Advanced Server 11 » EPAS_12
Bound Remote Servers	(none)

> CPU

Total CPU Cores: 2

Average CPU Utilization (%): 25.31

Core ID	Load Percentage
CPU0	25.267327
CPU1	25.353135

> Disk Utilization

Total Disk Size (MB): 32098

Disk Space Used (MB): 13791

Disk Space Available (MB): 16669

Disk Utilization (%): 42.97

Mount Point	File System	Size (MB)	Space Used (MB)	Space Available (MB)
/	/dev/sda3	31622	13657	16352
/boot	/dev/sda1	476	134	317

> Memory Details

Parameter	Value
Free RAM (MB)	1050
Memory Usage Percentage	72.17
Total Swap Memory (MB)	7999
Free Swap Memory (MB)	6589
Swap Usage Percentage	17.63

Agent: localhost.localdomain

> Agent Details

> CPU

> Disk Utilization

> Memory Details

> Group: PEM Server Directory

The **Group: PEM Server Directory**, provides details about:

- The database server version
- Host
- Port
- Database name
- Database size
- Tablespace size

System Configuration Report

Generated On: 2020-04-28 14:30:49

Go to: PEM Agents

Postgres Enterprise Manager Summary

Summary

Group: PEM Agents

Group: PEM Server Directory

Server: Postgres Enterprise Manager Server

Server Details

Parameter	Value
Agent	Postgres Enterprise Manager Host
Host	192.168.1.19
Port	5432
Database	postgres
Version	PostgreSQL 12.1 on x86_64-pc-linux-gnu, compiled by gcc (GCC) 4.8.5 20150623 (Red Hat 4.8.5-39), 64-bit
Service Id	postgresql-12
Remote Monitored?	False
Active	True

Database Details

Name	Size (MB)	Tablespace Name
postgres	8	pg_default
edbstore_temp	8	pg_default
hr	8	pg_default
pem	2407	pg_default
testdb	27	pg_default
db01	8	pg_default

Tablespace Details

Name	Size (MB)
pg_global	0
pg_default	2482

Server: EPAS_12

Server Details

Database Details

Tablespace Details

Server: EDB Postgres Advanced Server 11

Server Details

Database Details

Tablespace Details

Server: PostgreSQL12_Centos7_1

Server Details

Database Details

Tablespace Details

Please note that here Group Server Name depends on the group name to which the server is added.

Core Usage Report

The Core Usage report provides detailed information about number of cores specific to:

- The server type
- Database version
- Platform and group name

Also, gives detailed information about locally managed servers with:

- Type
- Host
- Port
- Platform
- Cores
- RAM

The screenshot shows the 'Core Usage Report' section of the EDB Postgres Enterprise Manager. It includes a header with the date (2020-02-18), time (16:08:47 IST), and version (7.13.0-dev). Below is a 'Core Summary' table:

Core Summary		
Total Number of Cores: 13		
Server Type	Number of Servers	Number of Cores
EDB Postgres Advanced Server	1	4
PostgreSQL	2	5
BART	1	4

Below is a 'Database Version' table:

Database Version	Number of Servers	Number of Cores
PostgreSQL 10	1	4
PostgreSQL 11	1	1
Advanced Server 11	1	4

Below is a 'Platform' table:

Platform	Number of Servers	Number of Cores
Windows-x64	1	1
Linux-x64	3	12

Below is a 'Group Name' table:

Group Name	Number of Servers	Number of Cores
PEM Server Directory	3	9

Below is a 'Server Core Summary' section:

Locally Managed Servers: 3

Name	Type	Host:Port	Platform	Cores	Total RAM (MB)
Windows PostgreSQL 11	PostgreSQL	127.0.0.1:5432	Windows-x64	1	2047
PostgreSQL-10-Local	PostgreSQL	localhost:5432	Linux-x64	4	5786
Postgres Enterprise Manager Server	EDB Postgres Advanced Server	127.0.0.1:5444	Linux-x64	4	5786
				9	13619

Remotely Managed Servers: 1

Name	Type	Host:Port
PostgreSQL-11-Remote	PostgreSQL	172.19.12.3:5432

Unmanaged Servers: 1

Name	Host:Port
Performance Diagnostics Server	172.16.254.22: 5444

7.5 Management Basics

PEM provides a graphical interface that you can use to simplify management of your Postgres servers and the objects that reside on them.

The Grant Wizard simplifies the task of privilege management; to open the Grant Wizard, highlight the name of a server, database, or schema in the PEM client tree control, and select **Grant Wizard...** from the **Tools** menu.

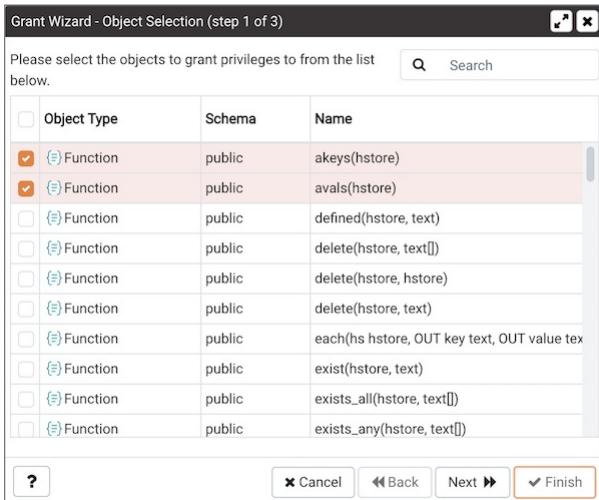
Contents:

7.5.1 Grant Wizard

The **Grant Wizard** tool is a graphical interface that allows you to manage the privileges of one or more database objects in a point-and-click environment. A search box, dropdown lists, and checkboxes facilitate quick selections of database objects, roles and privileges.

The wizard organizes privilege management through a sequence of windows: **Object Selection (step 1 of 3)**, **Privileges Selection (step 2 of 3)** and **Final (Review Selection) (step 3 of 3)**. The **Final (Review Selection)** window displays the SQL code generated by wizard selections.

To launch the **Grant Wizard** tool, select a database object in the **Browser** tree control, then navigate through **Tools** on the menu bar to click on the **Grant Wizard** option.



Use the fields in the **Object Selection (step 1 of 3)** window to select the object or objects on which you are modifying privileges. Use the **Search by object type or name** field to locate a database object, or use the scrollbar to scroll through the list of all accessible objects.

- Each row in the table lists object identifiers; check the checkbox in the left column to include an object as a target of the Grant Wizard. The table displays:
 - The object type in the **Object Type** field
 - The schema in which the object resides in the **Schema** field
 - The object name in the **Name** field.

Click the **Next** button to continue, or the **Cancel** button to close the wizard without modifying privileges.



Use the fields in the **Privileges Selection (step 2 of 3)** window to grant privileges. If you grant a privilege WITH GRANT OPTION, the Grantee will have the right to grant privileges on the object to others. If WITH GRANT OPTION is subsequently revoked, any role who received access to that object from that Grantee (directly or through a chain of grants) will lose their privileges on the object.

- Click the **Add** icon (+) to assign a set of privileges.
- Select the name of the role from the drop-down listbox in the **Grantee** field.
- Click inside the **Privileges** field. Check the boxes to the left of one or more privileges to grant the selected privileges to the specified user. If privileges have previously been granted on a database object, unchecking a privilege for a group or user will result in revoking that privilege.
- The current user, who is the default grantor for granting the privilege, is displayed in the **Grantor** field.
- Click the **Add** icon (+) to assign a set of privileges to another role; to discard a privilege, click the trash icon to the

left of the row and confirm deletion in the **Delete Row** dialog.

For more information about granting privileges on database objects, see the [PostgreSQL core documentation](#).

Click the **Next** button to continue, the **Back** button to select or deselect additional database objects, or the **Cancel** button to close the wizard without modifying privileges.

Your entries in the **Grant Wizard** tool generate a SQL command; you can review the command in the **Final (Review Selection) (step 3 of 3)** window (see an example below).

Example

The following is an example of the sql command generated by user selections in the **Grant Wizard** tool:



The screenshot shows a software window titled "Grant Wizard - Final (Review Selection) (step 3 of 3)". The main area contains a code editor with the following SQL command:

```

1 GRANT EXECUTE ON FUNCTION public.akeys(hstore) TO pem_admin;
2
3 GRANT EXECUTE ON FUNCTION public.avals(hstore) TO pem_admin;
4
5

```

Below the code editor, there is a message: "The SQL below will be executed on the database server to grant the selected privileges. Please click on Finish to complete the process." At the bottom of the window are buttons for "?", "Cancel", "Back", "Next", and "Finish". The "Finish" button is highlighted with an orange border.

The commands displayed assign a role named **Bob** **INSERT** and **UPDATE** privileges **WITH GRANT OPTION** on the **sales_meetings** and the **sales_territories** tables.

- Click the **Back** button to select or deselect additional database objects, roles and privileges.
- Click the **Cancel** button to exit without saving work.
- Click the **Finish** button to save selections and exit the wizard.

7.5.2 Add named restore point Dialog

Use the **Add named restore point** dialog to take a named snapshot of the state of the server for use in a recovery file. To create a named restore point, the server's postgresql.conf file must specify a **wal_level** value of **replica**, or **logical**. You must be a database superuser to create a restore point.



When the **Restore point name** window launches, use the field **Enter the name of the restore point to add** to provide a descriptive name for the restore point.

For more information about using a restore point as a recovery target, please see the [PostgreSQL documentation](#).

- Click the **OK** button to save the restore point.
- Click the **Cancel** button to exit without saving work.

7.5.3 Import/Export data Dialog

Use the **Import/Export data** dialog to copy data from a table to a file, or copy data from a file into a table.

The **Import/Export data** dialog organizes the import/export of data through the **Options** and **Columns** tabs.



Use the fields in the **Options** tab to specify import and export preferences:

- Move the **Import/Export** switch to the **Import** position to specify that the server should import data to a table from a file. The default is **Export**.
- Use the fields in the **File Info** field box to specify information about the source or target file:
 - Enter the name of the source or target file in the **Filename** field. Optionally, select the **Browser** icon (ellipsis) to the right to navigate into a directory and select a file.
 - Use the drop-down listbox in the **Format** field to specify the file type. Select:
 - **binary** for a .bin file.
 - **csv** for a .csv file.
 - **text** for a .txt file.
 - Use the drop-down listbox in the **Encoding** field to specify the type of character encoding.



- Use the fields in the **Miscellaneous** field box to specify additional information:

- Move the **OID** switch to the **Yes** position to include the **OID** column. The **OID** is a system-assigned value that may not be modified. The default is **No**.
- Move the **Header** switch to the **Yes** position to include the table header with the data rows. If you include the table header, the first row of the file will contain the column names.
- If you are exporting data, specify the delimiter that will separate the columns within the target file in the **Delimiter** field. The separating character can be a colon, semicolon, a vertical bar, or a tab.
- Specify a quoting character used in the **Quote** field. Quoting can be applied to string columns only (i.e. numeric columns will not be quoted) or all columns regardless of data type. The character used for quoting can be a single quote or a double quote.
- Specify a character that should appear before a data character that matches the **QUOTE** value in the **Escape** field.

Click the **Columns** tab to continue.



Use the fields in the **Columns** tab to select the columns that will be imported or exported:

- Click inside the **Columns to export/import** field to deselect one or more columns from the drop-down listbox. To delete a selection, click the **x** to the left of the column name. Click an empty spot inside the field to access the drop-down list.
- Use the **NULL Strings** field to specify a string that will represent a null value within the source or target file.
- If enabled, click inside the **Not null columns** field to select one or more columns that will not be checked for a NULL value. To delete a column, click the **x** to the left of the column name.

After completing the **Import/Export data** dialog, click the **OK** button to perform the import or export. PEM will inform you when the background process completes:



Use the Stop Process button to stop the Import/Export process.

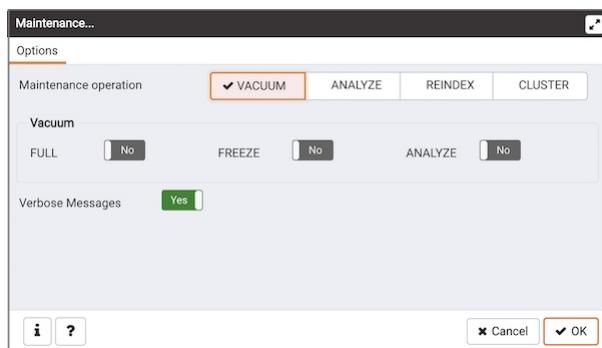
Use the **Click here for details** link on the notification to open the **Process Watcher** and review detailed information about the execution of the command that performed the import or export:



Note

You can click on the icon in the process watcher window to open the file location in the Storage Manager. You can use the [Storage Manager](#) to download the backup file on the client machine .

7.5.4 Maintain a database object



This tool allows to maintain the database in total, or only a selected table, or a selected index.

Maintenance comes in three flavors.

VACUUM

VACUUM will scan the database or table for rows, that are not in use any more. If a row is updated or deleted, the previous content isn't replaced, but rather marked invalid. The new data is inserted freshly into the database. You need to perform a garbage collection regularly, to insure that your database doesn't contain too much unused data, wasting disk space and ultimately degrading performance.

Please press the Help button to see the PostgreSQL help about the VACUUM command to learn more about the options.

The output of the database server is displayed in the messages page as they arrive. If Verbose is selected, the server will send very detailed info about what it did.

While this tool is very handy for ad-hoc maintenance purposes, you are encouraged to install an automatic job, that performs a VACUUM job regularly to keep your database in a neat state.

ANALYZE

ANALYZE investigates statistical values about the selected database or table. This enables the query optimizer to select the fastest query plan, to give optimal performance. Every time your data is changing radically, you should perform this task. It can be included in a VACUUM run, using the appropriate option.

REINDEX

REINDEX rebuilds the indexes in case these have degenerated caused by unusual data patterns inserted. This can happen for example if you insert many rows with increasing index values, and delete low index values.

The RECREATE option doesn't call the REINDEX SQL command internally, instead it drops the existing table and recreates it according to the current index definition. This doesn't lock the table exclusively, as REINDEX does, but will lock write access only.

7.5.4.1 Maintenance Dialog

Use the **Maintenance** dialog to VACUUM, ANALYZE, REINDEX or CLUSTER a database or selected database objects.



While this utility is useful for ad-hoc maintenance purposes, you are encouraged to perform automatic VACUUM jobs on a regular schedule.

Select a button next to **Maintenance operation** to specify the type of maintenance:

- Click **VACUUM** to scan the selected database or table to reclaim storage used by dead tuples.
 - Move the **FULL** switch to the **Yes** position to compact tables by writing a completely new version of the table file without dead space. The default is **No**.
 - Move the **FREEZE** switch to the **Yes** position to freeze data in a table when it will have no further updates. The default is **No**.
 - Move the **ANALYZE** switch to the **Yes** position to issue ANALYZE commands whenever the content of a table has changed sufficiently. The default is **No**.
- Click **ANALYZE** to update the stored statistics used by the query planner. This enables the query optimizer to select the fastest query plan for optimal performance.
- Click **REINDEX** to rebuild any index in case it has degenerated due to the insertion of unusual data patterns. This happens, for example, if you insert rows with increasing index values, and delete low index values.

- Click **CLUSTER** to instruct PostgreSQL to cluster the selected table.

To exclude status messages from the process output, move the **Verbose Messages** switch to the **No** position; by default, status messages are included.

When you've completed the dialog, click **OK** to start the background process; to exit the dialog without performing maintenance operations, click **Cancel**.

pgAdmin will inform you when the background process completes:



Use the Stop Process button to stop the Maintenance process.

Use the **Click here for details** link on the notification to open the **Process Watcher** and review detailed information about the execution of the command that performed the import or export:



7.5.5 Storage Manager

Storage Manager is a feature that helps you manage your system's storage device. You can use *Storage Manager* to:

- Download, upload, or manage operating system files.
- Download *backup* or *export* files (custom, tar and plain text format) on a client machine.
- Download *exportdump* files of tables.

You can access *Storage Manager* from the *Tools* Menu.



Use icons on the top of the *Storage Manager* window to manage storage:

Use the **Home** icon to return to the home directory.

Use the **Up Arrow** icon to return to the previous directory.

Use the **Refresh** icon to display the most-recent files available.

Select the **Download** icon to download the selected file.

Select the **Delete** icon to delete the selected file or folder.

Select the **Edit** icon to rename a file or folder.

Use the **Upload** icon to upload a file.

Use the **New Folder** icon to add a new folder.

Use the **Grid View** icon to display all the files and folders in a grid view.

Use the **Table View** icon to display all the files and folders in a list view.

Click on the check box next to *Show hidden files and folders* at the bottom of the window to view hidden files and folders.

Use the *Format* drop down list to select the format of the files to be displayed; choose from *sql*, *csv*, or *All Files*.

You can also download backup files through *Storage Manager* at the successful completion of the backups taken through [Backup Dialog](#), [Backup Global Dialog](#), or [Backup Server Dialog](#).

At the successful completion of a backup, click on the icon to open the current backup file in *Storage Manager* on the *process watcher* window.



7.5.6 Backup Dialog

PEM uses the `pg_dump` utility to provide an easy way to create a backup in a plain-text or archived format. You can then use a client application (like `psql` or the [Query Tool](#)) to restore a plain-text backup file, or use the Postgres `pg_restore` utility to restore an archived backup. The `pg_dump` utility must have read access to all database objects that you want to back up.

You can backup a single table, a schema, or a complete database. Select the name of the backup source in the [Browser](#) tree control, right click to open the context menu, and select [Backup...](#) to open the [Backup](#) dialog. The name of the object selected will appear in the dialog title bar.



Use the fields in the [General](#) tab to specify parameters for the backup:

- Enter the name of the backup file in the [Filename](#) field. Optionally, select the [Browser](#) icon (...) to the right to navigate into a directory and select a file that will contain the archive.
- Use the drop-down listbox in the [Format](#) field to select the format that is best suited for your application. Each format has advantages and disadvantages:
 - Select [Custom](#) to create a custom archive file that you can use with `pg_restore` to create a copy of a database. Custom archive file formats must be restored with `pg_restore`. This format offers the opportunity to select which database objects to restore from the backup file. [Custom](#) archive format is

- recommended for medium to large databases as it is compressed by default.
- Select **Tar** to generate a tar archive file that you can restore with **pg_restore**. The tar format does not support compression.
 - Select **Plain** to create a plain-text script file. A plain-text script file contains SQL statements and commands that you can execute at the **psql** command line to recreate the database objects and load the table data. A plain-text backup file can be edited in a text editor, if desired, before using the **psql** program to restore database objects. **Plain** format is normally recommended for smaller databases; script dumps are not recommended for blobs. The SQL commands within the script will reconstruct the database to the last saved state of the database. A plain-text script can be used to reconstruct the database on another machine, or (with modifications) on other architectures.
 - Select **Directory** to generate a directory-format archive suitable for use with **pg_restore**. This file format creates a directory with one file for each table and blob being dumped, plus a **Table of Contents** file describing the dumped objects in a machine-readable format that **pg_restore** can read. This format is compressed by default.
- Use the **Compression Ratio** field to select a compression level for the backup. Specify a value of zero to mean use no compression; specify a maximum compression value of 9. Please note that tar archives do not support compression.
 - Use the **Encoding** drop-down listbox to select the character encoding method that should be used for the archive.
 - Use the **Number of Jobs** field (when applicable) to specify the number of tables that will be dumped simultaneously in a parallel backup.
 - Use the dropdown listbox next to **Roolename** to specify the role that owns the backup.

Click the **Dump options** tab to continue. Use the box fields in the **Dump options** tab to provide options for **pg_dump**.



- Move switches in the Sections field box to select a portion of the object that will be backed up.
 - Move the switch next to **Pre-data** to the **Yes** position to include all data definition items not included in the data or post-data item lists.
 - Move the switch next to **Data** to the **Yes** position to backup actual table data, large-object contents, and sequence values.
 - Move the switch next to **Post-data** to the **Yes** position to include definitions of indexes, triggers, rules, and constraints other than validated check constraints.



- Move switches in the Type of objects field box to specify details about the type of objects that will be backed up.

- Move the switch next to **Only data** to the **Yes** position to limit the back up to data.
- Move the switch next to **Only schema** to limit the back up to schema-level database objects.
- Move the switch next to **Blobs** to the **No** position to exclude large objects in the backup.



- Move switches in the Do not save field box to select the objects that will not be included in the backup.

- Move the switch next to **Owner** to the **Yes** position to exclude commands that set object ownership.
- Move the switch next to **Privilege** to the **Yes** position to exclude commands that create access privileges.
- Move the switch next to **Tablespace** to the **Yes** position to exclude tablespaces.
- Move the switch next to **Unlogged table data** to the **Yes** position to exclude the contents of unlogged tables.
- Move the switch next to **Comments** to the **Yes** position to exclude commands that set the comments. Note: This option is visible only for database server greater than or equal to 11.



- Move switches in the Queries field box to specify the type of statements that should be included in the backup.

- Move the switch next to **Use Column Inserts** to the **Yes** position to dump the data in the form of INSERT statements and include explicit column names. Please note: this may make restoration from backup slow.
- Move the switch next to **Use Insert commands** to the **Yes** position to dump the data in the form of INSERT statements rather than using a COPY command. Please note: this may make restoration from backup slow.
- Move the switch next to **Include CREATE DATABASE statement** to the **Yes** position to include a command in the backup that creates a new database when restoring the backup.
- Move the switch next to **Include DROP DATABASE statement** to the **Yes** position to include a command in the backup that will drop any existing database object with the same name before recreating the object during a backup.
- Move the switch next to **Load Via Partition Root** to the **Yes** position, so when dumping a COPY or INSERT statement for a partitioned table, target the root of the partitioning hierarchy which contains it

rather than the partition itself. Note: This option is visible only for database server greater than or equal to 11.



- Move switches in the Disable field box to specify the type of statements that should be excluded from the backup.
 - Move the switch next to **Trigger** (active when creating a data-only backup) to the **Yes** position to include commands that will disable triggers on the target table while the data is being loaded.
 - Move the switch next to **\$ quoting** to the **Yes** position to enable dollar quoting within function bodies; if disabled, the function body will be quoted using SQL standard string syntax.



- Move switches in the Miscellaneous field box to specify miscellaneous backup options.
 - Move the switch next to **With OIDs** to the **Yes** position to include object identifiers as part of the table data for each table.
 - Move the switch next to **Verbose messages** to the **No** position to instruct `pg_dump` to exclude verbose messages.
 - Move the switch next to **Force double quotes on identifiers** to the **Yes** position to force the quoting of all identifiers.
 - Move the switch next to **Use SET SESSION AUTHORIZATION** to the **Yes** position to include a statement that will use a SET SESSION AUTHORIZATION command to determine object ownership (instead of an ALTER OWNER command).

When you've specified the details that will be incorporated into the `pg_dump` command:

- Click the **Backup** button to build and execute a command that builds a backup based on your selections on the **Backup** dialog.
- Click the **Cancel** button to exit without saving work.



Use the Stop Process button to stop the Backup process.

If the backup is successful, a popup window will confirm success. Click *More details* on the popup window to launch the *Process Watcher*. The *Process Watcher* logs all the activity associated with the backup and provides additional information for troubleshooting.



If the backup is unsuccessful, you can review the error messages returned by the backup command on the **Process Watcher**.

Note

You can click on the  icon in the process watcher window to open the file location in the Storage Manager. You can use the [Storage Manager](#) to download the backup file on the client machine .

7.5.7 Backup Globals Dialog

Use the **Backup Globals** dialog to create a plain-text script that recreates all of the database objects within a cluster, and the global objects that are shared by those databases. Global objects include tablespaces, roles, and object properties. You can use the PEM [Query Tool](#) to play back a plain-text script, and recreate the objects in the backup.



Use the fields in the **General** tab to specify the following:

- Enter the name of the backup file in the **Filename** field. Optionally, select the **Browser** icon (ellipsis) to the right to navigate into a directory and select a file that will contain the archive.

- Use the drop-down listbox next to **Role name** to specify a role with connection privileges on the selected server. The role will be used for authentication during the backup.

Move switches in the Miscellaneous field box to specify the type of statements that should be included in the backup.

- Move the **Verbose messages** switch to the **No** position to exclude status messages from the backup. The default is **Yes**.
- Move the **Force double quote on identifiers** switch to the **Yes** position to name identifiers without changing case. The default is **No**.

Click the **Backup** button to build and execute a command based on your selections; click the **Cancel** button to exit without saving work.



Use the Stop Process button to stop the Backup process.

If the backup is successful, a popup window will confirm success. Click [Click here for details](#) on the popup window to launch the **Process Watcher**. The **Process Watcher** logs all the activity associated with the backup and provides additional information for troubleshooting.



If the backup is unsuccessful, review the error message returned by the **Process Watcher** to resolve any issue.

Note

You can click on the  icon in the process watcher window to open the file location in the Storage Manager. You can use the [Storage Manager](#) to download the backup file on the client machine .

7.5.8 Backup Server Dialog

Use the **Backup Server** dialog to create a plain-text script that will recreate the selected server. You can use the

PEM [Query Tool](#) to play back a plain-text script, and recreate the server.



Use the fields in the **General** tab to specify the following:

- Enter the name of the backup file in the **Filename** field. Optionally, select the **Browser** icon (ellipsis) to the right to navigate into a directory and select a file that will contain the archive.
- Use the **Encoding** drop-down listbox to select the character encoding method that should be used for the archive. Note: This option is visible only for database server greater than or equal to 11.
- Use the drop-down listbox next to **Role name** to specify a role with connection privileges on the selected server. The role will be used for authentication during the backup.



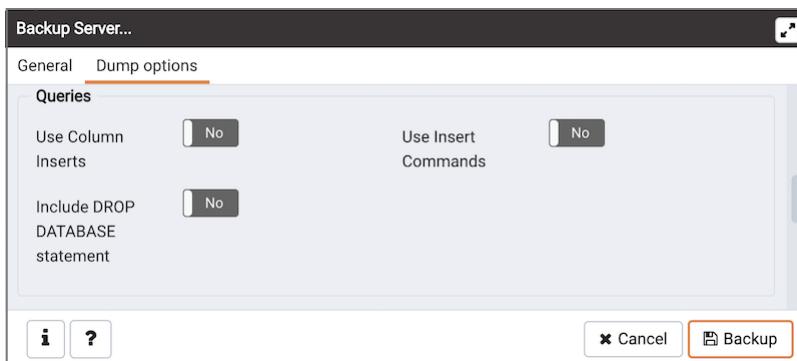
- Move switches in the Type of objects field box to specify details about the type of objects that will be backed up.
 - Move the switch next to **Only data** to the **Yes** position to limit the back up to data.
 - Move the switch next to **Only schema** to limit the back up to schema-level database objects.



- Move switches in the Do not save field box to select the objects that will not be included in the backup.
 - Move the switch next to **Owner** to the **Yes** position to exclude commands that set object ownership.
 - Move the switch next to **Privilege** to the **Yes** position to exclude commands that create access privileges.
 - Move the switch next to **Tablespace** to the **Yes** position to exclude tablespaces.
 - Move the switch next to **Unlogged table data** to the **Yes** position to exclude the contents of

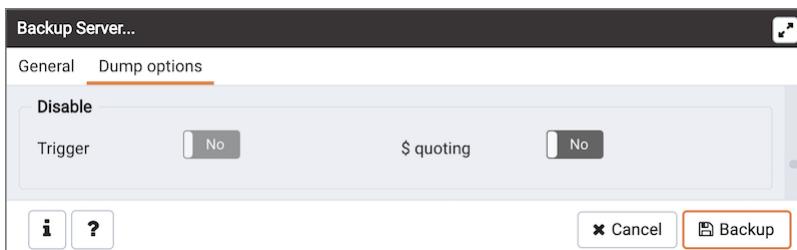
unlogged tables.

- Move the switch next to **Comments** to the **Yes** position to exclude commands that set the comments. Note: This option is visible only for database server greater than or equal to 11.



- Move switches in the Queries field box to specify the type of statements that should be included in the backup.

- Move the switch next to **Use Column Inserts** to the **Yes** position to dump the data in the form of INSERT statements and include explicit column names. Please note: this may make restoration from backup slow.
- Move the switch next to **Use Insert commands** to the **Yes** position to dump the data in the form of INSERT statements rather than using a COPY command. Please note: this may make restoration from backup slow.
- Move the switch next to **Include DROP DATABASE statement** to the **Yes** position to include a command in the backup that will drop any existing database object with the same name before recreating the object during a backup.



- Move switches in the Disable field box to specify the type of statements that should be excluded from the backup.

- Move the switch next to **Trigger** (active when creating a data-only backup) to the **Yes** position to include commands that will disable triggers on the target table while the data is being loaded.
- Move the switch next to **\$ quoting** to the **Yes** position to enable dollar quoting within function bodies; if disabled, the function body will be quoted using SQL standard string syntax.



- Move switches in the Miscellaneous field box to specify miscellaneous backup options.

- Move the switch next to **With OIDs** to the **Yes** position to include object identifiers as part of the table data for each table.
- Move the switch next to **Verbose messages** to the **No** position to instruct **pg_dump** to exclude verbose messages.
- Move the switch next to **Force double quotes on identifiers** to the **Yes** position to force the quoting of all identifiers.
- Move the switch next to **Use SET SESSION AUTHORIZATION** to the **Yes** position to include a statement that will use a SET SESSION AUTHORIZATION command to determine object ownership (instead of an ALTER OWNER command).

Click the **Backup** button to build and execute a command based on your selections; click the **Cancel** button to exit without saving work.



Use the Stop Process button to stop the Backup process.

If the backup is successful, a popup window will confirm success. Click **Click here for details** on the popup window to launch the **Process Watcher**. The **Process Watcher** logs all the activity associated with the backup and provides additional information for troubleshooting.



If the backup is unsuccessful, review the error message returned by the **Process Watcher** to resolve any issue.

Note

You can click on the  icon in the process watcher window to open the file location in the Storage Manager. You can use the **Storage Manager** to download the backup file on the client machine .

7.5.9 Restore Dialog

The **Restore** dialog provides an easy way to use a Custom, tar, or Directory format backup taken with the PEM **Backup** dialog to recreate a database or database object. The **Backup** dialog invokes options of the pg_dump client utility; the **Restore** dialog invokes options of the pg_restore client utility.

You can use the **Query Tool** to play back the script created during a plain-text backup made with the **Backup** dialog. For more information about backing up or restoring, please refer to the documentation for **pg_dump** or **pg_restore**.



Use the fields on the **General** tab to specify general information about the restore process:

- Use the drop-down listbox in the **Format** field to select the format of your backup file.
 - Select **Custom or tar** to restore from a custom archive file to create a copy of the backed-up object.
 - Select **Directory** to restore from a compressed directory-format archive.
- Enter the complete path to the backup file in the **Filename** field. Optionally, select the **Browser** icon (ellipsis) to the right to navigate into a directory and select the file that contains the archive.
- Use the **Number of Jobs** field to specify if pg_restore should use multiple (concurrent) jobs to process the restore. Each job uses a separate connection to the server.
- Use the drop-down listbox next to **Rolename** to specify the role that will be used to authenticate with the server during the restore process.

Click the **Restore options** tab to continue. Use the fields on the **Restore options** tab to specify options that correspond to **pg_restore** options.



- Use the switches in the Sections box to specify the content that will be restored:
 - Move the switch next to **Pre-data** to the **Yes** position to restore all data definition items not included in the data or post-data item lists.
 - Move the switch next to **Data** to the **Yes** position to restore actual table data, large-object contents, and sequence values.
 - Move the switch next to **Post-data** to the **Yes** position to restore definitions of indexes, triggers, rules,

and constraints (other than validated check constraints).



- Use the switches in the Type of objects box to specify the objects that will be restored:

- Move the switch next to **Only data** to the **Yes** position to limit the restoration to data.
- Move the switch next to **Only schema** to limit the restoration to schema-level database objects.



- Use the switches in the Do not save box to specify which objects will not be restored:

- Move the switch next to **Owner** to the **Yes** position to exclude commands that set object ownership.
- Move the switch next to **Privilege** to the **Yes** position to exclude commands that create access privileges.
- Move the switch next to **Tablespace** to the **Yes** position to exclude tablespaces.
- Move the switch next to **Comments** to the **Yes** position to exclude commands that set the comments. Note: This option is visible only for database server greater than or equal to 11.



- Use the switches in the Queries box to specify the type of statements that should be included in the restore:

- Move the switch next to **Include CREATE DATABASE statement** to the **Yes** position to include a command that creates a new database before performing the restore.
- Move the switch next to **Clean before restore** to the **Yes** position to drop each existing database object (and data) before restoring.
- Move the switch next to **Single transaction** to the **Yes** position to execute the restore as a single transaction (that is, wrap the emitted commands in **BEGIN/COMMIT**). This ensures that either all the commands complete successfully, or no changes are applied. This option implies **--exit-on-error**.



- Use the switches in the Disable box to specify the type of statements that should be excluded from the restore:
 - Move the switch next to **Trigger** (active when creating a data-only restore) to the **Yes** position to include commands that will disable triggers on the target table while the data is being loaded.
 - Move the switch next to **No data for Failed Tables** to the **Yes** position to ignore data that fails a trigger.



- Use the switches in the Miscellaneous/Behavior box to specify miscellaneous restore options:
 - Move the switch next to **Verbose messages** to the **No** position to instruct **pg_restore** to exclude verbose messages.
 - Move the switch next to **Use SET SESSION AUTHORIZATION** to the **Yes** position to include a statement that will use a SET SESSION AUTHORIZATION command to determine object ownership (instead of an ALTER OWNER command).
 - Move the switch next to **Exit on error** to the **Yes** position to instruct **pg_restore** to exit restore if there is an error in sending SQL commands. The default is to continue and to display a count of errors at the end of the restore.

When you've specified the details that will be incorporated into the **pg_restore** command, click the **Restore** button to start the process, or click the **Cancel** button to exit without saving your work. A popup will confirm if the restore is successful.



Use the Stop Process button to stop the Restore process.

Click **Click here for details** on the popup to launch the **Process Watcher**. The **Process Watcher** logs all the activity associated with the restore, and provides additional information for troubleshooting should the restore command encounter problems.



7.5.10 Managing Cluster Level Objects

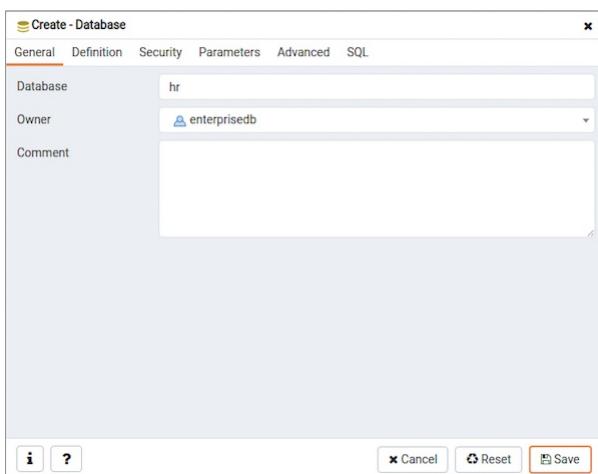
Some object definitions reside at the cluster level; PEM provides dialogs that allow you to create these objects, manage them, and control their relationships to each other. To access a dialog that allows you to create a database object, right-click on the object type in the Browser tree control, and select the **Create** option for that object. For example, to create a new database, right-click on the **Databases** node, and select **Create Database...**

Contents:

7.5.10.1 Database Dialog

Use the **Database** dialog to define or modify a database. To create a database, you must be a database superuser or have the CREATE privilege.

The **Database** dialog organizes the development of a database through the following dialog tabs: **General**, **Definition**, **Security**, and **Parameters**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the database:

- Use the **Database** field to add a descriptive name for the database. The name will be displayed in the **Browser** tree control.
- Select the owner of the database from the drop-down listbox in the **Owner** field.
- Store notes about the database in the **Comment** field.

Click the **Definition** tab to continue.

The screenshot shows the 'Create - Database' dialog box. The 'Definition' tab is active. The form contains the following fields:

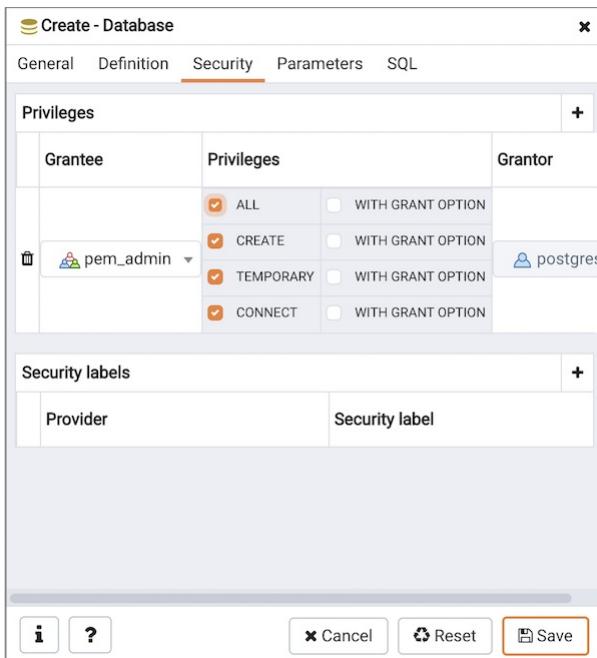
- Encoding:** UTF8
- Template:** Select an item...
- Tablespace:** Select an item...
- Collation:** Select from the list
- Character type:** Select from the list
- Connection limit:** -1

At the bottom are buttons for **Cancel**, **Reset**, and **Save**.

Use the **Definition** tab to set properties for the database:

- Select a character set from the drop-down listbox in the **Encoding** field. The default is **UTF8**.
- Select a template from the drop-down listbox in the **Template** field. If you do not specify a template, the database will use template1.
- Select a tablespace from the drop-down listbox in the **Tablespace** field. The selected tablespace will be the default tablespace used to contain database objects.
- Select the collation order from the drop-down listbox in the **Collation** field.
- Select the character classification from the drop-down listbox in the **Character Type** field. This affects the categorization of characters, e.g. lower, upper and digit. The default, or a blank field, uses the character classification of the template database.
- Specify a connection limit in the **Connection Limit** field to configure the maximum number of connection requests. The default value (**-1**) allows unlimited connections to the database.

Click the **Security** tab to continue.



Use the **Security** tab to assign privileges and define security labels.

Use the **Privileges** panel to assign privileges to a role. Click the **Add** icon (+) to set privileges for database objects:

- Select the name of the role from the drop-down listbox in the **Grantee** field.
- Click inside the **Privileges** field. Check the boxes to the left of one or more privileges to grant the selected privilege to the specified user.
- The current user, who is the default grantor for granting the privilege, is displayed in the **Grantor** field.

Click add to set additional privileges; to discard a privilege, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Use the **Security Labels** panel to define security labels applied to the database. Click the **Add** icon (+) to add each security label selection:

- Specify a security label provider in the **Provider** field. The named provider must be loaded and must consent to the proposed labeling operation.
- Specify a security label in the **Security Label** field. The meaning of a given label is at the discretion of the label provider. PostgreSQL places no restrictions on whether or how a label provider must interpret security labels; it merely provides a mechanism for storing them.

To discard a security label, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **Parameters** tab to continue.



Use the **Parameters** tab to set parameters for the database. Click the **Add** icon (+) to add each parameter:

- Use the drop-down listbox in the **Name** field to select a parameter.
- Use the **Value** field to set a value for the parameter.
- Use the drop-down listbox next to **Role** to select a role to which the parameter setting specified will apply.

Follow these steps to add additional parameter value definitions; to discard a parameter, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **Advanced** tab to continue.



Use the **Advanced** tab to set advanced parameters for the database.

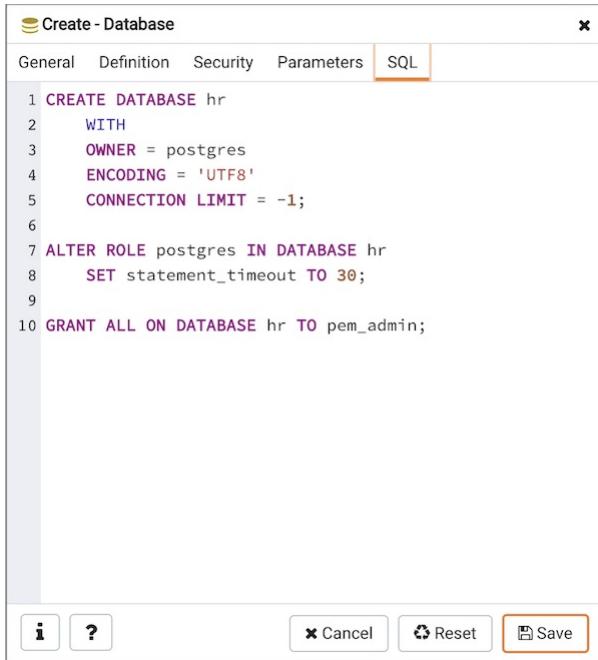
- Use **Schema restriction** field to provide a SQL restriction that will be used against the pg_namespace table to limit the schemas that you see. For example, you might enter: `public` so that only `public` are shown in the pgAdmin browser. Separate entries with a comma or tab as you type.

Click the **SQL** tab to continue.

Your entries in the **Database** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Database** dialog:



The screenshot shows the 'Create - Database' dialog window. The title bar says 'Create - Database'. Below it is a tab bar with 'General', 'Definition', 'Security', 'Parameters', and 'SQL'. The 'SQL' tab is selected and highlighted in orange. The main area contains the following SQL code:

```

1 CREATE DATABASE hr
2   WITH
3     OWNER = postgres
4     ENCODING = 'UTF8'
5     CONNECTION LIMIT = -1;
6
7 ALTER ROLE postgres IN DATABASE hr
8   SET statement_timeout TO 30;
9
10 GRANT ALL ON DATABASE hr TO pem_admin;

```

At the bottom of the dialog are several buttons: 'Info' (greyed out), 'Cancel', 'Reset', and 'Save' (highlighted in orange).

The example creates a database named `hr` that is owned by `postgres`. It allows unlimited connections, and is available to all authenticated users.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.10.2 Move Objects Dialog

Use the **Move Objects** dialog to move database objects from one tablespace to another tablespace.

The **Move Objects** dialog organizes the movement of database objects with the **General** tab; the **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the items that will be moved and the tablespace to which they will be moved:

- Use the **New tablespace** drop-down listbox to select a pre-existing tablespace to which the object will be moved. (To create a tablespace, use the **Tablespace** dialog; access the dialog by right clicking **Tablespaces** in the **Browser** tree control and selecting **Create Tablespace...** from the context-menu.)
- Use the **Object type** drop-down listbox to select from the following:
 - Select **All** to move all tables, indexes, and materialized views from the current tablespace (currently selected in the **Browser** tree control) to the new tablespace.
 - Select **Tables** to move tables from the current tablespace to the new tablespace.
 - Select **Indexes** to move indexes from the current tablespace to the new tablespace.
 - Select **Materialized views** to move materialized views from the current tablespace to the new tablespace.
- Use the **Object owner** drop-down listbox to select the role that owns the objects selected in the **Object type** field. This field is optional.

Click the **SQL** tab to continue.

Your entries in the **Move Objects** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit the **General** tab to modify the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Move Objects** dialog:



The example shown demonstrates moving materialized views owned by Alice from tablespace `tbspace_01` to `tbspace_02`.

- Click the **Help** button (?) to access online help.
- Click the **OK** button to save work.
- Click the **Cancel** button to exit without saving work.

7.5.10.3 Resource Group Dialog

Use the **Resource Group** dialog to create a resource group and set values for its resources. A resource group is a named, global group on which various resource usage limits can be defined. The resource group is accessible from all databases in the cluster. To use the **Resource Group** dialog, you must have superuser privileges. Please note that resource groups are supported when connected to EDB Postgres Advanced Server; for more information about using resource groups, please see the EDB Postgres Advanced Server Guide, available at:

<http://www.enterprisedb.com/>

Fields used to create a resource group are located on the **General** tab. The **SQL** tab displays the SQL code generated by your selections on the **Resource Group** dialog.



Use the fields on the **General** tab to specify resource group attributes:

- Use the **Name** field to add a descriptive name for the resource group. This name will be displayed in the tree control.
- Use the **CPU rate limit (%)** field to set the value of the CPU rate limit resource type assigned to the resource group. The valid range for a CPU rate limit is from 0 to 1.67772e+07. The default value is 0.
- Use the **Dirty rate limit (KB)** field to set the value of the dirty rate limit resource type assigned to the resource group. The valid range for a dirty rate limit is from 0 to 1.67772e+07. The default value is 0.

Click the **SQL** tab to continue.

Your entries in the **Resource Group** dialog generate a SQL command. Use the **SQL** tab for review; revisit the

General tab to make any changes to the SQL command.

Example

The following is an example of the sql command generated by selections made in the **Resource Group** dialog:



The screenshot shows the 'Create - Resource Group' dialog window. The 'SQL' tab is selected. The main area contains the following SQL code:

```

1 CREATE RESOURCE GROUP acctg;
2
3 -- Following query will be executed in a separate transaction
4 ALTER RESOURCE GROUP acctg
5     SET cpu_rate_limit = 0, dirty_rate_limit = 0;

```

At the bottom of the dialog, there are several buttons: Info (i), Help (?), Cancel, Reset, and Save. The 'Save' button is highlighted with a red border.

The example creates a resource group named `acctg` that sets `cpu_rate_limit` to `2`, and `dirty_rate_limit` to `6144`.

- Click the Info button (`i`) to access online SQL syntax reference material.
- Click the Help button (`?`) to access online documentation about Resource Groups.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.10.4 Login/Group Role Dialog

Use the **Login/Group Role** dialog to define a role. A role may be an individual user (with or without login privileges) or a group of users. Note that roles defined at the cluster level are shared by all databases in the cluster.

The **Login/Group Role** dialog organizes the creation and management of roles through the following dialog tabs: **General**, **Definition**, **Privileges**, **Parameters**, and **Security**. The **SQL** tab displays the SQL code generated by dialog selections.

The screenshot shows the 'Create - Login/Group Role' dialog box. The 'General' tab is selected. In the 'Name' field, the value 'alice' is entered. The 'Comments' field is empty. At the bottom of the dialog, there are buttons for 'Cancel', 'Reset', and 'Save'.

Use the fields on the **General** tab to identify the role.

- Use the **Name** field to provide the name of the role. The name will be displayed in the tree control.
- Provide a note about the role in the **Comments** field.

Click the **Definition** tab to continue.

The screenshot shows the 'Create - Login/Group Role' dialog box. The 'Definition' tab is selected. In the 'Password' field, three dots (...) are shown. The 'Account expires' field contains 'YYYY-MM-DD HH:mm:ss Z' and has a calendar icon. The 'Connection limit' field contains '-1'. At the bottom of the dialog, there are buttons for 'Cancel', 'Reset', and 'Save'.

Use the **Definition** tab to set a password and configure connection rules:

- Provide a password that will be associated with the role in the **Password** field.
- Provide an expiration date for the password in the **Account Expires** field (the role does not expire). The expiration date is not enforced when a user logs in with a non-password-based authentication method.
- If the role is a login role, specify how many concurrent connections the role can make in the **Connection Limit** field. The default value (**-1**) allows unlimited connections.

Click the **Privileges** tab to continue.



Use the **Privileges** tab to grant privileges to the role.

- Move the **Can login?** switch to the **Yes** position if the role has login privileges. The default value is **No**.
- Move the **Superuser** switch to the **Yes** position if the role is a superuser within the database. The default value is **No**.
- Move the **Create roles?** switch to the **Yes** position to specify whether a role is permitted to create roles. A role with this privilege can alter and drop roles. The default value is **No**.
- Move the **Create databases** switch to the **Yes** position to control whether a role can create databases. The default value is **No**.
- The **Update catalog?** switch is disabled until the role is given superuser privileges. Move the **Update catalogs?** switch to the **No** position to control whether a role can update catalogs. The default value is **Yes** when the **Superuser** switch is in the **Yes** position.
- Move the **Inherit rights from the parent roles?** switch to the **No** position if a role does not inherit privileges. The default value is **Yes**.
- Move the **Can initiate streaming replication and backups?** switch to the **Yes** position to control whether a role can initiate streaming replication or put the system in and out of backup mode. The default value is **No**.



- Specify members of the role in the **Role Membership** field. Click inside the **Roles** field to select role names from a drop down list. Confirm each selection by checking the checkbox to the right of the role name; delete a selection by clicking the **x** to the left of the role name. Membership conveys the privileges granted to the specified role to each of its members.

Click the **Parameters** tab to continue.



Use the fields on the **Parameters** tab to set session defaults for a selected configuration parameter when the role is connected to a specified database. This tab invokes the ALTER ROLE... SET configuration_parameter syntax. Click the **Add** icon (+) to assign a value for a parameter.

- Use the drop-down listbox in the **Name** field to select a parameter.
- Use the **Value** field to specify a value for the parameter.
- Use the drop-down listbox in the **Database** field to select a database.

Click the **Add** icon (+) to specify each additional parameter; to discard a parameter, click the trash icon to the left of the row and confirm the deletion in the **Delete Row** popup.

Click the **Security** tab to continue.



Use the **Security** tab to define security labels applied to the role. Click the **Add** icon (+) to add each security label selection.

- Specify a security label provider in the **Provider** field. The named provider must be loaded and must consent to the proposed labeling operation.
- Specify a security label in the **Security Label** field. The meaning of a given label is at the discretion of the label provider. PostgreSQL places no restrictions on whether or how a label provider must interpret security labels; it merely provides a mechanism for storing them.

To discard a security label, click the trash icon to the left of the row and confirm the deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **Login/Group Role** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Login/Group Role** dialog:

```

1 CREATE ROLE alice WITH
2   LOGIN
3   NOSUPERUSER
4   NOCREATEDBD
5   NOCREATEROLE
6   INHERIT
7   NOREPLICATION
8   CONNECTION LIMIT -1
9   PASSWORD 'xxxxxx';
10 GRANT pg_monitor TO alice WITH ADMIN OPTION;

```

The example creates a login role named `alice` with `pem_user` privileges; the role can make unlimited connections to the server at any given time.

- Click the Info button (`i`) to access online SQL help.
- Click the Help button (`?`) to access the documentation for the dialog.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.10.5 Tablespace Dialog

Use The **Tablespace** dialog to define a tablespace. A tablespace allows superusers to define an alternative location on the file system where the data files containing database objects (such as tables and indexes) reside. Tablespaces are only supported on systems that support symbolic links. Note that a tablespace cannot be used independently of the cluster in which it is defined.

The **Tablespace** dialog organizes the definition of a tablespace through the following tabs: **General**, **Definition**, **Parameters**, and **Security**. The **SQL** tab displays the SQL code generated by dialog selections.



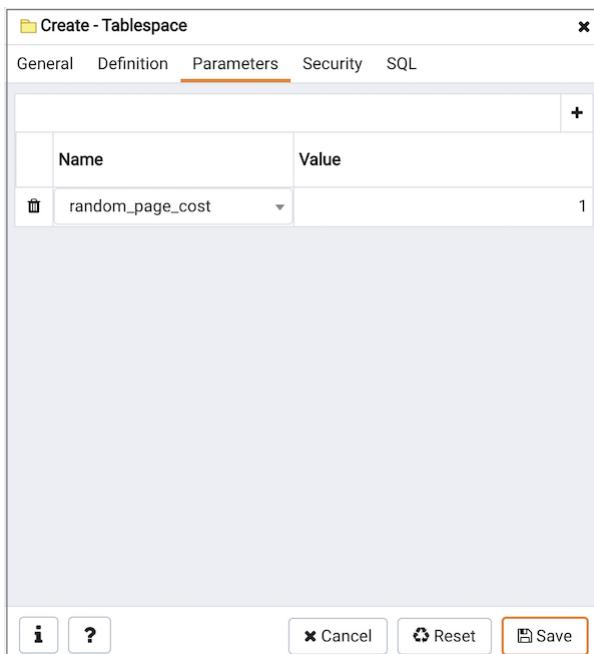
- Use the **Name** field to identify the tablespace with a descriptive name. The name cannot begin with pg_ ; these names are reserved for system tablespaces.
- Select the owner of the tablespace from the drop-down listbox in the *Owner* field.
- Store notes about the tablespace in the **Comment** field.

Click the **Definition** tab to continue.



- Use the **Location** field to specify an absolute path to a directory that will contain the tablespace.

Click the **Parameters** tab to continue.

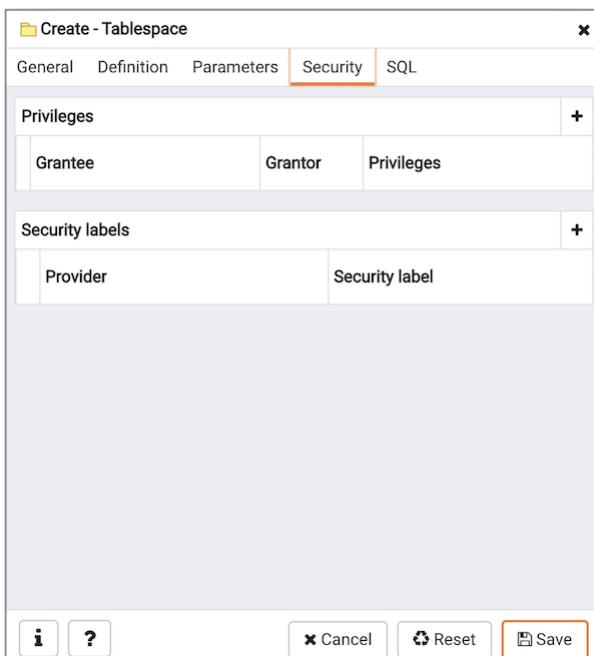


Use the **Parameters** tab to set parameters for the tablespace. Click the *Add* icon (+) to add a row to the table below.

- Use the drop-down listbox next to **Name** to select a parameter.
- Use the **Value** field to set a value for the parameter.

Click the *Add* icon (+) to specify each additional parameter; to discard a parameter, click the trash icon to the left of the row and confirm deletion in the **Delete Row** dialog.

Click the **Security** tab to continue.



Use the **Security** tab to assign privileges and define security labels for the tablespace.

Use the **Privileges** panel to assign security privileges. Click the *Add* icon (+) to assign a set of privileges:

- Select the name of the role from the drop-down listbox in the **Grantee** field.
- The current user, who is the default grantor for granting the privilege, is displayed in the **Grantor** field.
- Click inside the **Privileges** field. Check the boxes to the left of one or more privileges to grant the selected

privileges to the specified user.

Click the **Add** icon to assign additional sets of privileges; to discard a privilege, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Use the **Security Labels** panel to define security labels applied to the tablespace. Click the **Add** icon (+) to add each security label selection:

- Specify a security label provider in the **Provider** field. The named provider must be loaded and must consent to the proposed labeling operation.
- Specify a security label in the **Security Label** field. The meaning of a given label is at the discretion of the label provider. PostgreSQL places no restrictions on whether or how a label provider must interpret security labels; it merely provides a mechanism for storing them.

To discard a security label, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **Tablespace** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Tablespace** dialog:

```

CREATE TABLESPACE new_tab
OWNER postgres
LOCATION '/home/postgres/dbs';

ALTER TABLESPACE new_tab
OWNER TO postgres;

ALTER TABLESPACE new_tab
SET (random_page_cost=1);

```

The example shown demonstrates creating a tablespace named **space_01**. It has a *random_page_cost* value equal to **1**.

- Click the **Info** button (i) to access online help.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.11 Managing Database Objects

PEM provides simple but powerful dialogs that you can use to design and create database objects. Each dialog contains a series of tabs that you use to describe the object that will be created by the dialog; the SQL tab displays the SQL command that the server will execute when creating the object.

To access a dialog that allows you to create a database object, right-click on the object type in the Browser tree control, and select the **Create** option for that object. For example, to create a new cast, right-click on the **Casts** node, and select Create Cast...

Contents:

7.5.11.1 Cast Dialog

Use the **Cast** dialog to define a cast. A cast specifies how to convert a value from one data type to another.

The **Cast** dialog organizes the development of a cast through the following dialog tabs: **General** and **Definition**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the cast:

- The **Name** field is disabled. The name that will be displayed in the **Browser** tree control is the **Source** type concatenated with the **Target** type, and is generated automatically when you make selections on the **Cast** dialog **Definition** tab.
- Store notes about the cast in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define parameters:

- Use the drop-down listbox next to **Source type** to select the name of the source data type of the cast.
- Use the drop-down listbox next to **Target type** to select the name of the target data type of the cast.
- Use the drop-down listbox next to **Function** to select the function used to perform the cast. The function's result data type must match the target type of the cast.
- Move the **Context** switch to the **Implicit** position if the cast is implicit. By default, a cast can be invoked only by an explicit cast request. If the cast is marked **Implicit** then it can be invoked implicitly in any context, whether by assignment or internally in an expression.

Click the **SQL** tab to continue.

Your entries in the **Cast** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Cast** dialog:



The cast uses a function named `int4(bigint)` to convert a bigint data type to an integer.

- Click the `Info` button (i) to access online help. View context-sensitive help in the `Tabbed browser`, where a new tab displays the PostgreSQL core documentation.
- Click the `Save` button to save work.
- Click the `Cancel` button to exit without saving work.
- Click the `Reset` button to restore configuration parameters.

7.5.11.2 Collation Dialog

Use the `Collation` dialog to define a collation. A collation is an SQL schema object that maps a SQL name to operating system locales. To create a collation, you must have a CREATE privilege on the destination schema.

The `Collation` dialog organizes the development of a collation through the following dialog tabs: `General` and `Definition`. The `SQL` tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the collation:

- Use the **Name** field to provide a name for the collation. The collation name must be unique within a schema. The name will be displayed in the Browser tree control.
- Select the name of the owner from the drop-down listbox in the **Owner** field.
- Select the name of the schema in which the collation will reside from the drop-down listbox in the **Schema** field.
- Store notes about the collation in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to specify the operating system locale settings:

- Use the drop-down listbox next to **Copy collation** to select the name of an existing collation to copy. The new collation will have the same properties as the existing one, but will be an independent object. If you choose to copy an existing collation, you cannot modify the collation properties displayed on this tab.
- Use the **Locale** field to specify a locale; a locale specifies language and language formatting characteristics. If you specify this, you cannot specify either of the following parameters. To view a list of locales supported by your Linux

system use the command `locale -a`.

- Use the `LC_COLLATE` field to specify a locale with specified string sort order. The locale must be applicable to the current database encoding. (See CREATE DATABASE for details.)
- Use the `LC_CTYPE` field to specify a locale with specified character classification. The locale must be applicable to the current database encoding. (See CREATE DATABASE for details.)

Click the `SQL` tab to continue.

Your entries in the `Collation` dialog generate a SQL command (see an example below). Use the `SQL` tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the `Collation` dialog:

```

A↓ Create - Collation
General Definition SQL
1 CREATE COLLATION public.french
2   FROM pg_catalog.\"C\";
3
4 ALTER COLLATION public.french
5   OWNER TO postgres;

Cancel Reset Save

```

The example shown demonstrates creating a collation named `french` that uses the rules specified for the locale, `fr-BI-x-icu`. The collation is owned by `postgres`.

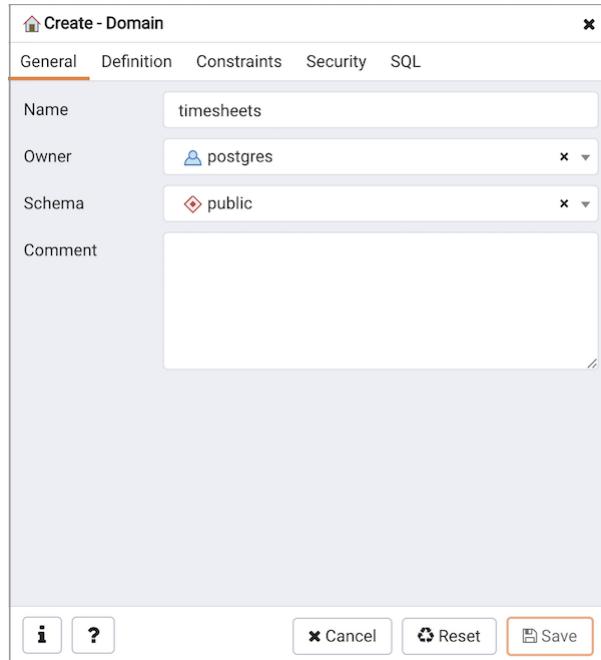
- Click the `Info` button (i) to access online help. For more information about setting a locale, see Chapter 22.1 Locale Support of the PostgreSQL core documentation:

`http://www.postgresql.org/docs/current/static/locale.html`
- Click the `Save` button to save work.
- Click the `Cancel` button to exit without saving work.
- Click the `Reset` button to restore configuration parameters.

7.5.11.3 Domain Dialog

Use the **Domain** dialog to define a domain. A domain is a data type definition that may constrain permissible values. Domains are useful when you are creating multiple tables that contain comparable columns; you can create a domain that defines constraints that are common to the columns and re-use the domain definition when creating the columns, rather than individually defining each set of constraints.

The **Domain** dialog organizes the development of a domain through the following tabs: **General**, **Definition**, **Constraints**, and **Security**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields on the **General** tab to identify a domain:

- Use the **Name** field to add a descriptive name for the domain. The name will be displayed in the **Browser** tree control.
- Use the drop-down listbox next to **Owner** to select a role that will own the domain.
- Select the name of the schema in which the domain will reside from the drop-down listbox in the **Schema** field.
- Store notes about the domain in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to describe the domain:

- Use the drop-down listbox next to **Base type** to specify a data type.
- Use the context-sensitive **Length** field to specify a numeric length for a numeric type.
- Use the context-sensitive **Precision** field to specify the total count of significant digits for a numeric type.
- Specify a default value for the domain data type in the **Default** field. The data type of the default expression must match the data type of the domain. If no default value is specified, then the default value is the null value.
- Move the **Not Null** switch to specify the values of this domain are prevented from being null.
- Use the drop-down listbox next to **Collation** to apply a collation cast. If no collation is specified, the underlying data type's default collation is used. The underlying type must be collatable if COLLATE is specified.

Click the **Constraints** tab to continue.

Name	Check	Validate?
chk	value in ('Mon','Tue','Wed','Thur','Fri')	<input checked="" type="checkbox"/>

Use the fields in the **Constraints** tab to specify rules for the domain. Click the **Add** icon (+) to set constraints:

- Use the **Name** field to specify a name for the constraint.
- Use the **Check** field to provide an expression for the constraint.
- Use the **Validate** checkbox to determine whether the constraint will be validated. The default checkbox is checked and sets a validation requirement.

A CHECK clause specifies an integrity test which values of the domain must satisfy. Each constraint must be an expression that produces a Boolean result. Use the key word VALUE to refer to the value being tested. Expressions evaluating to TRUE or UNKNOWN succeed. If the expression produces a FALSE result, an error is reported and the value is not allowed to be converted to the domain type. A CHECK expression cannot contain subqueries nor refer to variables other than VALUE. If a domain has multiple CHECK constraints, they will be tested in alphabetical order by name.

Click the **Add** icon (+) to set additional constraints; to discard a constraint, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **Security** tab to continue.



Use the **Security Labels** panel to assign security labels. Click the **Add** icon (+) to add a label:

- Specify a security label provider in the **Provider** field. The named provider must be loaded and must consent to the proposed labeling operation.
- Specify a security label in the **Security Label** field. The meaning of a given label is at the discretion of the label provider. PostgreSQL places no restrictions on whether or how a label provider must interpret security labels; it merely provides a mechanism for storing them.

Click the **Add** icon (+) to specify each additional label; to discard a label, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **Domain** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by selections made in the **Domain** dialog:



```

1 CREATE DOMAIN public.timesheets
2   AS character varying;
3
4 ALTER DOMAIN public.timesheets OWNER TO postgres;
5
6 ALTER DOMAIN public.timesheets
7   ADD CONSTRAINT chk CHECK (value IN ('Mon', 'Tue', 'Wed', 'Thur', 'Fri'));

```

The example shown demonstrates creating a domain named **minimum-wage** that confirms that the value entered is greater than or equal to **7.25**.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.11.4 Domain Constraints Dialog

Use the **Domain Constraints** dialog to create or modify a domain constraint. A domain constraint confirms that the values provided for a domain meet a defined criteria. The **Domain Constraints** dialog implements options of the ALTER DOMAIN command.

The **Domain Constraints** dialog organizes the development of a domain constraint through the following dialog tabs: **General** and **Definition**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the domain constraint:

- Use the **Name** field to add a descriptive name for the constraint. The name will be displayed in the **Browser** tree control.
- Store notes about the constraint in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define the domain constraint:

- Use the **Check** field to provide a CHECK expression. A CHECK expression specifies a constraint that the domain must satisfy. A constraint must produce a Boolean result; include the key word VALUE to refer to the value being tested. Only those expressions that evaluate to TRUE or UNKNOWN will succeed. A CHECK expression cannot contain subqueries or refer to variables other than VALUE. If a domain has multiple CHECK constraints, they will be tested in alphabetical order.
- Move the **Validate?** switch to the **No** position to mark the constraint NOT VALID. If the constraint is marked NOT VALID, the constraint will not be applied to existing column data. The default value is **Yes**.

Click the **SQL** tab to continue.

Your entries in the **Domain Constraints** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Domain Constraints** dialog:

```

1 ALTER DOMAIN public.timesheets
2   ADD CONSTRAINT chk CHECK (value in ('Mon', 'Tue', 'Wed', 'Thur', 'Fri'));

```

The example shown demonstrates creating a domain constraint on the domain **timesheets** named **weekday**. It constrains a value to equal **Friday**.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.11.5 Event Trigger Dialog

Use the **Domain Trigger** dialog to define an event trigger. Unlike regular triggers, which are attached to a single table and capture only DML events, event triggers are global to a particular database and are capable of capturing DDL events. Like regular triggers, event triggers can be written in any procedural language that includes event trigger support, or in C, but not in SQL.

The **Domain Trigger** dialog organizes the development of a event trigger through the following dialog tabs: **General**, **Definition**, and **Security Labels**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the event trigger:

- Use the **Name** field to add a descriptive name for the event trigger. The name will be displayed in the **Browser** tree control.
- Use the drop-down listbox next to **Owner** to specify the owner of the event trigger.
- Store notes about the event trigger in the **Comment** field.

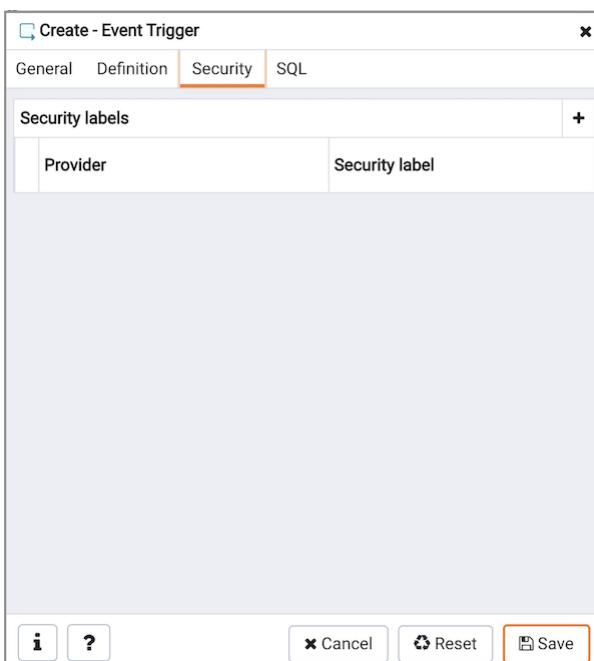
Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define the event trigger:

- Select a value from the drop down of **Trigger Enabled** field to specify a status
- Use the drop-down listbox next to **Trigger function** to specify an existing function. A trigger function takes an empty argument list, and returns a value of type event_trigger.
- Select a value from the drop down of **Events** field to specify when the event trigger will fire: **DDL COMMAND START**, **DDL COMMAND END**, or **SQL DROP**.
- Use the **When TAG in** field to enter filter values for TAG for which the trigger will be executed. The values must be in single quotes separated by comma.

Click the **Security Labels** tab to continue.



Use the **Security** tab to define security labels applied to the trigger. Click the **Add** icon (+) to add each security label.

- Specify a security label provider in the **Provider** field. The named provider must be loaded and must consent to the proposed labeling operation.

- Specify a security label in the **Security Label** field. The meaning of a given label is at the discretion of the label provider. PostgreSQL places no restrictions on whether or how a label provider must interpret security labels; it merely provides a mechanism for storing them.

Click the **Add** icon (+) to assign additional security labels; to discard a security label, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **Domain Trigger** dialog generate a generate a SQL command. Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Domain Trigger** dialog:

```

CREATE - Event Trigger
General Definition Security SQL
1 CREATE EVENT TRIGGER accounts ON DDL_COMMAND_START
2   EXECUTE PROCEDURE test_func();
3
4 ALTER EVENT TRIGGER accounts
5   OWNER TO postgres;

Cancel Reset Save

```

The command creates an event trigger named **accounts** that invokes the procedure named **acct_due**.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.11.6 Extension Dialog

Use the **Extension** dialog to install a new extension into the current database. An extension is a collection of SQL objects that add targeted functionality to your Postgres installation. The **Extension** dialog adds the functionality of an extension to the current database only; you must register the extension in each database that use the extension. Before you load an extension into a database, you should confirm that any pre-requisite files are installed.

The **Extension** dialog allows you to implement options of the CREATE EXTENSION command through the following dialog tabs: **General** and **Definition**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify an extension:

- Use the drop-down listbox in the **Name** field to select the extension. Each extension must have a unique name.
- Store notes about the extension in the **Comment** field.

Click the **Definition** tab to continue.



Use the **Definition** tab to select the **Schema** and **Version**:

- Use the drop-down listbox next to **Schema** to select the name of the schema in which to install the extension's objects.
- Use the drop-down listbox next to **Version** to select the version of the extension to install.

Click the **SQL** tab to continue.

Your entries in the **Extension** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Extension** dialog:



The screenshot shows a dialog window titled "Create - Extension". The "SQL" tab is selected. The content area contains the following SQL code:

```

1 CREATE EXTENSION file_fdw
2   SCHEMA public
3   VERSION "1.0";

```

At the bottom of the dialog are several buttons: "Info" (with a question mark icon), "Cancel", "Reset", and "Save" (highlighted with a red border).

The command creates the **chkpass** extension in the **public** schema. It is version **1.0** of **chkpass**.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.11.7 Foreign Data Wrapper Dialog

Use the **Foreign Data Wrapper** dialog to create or modify a foreign data wrapper. A foreign data wrapper is an adapter between a Postgres database and data stored on another data source.

You must be a superuser to create a foreign data wrapper.

The **Foreign Data Wrapper** dialog organizes the development of a foreign data wrapper through the following dialog tabs: **General**, **Definition**, **Options**, and **Security**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the foreign data wrapper:

- Use the **Name** field to add a descriptive name for the foreign data wrapper. A foreign data wrapper name must be unique within the database. The name will be displayed in the **Browser** tree control.
- Use the drop-down listbox next to **Owner** to select the name of the role that will own the foreign data wrapper.
- Store notes about the foreign data wrapper in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to set parameters:

- Select the name of the handler function from the drop-down listbox in the **Handler** field. This is the name of an existing function that will be called to retrieve the execution functions for foreign tables.
- Select the name of the validator function from the drop-down listbox in the **Validator** field. This is the name of an existing function that will be called to check the generic options given to the foreign data wrapper, as well as options for foreign servers, user mappings and foreign tables using the foreign data wrapper.

Click the **Options** tab to continue.



Use the fields in the **Options** tab to specify options:

- Click the the **Add** icon (+) button to add an option/value pair for the foreign data wrapper. Supported option/value pairs will be specific to the selected foreign data wrapper.
- Specify the option name in the **Option** field and provide a corresponding value in the **Value** field.

Click the **Add** icon (+) to specify each additional pair; to discard an option, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **Security** tab to continue.



Use the **Security** tab to assign security privileges. Click the **Add** icon (+) to assign a set of privileges.

- Select the name of the role from the drop-down listbox in the **Grantee** field.
- Click inside the **Privileges** field. Check the boxes to the left of one or more privileges to grant the selected

privileges to the specified user.

- The current user, who is the default grantor for granting the privilege, is displayed in the **Grantor** field.

Click **Add** to assign additional privileges; to discard a privilege, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **Foreign Data Wrapper** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Foreign Data Wrapper** dialog:

```

CREATE FOREIGN DATA WRAPPER lib_qp_debug
VALIDATOR public.file_fdw_validator
HANDLER public.file_fdw_handler;

ALTER FOREIGN DATA WRAPPER lib_qp_debug
OWNER TO postgres;

COMMENT ON FOREIGN DATA WRAPPER lib_qp_debug
IS 'This FDW enables debugging';

```

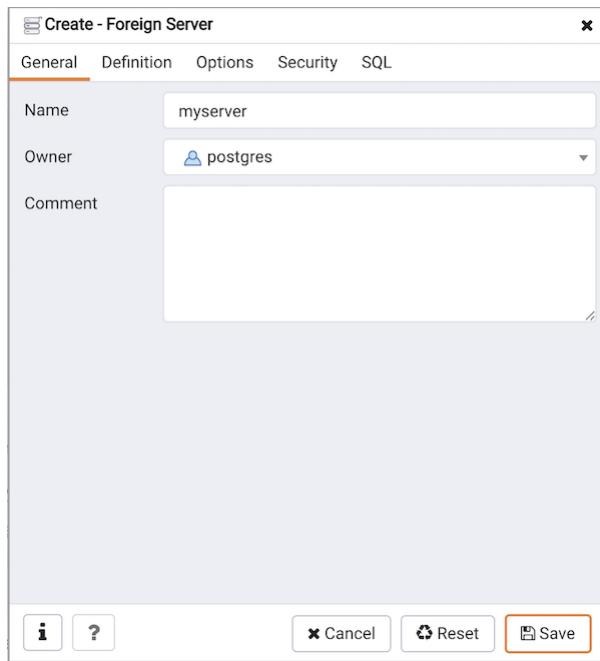
The example creates a foreign data wrapper named `libpq_debug` that uses pre-existing validator and handler functions, `dblink_fdw_validator` and `libpg_fdw_handler`. Selections on the **Options** tab set `debug` equal to `true`. The foreign data wrapper is owned by `postgres`.

- Click the **Help** button (?) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.11.8 Foreign Server Dialog

Use the **Foreign Server** dialog to create a foreign server. A foreign server typically encapsulates connection information that a foreign-data wrapper uses to access an external data resource. Each foreign data wrapper may connect to a different foreign server; in the **Browser** tree control, expand the node of the applicable foreign data wrapper to launch the **Foreign Server** dialog.

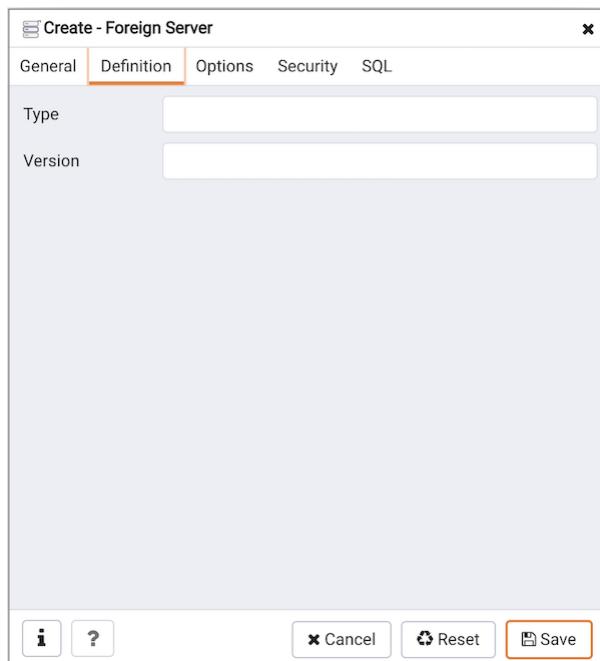
The **Foreign Server** dialog organizes the development of a foreign server through the following dialog tabs: **General**, **Definition**, **Options**, and **Security**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the foreign server:

- Use the **Name** field to add a descriptive name for the foreign server. The name will be displayed in the **Browser** tree control. It must be unique within the database.
- Use the drop-down listbox next to **Owner** to select a role.
- Store notes about the foreign server in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to set parameters:

- Use the **Type** field to specify a server type.
- Use the **Version** field to specify a server version.

Click the **Options** tab to continue.

Options	Value
host	127.0.0.1
port	5432
dbname	db01

Use the fields in the **Options** tab to specify options. Click the **Add** button to create an option clause for the foreign server.

- Specify the option name in the **Option** field.
- Provide a corresponding value in the **Value** field.

Click **Add** to create each additional clause; to discard an option, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **Security** tab to continue.

Grantee	Privileges	Grantor
---------	------------	---------

Use the **Security** tab to assign security privileges to the foreign server. Click **Add** before you assign a set of privileges.

- Select the name of the role from the drop-down listbox in the **Grantee** field.
- Click inside the **Privileges** field. Check the boxes to the left of one or more privileges to grant the selected

privileges to the specified user.

- The current user, who is the default grantor for granting the privilege, is displayed in the **Grantor** field.

Click **Add** to assign a new set of privileges; to discard a privilege, click the trash icon to the left of the row and confirm deletion in the **Delete Row** dialog.

Click the **SQL** tab to continue.

Your entries in the **Foreign Server** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Foreign Server** dialog:

```

CREATE SERVER myserver
  FOREIGN DATA WRAPPER postgres_fdw
  OPTIONS (host '127.0.0.1', port '5432', dbname 'db01');

ALTER SERVER myserver
  OWNER TO postgres;

```

The example shown demonstrates creating a foreign server for the foreign data wrapper **hdfs_fdw**. It has the name **hdfs_server**; its type is **hiveserver2**. Options for the foreign server include a host and a port.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.11.9 Foreign Table Dialog

Use the **Foreign Table** dialog to define a foreign table in the current database. Foreign tables define the structure of an external data source that resides on a foreign server.

The **Foreign Table** dialog organizes the development of a foreign table through the following dialog tabs: **General**, **Definition**, **Columns**, **Constraints**, **Options**, and **Security**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the foreign table:

- Use the **Name** field to add a descriptive name for the foreign table. The name of the foreign table must be distinct from the name of any other foreign table, table, sequence, index, view, existing data type, or materialized view in the same schema. The name will be displayed in the **Browser** tree control.
- Use the drop-down listbox next to **Owner** to select the name of the role that will own the foreign table.
- Select the name of the schema in which the foreign table will reside from the drop-down listbox in the **Schema** field.
- Store notes about the foreign table in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define the external data source:

- Use the drop-down listbox next to **Foreign server** to select a foreign server. This list is populated with servers defined through the **Foreign Server** dialog.
- Use the drop-down listbox next to **Inherits** to specify a parent table. The foreign table will inherit all of its columns. This field is optional.

Click the **Columns** tab to continue.

Name	Data type
No	numeric

Use the fields in the **Columns** tab to add columns and their attributes to the table. Click the **Add** icon (+) to define a column:

- Use the **Name** field to add a descriptive name for the column.
- Use the drop-down listbox in the **Data Type** field to select a data type for the column. This can include array specifiers. For more information on which data types are supported by PostgreSQL, refer to Chapter 8 of the core documentation.

Click the **Add** icon (+) to specify each additional column; to discard a column, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **Constraints** tab to continue.

Name	Check	No inherit?	Validate?
------	-------	-------------	-----------

Use the fields in the **Constraints** tab to apply a table constraint to the foreign table. Click the **Add** icon (+) to define a constraint:

- Use the **Name** field to add a descriptive name for the constraint. If the constraint is violated, the constraint name is present in error messages, so constraint names like `col must be positive` can be used to communicate helpful information.
- Use the **Check** field to write a check expression producing a Boolean result. Each row in the foreign table is expected to satisfy the check expression.
- Check the **No Inherit** checkbox to specify that the constraint will not propagate to child tables.
- Uncheck the **Validate** checkbox to disable validation. The database will not assume that the constraint holds for all rows in the table.

Click the **Add** icon (+) to specify each additional constraint; to discard a constraint, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **Options** tab to continue.



Use the fields in the **Options** tab to specify options to be associated with the new foreign table or one of its columns; the accepted option names and values are specific to the foreign data wrapper associated with the foreign server. Click the **Add** icon (+) to add an option/value pair.

- Specify the option name in the **Option** field. Duplicate option names are not allowed.
- Provide a corresponding value in the **Value** field.

Click the **Add** icon (+) to specify each additional option/value pair; to discard an option, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **Security** tab to continue.



Use the **Security** tab to assign privileges and define security labels.

Use the **Privileges** panel to assign privileges to a role. Click the **Add** icon (+) to set privileges for database objects:

- Select the name of the role to which privileges will be assigned from the drop-down listbox in the **Grantee** field.
- Click inside the **Privileges** field. Check the boxes to the left of one or more privileges to grant the selected privilege to the specified user.
- The current user, who is the default grantor for granting the privilege, is displayed in the **Grantor** field.

Click the **Add** icon (+) to assign additional privileges; to discard a privilege, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Use the **Security Labels** panel to define security labels applied to the function. Click the **Add** icon (+) to add each security label selection:

- Specify a security label provider in the **Provider** field. The named provider must be loaded and must consent to the proposed labeling operation.
- Specify a security label in the **Security Label** field. The meaning of a given label is at the discretion of the label provider. PostgreSQL places no restrictions on whether or how a label provider must interpret security labels; it merely provides a mechanism for storing them.

Click the **Add** icon (+) to assign additional security labels; to discard a security label, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **Foreign Table** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Foreign Table** dialog:



The example shown demonstrates creating a foreign table `weblogs` with multiple columns and two options.

- Click the `Info` button (i) to access online help. View context-sensitive help in the `Tabbed browser`, where a new tab displays the PostgreSQL core documentation.
- Click the `Save` button to save work.
- Click the `Cancel` button to exit without saving work.
- Click the `Reset` button to restore configuration parameters.

7.5.11.10 FTS Configuration dialog

Use the `FTS Configuration` dialog to configure a full text search. A text search configuration specifies a text search parser that can divide a string into tokens, along with dictionaries that can identify searchable tokens.

The `FTS Configuration` dialog organizes the development of a FTS configuration through the following dialog tabs: `"General`, `Definition`, and `Tokens`. The `SQL` tab displays the SQL code generated by dialog selections.

Click the `General` tab to begin.

Create - FTS Configuration

General Definition Tokens SQL

Name: meme_phrases

Owner: postgres

Schema: public

Comment:

Buttons: i ? Cancel Reset Save

Use the fields in the **General** tab to identify a FTS configuration:

- Use the **Name** field to add a descriptive name for the FTS configuration. The name will be displayed in the **Browser** tree control.
- Use the drop-down listbox next to **Owner** to specify the role that will own the configuration.
- Select the name of the schema in which the FTS configuration will reside from the drop-down listbox in the **Schema** field.
- Store notes about the FTS configuration in the **Comment** field.

Click the **Definition** tab to continue.

Create - FTS Configuration

General **Definition** Tokens SQL

Parser: default

Copy config:

Buttons: i ? Cancel Reset Save

Use the fields in the **Definition** tab to define parameters:

- Select the name of the text search parser from the drop-down listbox in the **Parser** field.
- Select a language from the drop-down listbox in the **Copy Config** field.

Click the **Tokens** tab to continue.



Use the fields in the **Tokens** tab to add a token:

- Use the **Tokens** field to specify the name of a token.
- Click the **Add** icon (+) to create a token.
- Use the **Dictionaries** field to specify a dictionary.

Repeat these steps to add additional tokens; to discard a token, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **FTS Configuration** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **FTS Configuration** dialog:

```

1 CREATE TEXT SEARCH CONFIGURATION public.meme_phrases (
2     PARSER = default
3 );
4
5 ALTER TEXT SEARCH CONFIGURATION public.meme_phrases OWNER TO postgres;

```

The example shown demonstrates creating a FTS configuration named **meme_phrases**. It uses the **default** parser.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new

tab displays the PostgreSQL core documentation.

- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.11.11 FTS Dictionary Dialog

Use the **FTS Dictionary** dialog to create a full text search dictionary. You can use a predefined templates or create a new dictionary with custom parameters.

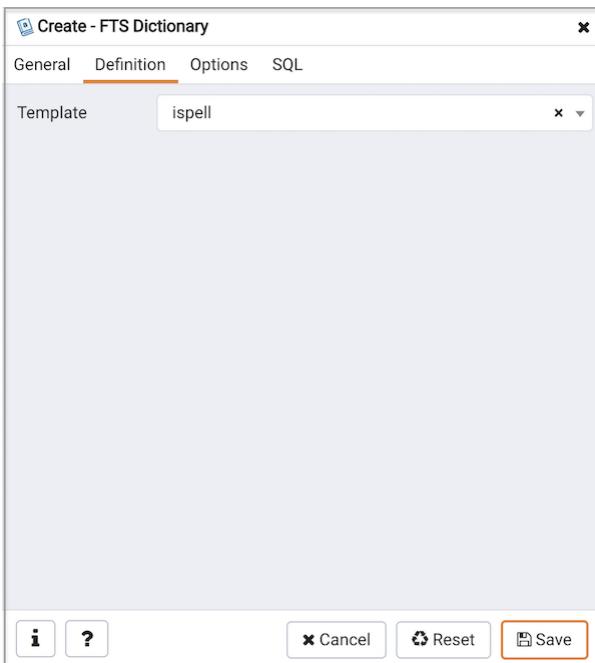
The **FTS Dictionary** dialog organizes the development of a FTS dictionary through the following dialog tabs: **General**, **Definition**, and **Options**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the dictionary:

- Use the **Name** field to add a descriptive name for the dictionary. The name will be displayed in the **Browser** tree control.
- Use the drop-down listbox next to **Owner** to select the role that will own the FTS Dictionary.
- Select the name of the schema in which the dictionary will reside from the drop-down listbox in the **Schema** field.
- Store notes about the dictionary in the **Comment** field.

Click the **Definition** tab to continue.



Use the field in the **Definition** tab to choose a template from the drop-down listbox:

- Select **ispell** to select the Ispell template. The Ispell dictionary template supports morphological dictionaries, which can normalize many different linguistic forms of a word into the same lexeme. For example, an English Ispell dictionary can match all declensions and conjugations of the search term bank, e.g., banking, banked, banks, banks', and bank's. Ispell dictionaries usually recognize a limited set of words, so they should be followed by another broader dictionary; for example, a Snowball dictionary, which recognizes everything.
- Select **simple** to select the simple template. The simple dictionary template operates by converting the input token to lower case and checking it against a file of stop words. If it is found in the file then an empty array is returned, causing the token to be discarded. If not, the lower-cased form of the word is returned as the normalized lexeme. Alternatively, the dictionary can be configured to report non-stop-words as unrecognized, allowing them to be passed on to the next dictionary in the list.
- Select **snowball** to select the Snowball template. The Snowball dictionary template is based on a project by Martin Porter, inventor of the popular Porter's stemming algorithm for the English language. Snowball now provides stemming algorithms for many languages (see the Snowball site for more information). Each algorithm understands how to reduce common variant forms of words to a base, or stem, spelling within its language. A Snowball dictionary recognizes everything, whether or not it is able to simplify the word, so it should be placed at the end of the dictionary list. It is useless to have it before any other dictionary because a token will never pass through it to the next dictionary.
- Select **synonym** to select the synonym template. This dictionary template is used to create dictionaries that replace a word with a synonym. Phrases are not supported (use the thesaurus template (Section 12.6.4) for that). A synonym dictionary can be used to overcome linguistic problems, for example, to prevent an English stemmer dictionary from reducing the word Paris to pari.
- Select **thesaurus** to select the thesaurus template. A thesaurus dictionary replaces all non-preferred terms by one preferred term and, optionally, preserves the original terms for indexing as well. PostgreSQL's current implementation of the thesaurus dictionary is an extension of the synonym dictionary with added phrase support.

Click the **Options** tab to continue.



Use the fields in the **Options** tab to provide template-specific options. Click the **Add** icon (+) to add an option clause:

- Specify the name of an option in the **Option** field
- Provide a value for the option in the **Value** field.

Click the **Add** icon (+) to specify each additional option/value pair; to discard an option, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **FTS Dictionary** dialog generate a generate a SQL command. Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **FTS Dictionary** dialog:

```

CREATE TEXT SEARCH DICTIONARY public.more_stopwords (
  TEMPLATE = ispell,
  data_option = 'data_value'
);

```

The screenshot shows the 'Create - FTS Dictionary' dialog with the 'SQL' tab selected. The generated SQL command is displayed in the main area:

```

CREATE TEXT SEARCH DICTIONARY public.more_stopwords (
  TEMPLATE = ispell,
  data_option = 'data_value'
);

```

The example shown demonstrates creating a custom dictionary named `more_stopwords` which is based on the simple template and is configured to use standard English.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.11.12 FTS Parser Dialog

Use the **FTS Parser** dialog to create a new text search parser. A text search parser defines a method for splitting a text string into tokens and assigning types (categories) to the tokens.

The **FTS Parser** dialog organizes the development of a text search parser through the following dialog tabs: **General**, and **Definition**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify a text search parser:

- Use the **Name** field to add a descriptive name for the parser. The name will be displayed in the **Browser** tree control.
- Select the name of the schema in which the parser will reside from the drop-down listbox in the **Schema** field.
- Store notes about the domain in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define parameters:

- Use the drop-down listbox next to **Start function** to select the name of the function that will initialize the parser.
- Use the drop-down listbox next to **Get next token function** to select the name of the function that will return the next token.
- Use the drop-down listbox next to **End function** to select the name of the function that is called when the parser is finished.
- Use the drop-down listbox next to **Lextypes function** to select the name of the lextypes function for the parser. The lextypes function returns an array that contains the id, alias, and a description of the tokens used by the parser.
- Use the drop-down listbox next to **Headline function** to select the name of the headline function for the parser. The headline function returns an excerpt from the document in which the terms of the query are highlighted.

Click the **SQL** tab to continue.



Your entries in the **FTS Parser** dialog generate a generate a SQL command. Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.11.13 FTS Template Dialog

Use the **FTS Template** dialog to create a new text search template. A text search template defines the functions that implement text search dictionaries.

The **FTS Template** dialog organizes the development of a text search Template through the following dialog tabs: **General**, and **Definition**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify a template:

- Use the **Name** field to add a descriptive name for the template. The name will be displayed in the **Browser** tree control.
- Select the name of the schema in which the template will reside from the drop-down listbox in the **Schema** field.
- Store notes about the template in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define function parameters:

- Use the drop-down listbox next to **Init function** to select the name of the init function for the template. The init function is optional.
- Use the drop-down listbox next to **Lexize function** to select the name of the lexize function for the template. The lexize function is required.

Click the **SQL** tab to continue.

Your entries in the **FTS Template** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **FTS Template** dialog:

```
1 CREATE TEXT SEARCH TEMPLATE public.\"ru_template\" (
2   INIT = brin_inclusion_opcinfo,   LEXIZE = disspell_lexize
3 );
```

The example shown demonstrates creating a fts template named **ru_template** that uses the ispell dictionary.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new

tab displays the PostgreSQL core documentation.

- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.11.14 Function Dialog

Use the **Function** dialog to define a function. If you drop and then recreate a function, the new function is not the same entity as the old; you must drop existing rules, views, triggers, etc. that refer to the old function.

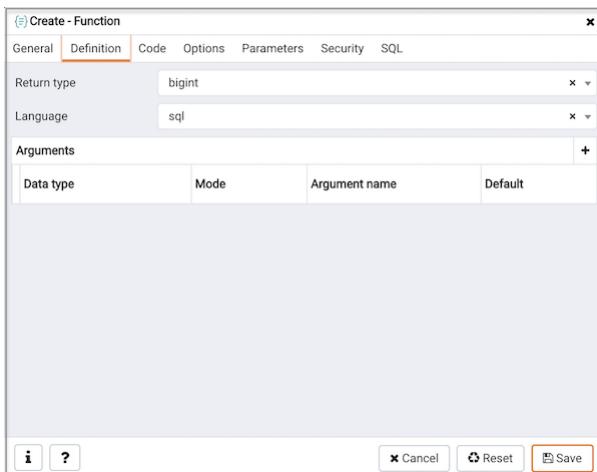
The **Function** dialog organizes the development of a function through the following dialog tabs: **General**, **Definition**, **Code**, **Options**, **Parameters**, and **Security**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify a function:

- Use the **Name** field to add a descriptive name for the function. The name will be displayed in the **Browser** tree control.
- Use the drop-down listbox next to **Owner** to select the name of the role that will own the function.
- Use the drop-down listbox next to **Schema** to select the schema in which the function will be created.
- Store notes about the function in the **Comment** field.

Click the **Definition** tab to continue.

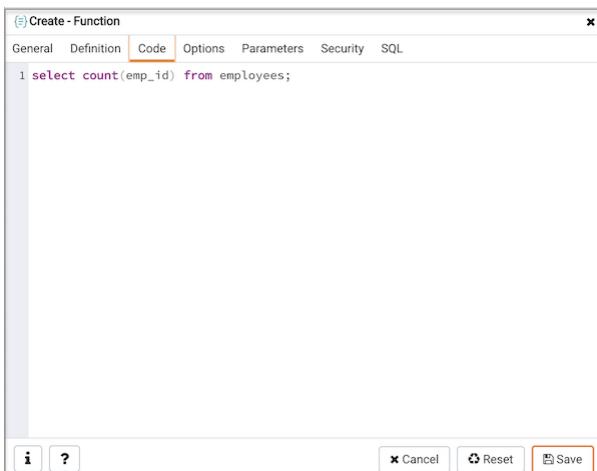


Use the fields in the **Definition** tab to define the function:

- Use the drop-down listbox next to **Return type** to select the data type returned by the function, if any.
- Use the drop-down listbox next to **Language** to select the implementation language. The default is **sql**.
- Use the fields in the **Arguments** to define an argument. Click the Add icon (+) to set parameters and values for the argument:
 - Use the drop-down listbox in the **Data type** field to select a data type.
 - Use the drop-down listbox in the **Mode** field to select a mode. Select IN for an input parameter; select OUT for an output parameter; select INOUT for both an input and an output parameter; or, select VARIADIC to specify a VARIADIC parameter.
 - Provide a name for the argument in the **Argument Name** field.
 - Specify a default value for the argument in the **Default Value** field.

Click the **Add** icon (+) to define another argument; to discard an argument, click the trash icon to the left of the row and confirm deletion in the *Delete Row* popup.

Click the **Code** tab to continue.



- Use the **Code** field to write the code that will execute when the function is called.

Click the **Options** tab to continue.



Use the fields in the **Options** tab to describe or modify the action of the function:

- Use the drop-down listbox next to **Volatility** to select one of the following. **VOLATILE** is the default value.
 - **VOLATILE** indicates that the function value can change even within a single table scan, so no optimizations can be made.
 - **STABLE** indicates that the function cannot modify the database, and that within a single table scan it will consistently return the same result for the same argument values.
 - **IMMUTABLE** indicates that the function cannot modify the database and always returns the same result when given the same argument values.
- Move the **Returns a Set?** switch to indicate if the function returns a set that includes multiple rows. The default is **No**.
- Move the **Strict?** switch to indicate if the function always returns NULL whenever any of its arguments are NULL. If **Yes**, the function is not executed when there are NULL arguments; instead a NULL result is assumed automatically. The default is **No**.
- Move the **Security of definer?** switch to specify that the function is to be executed with the privileges of the user that created it. The default is **No**.
- Move the **Window?** switch to indicate that the function is a window function rather than a plain function. The default is **No**. This is currently only useful for functions written in C. The WINDOW attribute cannot be changed when replacing an existing function definition. For more information about the CREATE FUNCTION command, see the PostgreSQL core documentation available at:
 - <http://www.postgresql.org/docs/current/static/functions-window.html>
- Use the **Estimated cost** field to specify a positive number representing the estimated execution cost for the function, in units of cpu_operator_cost. If the function returns a set, this is the cost per returned row.
- Use the **Estimated rows** field to specify a positive number giving the estimated number of rows that the query planner should expect the function to return. This is only allowed when the function is declared to return a set. The default assumption is 1000 rows.
- Move the **Leak proof?** switch to indicate whether the function has side effects. The default is **No**. This option can only be set by the superuser.
- Use the **Support function** field to specify a planner support function to use for the function.

Click the **Parameters** tab to continue.



Use the fields in the **Parameters** tab to specify settings that will be applied when the function is invoked. Click the **Add** icon (+) to add a **Name / Value** field in the table.

- Use the drop-down listbox in the **Name** column in the **Parameters** panel to select a parameter.
- Use the **Value** field to specify the value that will be associated with the selected variable. This field is context-sensitive.

Click the **Security** tab to continue.



Use the **Security** tab to assign privileges and define security labels.

Use the **Privileges** panel to assign usage privileges for the function to a role.

- Select the name of the role from the drop-down listbox in the **Grantee** field.
- Click inside the **Privileges** field. Check the boxes to the left of one or more privileges to grant the selected privilege to the specified user.
- The current user, who is the default grantor for granting the privilege, is displayed in the **Grantor** field.

Click the **Add** icon (+) to assign additional privileges; to discard a privilege, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Use the **Security Labels** panel to define security labels applied to the function. Click the **Add** icon (+) to add each security label selection:

- Specify a security label provider in the **Provider** field. The named provider must be loaded and must consent to the proposed labeling operation.
- Specify a security label in the **Security Label** field. The meaning of a given label is at the discretion of the label provider. PostgreSQL places no restrictions on whether or how a label provider must interpret security labels; it

merely provides a mechanism for storing them.

Click the **Add** icon (+) to assign additional security labels; to discard a security label, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **Function** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by selections made in the **Function** dialog:



```

Create - Function
General Definition Code Options Parameters Security SQL
x
1 CREATE FUNCTION public.emp_count()
2   RETURNS bigint
3   LANGUAGE 'sql'
4   VOLATILE |
5 AS $BODY$select count(emp_id) from employees;$BODY$;
6
7 ALTER FUNCTION public.emp_count()
8   OWNER TO postgres;
9
10 COMMENT ON FUNCTION public.emp_count()
11   IS 'This function gets the employee count';

```

The screenshot shows a software interface titled "Create - Function". The "SQL" tab is active. The main area contains a block of SQL code. At the bottom are buttons for "Info", "?", "Cancel", "Reset", and "Save", with "Save" being highlighted.

The example demonstrates creating an **edbsql** function named **emp_comp**. The function adds two columns (p_sal and p_comm), and then uses the result to compute a yearly salary, returning a NUMERIC value.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.11.15 Language Dialog

Use the CREATE LANGUAGE dialog to register a new procedural language.

The **Language** dialog organizes the registration of a procedural language through the following dialog tabs: **General**, **Definition**, and **Security**. The **SQL** tab displays the SQL code generated by dialog selections.

The screenshot shows the 'Create - Language' dialog box with the 'General' tab selected. The 'Name' field contains 'plperl', the 'Owner' field shows 'postgres', and the 'Comment' field is empty. At the bottom, there are buttons for 'Cancel', 'Reset', and 'Save'.

Use the fields in the **General** tab to identify a language:

- Use the drop-down listbox next to **Name** to select a language script.
- Use the drop-down listbox next to **Owner** to select a role.
- Store notes about the language in the **Comment** field.

Click the **Definition** tab to continue.

The screenshot shows the 'Create - Language' dialog box with the 'Definition' tab selected. The 'Trusted?' switch is set to 'Yes'. Below it are four dropdown menus for 'Handler function', 'Inline function', and 'Validator function', all of which are currently empty. At the bottom, there are buttons for 'Cancel', 'Reset', and 'Save'.

Use the fields in the **Definition** tab to define parameters:

- Move the **Trusted?** switch to the **No** position to specify only users with PostgreSQL superuser privilege can use this language. The default is **Yes**.
- When enabled, use the drop-down listbox next to **Handler Function** to select the function that will be called to execute the language's functions.
- When enabled, use the drop-down listbox next to **Inline Function** to select the function that will be called to execute an anonymous code block (DO command) in this language.
- When enabled, use the drop-down listbox next to **Validator Function** to select the function that will be called

when a new function in the language is created, to validate the new function.

Click the **Security** tab to continue.



Use the **Security** tab to assign privileges and define security labels.

Use the **Privileges** panel to assign privileges to a role. Click the **Add** icon (+) to set privileges for database objects:

- Select the name of the role from the drop-down listbox in the **Grantee** field.
- Click inside the **Privileges** field. Check the boxes to the left of one or more privileges to grant the selected privilege to the specified user.
- The current user, who is the default grantor for granting the privilege, is displayed in the **Grantor** field.

Click the **Add** icon (+) to assign additional privileges; to discard a privilege, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Use the **Security Labels** panel to define security labels applied to the function. Click the **Add** icon (+) to add each security label selection:

- Specify a security label provider in the **Provider** field. The named provider must be loaded and must consent to the proposed labeling operation.
- Specify a security label in the **Security Label** field. The meaning of a given label is at the discretion of the label provider. PostgreSQL places no restrictions on whether or how a label provider must interpret security labels; it merely provides a mechanism for storing them.

Click the **Add** icon (+) to assign additional security labels; to discard a security label, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **Language** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Language** dialog:



"The example shown demonstrates creating the procedural language named `plperl`."

- Click the `Info` button (i) to access online help. View context-sensitive help in the `Tabbed browser`, where a new tab displays the PostgreSQL core documentation.
- Click the `Save` button to save work.
- Click the `Cancel` button to exit without saving work.
- Click the `Reset` button to restore configuration parameters.

7.5.11.16 Materialized View Dialog

Use the `Materialized View` dialog to define a materialized view. A materialized view is a stored or cached view that contains the result set of a query. Use the `REFRESH MATERIALIZED VIEW` command to update the content of a materialized view.

The `Materialized View` dialog organizes the development of a `materialized_view` through the following dialog tabs: `General`, `Definition`, `Storage`, `Parameter`, and `Security`. The `SQL` tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the materialized view:

- Use the **Name** field to add a descriptive name for the materialized view. The name will be displayed in the **Browser** tree control.
- Use the drop-down listbox next to **Owner** to select the role that will own the materialized view.
- Select the name of the schema in which the materialized view will reside from the drop-down listbox in the **Schema** field.
- Store notes about the materialized view in the **Comment** field.

Click the **Definition** tab to continue.



Use the text editor field in the **Definition** tab to provide the query that will populate the materialized view. Please note that updating the definition of existing materialized view would result in loss of Parameter(Table, Toast), Security(Privileges & Security labels), Indexes and other dependent objects.

Click the **Storage** tab to continue.



Use the fields in the **Storage** tab to maintain the materialized view:

- Move the **With Data** switch to the **Yes** position to specify the materialized view should be populated at creation time. If not, the materialized view cannot be queried until you invoke REFRESH MATERIALIZED VIEW.
- Use the drop-down listbox next to **Tablespace** to select a location for the materialized view.
- Use the **Fill Factor** field to specify a fill factor for the materialized view. The fill factor for a table is a percentage between 10 and 100. 100 (complete packing) is the default.

Click the **Parameter** tab to continue.



Use the tabs nested inside the **Parameter** tab to specify VACUUM and ANALYZE thresholds; use the **Table** tab and the **Toast Table** tab to customize values for the table and the associated toast table. To change the default values:

- Move the **Custom auto-vacuum?** switch to the **Yes** position to perform custom maintenance on the materialized view and to select values in the **Vacuum table**. The **Vacuum Table** provides default values for maintenance operations.
- Changing **Autovacuum enabled?** to **Not set** will reset autovacuum_enabled.

Click the **Security** tab to continue.



Use the **Security** tab to assign privileges and define security labels.

Use the **Privileges** panel to assign privileges to a role. Click the **Add** icon (+) to set privileges for the materialized view:

- Select the name of the role from the drop-down listbox in the **Grantee** field.
- Click inside the **Privileges** field. Check the boxes to the left of one or more privileges to grant the selected privilege to the specified user.
- The current user, who is the default grantor for granting the privilege, is displayed in the **Grantor** field.

Click the **Add** icon (+) to assign additional privileges; to discard a privilege, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Use the **Security Labels** panel to define security labels applied to the materialized view. Click the **Add** icon (+) to add each security label selection:

- Specify a security label provider in the **Provider** field. The named provider must be loaded and must consent to the proposed labeling operation.
- Specify a security label in the **Security Label** field. The meaning of a given label is at the discretion of the label provider. PostgreSQL places no restrictions on whether or how a label provider must interpret security labels; it merely provides a mechanism for storing them.

Click the **Add** icon (+) to assign additional security labels; to discard a security label, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **Materialized View** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Materialized View** dialog:



The example shown creates a query named `new_hires` that stores the result of the displayed query in the `pg_default` tablespace.

- Click the `Info` button (i) to access online help. View context-sensitive help in the `Tabbed browser`, where a new tab displays the PostgreSQL core documentation.
- Click the `Save` button to save work.
- Click the `Cancel` button to exit without saving work.
- Click the `Reset` button to restore configuration parameters.

7.5.11.17 Package Dialog

Use the `Package` dialog to create a (user-defined) package specification.

The `Package` dialog organizes the management of a package through the following dialog tabs: `General`, `Header`, `Body`, and `Security`. The `SQL` tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the package:

- Use the **Name** field to add a descriptive name for the package. The name of a new package must not match any existing package in the same schema.
- Select the schema in which the package will reside from the drop-down listbox in the **Schema** field.
- Store notes about the package in the **Comment** field.

Click the **Header** tab to continue.



Use the **Header** field to define the public interface for the package.

Click the **Body** tab to continue.

```

1 FUNCTION get_dept_name (
2     p_deptno      IN NUMBER
3 ) RETURN VARCHAR2
4 IS
5     v_dname        VARCHAR2(14);
6 BEGIN
7     SELECT dname INTO v_dname FROM dept WHERE deptno = p_deptno;
8     RETURN v_dname;
9 EXCEPTION
10    WHEN NO_DATA_FOUND THEN
11        DBMS_OUTPUT.PUT_LINE('Invalid department number ' || p_deptno);
12        RETURN '';
13 END;
14
15 PROCEDURE fire_emp (
16     p_empno      NUMBER
17 )
18 AS
19 BEGIN
20     DELETE FROM emp WHERE empno = p_empno;
21 END;

```

Buttons: i, ?, Cancel, Reset, Save

Use the **Body** field to provide the code that implements each package object.

Click the **Security** tab to continue.

Privileges		
Grantee	Privileges	Grantor

Buttons: i, ?, Cancel, Reset, Save

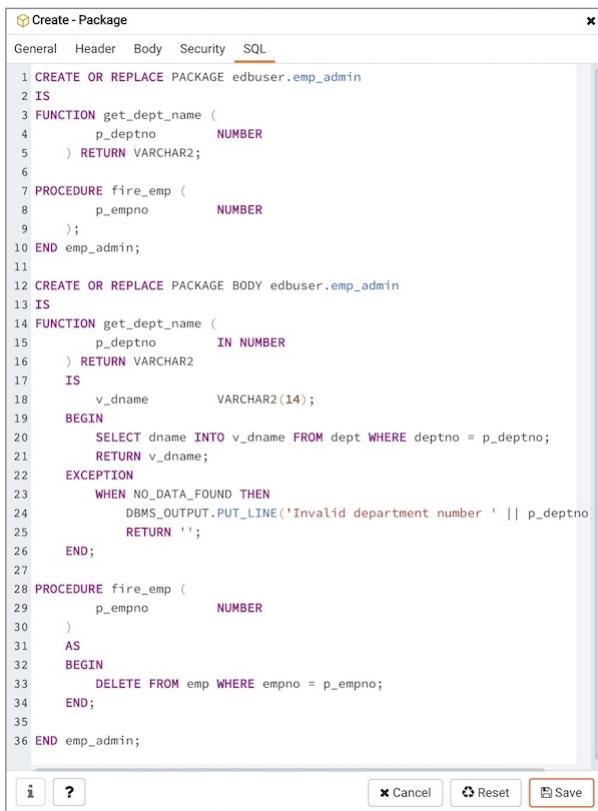
Use the fields in the **Security** tab to assign EXECUTE privileges for the package to a role. Click the **Add** icon (+) to set privileges for the package:

- Select the name of the role from the drop-down listbox in the **Grantee** field.
- Click inside the **Privileges** field. Check the boxes to the left of a privilege to grant the selected privilege to the specified user.
- The current user, who is the default grantor for granting the privilege, is displayed in the **Grantor** field.

Click the **Add** icon (+) to assign additional privileges; to discard a privilege, click the trash icon to the left of the row, and confirm the deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **Package** dialog generate a SQL command that creates or modifies a package definition:



```

1 CREATE OR REPLACE PACKAGE edbuser.emp_admin
2 IS
3 FUNCTION get_dept_name (
4     p_deptno      NUMBER
5 ) RETURN VARCHAR2;
6
7 PROCEDURE fire_emp (
8     p_empno       NUMBER
9 );
10 END emp_admin;
11
12 CREATE OR REPLACE PACKAGE BODY edbuser.emp_admin
13 IS
14 FUNCTION get_dept_name (
15     p_deptno      IN NUMBER
16 ) RETURN VARCHAR2
17 IS
18     v_dname        VARCHAR2(14);
19 BEGIN
20     SELECT dname INTO v_dname FROM dept WHERE deptno = p_deptno;
21     RETURN v_dname;
22 EXCEPTION
23     WHEN NO_DATA_FOUND THEN
24         DBMS_OUTPUT.PUT_LINE('Invalid department number ' || p_deptno
25         RETURN '';
26 END;
27
28 PROCEDURE fire_emp (
29     p_empno      NUMBER
30 )
31 AS
32 BEGIN
33     DELETE FROM emp WHERE empno = p_empno;
34 END;
35
36 END emp_admin;

```

The dialog has three buttons at the bottom: 'Cancel', 'Reset', and 'Save' (highlighted with a red border).

The example shown demonstrates creating a package named `empinfo` that includes two function and two procedure.

- Click the `Save` button to save work.
- Click the `Cancel` button to exit without saving work.
- Click the `Reset` button to delete any changes to the dialog.

7.5.11.18 Procedure Dialog

Use the `Procedure` dialog to create a procedure; procedures are supported by PostgreSQL v11+ and EDB Postgres Advanced Server. The `Procedure` dialog allows you to implement options of the CREATE PROCEDURE command.

The `Procedure` dialog organizes the development of a procedure through the following dialog tabs: `General`, `Definition`, `Options`, `Arguments`, `Parameters`, and `Security`. The `SQL` tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify a procedure:

- Use the **Name** field to add a descriptive name for the procedure. The name will be displayed in the ** tree control.
- Use the drop-down listbox next to **Owner** to select a role.
- Select the name of the schema in which the procedure will reside from the drop-down listbox in the **Schema** field.
- Store notes about the procedure in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define the procedure:

- Use the drop-down listbox next to **Language** to select a language. The default is **edbspl**.
- Use the fields in the **Arguments** section to define an argument. Click **Add** to set parameters and values for the argument:
 - Use the drop-down listbox next to **Data type** to select a data type.
 - Use the drop-down listbox next to **Mode** to select a mode. Select **IN** for an input parameter; select **OUT** for an output parameter; select **INOUT** for both an input and an output parameter; or, select **VARIADIC** to specify a VARIADIC parameter.
 - Write a name for the argument in the **Argument Name** field.
 - Specify a default value for the argument in the **Default Value** field.

Click **Add** to define another argument; to discard an argument, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **Code** tab to continue.



- Use the **Code** field to specify the code that will execute when the procedure is called.

Click the **Options** tab to continue.



Use the fields in the **Options** tab to describe or modify the behavior of the procedure:

- Use the drop-down listbox under **Volatility** to select one of the following. **VOLATILE** is the default value.
 - **VOLATILE** indicates that the value can change even within a single table scan, so no optimizations can be made.
 - **STABLE** indicates that the procedure cannot modify the database, and that within a single table scan it will consistently return the same result for the same argument values, but that its result could change across SQL statements.
 - **IMMUTABLE** indicates that the procedure cannot modify the database and always returns the same result when given the same argument values.
- Move the **Strict?** switch to indicate if the procedure always returns NULL whenever any of its arguments are NULL. If **Yes**, the procedure is not executed when there are NULL arguments; instead a NULL result is assumed automatically. The default is **No**.
- Move the **Security of definer?** switch to specify that the procedure is to be executed with the privileges of the user that created it. The default is **No**.
- Use the **Estimated cost** field to specify a positive number representing the estimated execution cost for the procedure, in units of cpu_operator_cost. If the procedure returns a set, this is the cost per returned row.
- Move the **Leak proof?** switch to indicate whether the procedure has side effects – it reveals no information

about its arguments other than by its return value. The default is **No**.

Click the **Parameters** tab to continue.



Use the fields in the **Parameters** tab to specify settings that will be applied when the procedure is invoked:

- Use the drop-down listbox next to **Parameter Name** in the **Parameters** panel to select a parameter.
- Click the **Add** button to add the variable to **Name** field in the table.
- Use the **Value** field to specify the value that will be associated with the selected variable. This field is context-sensitive.

Click the **Security** tab to continue.



Use the **Security** tab to assign privileges and define security labels.

Use the **Privileges** panel to assign execute privileges for the procedure to a role:

- Select the name of the role from the drop-down listbox in the **Grantee** field.
- Click inside the **Privileges** field. Check the boxes to the left of one or more privileges to grant the selected privilege to the specified user.
- The current user, who is the default grantor for granting the privilege, is displayed in the **Grantor** field.

Click **Add** to assign additional privileges; to discard a privilege, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Use the **Security Labels** panel to define security labels applied to the procedure. Click **Add** to add each security label selection:

- Specify a security label provider in the **Provider** field. The named provider must be loaded and must consent to the proposed labeling operation.
- Specify a security label in the **Security Label** field. The meaning of a given label is at the discretion of the label provider. PostgreSQL places no restrictions on whether or how a label provider must interpret security labels; it merely provides a mechanism for storing them.

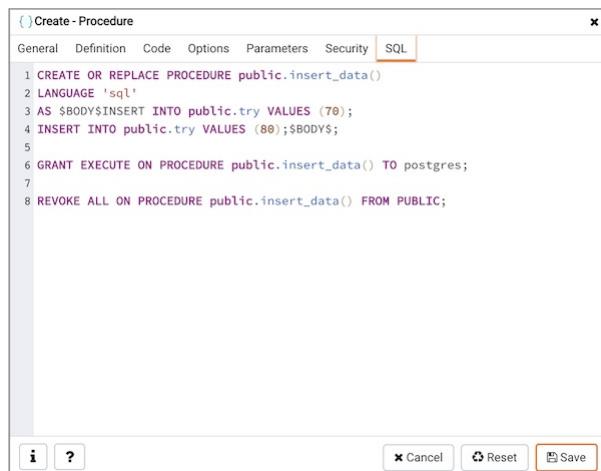
Click **Add** to assign additional security labels; to discard a security label, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **Procedure** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by selections made in the **Procedure** dialog:



```

CREATE OR REPLACE PROCEDURE public.insert_data()
LANGUAGE 'sql'
AS $BODY$INSERT INTO public.try VALUES (70);
INSERT INTO public.try VALUES (80);$BODY$;
GRANT EXECUTE ON PROCEDURE public.insert_data() TO postgres;
REVOKE ALL ON PROCEDURE public.insert_data() FROM PUBLIC;

```

The example demonstrates creating a procedure that returns a list of employees from a table named **emp**. The procedure is a SECURITY DEFINER, and will execute with the privileges of the role that defined the procedure.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.11.19 Schema Dialog

Use the **Schema** dialog to define a schema. A schema is the organizational workhorse of a database, similar to directories or namespaces. To create a schema, you must be a database superuser or have the CREATE privilege.

The **Schema** dialog organizes the development of schema through the following dialog tabs: **General** and **Security**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields on the **General** tab to identify the schema.

- Use the **Name** field to add a descriptive name for the schema. The name will be displayed in the **Browser** tree control.
- Select the owner of the schema from the drop-down listbox in the **Owner** field.
- Store notes about the schema in the **Comment** field.

Click the **Security** tab to continue.



Use the **Security** tab to assign privileges and security labels for the schema.

Click the **Add** icon (+) to assign a set of privileges in the **Privileges** panel:

- Select the name of the role from the drop-down listbox in the **Grantee** field.
- Click inside the **Privileges** field. Check the boxes to the left of one or more privileges to grant the selected privileges to the specified user.
- Select the name of the role that is granting the privilege from the drop-down listbox in the **Grantor** field. The

default grantor is the owner of the schema.

Click the **Add** icon (+) to assign additional sets of privileges; to discard a privilege, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **Add** icon (+) to assign a security label in the **Security Labels** panel:

- Specify a security label provider in the **Provider** field. The named provider must be loaded and must consent to the proposed labeling operation.
- Specify a security label in the **Security Label** field. The meaning of a given label is at the discretion of the label provider. PostgreSQL places no restrictions on whether or how a label provider must interpret security labels; it merely provides a mechanism for storing them.

Click the **Add** icon (+) to assign additional security labels; to discard a security label, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **Default Privileges** tab to continue.



Use the **Default Privileges** tab to grant privileges for tables, sequences, functions and types. Use the tabs nested inside the **Default Privileges** tab to specify the database object and click the **Add** icon (+) to assign a set of privileges:

- Select the name of a role that will be granted privileges in the schema from the drop-down listbox in the **Grantee** field.
- Click inside the **Privileges** field. Check the boxes to the left of one or more privileges to grant the selected privileges to the specified user.
- The current user, who is the default grantor for granting the privilege, is displayed in the **Grantor** field.

Click the **SQL** tab to continue.

Your entries in the **Schema** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by selections made in the **Schema** dialog:



The example creates a schema named hr; the command grants **USAGE** privileges to **public** and assigns the ability to grant privileges to **alice**.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.11.20 Sequence Dialog

Use the **Sequence** dialog to create a sequence. A sequence generates unique values in a sequential order (not necessarily contiguous).

The **Sequence** dialog organizes the development of a sequence through the following dialog tabs: **General**, **Definition**, and **Security**. The **SQL** tab displays the SQL code generated by dialog selections.

1..3 Create - Sequence

General Definition Security SQL

Name: seq1

Owner: postgres

Schema: public

Comment:

Buttons: Cancel, Reset, Save

Use the fields in the **General** tab to identify a sequence:

- Use the **Name** field to add a descriptive name for the sequence. The name will be displayed in the **Browser** tree control. The sequence name must be distinct from the name of any other sequence, table, index, view, or foreign table in the same schema.
- Use the drop-down listbox next to **Owner** to select the name of the role that will own the sequence.
- Use the drop-down listbox next to **Schema** to select the schema in which the sequence will reside.
- Store notes about the sequence in the **Comment** field.

Click the **Definition** tab to continue.

1..3 Create - Sequence

General Definition Security SQL

Increment: 10

Start: 1

Minimum:

Maximum: 100

Cache:

Cycled: No

Buttons: Cancel, Reset, Save

Use the fields in the **Definition** tab to define the sequence:

- Use the **Increment** field to specify which value is added to the current sequence value to create a new value.
- Provide a value in the **Start** field to specify the beginning value of the sequence. The default starting value is MINVALUE for ascending sequences and MAXVALUE for descending ones.
- Provide a value in the **Minimum** field to specify the minimum value a sequence can generate. If this clause is not

supplied or NO MINVALUE is specified, then defaults will be used. The defaults are 1 and -263-1 for ascending and descending sequences, respectively.

- Provide a value in the **Maximum** field to specify the maximum value for the sequence. If this clause is not supplied or NO MAXVALUE is specified, then default values will be used. The defaults are 263-1 and -1 for ascending and descending sequences, respectively.
- Provide a value in the **Cache** field to specify how many sequence numbers are to be preallocated and stored in memory for faster access. The minimum value is 1 (only one value can be generated at a time, i.e., no cache), and this is also the default.
- Move the **Cycled** switch to the **Yes** position to allow the sequence to wrap around when the MAXVALUE or the MINVALUE has been reached by an ascending or descending sequence respectively. If the limit is reached, the next number generated will be the MINVALUE or MAXVALUE, respectively. The default is **No**.

Click the **Security** tab to continue.

The screenshot shows the 'Create - Sequence' dialog box with the 'Security' tab selected. The 'Privileges' panel is active, displaying grants for the 'PUBLIC' role. Under 'Grantee', 'PUBLIC' is selected. Under 'Privileges', 'ALL', 'SELECT', 'UPDATE', and 'USAGE' are checked, while 'WITH GRANT OPTION' is unchecked for all. The 'Grantor' field shows 'postgres'. The 'Security labels' panel is partially visible below it. At the bottom, there are 'Cancel', 'Reset', and 'Save' buttons, with 'Save' being highlighted.

Use the **Security** tab to assign privileges and define security labels for the sequence.

Use the **Privileges** panel to assign privileges. Click the **Add** icon (+) to set privileges:

- Select the name of a role that will be granted privileges from the drop-down listbox in the **Grantee** field.
- Click inside the **Privileges** field. Check the boxes to the left of one or more privileges to grant the selected privilege to the specified user.
- The current user, who is the default grantor for granting the privilege, is displayed in the **Grantor** field.

Click the **Add** icon (+) to assign additional privileges; to discard a privilege, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Use the **Security Labels** panel to define security labels applied to the sequence. Click the **Add** icon (+) to add each security label selection:

- Specify a security label provider in the **Provider** field. The named provider must be loaded and must consent to the proposed labeling operation.
- Specify a security label in the **Security Label** field. The meaning of a given label is at the discretion of the label provider. PostgreSQL places no restrictions on whether or how a label provider must interpret security labels; it merely provides a mechanism for storing them.

Click the **Add** icon (+) to assign additional security labels; to discard a security label, click the trash icon to the left of

the row and confirm deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **Sequence** dialog generate a generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Sequence** dialog:



```
1.3 Create - Sequence
General Definition Security SQL
x
1 CREATE SEQUENCE public.seq1
2   INCREMENT 10
3   START 1
4   MINVALUE 0
5   MAXVALUE 100;
6
7 ALTER SEQUENCE public.seq1
8   OWNER TO postgres;
9
10 GRANT ALL ON SEQUENCE public.seq1 TO PUBLIC;
```

The dialog has tabs for General, Definition, Security, and SQL. The SQL tab is selected and highlighted with an orange border. Below the tabs is a large text area containing the generated SQL code. At the bottom are buttons for Info, Cancel, Reset, and Save, with Save being the active button (orange border).

The example shown demonstrates a sequence named **seconds**. The sequence will increase in **5** second increments, and stop when it reaches a maximum value equal of **60**.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.11.21 Synonym Dialog

Use the **Synonym** dialog to substitute the name of a target object with a user-defined synonym.

The **Synonym** dialog organizes the development of a synonym through the **General** tab. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the synonym:

- Use the **Name** field to specify the name of synonym. The name will be displayed in the **Browser** tree control.
- Select the name of the schema in which the synonym will reside from the drop-down listbox in the **Schema** field.

In the definition panel, identify the target:

- Use the drop-down listbox next to **Target Type** to select the type of object referenced by the synonym.
- Use the drop-down listbox next to **Target Schema** to select the name of the schema in which the object resides.
- Use the drop-down listbox next to **Target Object** to select the name of the object referenced by the synonym.

Click the **SQL** tab to continue.

Your selections and entries in the **Synonym** dialog generate a SQL command.



The example creates a synonym for the **emp** table named **emp_hist**.

- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.11.22 Trigger function Dialog

Use the **Trigger function** dialog to create or manage a trigger_function. A trigger function defines the action that will be invoked when a trigger fires.

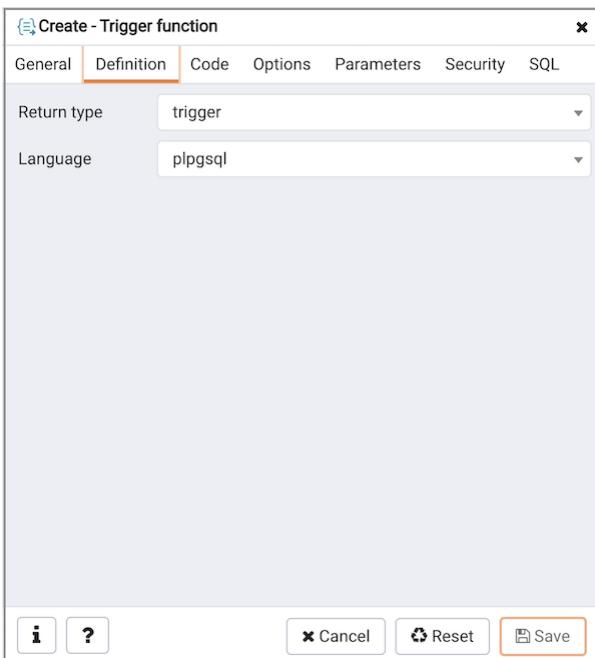
The **Trigger function** dialog organizes the development of a trigger function through the following dialog tabs: **General**, **Definition**, **Code**, **Options**, **Parameters** and **Security**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the trigger function:

- Use the **Name** field to add a descriptive name for the trigger function. The name will be displayed in the **Browser** tree control. Please note that trigger functions will be invoked in alphabetical order.
- Use the drop-down listbox next to **Owner** to select the role that will own the trigger function.
- Select the name of the schema in which the trigger function will reside from the drop-down listbox in the **Schema** field.
- Store notes about the trigger function in the **Comment** field.

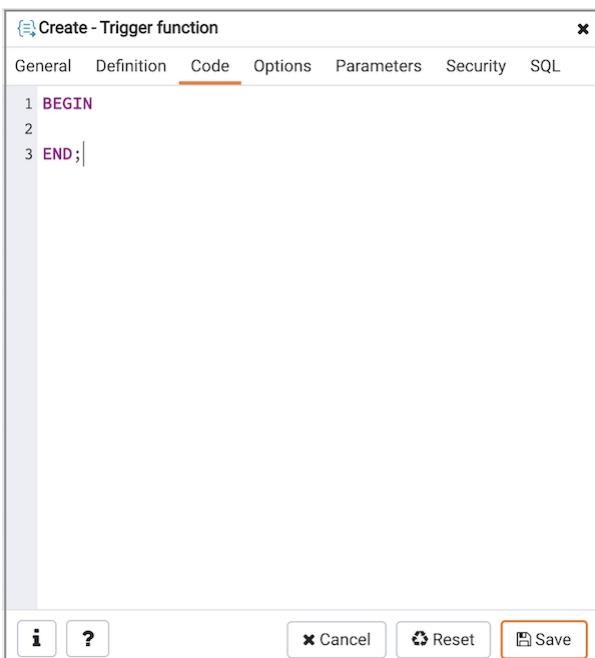
Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define the trigger function:

- Use the drop-down listbox next to **Return type** to specify the pseudotype that is associated with the trigger function:
 - Select **trigger** if you are creating a DML trigger.
 - Select **event_trigger** if you are creating a DDL trigger.
- Use the drop-down listbox next to **Language** to select the implementation language. The default is **plpgsql**.

Click the **Code** tab to continue.



- Use the **Code** field to write the code that will execute when the trigger function is called.

Click the **Options** tab to continue.



Use the fields in the **Options** tab to describe or modify the action of the trigger function:

- Use the drop-down listbox next to **Volatility** to select one of the following:
 - **VOLATILE** indicates that the trigger function value can change even within a single table scan.
 - **STABLE** indicates that the trigger function cannot modify the database, and that within a single table scan it will consistently return the same result for the same argument values.
 - **IMMUTABLE** indicates that the trigger function cannot modify the database and always returns the same result when given the same argument values.
- Move the **Returns a Set?** switch to indicate if the trigger function returns a set that includes multiple rows. The default is **No**.
- Move the **Strict?** switch to indicate if the trigger function always returns NULL whenever any of its arguments are NULL. If **Yes**, the function is not executed when there are NULL arguments; instead a NULL result is assumed automatically. The default is **No**.
- Move the **Security of definer?** switch to specify that the trigger function is to be executed with the privileges of the user that created it. The default is **No**.
- Move the **Window?** switch to indicate that the trigger function is a window function rather than a plain function. The default is **No**. This is currently only useful for trigger functions written in C.
- Use the **Estimated cost** field to specify a positive number representing the estimated execution cost for the trigger function, in units of cpu_operator_cost. If the function returns a set, this is the cost per returned row.
- Use the **Estimated rows** field to specify a positive number giving the estimated number of rows that the query planner should expect the trigger function to return. This is only allowed when the function is declared to return a set. The default assumption is 1000 rows.
- Move the **Leak proof?** switch to indicate whether the trigger function has side effects. The default is **No**. This option can only be set by the superuser.

Click the **Parameters** tab to continue.



Use the fields in the **Parameters** tab to specify settings that will be applied when the trigger function is invoked. Click the **Add** icon (+) to add a **Name**/**Value** pair to the table below.

- Use the drop-down listbox in the **Name** field to select a parameter.
- Use the **Value** field to specify the value that will be associated with the selected parameter. This field is context-sensitive.

Click the **Add** icon (+) to set additional parameters; to discard a parameter, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **Security** tab to continue.



Use the **Security** tab to assign privileges and define security labels.

Use the **Privileges** panel to assign usage privileges for the trigger function to a role. Click the **Add** icon (+) to add a role to the table.

- Select the name of the role from the drop-down listbox in the **Grantee** field.
- Click inside the **Privileges** field. Check the boxes to the left of one or more privileges to grant the selected privilege to the specified user.
- The current user, who is the default grantor for granting the privilege, is displayed in the **Grantor** field.

Click the **Add** icon (+) to assign additional privileges; to discard a privilege, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Use the **Security Labels** panel to define security labels applied to the trigger function. Click the **Add** icon (+) to add each security label selection:

- Specify a security label provider in the **Provider** field. The named provider must be loaded and must consent to the proposed labeling operation.
- Specify a security label in the **Security Label** field. The meaning of a given label is at the discretion of the label provider. PostgreSQL places no restrictions on whether or how a label provider must interpret security labels; it merely provides a mechanism for storing them.

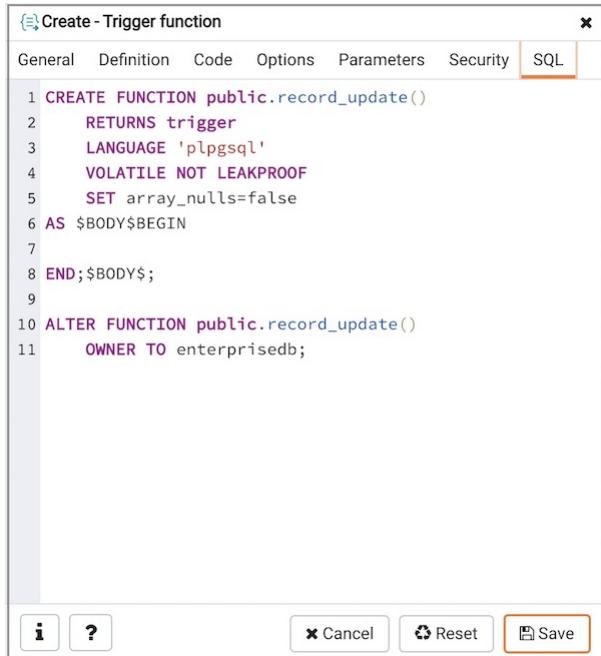
Click the **Add** icon (+) to assign additional security labels; to discard a security label, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **Trigger function** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit other tabs to modify the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Trigger function** dialog:



```

CREATE FUNCTION public.record_update()
RETURNS trigger
LANGUAGE 'plpgsql'
VOLATILE NOT LEAKPROOF
SET array_nulls=false
AS $BODY$BEGIN
END;$BODY$;
ALTER FUNCTION public.record_update()
OWNER TO enterprisedb;

```

The example shown demonstrates creating a trigger function named **emp_stamp** that checks for a new employee's name, and checks that the employee's salary is a positive value.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.

- Click the **Reset** button to restore configuration parameters.

7.5.11.23 Type Dialog

Use the **Type** dialog to register a custom data type.

The **Type** dialog organizes the development of a data type through the following dialog tabs: **General**, **Definition**, and **Security**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the custom data type:

- Use the **Name** field to add a descriptive name for the type. The name will be displayed in the **Browser** tree control. The type name must be distinct from the name of any existing type, domain, or table in the same schema.
- Use the drop-down listbox next to **Owner** to select the role that will own the type.
- Select the name of the schema in which the type will reside from the drop-down listbox in the **Schema** field.
- Store notes about the type in the **Comments** field.

Click the **Definition** tab to continue.

Select a data type from the drop-down listbox next to **Type** on the **Definition** tab; the panel below changes to display the options appropriate for the selected data type. Use the fields in the panel to define the data type.

There are five data types:

- Composite Type*
- Enumeration Type*
- Range Type*
- External Type* (or *Base Type*)
- Shell Type*

If you select **Composite** in the **Type** field, the **Definition** tab displays the **Composite Type** panel:

The screenshot shows the 'Create - Type' dialog box. The 'Definition' tab is active. Under 'Type', 'Composite' is selected. In the 'Composite Type' section, there is a table with one row. The columns are 'Member Name', 'Type', 'Length/Precision', 'Scale', and 'Collation'. The 'Member Name' column contains 'data_member', the 'Type' column contains '"char"[]', and the 'Collation' column has a dropdown menu. At the bottom right, there are 'Cancel', 'Reset', and 'Save' buttons.

Click the **Add** icon (+) to provide attributes of the type. Fields on the **General** panel are context sensitive and may be disabled.

- Use the **Member Name** field to add an attribute name.
- Use the drop-down listbox in the **Type** field to select a datatype.
- Use the **Length/Precision** field to specify the maximum length of a non-numeric type, or the total count of significant digits in a numeric type.
- Use the **Scale** field to specify the number of digits to the right of the decimal point.
- Use the drop-down listbox in the **Collation** field to select a collation (if applicable).

Click the **Add** icon (+) to define an additional member; click the trash icon to the left of the row to discard a row.

If you select the **Enumeration** in the **Type** field, the **Definition** tab displays the **Enumeration Type** panel:

The screenshot shows the 'Create - Type' dialog box. The 'Definition' tab is active. Under 'Type', 'Enumeration' is selected. In the 'Enumeration type' section, there is a table with one row. The columns are 'Label' and 'data_label'. At the bottom right, there are 'Cancel', 'Reset', and 'Save' buttons.

Click the **Add** icon (+) to provide a label for the type.

- Use the **Label** field to add a label, which must be less than 64 bytes long.

Click the **Add** icon (+) after each selection to create additional labels; to discard a label, click the trash icon to the left of the row.

If you select **External**, the **Definition** tab displays the **External Type** panel:



On the **Required** tab:

- Use the drop-down listbox next to the **Input function** field to add an input_function. The input_function converts the type's external textual representation to the internal representation used by the operators and functions defined for the type.
- Use the drop-down listbox next to the **Output function** field to add an output_function. The output_function converts the type's internal representation used by the operators and functions defined for the type to the type's external textual representation.

On the **Optional-1** tab:

- Use the drop-down listbox next to the optional **Receive Function** field to select a receive_function. The optional receive_function converts the type's external binary representation to the internal representation. If this function is not supplied, the type cannot participate in binary input.
- Use the drop-down listbox next to the optional **Send function** field to select a send_function. The optional send_function converts from the internal representation to the external binary representation. If this function is not supplied, the type cannot participate in binary output.
- Use the drop-down listbox next to the optional **Typmod in function** field tab to select a type_modifier_input_function.
- Use the drop-down listbox next to the optional **Typmod out function** field tab to select a type_modifier_output_function. It is allowed to omit the type_modifier_output_function, in which case the default display format is the stored typmod integer value enclosed in parentheses.
- Use the optional **Internal length** to specify a value for internal representation.
- Move the **Variable?** switch to specify the internal representation is of variable length (VARIABLE). The default is a fixed length positive integer.
- Specify a default value in the optional **Default** field in cases where a column of the data type defaults to something other than the null value. Specify the default with the DEFAULT key word. (A default can be overridden by an explicit DEFAULT clause attached to a particular column.)
- Use the drop-down listbox next to the optional **Analyze function** field to select a function for performing type-specific statistics collection for columns of the data type.
- Use the drop-down listbox next to the optional **Category type** field to help control which implicit cast will be

applied in ambiguous situations.

- Move the **Preferred?** switch to **Yes** to specify the selected category type is preferred. The default is **No**.

On the **Optional-2** tab:

- Use the drop-down listbox next to the optional **Element type** field to specify a data type.
- Use the optional **Delimiter** field to indicate the delimiter to be used between values in the external representation of arrays for this data type. The default delimiter is the comma (,). Note that the delimiter is associated with the array element type, not the array type itself.
- Use the drop-down listbox next to **Alignment type** to specify the storage alignment required for the data type. The allowed values (char, int2, int4, and double) correspond with alignment on 1, 2, 4, or 8 byte boundaries.
- Use the drop-down listbox next to optional **Storage type** to select a strategy for storing data.
- Move the **Passed by value?** switch to **Yes** to override the existing data type value. The default is **No**.
- Move the **Collatable?** switch to **Yes** to specify column definitions and expressions of the type may carry collation information through use of the COLLATE clause. The default is **No**.

If you select **Range** in the **Type** field, the **Definition** tab displays the **Range** panel. Fields on the **Range** panel are context-sensitive and may be disabled.

Create - Type	
General	Definition
Security	SQL
Type	Range
Subtype	anyrange
Subtype operator class	(disabled)
Collation	(disabled)
Canonical function	(disabled)
Subtype diff function	(disabled)

Buttons: i ? Cancel Reset **Save**

- Use the drop-down listbox next to **Sub-type** to select an associated b-tree operator class (to determine the ordering of values for the range type).
- Use the drop-down listbox next to **Sub-type operator class** to use a non-default operator class.
- Use the drop-down listbox next to **Collation** to use a non-default collation in the range's ordering if the sub-type is collatable.
- Use the drop-down listbox next to **Canonical function** to convert range values to a canonical form.
- Use the drop-down listbox next to **Sub-type diff function** to select a user-defined subtype_diff function.

If you select **Shell** in the **Type** field, the **Definition** tab displays the **Shell** panel:



A shell type is a placeholder for a type and has no parameters.

Click the **Security** tab to continue.



Use the **Security** tab to assign privileges and define security labels.

Use the **Privileges** panel to assign privileges for the type; click the **Add** icon (+) to grant privileges:

- Select the name of the role that will be granted privileges on the type from the drop-down listbox in the **Grantee** field.
- Click inside the **Privileges** field. Check the boxes to the left of one or more privileges to grant the selected privilege to the specified user.
- The current user, who is the default grantor for granting the privilege, is displayed in the **Grantor** field.

Click the **Add** icon (+) to assign additional privileges; to discard a privilege, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Use the **Security Labels** panel to define security labels applied to the type. Click the **Add** icon (+) to add each security label selection:

- Specify a security label provider in the **Provider** field. The named provider must be loaded and must consent to the proposed labeling operation.
- Specify a security label in the **Security Label** field. The meaning of a given label is at the discretion of the label provider. PostgreSQL places no restrictions on whether or how a label provider must interpret security labels; it merely provides a mechanism for storing them.

Click the **Add** icon (+) to assign additional security labels; to discard a security label, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **Type** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of a sql command generated by user selections made in the **Type** dialog:

```

CREATE TYPE public.work_order AS ENUM
('data_label');

ALTER TYPE public.work_order
OWNER TO postgres;

GRANT USAGE ON TYPE public.work_order TO PUBLIC;

```

The example shown demonstrates creating a data type named **work_order**. The data type is an enumerated type with three labels: new, open and closed.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.11.24 User Mapping Dialog

Use the **User Mapping** dialog to define a new mapping of a user to a foreign server.

The **User Mapping** dialog organizes the development of a user mapping through the following dialog tabs: **General** and **Options**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the drop-down listbox in the **User** field in the **General** tab to identify the connecting role:

- Select **CURRENT_USER** to use the name of the current role.
- Select **PUBLIC** if no other user-specific mapping is applicable.
- Select a pre-defined role name to specify the name of an existing user.

Click the **Options** tab to continue.



Use the fields in the **Options** tab to specify connection options; the accepted option names and values are specific to the foreign data wrapper associated with the server specified in the user mapping. Click the **Add** button to add an option/value pair.

- Specify the option name in the **Option** field.
- Provide a corresponding value in the **Value** field.

Click **Add** to specify each additional option/value pair; to discard an option, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **User Mapping** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **User Mapping** dialog:



```
CREATE USER MAPPING FOR postgres SERVER myserver
OPTIONS ("user" 'postgres', password 'postgres');
```

The example shown demonstrates a user mapping for the **hdfs_server**. The user is **CURRENT_USER** with a password **secret**.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.11.25 View Dialog

Use the **View** dialog to define a view. The view is not physically materialized; the query is executed each time the view is referenced in a query.

The **View** dialog organizes the development of a View through the following dialog tabs: **General**, **Definition**, **Code** and **Security**. The **SQL** tab displays the SQL code generated by dialog selections.

Click the **General** tab to begin.



Use the fields in the **General** tab to identify a view:

- Use the **Name** field to add a descriptive name for the view. The name of the view must be distinct from the name of any other view, table, sequence, index or foreign table in the same schema. The name will be displayed in the **Browser** tree control.
- Use the drop-down listbox next to **Owner** to select the role that will own the view.
- If applicable, select the name of the schema in which the view will reside from the drop-down listbox in the **Schema** field.
- Store notes about the view in the **Comments** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define properties of the view:

- Set the **Security Barrier** switch to **Yes** to indicate that the view is to act as a security barrier. For more

information about defining and using a security barrier rule, see Section 38.5 of the PostgreSQL documentation.

- Use the drop-down listbox next to **Check options** to select from **No**, **Local** or **Cascaded**:
 - The **Local** option specifies that new rows are only checked against the conditions defined in the view. Any conditions defined on underlying base views are not checked (unless you specify the CHECK OPTION).
 - The **Cascaded** option specifies new rows are checked against the conditions of the view and all underlying base views.

Click the **Code** tab to continue.



Use the workspace in the **Code** tab to write a query to create a view.

Click the **Security** tab to continue.



Use the **Security** tab to assign privileges and define security labels.

Use the **Privileges** panel to assign privileges to a role. Click the **Add** icon (+) to set privileges for the view:

- Select the name of the role that will be granted privileges from the drop-down listbox in the **Grantee** field.
- Click inside the **Privileges** field. Check the boxes to the left of one or more privileges to grant the selected privilege to the specified user.
- The current user, who is the default grantor for granting the privilege, is displayed in the **Grantor** field.

Click the **Add** icon (+) to assign additional privileges; to discard a privilege, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Use the **Security Labels** panel to define security labels applied to the view. Click the **Add** icon (+) to add each security label selection:

- Specify a security label provider in the **Provider** field. The named provider must be loaded and must consent to the proposed labeling operation.
- Specify a security label in the **Security Label** field. The meaning of a given label is at the discretion of the label provider. PostgreSQL places no restrictions on whether or how a label provider must interpret security labels; it merely provides a mechanism for storing them.

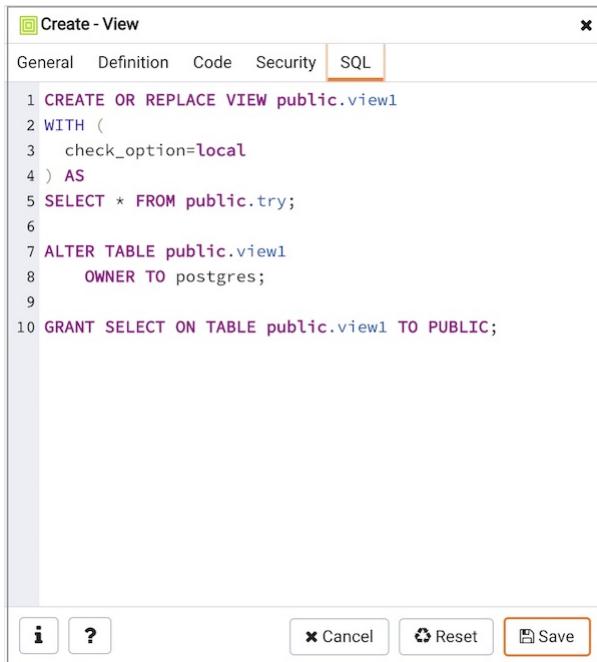
Click the **Add** icon (+) to assign additional security labels; to discard a security label, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **View** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **View** dialog:



```

CREATE OR REPLACE VIEW public.view1
WITH (
check_option=local
)
AS
SELECT * FROM public.try;

ALTER TABLE public.view1
OWNER TO postgres;

GRANT SELECT ON TABLE public.view1 TO PUBLIC;

```

The example shown demonstrates creating a view named **distributor_codes** that includes the content of the **code** column from the **distributors** table.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.

- Click the **Reset** button to restore configuration parameters.

7.5.12 Creating or Modifying a Table

PEM provides dialogs that allow you to modify all table properties and attributes.

To access a dialog that allows you to create a database object, right-click on the object type in the **Browser** tree control, and select the **Create** option for that object. For example, to create a new table, Select a database from the tree control, select the schema under the database, right-click on the *Tables* node, and select *Create Table...*

Contents:

7.5.12.1 Check Dialog

Use the **Check** dialog to define or modify a check constraint. A check constraint specifies an expression that produces a Boolean result that new or updated rows must satisfy for an insert or update operation to succeed.

The **Check** dialog organizes the development of a check constraint through the **General** and **Definition** tabs. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the check constraint:

- Use the **Name** field to provide a descriptive name for the check constraint that will be displayed in the **Browser** tree control. With PostgreSQL 9.5 forward, when a table has multiple check constraints, they will be tested for each row in alphabetical order by name and after NOT NULL constraints.
- Store notes about the check constraint in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define the check constraint:

- Provide the expression that a row must satisfy in the **Check** field.
- Move the **No Inherit?** switch to the **Yes** position to specify this constraint is automatically inherited by a table's children. The default is **No**.
- Move the **Don't validate?** switch to the **No** position to skip validation of existing data; the constraint may not hold for all rows in the table. The default is **Yes**.

Click the **SQL** tab to continue.

Your entries in the **Check** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Check** dialog:



The example shown demonstrates creating a check constraint named `check_price` on the `price` column of the `products` table. The constraint confirms that any values added to the column are greater than 0.

- Click the `Info` button (i) to access online help. View context-sensitive help in the `Tabbed browser`, where a new tab displays the PostgreSQL core documentation.
- Click the `Save` button to save work.
- Click the `Cancel` button to exit without saving work.
- Click the `Reset` button to restore configuration parameters.

7.5.12.2 Column Dialog

Use the `Column` dialog to add a column to an existing table or modify a column definition.

The `Column` dialog organizes the development of a column through the following dialog tabs: `General`, `Definition`, and `Security`. The `SQL` tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the column:

- Use the **Name** field to add a descriptive name for the column. The name will be displayed in the **Browser** tree control. This field is required.
- Store notes about the column in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to add parameters for the column. (Fields are disabled if inapplicable.)

- Use the drop-down listbox next to **Data Type** to select a data type for the column. For more information on the data types that are supported by PostgreSQL, refer to Chapter 8 of the Postgres core documentation. This field is required.
- Use the **Length/Precision** and **Scale** fields to specify the maximum number of significant digits in a numeric value, or the maximum number of characters in a text value.
- Use the drop-down listbox next to **Collation** to apply a collation setting to the column.

Click the **Constraints** tab to continue.

The screenshot shows the 'Create - Column' dialog box with the 'Constraints' tab selected. The 'Type' field is set to 'NONE'. Other fields like 'Default' and 'Not NULL?' are also visible.

Use the fields in the **Constraints** tab to specify constraints for the column. (Fields are disabled if inapplicable.)

- Use the **Default Value** field to specify a default data value.
- Move the **Not Null** switch to the **Yes** position to specify the column may not contain null values. The default is **No**.
- Use the **Type** field to specify the column type (NONE/IDENTITY/GENERATED). The default is **NONE**.

Click the **IDENTITY** type to create Identity column.

The screenshot shows the 'Create - Column' dialog box with the 'Constraints' tab selected. The 'Type' field is set to 'IDENTITY' and the 'Identity' dropdown is set to 'ALWAYS'. Other fields like 'Increment', 'Start', 'Minimum', 'Maximum', 'Cache', and 'Cycled' are also visible.

Use the following fields to create **IDENTITY** column. Identity columns are applicable for PG/EPAS version 10 and above.

- Use the **Identity** field to specify ALWAYS or BY DEFAULT. This clause is used to determine how the sequence value is given precedence over a user-specified value in an INSERT statement.
- Use the **Increment** field to specify which value is added to the current sequence value to create a new value.

- Provide a value in the **Start** field to specify the beginning value of the sequence. The default starting value is MINVALUE for ascending sequences and MAXVALUE for descending ones.
- Provide a value in the **Minimum** field to specify the minimum value a sequence can generate. If this clause is not supplied or NO MINVALUE is specified, then defaults will be used. The defaults are 1 and -263-1 for ascending and descending sequences, respectively.
- Provide a value in the **Maximum** field to specify the maximum value for the sequence. If this clause is not supplied or NO MAXVALUE is specified, then default values will be used. The defaults are 263-1 and -1 for ascending and descending sequences, respectively.
- Provide a value in the **Cache** field to specify how many sequence numbers are to be preallocated and stored in memory for faster access. The minimum value is 1 (only one value can be generated at a time, i.e., no cache), and this is also the default.
- Move the **Cycled** switch to the **Yes** position to allow the sequence to wrap around when the MAXVALUE or the MINVALUE has been reached by an ascending or descending sequence respectively. If the limit is reached, the next number generated will be the MINVALUE or MAXVALUE, respectively. The default is **No**.

Click the **GENERATED** type to create Generated column.



Use the following fields to create **GENERATED** column. Generated columns are applicable for PG/EPAS version 12 and above.

- Use the **Expression** field to specify the generation expression. It can refer to other columns in the table, but not other generated columns. Any functions and operators used must be immutable. References to other tables are not allowed.

Click the **Variables** tab to continue.



Use the **Variables** tab to specify the number of distinct values that may be present in the column; this value overrides estimates made by the ANALYZE command. Click the **Add** icon (+) to add a **Name / Value** pair:

- Select the name of the variable from the drop-down listbox in the **Name** field.
 - Select **n_distinct** to specify the number of distinct values for the column.
 - Select **n_distinct_inherited** to specify the number of distinct values for the table and its children.
- Specify the number of distinct values in the **Value** field. For more information, see the documentation for [ALTER TABLE](#).

Click the **Add** icon (+) to specify each additional **Name / Value** pair; to discard a variable, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **Security** tab to continue.



Use the **Security** tab to assign attributes and define security labels. Click the **Add** icon (+) to add each security label

selection:

- Specify a security label provider in the **Provider** field. The named provider must be loaded and must consent to the proposed labeling operation.
- Specify a security label in the **Security Label** field. The meaning of a given label is at the discretion of the label provider. PostgreSQL places no restrictions on whether or how a label provider must interpret security labels; it merely provides a mechanism for storing them.

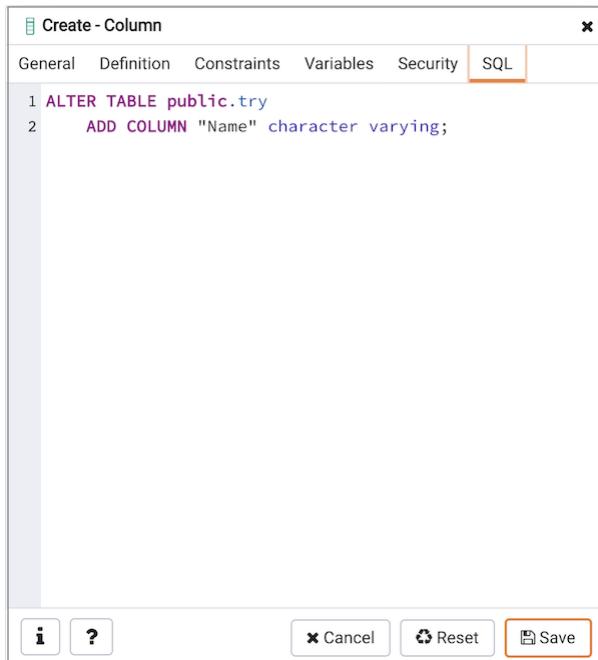
Click the **Add** icon (+) to assign additional security labels; to discard a security label, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **Column** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Column** dialog:



```
1 ALTER TABLE public.try
2     ADD COLUMN "Name" character varying;
```

The example shown demonstrates creating a column named **territory** in the table named **distributors**.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.12.3 Compound Trigger Dialog

Use the **Compound Trigger** dialog to create a compound trigger or modify an existing compound trigger. **Compound**

Trigger is supported only for EPAS server 12 and above. A compound trigger executes a specified code when certain events occur.

The **Compound Trigger** dialog organizes the development of a compound trigger through the following dialog tabs: **General**, **Events**, and **Code**. The SQL tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the compound trigger:

- Use the **Name** field to add a descriptive name for the compound trigger. This must be distinct from the name of any other compound trigger for the same table. The name will be displayed in the **Browser** tree control.
- Store notes about the compound trigger in the **Comment** field.



- **Trigger enabled** field is available in compound trigger dialog once the trigger is created by selecting the **properties** of the trigger. You can select one of the four options available.

Click the **Events** tab to continue.



Use the fields in the **Events** tab to specify how and when the compound trigger fires:

- Select the type of event(s) that will invoke the compound trigger; to select an event type, move the switch next to the event to the **YES** position. The supported event types are **INSERT**, **UPDATE**, **DELETE** and **TRUNCATE**. Views cannot have TRUNCATE triggers.
- Use the **When** field to provide a boolean condition that will invoke the compound trigger.
- If defining a column-specific compound trigger, use the **Columns** field to specify the columns or columns that are the target of the compound trigger.

Click the **Code** tab to continue.



Use the **Code** field to specify the code for the five timing events **BEFORE STATEMENT**, **AFTER STATEMENT**, **BEFORE EACH ROW**, **AFTER EACH ROW**, **INSTEAD OF EACH ROW** that will be invoked when the compound trigger fires. Basic template is provided with place holders.

Click the **SQL** tab to continue.

Your entries in the **Compound Trigger** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Compound Trigger** dialog:



```

CREATE OR REPLACE TRIGGER hr_ct
FOR INSERT OR DELETE OR UPDATE
ON edbuser.employees
COMPOUND TRIGGER
5 var_sal NUMBER := 10000;
6
7 BEFORE STATEMENT IS
8 BEGIN
9     var_sal := var_sal + 1000;
10    DBMS_OUTPUT.PUT_LINE('Before Statement: ' || var_sal);
11 END;
12
13 BEFORE EACH ROW IS
14 BEGIN
15     var_sal := var_sal + 1000;
16    DBMS_OUTPUT.PUT_LINE('Before Each Row: ' || var_sal);
17 END BEFORE EACH ROW;
18
19 AFTER EACH ROW IS
20 BEGIN
21     var_sal := var_sal + 1000;
22    DBMS_OUTPUT.PUT_LINE('After Each Row: ' || var_sal);
23 END AFTER EACH ROW;
24
25 AFTER STATEMENT IS
26 BEGIN
27     var_sal := var_sal + 1000;
28    DBMS_OUTPUT.PUT_LINE('After Statement: ' || var_sal);
29 END AFTER STATEMENT;
30
31 END hr_ct;
32
33 COMMENT ON TRIGGER hr_ct ON edbuser.employees
34 IS 'This is a compound trigger';

```

The screenshot shows the 'Create - Compound Trigger' dialog window. The title bar says 'Create - Compound Trigger'. Below it is a tab bar with 'General', 'Events', 'Code', and 'SQL', where 'SQL' is selected. The main area contains the generated SQL code for a compound trigger named 'hr_ct'. The code includes logic for BEFORE STATEMENT, BEFORE EACH ROW, AFTER EACH ROW, and AFTER STATEMENT triggers, all incrementing a variable 'var_sal' by 1000 and printing its value using DBMS_OUTPUT.PUT_LINE. At the bottom of the dialog are buttons for 'Info' (i), 'Cancel', 'Reset', and 'Save' (which is highlighted in orange).

The example demonstrates creating a compound trigger named **test_ct**.

- Click the **Info** button (i) to access online help.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.12.4 Exclusion constraint Dialog

Use the **Exclusion constraint** dialog to define or modify the behavior of an exclusion constraint. An exclusion constraint guarantees that if any two rows are compared on the specified column or expression (using the specified operator), at least one of the operator comparisons will return false or null.

The **Exclusion constraint** dialog organizes the development of an exclusion constraint through the following dialog tabs: **General**, **Definition**, and **Columns**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the exclusion constraint:

- Use the **Name** field to provide a descriptive name for the exclusion constraint. The name will be displayed in the **Browser** tree control.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define the exclusion constraint:

- Use the drop-down listbox next to **Tablespace** to select the tablespace in which the index associated with the exclude constraint will reside.
- Use the drop-down listbox next to **Access method** to specify the type of index that will be used when implementing the exclusion constraint:
 - Select **gist** to specify a GiST index.
 - Select **spgist** to specify a space-partitioned GiST index.

- Select **btree** to specify a B-tree index.
- Select **hash** to specify a hash index.
- Use the **Fill Factor** field to specify a fill factor for the table and associated index. The fill factor is a percentage between 10 and 100. 100 (complete packing) is the default.
- Move the **Deferrable?** switch to the **Yes** position to specify that the timing of the constraint is deferrable, and can be postponed until the end of the statement. The default is **No**.
- If enabled, move the **Deferred?** switch to the **Yes** position to specify the timing of the constraint is deferred to the end of the statement. The default is **No**.
- Use the **Constraint** field to provide a condition that a row must satisfy to be included in the table.

Click the **Columns** tab to continue.



Use the fields in the *Columns* tab to specify the column(s) or expression(s) to which the constraint applies. Use the */s expression ?* switch to enable expression text input. Use the drop-down listbox next to *Column* to select a column. Once the *Column* is selected or the *Expression* is entered then click the *Add* icon (+) to provide details of the action on the column/expression:

- The *Col/Exp* field is populated with the selection made in the *Column* drop-down listbox or the *Expression* entered.
- If applicable, use the drop-down listbox in the *Operator class* to specify the operator class that will be used by the index for the column.
- Move the **DESC** switch to **DESC** to specify a descending sort order. The default is **ASC** which specifies an ascending sort order.
- Use the **NULLS order** column to specify the placement of NULL values (when sorted). Specify **FIRST** or **LAST**.
- Use the drop-down list next to *Operator* to specify a comparison or conditional operator.

Use **Include columns** field to specify columns for **INCLUDE** clause of the constraint. This option is available in Postgres 11 and later.

Click the **SQL** tab to continue.

Your entries in the **Exclusion Constraint** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Exclusion Constraint** dialog:



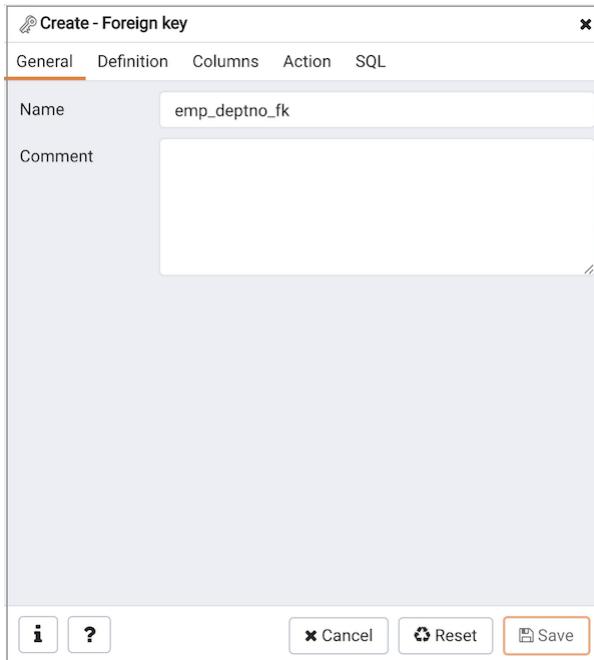
The example shown demonstrates creating an exclusion constraint named `exclude_department` that restricts additions to the dept table to those additions that are not equal to the value of the `deptno` column. The constraint uses a btree index.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.12.5 Foreign key Dialog

Use the **Foreign key** dialog to specify the behavior of a foreign key constraint. A foreign key constraint maintains referential integrity between two tables. A foreign key constraint cannot be defined between a temporary table and a permanent table.

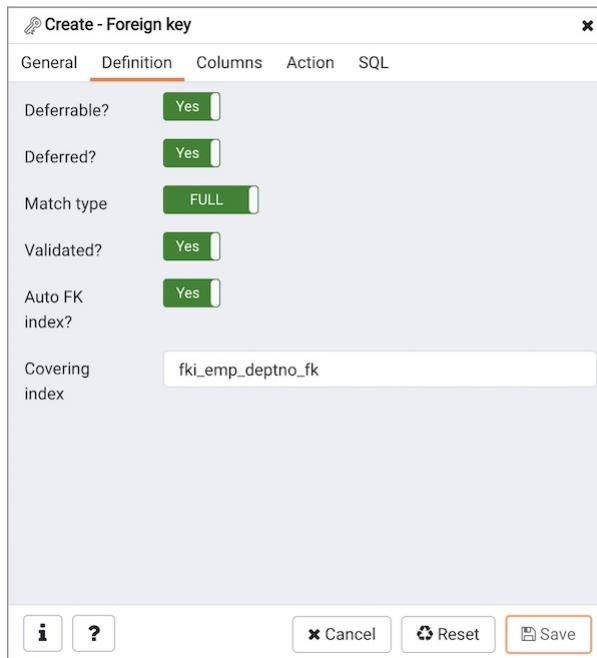
The **Foreign key** dialog organizes the development of a foreign key constraint through the following dialog tabs: **General**, **Definition**, **Columns**, and **Action**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the foreign key constraint:

- Use the **Name** field to add a descriptive name for the foreign key. The name will be displayed in the **Browser** tree control.
- Store notes about the foreign key constraint in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define the foreign key constraint:

- Move the **Deferrable?** switch to the **Yes** position to specify the timing of the constraint is deferrable and can be postponed until the end of the statement. The default is **No**.
- If enabled, move the **Deferred?** switch to the **Yes** position to specify the timing of the constraint is deferred to the end of the statement. The default is **No**.
- Move the **Match type** switch to specify the type of matching that is enforced by the constraint:

- Select **Full** to indicate that all columns of a multicolumn foreign key must be null if any column is null; if all columns are null, the row is not required to have a match in the referenced table.
- Select **Simple** to specify that a single foreign key column may be null; if any column is null, the row is not required to have a match in the referenced table.
- Move the **Validated** switch to the **Yes** position to instruct the server to validate the existing table content (against a foreign key or check constraint) when you save modifications to this dialog.
- Move the **Auto FK Index** switch to the **No** position to disable the automatic index feature.
- The field next to **Covering Index** generates the name of an index if the **Auto FK Index** switch is in the **Yes** position; or, this field is disabled.

Click the **Columns** tab to continue.



Use the fields in the **Columns** tab to specify one or more reference column(s). A Foreign Key constraint requires that one or more columns of a table must only contain values that match values in the referenced column(s) of a row of a referenced table:

- Use the drop-down listbox next to **Local column** to specify the column in the current table that will be compared to the foreign table.
- Use the drop-down listbox next to **References** to specify the name of the table in which the comparison column(s) resides.
- Use the drop-down listbox next to **Referencing** to specify a column in the foreign table.

Click the **Add** icon (+) to add a column to the list; repeat the steps above and click the **Add** icon (+) to add additional columns. To discard an entry, click the trash icon to the left of the entry and confirm deletion in the **Delete Row** popup.

Click the **Action** tab to continue.



Use the drop-down listboxes on the **Action** tab to specify behavior related to the foreign key constraint that will be performed when data within the table is updated or deleted:

- Use the drop-down listbox next to **On update** to select an action that will be performed when data in the table is updated.
- Use the drop-down listbox next to **On delete** to select an action that will be performed when data in the table is deleted.

The supported actions are:

NO ACTION	Produce an error indicating that the deletion or update will create a foreign key constraint violation. If the constraint is deferred, this error will be produced at constraint check time if any referencing rows still exist. This is the default.
RESTRICT	Throw an error indicating that the deletion or update would create a foreign key constraint violation. This is the same as NO ACTION except that the check is not deferrable.
CASCADE	Delete any rows referencing the deleted row, or update the values of the referencing column(s) to the new values of the referenced columns, respectively.
SET NULL	Set the referencing column(s) to null.
SET DEFAULT	Set the referencing column(s) to their default values. There must be a row in the referenced table that matches the default values (if they are not null), or the operation will fail.

Click the **SQL** tab to continue.

Your entries in the **Foreign key** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Foreign key** dialog:



The example shown demonstrates creating a foreign key constraint named `territory_fkey` that matches values in the `distributors` table `territory` column with those of the `sales_territories` table `region` column.

- Click the `Info` button (i) to access online help. View context-sensitive help in the `Tabbed browser`, where a new tab displays the PostgreSQL core documentation.
- Click the `Save` button to save work.
- Click the `Cancel` button to exit without saving work.
- Click the `Reset` button to restore configuration parameters.

7.5.12.6 Index Dialog

Use the `Index` dialog to create an index on a specified table or materialized view.

The `Index` dialog organizes the development of an index through the following dialog tabs: `General` and `Definition`. The `SQL` tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the index:

- Use the **Name** field to add a descriptive name for the index. The name will be displayed in the **Browser** tree control.
- Use the drop-down listbox next to **Tablespace** to select the tablespace in which the index will reside.
- Store notes about the index in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define the index:

- Use the drop-down listbox next to **Access Method** to select an index type:
 - Select **btree** to create a B-tree index. A B-tree index may improve performance when managing equality and range queries on data that can be sorted into some ordering (the default).
 - Select **hash** to create a hash index. A hash index may improve performance when managing simple equality comparisons.
 - Select **gist** to create a GiST index. A GiST index may improve performance when managing two-dimensional geometric data types and nearest-neighbor searches
 - Select **gin** to create a GIN index. A GIN index may perform well when managing values with more than one key.

- Select **spgist** to create a space-partitioned GiST index. A SP-GiST index may improve performance when managing non-balanced data structures.
- Select **brin** to create a BRIN index. A BRIN index may improve performance when managing minimum and maximum values and ranges.
- Use the **Fill Factor** field to specify a fill factor for the index. The fill factor specifies how full the selected method will try to fill each index page.
- Move the **Unique?** switch to the **Yes** position to check for duplicate values in the table when the index is created and when data is added. The default is **No**.
- Move the **Clustered?** switch to the **Yes** position to instruct the server to cluster the table.
- Move the **Concurrent build?** switch to the **Yes** position to build the index without taking any locks that prevent concurrent inserts, updates, or deletes on the table.
- Use the **Constraint** field to provide a constraint expression; a constraint expression limits the entries in the index to those rows that satisfy the constraint.

Use the context-sensitive fields in the **Columns** panel to specify which column(s) the index queries. Click the **Add** icon (+) to add a column:

- Use the drop-down listbox in **Column** field to select the name of the column from the table.
- If enabled, use the drop-down listbox to select an available **Operator class** to specify the type of action performed on the column.
- If enabled, move the **Sort order** switch to specify the sort order:
 - Select **ASC** to specify an ascending sort order (the default);
 - Select **DESC** to specify a descending sort order.
- If enabled, move the **Nulls** switch to specify the sort order of nulls:
 - Select **First** to specify nulls sort before non-nulls;
 - Select **Last** to specify nulls sort after non-nulls (the default).
- Use the drop-down listbox in the **Collation** field to select a collation to use for the index.

Use **Include columns** field to specify columns for **INCLUDE** clause of the index. This option is available in Postgres 11 and later.

Click the **SQL** tab to continue.

Your entries in the **Index** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Index** dialog:



The example shown demonstrates creating an index named `dist_codes` that indexes the values in the `code` column of the `distributors` table.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.12.7 Primary key Dialog

Use the **Primary key** dialog to create or modify a primary key constraint. A primary key constraint indicates that a column, or group of columns, uniquely identifies rows in a table. This requires that the values in the selected column(s) be both unique and not null.

The **Primary key** dialog organizes the development of a primary key constraint through the **General** and **Definition** tabs. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the primary key:

- Use the **Name** field to add a descriptive name for the primary key constraint. The name will be displayed in the **Browser** tree control.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define the primary key constraint:

- Click inside the **Columns** field and select one or more column names from the drop-down listbox. To delete a selection, click the **x** to the left of the column name. The primary key constraint should be different from any unique constraint defined for the same table; the selected column(s) for the constraints must be distinct.
- Use **Include columns** field to specify columns for **INCLUDE** clause of the index. This option is available in Postgres 11 and later.
- Select the name of the tablespace in which the primary key constraint will reside from the drop-down listbox in the **Tablespace** field.
- Select the name of an index from the drop-down listbox in the **Index** field. This field is optional. Adding a primary

key will automatically create a unique B-tree index on the column or group of columns listed in the primary key, and will force the column(s) to be marked NOT NULL.

- Use the **Fill Factor** field to specify a fill factor for the table and index. The fill factor for a table is a percentage between 10 and 100. 100 (complete packing) is the default.
- Move the **Deferrable?** switch to the **Yes** position to specify the timing of the constraint is deferrable and can be postponed until the end of the statement. The default is **No**.
- If enabled, move the **Deferred?** switch to the **Yes** position to specify the timing of the constraint is deferred to the end of the statement. The default is **No**.

Click the **SQL** tab to continue.

Your entries in the **Primary key** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Primary key** dialog:

```

Create - Primary key
General Definition SQL
1 ALTER TABLE edbuser.employees
2   ADD CONSTRAINT emp_empno_pk PRIMARY KEY (empno);

Cancel Reset Save

```

The example shown demonstrates creating a primary key constraint named **dept_pkey** on the **dept_id** column of the **dept** table.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.12.8 RLS Policy Dialog

Use the **RLS Policy** dialog to Create a Row Level Security Policy.

Note

If the Row Level Security is enabled at table level and no policy is created then by default **Deny Policy** is applied. That means, no rows are visible or can be modified for that table.

The **RLS Policy** dialog creates a Row Level Security Policy through the following dialog tabs: **General**, and **Commands**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to define the RLS Policy:

- Use the **Name** field to add a descriptive name for the RLS Policy. The name will be displayed in the pgAdmin tree control.
- Use the drop-down listbox next to **Role** to select the Role to which the RLS Policy is to be applied.
- Use the drop-down listbox next to **Type** to select the type of the policy.

Click the **Commands** tab to continue.



Use the fields in the **Commands** tab to define the RLS Policy:

- Use the drop-down listbox next to **Event** to select the command to which policy applies. Valid options are ALL, SELECT, INSERT, UPDATE, and DELETE. Default is ALL.
- Use the **Using** field to add a SQL conditional expression returning boolean. This expression will be added to queries that refer to the table if row level security is enabled.
- Use the **With check** field to add a SQL conditional expression returning boolean. This expression will be used in INSERT and UPDATE queries against the table if row level security is enabled.

Click the **SQL** tab to continue.

Your entries in the **RLS Policy** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **RLS Policy** dialog:

```

CREATE POLICY test
ON pem.agent
AS PERMISSIVE
FOR ALL
TO public;

```

The example shown demonstrates creating a RLS Policy named `account_managers` that applies the Row Level Security on the `accounts` table.

- Click the **Info** button (i) to access online help.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.12.9 Rule Dialog

Use the **Rule** dialog to define or modify a rule for a specified table or view. A PostgreSQL rule allows you to define an additional action that will be performed when a SELECT, INSERT, UPDATE, or DELETE is performed against a table.

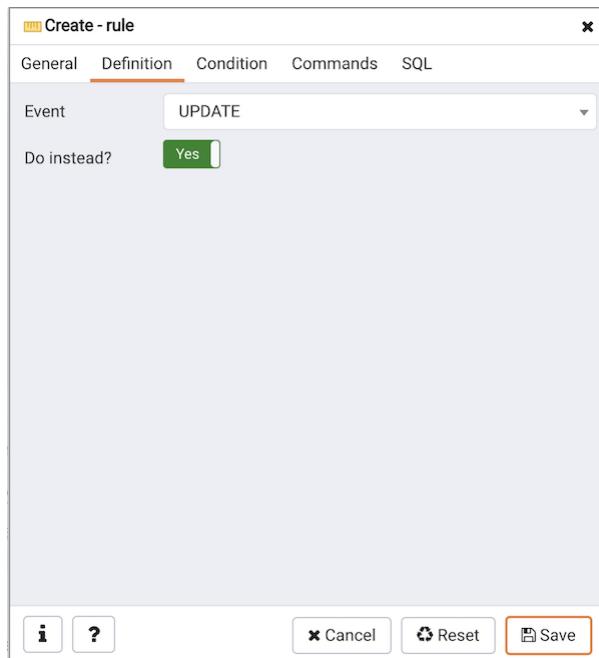
The **Rule** dialog organizes the development of a rule through the **General**, **Definition**, **Condition**, **Commands** tabs. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the rule:

- Use the **Name** field to add a descriptive name for the rule. The name will be displayed in the **Browser** tree control. Multiple rules on the same table are applied in alphabetical name order.
- Store notes about the rule in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to write parameters:

- Click inside the **Event** field to select the type of event that will invoke the rule; event may be **Select**, **Insert**, **Update**, or **Delete**.
- Move the **Do Instead** switch to **Yes** indicate that the commands should be executed instead of the original command; if Do Instead specifies **No**, the rule will be invoked in addition to the original command.

Click the **Condition** tab to continue.



Specify a SQL conditional expression that returns a boolean value in the editor.

Click the **Commands** tab to continue.



Provide a command in the editor that defines the action performed by the rule.

Click the **SQL** tab to continue.

Your entries in the **Rule** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Rule** dialog:



The example sends a notification when an UPDATE executes against a table.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.12.10 Table Dialog

Use the **Table** dialog to create or modify a table.

The **Table** dialog organizes the development of a table through the following dialog tabs: **General**, **Columns**, **Constraints**, **Advanced**, **Parameter**, and **Security**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the table:

- Use the **Name** field to add a descriptive name for the table. A table cannot have the same name as any existing table, sequence, index, view, foreign table, or data type in the same schema. The name specified will be displayed in the **Browser** tree control. This field is required.
- Select the owner of the table from the drop-down listbox in the **Owner** field. By default, the owner of the table is the role that creates the table.
- Select the name of the schema in which the table will reside from the drop-down listbox in the **Schema** field.
- Use the drop-down listbox in the **Tablespace** field to specify the tablespace in which the table will be stored.
- Move the **Partitioned Table?** switch to the **Yes** in case you want to create a partitioned table. Option is available for PostgreSQL 10 and above.
- Store notes about the table in the **Comment** field.

Click the **Columns** tab to continue.

Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?
empno	numeric			<input checked="" type="checkbox"/> No	<input checked="" type="checkbox"/> No
ename	character varying			<input checked="" type="checkbox"/> No	<input checked="" type="checkbox"/> No
job	character varying			<input checked="" type="checkbox"/> No	<input checked="" type="checkbox"/> No
sal	numeric			<input checked="" type="checkbox"/> No	<input checked="" type="checkbox"/> No
mgr	numeric			<input checked="" type="checkbox"/> No	<input checked="" type="checkbox"/> No
email	character varying			<input checked="" type="checkbox"/> No	<input checked="" type="checkbox"/> No
hiredate	date			<input checked="" type="checkbox"/> No	<input checked="" type="checkbox"/> No
deptno	numeric			<input checked="" type="checkbox"/> No	<input checked="" type="checkbox"/> No

Use the drop-down listbox next to **Inherited from table(s)** to specify any parent table(s); the table will inherit columns from the selected parent table(s). Click inside the **Inherited from table(s)** field to select a table name from a drop-down list. Repeat to add any other parent tables. Delete a selected table by clicking the **x** to the left of the parent name. Note that inherited column names and datatypes are not editable in the current dialog; they must be modified at the parent level.

Click the **Add** icon (+) to specify the names of columns and their datatypes in the **Columns** table:

- Use the **Name** field to add a descriptive name for the column.
- Use the drop-down listbox in the **Data type** field to select a data type for the column. This can include array specifiers. For more information on the data types supported by PostgreSQL, refer to Chapter 8 of the core documentation.
- If enabled, use the **Length** and **Precision** fields to specify the maximum number of significant digits in a numeric value, or the maximum number of characters in a text value.
- Move the **Not NULL?** switch to the **Yes** position to require a value in the column field.
- Move the **Primary key?** switch to the **Yes** position to specify the column is the primary key constraint.

Click the **Add** icon (+) to add additional columns; to discard a column, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **Constraints** tab to continue.



Use the fields in the **Constraints** tab to provide a table or column constraint. Optional constraint clauses specify constraints (tests) that new or updated rows must satisfy for an **INSERT** or **UPDATE** operation to succeed. Select the appropriate constraint type by selecting one of the following tabs on the **Constraints** panel:

Tab Name Constraint

Primary Key	Provides a unique identifier for each row in the table.
Foreign Key	Maintains referential integrity between two tables.
Check	Requires data satisfies an expression or condition before insertion or modification.
Unique	Ensures that the data contained in a column, or a group of columns, is unique among all the rows in the table.
Exclude	Guarantees that if any two rows are compared on the specified column or expression (using the specified operator), at least one of the operator comparisons will return false or null.

To add a primary key for the table, select the **Primary Key** tab, and click the **Add** icon (+). To define the primary key, click the **Edit** icon to the left of the **Trash** icon. A dialog similar to the **Primary key** dialog (accessed by right clicking on **Constraints** in the **Browser** tree control) opens.

Use the fields in the **General** tab to identify the primary key:

- Use the **Name** field to add a descriptive name for the primary key constraint. The name will be displayed in the **Browser** tree control.
- Provide notes about the primary key in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define the primary key constraint:

- Click inside the **Columns** field and select one or more column names from the drop-down listbox. To delete a selection, click the **x** to the left of the column name. The primary key constraint should be different from any unique constraint defined for the same table; the selected column(s) for the constraints must be distinct.
- Select the name of the tablespace in which the primary key constraint will reside from the drop-down listbox in the **Tablespace** field.
- Use the **Fill Factor** field to specify a fill factor for the table and index. The fill factor for a table is a percentage between 10 and 100. 100 (complete packing) is the default.
- Move the **Deferrable?** switch to the **Yes** position to specify the timing of the constraint is deferrable and can be postponed until the end of the statement. The default is **No**.
- If enabled, move the **Deferred?** switch to the **Yes** position to specify the timing of the constraint is deferred to the end of the statement. The default is **No**.



To add a foreign key constraint, select the **Foreign Key** tab, and click the **Add** icon (+). To define the constraint, click the **Edit** icon to the left of the **Trash** icon. A dialog similar to the **Foreign key** dialog (accessed by right clicking on **Constraints** in the **Browser** tree control) opens.

Use the fields in the **General** tab to identify the foreign key constraint:

- Use the **Name** field to add a descriptive name for the foreign key constraint. The name will be displayed in the **Browser** tree control.
- Provide notes about the foreign key in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define the foreign key constraint:

- Move the **Deferrable?** switch to the **Yes** position to specify the timing of the constraint is deferrable and can be postponed until the end of the statement. The default is **No**.
- If enabled, move the **Deferred?** switch to the **Yes** position to specify the timing of the constraint is deferred to the end of the statement. The default is **No**.
- Move the **Match type** switch specify the type of matching that is enforced by the constraint:
 - Select **Full** to indicate that all columns of a multicolumn foreign key must be null if any column is null; if all columns are null, the row is not required to have a match in the referenced table.
 - Select **Simple** to specify that a single foreign key column may be null; if any column is null, the row is not required to have a match in the referenced table.
- Move the **Validated** switch to the **Yes** position to instruct the server to validate the existing table content (against a foreign key or check constraint) when you save modifications to this dialog.
- Move the **Auto FK Index** switch to the **No** position to disable the automatic index feature.
- The field next to **Covering Index** generates the name of an index if the **Auto FK Index** switch is in the **Yes** position; or, this field is disabled.

Click the **Columns** tab to continue.



Use the fields in the **Columns** tab to specify one or more reference column(s). A Foreign Key constraint requires that

one or more columns of a table must only contain values that match values in the referenced column(s) of a row of a referenced table:

- Use the drop-down listbox next to **Local column** to specify the column in the current table that will be compared to the foreign table.
- Use the drop-down listbox next to **References** to specify the name of the table in which the comparison column(s) resides.
- Use the drop-down listbox next to **Referencing** to specify a column in the foreign table.

Click the **Add** icon (+) to add a column to the list; repeat the steps above and click the **Add** icon (+) to add additional columns. To discard an entry, click the trash icon to the left of the entry and confirm deletion in the **Delete Row** popup.

Click the **Action** tab to continue.



Use the drop-down listboxes on the **Action** tab to specify behavior related to the foreign key constraint that will be performed when data within the table is updated or deleted:

- Use the drop-down listbox next to **On update** to select an action that will be performed when data in the table is updated.
- Use the drop-down listbox next to **On delete** to select an action that will be performed when data in the table is deleted.

The supported actions are:

NO ACTION	Produce an error indicating that the deletion or update will create a foreign key constraint violation. If the constraint is deferred, this error will be produced at constraint check time if any referencing rows still exist. This is the default.
RESTRICT	Throw an error indicating that the deletion or update would create a foreign key constraint violation. This is the same as NO ACTION except that the check is not deferrable.
CASCADE	Delete any rows referencing the deleted row, or update the values of the referencing column(s) to the new values of the referenced columns, respectively.
SET NULL	Set the referencing column(s) to null.
SET DEFAULT	Set the referencing column(s) to their default values. There must be a row in the referenced table that matches the default values (if they are not null), or the operation will fail.



To add a check constraint, select the **Check** tab on the panel, and click the **Add** icon (+). To define the check constraint, click the **Edit** icon to the left of the **Trash** icon. A dialog similar to the **Check** dialog (accessed by right clicking on **Constraints** in the **Browser** tree control) opens.

Use the fields in the **General** tab to identify the check constraint:

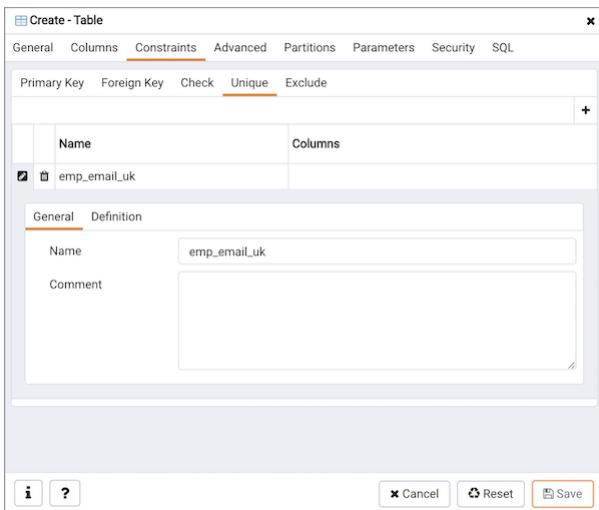
- Use the **Name** field to add a descriptive name for the check constraint. The name will be displayed in the **Browser** tree control. With PostgreSQL 9.5 forward, when a table has multiple check constraints, they will be tested for each row in alphabetical order by name and after NOT NULL constraints.
- Provide notes about the check constraint in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define the check constraint:

- Provide the expression that a row must satisfy in the **Check** field. This field is required.
- Move the **No Inherit?** switch to the **Yes** position to specify this constraint is automatically inherited by a table's children. The default is **No**.
- Move the **Don't validate?** switch to the **No** position to skip validation of existing data; the constraint may not hold for all rows in the table. The default is **Yes**.

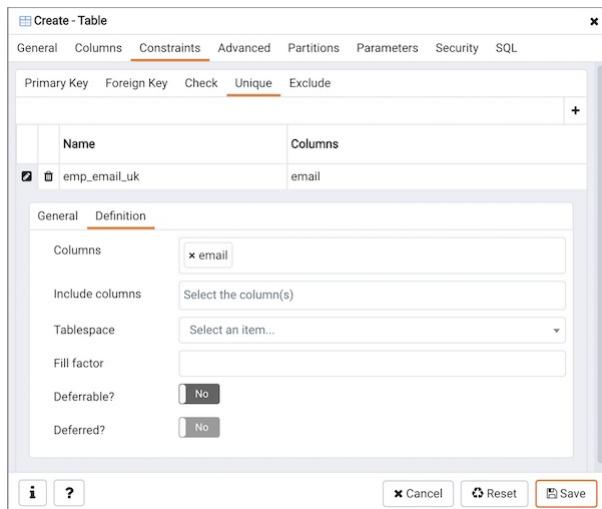


To add a unique constraint, select the **Unique** tab on the panel, and click the **Add** icon (+). To define the constraint, click the **Edit** icon to the left of the **Trash** icon. A dialog similar to the **Unique constraint** dialog (accessed by right clicking on **Constraints** in the **Browser** tree control) opens.

Use the fields in the **General** tab to identify the unique constraint:

- Use the **Name** field to add a descriptive name for the unique constraint. The name will be displayed in the **Browser** tree control.
- Provide notes about the unique constraint in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define the unique constraint:

- Click inside the **Columns** field and select one or more column names from the drop-down listbox. To delete a selection, click the **x** to the left of the column name. The unique constraint should be different from the primary key constraint defined for the same table; the selected column(s) for the constraints must be distinct.
- Select the name of the tablespace in which the unique constraint will reside from the drop-down listbox in the **Tablespace** field.
- Use the **Fill Factor** field to specify a fill factor for the table and index. The fill factor for a table is a percentage between 10 and 100. 100 (complete packing) is the default.
- Move the **Deferrable?** switch to the **Yes** position to specify the timing of the constraint is deferrable and can be postponed until the end of the statement. The default is **No**.
- If enabled, move the **Deferred?** switch to the **Yes** position to specify the timing of the constraint is deferred to the end of the statement. The default is **No**.



To add an exclusion constraint, select the **Exclude** tab on the panel, and click the **Add** icon (+). To define the constraint, click the **Edit** icon to the left of the **Trash** icon. A dialog similar to the **Exclusion constraint** dialog (accessed by right clicking on **Constraints** in the **Browser** tree control) opens.

Use the fields in the **General** tab to identify the exclusion constraint:

- Use the **Name** field to provide a descriptive name for the exclusion constraint. The name will be displayed in the **Browser** tree control.
- Provide notes about the exclusion constraint in the **Comment** field.

Click the **Definition** tab to continue.

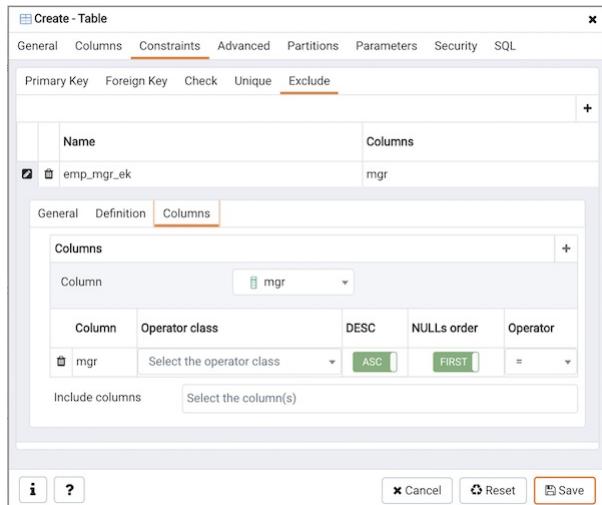


Use the fields in the **Definition** tab to define the exclusion constraint:

- Use the drop-down listbox next to **Tablespace** to select the tablespace in which the index associated with the exclude constraint will reside.
- Use the drop-down listbox next to **Access method** to specify the type of index that will be used when implementing the exclusion constraint:
 - Select **gist** to specify a GiST index (the default).
 - Select **spgist** to specify a space-partitioned GiST index.
 - Select **btree** to specify a B-tree index.
 - Select **hash** to specify a hash index.

- Use the **Fill Factor** field to specify a fill factor for the table and associated index. The fill factor is a percentage between 10 and 100. 100 (complete packing) is the default.
- Move the **Deferrable?** switch to the **Yes** position to specify that the timing of the constraint is deferrable, and can be postponed until the end of the statement. The default is **No**.
- If enabled, move the **Deferred?** switch to the **Yes** position to specify the timing of the constraint is deferred to the end of the statement. The default is **No**.
- Use the **Constraint** field to provide a condition that a row must satisfy to be included in the table.

Click the **Columns** tab to continue.



Use the fields in the **Columns** tab to to specify the column(s) to which the constraint applies. Use the drop-down listbox next to **Column** to select a column and click the **Add** icon (+) to provide details of the action on the column:

- The **Column** field is populated with the selection made in the **Column** drop-down listbox.
- If applicable, use the drop-down listbox in the **Operator class** to specify the operator class that will be used by the index for the column.
- Move the **DESC** switch to **DESC** to specify a descending sort order. The default is **ASC** which specifies an ascending sort order.
- Move the **NULLs order** switch to **LAST** to define an ascending sort order for NULLs. The default is **FIRST** which specifies a descending order.
- Use the drop-down list next to **Operator** to specify a comparison or conditional operator.

Click the **Advanced** tab to continue.



Use the fields in the **Advanced** tab to define advanced features for the table:

- Move the **RLS Policy?** switch to the **Yes** position to enable the Row Level Security.
- Move the **Force RLS Policy?** to the **Yes** position to force the policy on the owner of the table.
- Use the drop-down listbox next to **Of type** to copy the table structure from the specified composite type. Please note that a typed table will be dropped if the type is dropped (with DROP TYPE ... CASCADE).
- Use the **Fill Factor** field to specify a fill factor for the table. The fill factor for a table is a percentage between 10 and 100. 100 (complete packing) is the default.
- Use the **Toast tuple target** field to set toast_tuple_target storage parameter of the table. The toast_tuple_target value is in bytes and has minimum value of 128. This field will be enabled only for PostgreSQL version >= 11
- Use the **Parallel workers** field to set parallel_workers storage parameter of the table. The parallel_workers sets the number of workers that should be used to assist a parallel scan of the table. This field will be enabled only for PostgreSQL version >= 9.6
- Move the **Has OIDs?** switch to the **Yes** position to specify that each row within a table has a system-assigned object identifier. The default is **No**.
- Move the **Unlogged?** switch to the **Yes** position to disable logging for the table. Data written to an unlogged table is not written to the write-ahead log. Any indexes created on an unlogged table are automatically unlogged as well. The default is **No**.

Use the fields in the Like box to specify which attributes of an existing table from which a table will automatically copy column names, data types, and not-null constraints; after saving the new or modified table, any changes to the original table will not be applied to the new table.

- Use the drop-down listbox next to **Relation** to select a reference table.
- Move the **With default values?** switch to the **Yes** position to copy default values.
- Move the **With constraints?** switch to the **Yes** position to copy table and column constraints.
- Move the **With indexes?** switch to the **Yes** position to copy indexes.
- Move the **With storage?** switch to the **Yes** position to copy storage settings.
- Move the **With comments?** switch to the **Yes** position to copy comments.

With PostgreSQL 10 forward, the **Partition** tab will be visible.

Click the **Partition** tab to continue.



Use the fields in the **partition** tab to create the partitions for the table:

- Select a partition type from the **Partition Type** selection box. There are 3 options available; Range, List and Hash. Hash option will only enable for PostgreSQL version ≥ 11 .

Use the **Partition Keys** panel to define the partition keys. Click the **Add** icon (+) to add each partition keys selection:

- Select a partition key type in the **Keytype** field.
- Select a partition column in the **Column** field if Column option selected for **Keytype** field .
- Specify the expression in the **Expression** field if Expression option selected for the **Keytype** field.

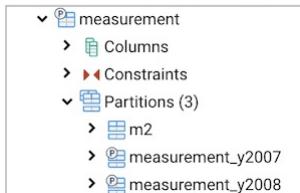
Use the **Partitions** panel to define the partitions of a table. Click the **Add** icon (+) to add each partition:

- Move the **Operation** switch to **attach** to attach the partition, by default it is **create**.
- Use the **Name** field to add the name of the partition.
- If partition type is Range or List then **Default** field will be enabled.
- If partition type is Range then **From** and **To** fields will be enabled.
- If partition type is List then **In** field will be enabled.
- If partition type is Hash then **Modulus** and **Remainder** fields will be enabled.

Users can create a partition and define them as a partitioned table. Click the **Edit** icon to expand the properties of a partition. Use the **Partition** tab to create that partition as a partitioned table.

- Move the **Partitioned Table?** switch to the **Yes** in case you want to create a partitioned table.
- Select a partition type from the **Partition Type** selection box.
- Use the **Partition Keys** panel to define the partition keys.

View of multi level Partitioned Table in browser tree:



Click the **Parameter** tab to continue.



Use the tabs nested inside the **Parameter** tab to specify VACUUM and ANALYZE thresholds; use the **Table** tab and the **Toast Table** tab to customize values for the table and the associated toast table:

- Move the **Custom auto-vacuum?** switch to the **Yes** position to perform custom maintenance on the table and to select values in the **Vacuum table**. The **Vacuum Table** provides default values for maintenance operations.
- Changing **Autovacuum enabled?** to **Not set** will reset autovacuum_enabled.

Provide a custom value in the **Value** column for each metric listed in the **Label** column.

Click the **Security** tab to continue.



Use the **Security** tab to assign privileges and define security labels.

Use the **Privileges** panel to assign privileges to a role. Click the **Add** icon (+) to set privileges for database objects:

- Select the name of the role from the drop-down listbox in the **Grantee** field.
- Click inside the **Privileges** field. Check the boxes to the left of one or more privileges to grant the selected privilege to the specified user.
- The current user, who is the default grantor for granting the privilege, is displayed in the **Grantor** field.

Click the **Add** icon (+) to assign additional privileges; to discard a privilege, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Use the **Security Labels** panel to define security labels applied to the function. Click the **Add** icon (+) to add each

security label selection:

- Specify a security label provider in the **Provider** field. The named provider must be loaded and must consent to the proposed labeling operation.
- Specify a security label in the **Security Label** field. The meaning of a given label is at the discretion of the label provider. PostgreSQL places no restrictions on whether or how a label provider must interpret security labels; it merely provides a mechanism for storing them.

Click the **Add** icon (+) to assign additional security labels; to discard a security label, click the trash icon to the left of the row and confirm deletion in the **Delete Row** popup.

Click the **SQL** tab to continue.

Your entries in the **Table** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Table** dialog:



```

CREATE TABLE public.emp
(
    empno numeric,
    ename character varying,
    job character varying,
    sal numeric,
    mgr numeric,
    email character varying,
    hiredate date,
    deptno numeric,
    CONSTRAINT emp_empno_pk PRIMARY KEY (empno),
    CONSTRAINT emp_email_uk UNIQUE (email),
    CONSTRAINT emp_deptno_fk FOREIGN KEY (deptno)
        REFERENCES public.dept (deptno) MATCH FULL
        ON UPDATE NO ACTION
        ON DELETE NO ACTION
        DEFERRABLE INITIALLY DEFERRED,
    CONSTRAINT emp_sal_ck CHECK (sal > 0) NOT VALID,
    CONSTRAINT emp_mgr_exk EXCLUDE USING btree (
        mgr WITH =
    )
)
TABLESPACE pg_default;
ALTER TABLE public.emp
OWNER to postgres;

```

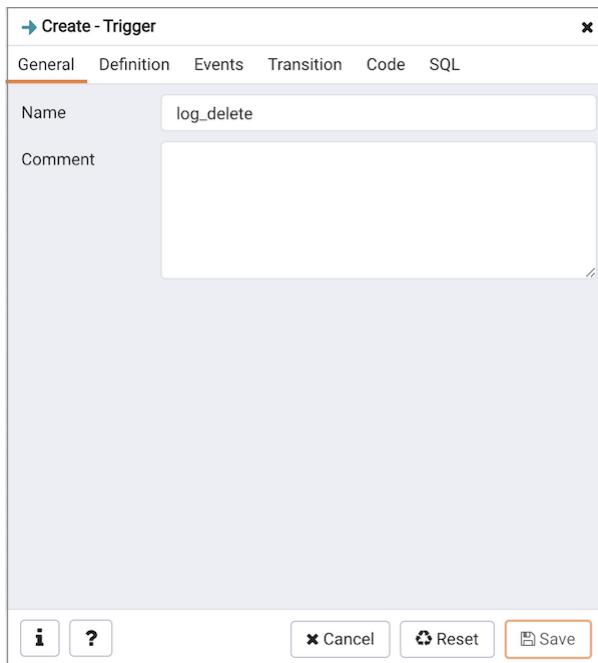
The example shown demonstrates creating a table named **product_category**. It has three columns and a primary key constraint on the **category_id** column.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.5.12.11 Trigger Dialog

Use the **Trigger** dialog to create a trigger or modify an existing trigger. A trigger executes a specified function when certain events occur.

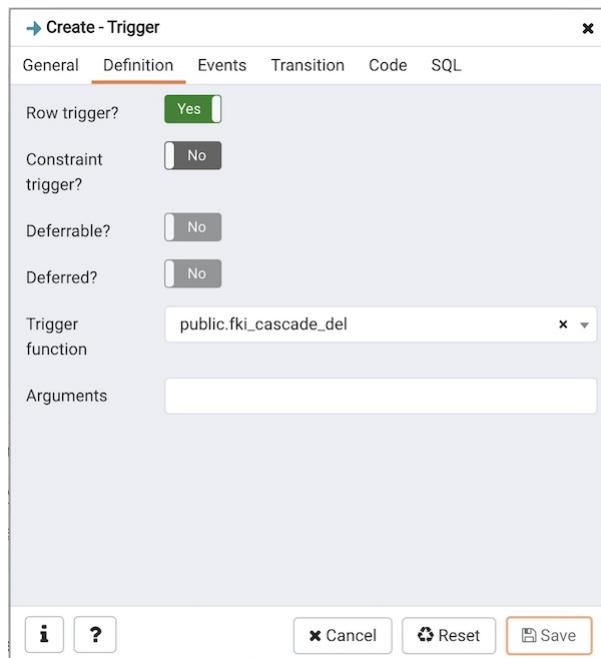
The **Trigger** dialog organizes the development of a trigger through the following dialog tabs: **General**, **Definition**, **Events**, and **Code**. The **SQL** tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the trigger:

- Use the **Name** field to add a descriptive name for the trigger. This must be distinct from the name of any other trigger for the same table. The name will be displayed in the **Browser** tree control. Note that if multiple triggers of the same kind are defined for the same event, they will be fired in alphabetical order by name.
- Store notes about the trigger in the **Comment** field.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define the trigger:

- Move the **Row trigger?** switch to the **No** position to disassociate the trigger from firing on each row in a table. The default is **Yes**.
- Move the **Constraint trigger?** switch to the **Yes** position to specify the trigger is a constraint trigger.

- If enabled, move the **Deferrable?** switch to the **Yes** position to specify the timing of the constraint trigger is deferrable and can be postponed until the end of the statement. The default is **No**.
- If enabled, move the **Deferred?** switch to the **Yes** position to specify the timing of the constraint trigger is deferred to the end of the statement causing the triggering event. The default is **No**.
- Use the drop-down listbox next to **Trigger Function** to select a trigger function or procedure.
- Use the **Arguments** field to provide an optional (comma-separated) list of arguments to the function when the trigger is executed. The arguments are literal string constants.



- **Trigger enabled** field is available in trigger dialog once the trigger is created. You can select one of the four options available.

Click the **Events** tab to continue.



Use the fields in the **Events** tab to specify how and when the trigger fires:

- Use the drop-down listbox next to the **Fires** fields to determine if the trigger fires **BEFORE** or **AFTER** a specified event. The default is **BEFORE**.

- Select the type of event(s) that will invoke the trigger; to select an event type, move the switch next to the event to the **YES** position. The supported event types are **INSERT**, **UPDATE**, **DELETE**, and **TRUNCATE**.
- Use the **When** field to provide a boolean condition that will invoke the trigger.
- If defining a column-specific trigger, use the **Columns** field to specify the columns or columns that are the target of the trigger.

Click the **Code** tab to continue.



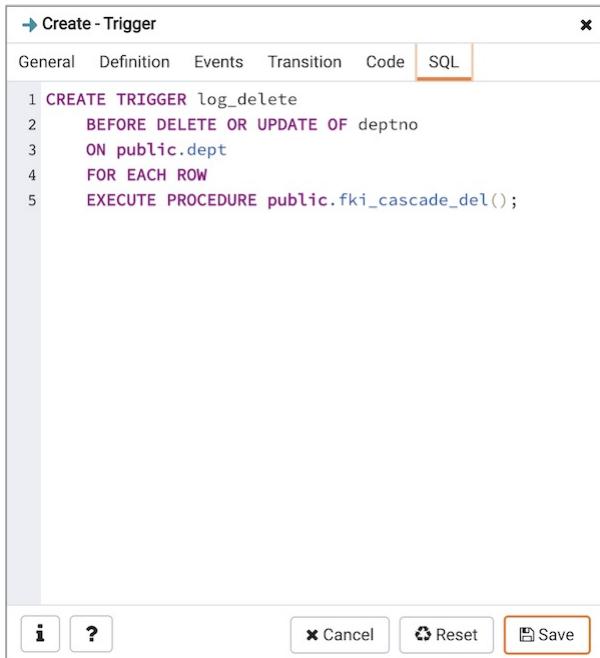
Use the **Code** field to specify any additional code that will be invoked when the trigger fires.

Click the **SQL** tab to continue.

Your entries in the **Trigger** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Trigger** dialog:



The example demonstrates creating a trigger named `log_update` that calls a procedure named `log_account_update` that logs any updates to the `distributors` table.

- Click the `Info` button (i) to access online help. View context-sensitive help in the `Tabbed browser`, where a new tab displays the PostgreSQL core documentation.
- Click the `Save` button to save work.
- Click the `Cancel` button to exit without saving work.
- Click the `Reset` button to restore configuration parameters.

7.5.12.12 Unique Constraint Dialog

Use the `Unique constraint` dialog to define a unique constraint for a specified table. Unique constraints ensure that the data contained in a column, or a group of columns, is unique among all the rows in the table.

The `Unique constraint` dialog organizes the development of a unique constraint through the following dialog tabs: `General` and `Definition`. The `SQL` tab displays the SQL code generated by dialog selections.



Use the fields in the **General** tab to identify the unique constraint:

- Use the **Name** field to add a descriptive name for the unique constraint. The name will be displayed in the **Browser** tree control.

Click the **Definition** tab to continue.



Use the fields in the **Definition** tab to define the unique constraint:

- Click inside the **Columns** field and select one or more column names from the drop-down listbox. To delete a selection, click the **x** to the left of the column name. The unique constraint should be different from the primary key constraint defined for the same table; the selected column(s) for the constraints must be distinct.
- Use **Include columns** field to specify columns for **INCLUDE** clause of the constraint. This option is available in Postgres 11 and later.
- Select the name of the tablespace in which the unique constraint will reside from the drop-down listbox in the **Tablespace** field.
- Select the name of an index from the drop-down listbox in the **Index** field. This field is optional. Adding a unique

constraint will automatically create a unique B-tree index on the column or group of columns listed in the constraint, and will force the column(s) to be marked NOT NULL.

- Use the **Fill Factor** field to specify a fill factor for the table and index. The fill factor for a table is a percentage between 10 and 100. 100 (complete packing) is the default.
- Move the **Deferrable?** switch to the **Yes** position to specify the timing of the constraint is deferrable and can be postponed until the end of the statement. The default is **No**.
- If enabled, move the **Deferred?** switch to the **Yes** position to specify the timing of the constraint is deferred to the end of the statement. The default is **No**.

Click the **SQL** tab to continue.

Your entries in the **Unique constraint** dialog generate a SQL command (see an example below). Use the **SQL** tab for review; revisit or switch tabs to make any changes to the SQL command.

Example

The following is an example of the sql command generated by user selections in the **Unique constraint** dialog:

```
① Create - Unique constraint
General Definition SQL
1 ALTER TABLE public.dept
2   ADD CONSTRAINT dept_loc_uk UNIQUE (loc);

Cancel Reset Save
```

The example shown demonstrates creating a unique constraint named **name_con** on the **name** column of the **distributors** table.

- Click the **Info** button (i) to access online help. View context-sensitive help in the **Tabbed browser**, where a new tab displays the PostgreSQL core documentation.
- Click the **Save** button to save work.
- Click the **Cancel** button to exit without saving work.
- Click the **Reset** button to restore configuration parameters.

7.6 Managing a BART Server

Postgres Enterprise Manager (PEM) is designed to assist database administrators, system architects, and performance analysts when administering, monitoring, and tuning PostgreSQL and Advanced Server database servers.

The EDB Backup and Recovery Tool (BART) is an administrative utility providing simplified backup and recovery management for multiple local or remote EDB Postgres Advanced Server and PostgreSQL database servers. For more information about BART, please visit the EnterpriseDB website at:

<https://www.enterprisedb.com/enterprise-postgres/edb-postgres-backup-and-recovery-tool>

From PEM version 7.10 onwards, you can manage a BART server through PEM console. PEM provides a user-friendly interface that allows you to manage your BART server and perform all the BART operations from PEM console.

Before you manage a BART server through PEM console, you must ensure that your system meets certain requirements. For more information, see:

7.6.1 Prerequisites for managing BART

- Before adding a BART server to the PEM console, you must manually install and configure BART on the BART host. For more information about installing and configuring BART, please see the [BART Installation Guide](#) available at:

<https://www.enterprisedb.com/docs>

- Before associating a database server with a BART server, you must install SSH on the database server and the BART server.
- Before restoring a BART backup, you must install BART, PEM agent, and SSH on the target server. SSH must also be installed on the BART server that you plan to use for restore.
- To take a backup of the replica database servers, you must ensure that the latest [pg_basebackup](#) utility is installed on the database server that you want to manage through BART.

7.6.2 Configuring a BART Server

You can use the [Create-BART server](#) dialog to register an existing BART server with the PEM server. To access the dialog, right-click on the [BART Servers](#) node and select [Create-BART Server](#).

The screenshot shows the 'Create - BART server' dialog box. The 'General' tab is active. The fields are as follows:

- Agent name:** Select from the list
- Server name:** (empty)
- Host:** Enter/Select from the list
IP address of the BART host
- User:** (empty)
Username for the BART host
- Installation path:** Enter/Select from the list
Path to BART installation directory
- Backup path:** (empty)
Path where all the backups will be stored
- pg_basebackup path:** (empty)
Path to pg_basebackup binary, eg: /usr/edb/as11/bin/pg_basebackup
- Xlog/WAL method?**: Fetch (button)
- Retention policy:** 1 Backups
- Log file:** (empty)
Path to the BART log file

At the bottom are buttons for Cancel, Reset, and Save.

Use the fields on the **General** tab to describe the general properties of the BART Server:

- Use the **Agent Name** field to select the agent that you want to configure as a BART server. Only those PEM agents that are supported for BART are listed in the drop-down list.
- Use the **Server Name** field to specify a user-friendly name for the server. The name specified will identify the server in the Browser tree.
- Use the **Host** field to specify the IP address of the host or agent where BART is installed.
- Use the **User** field to specify the user name that will be used for performing all the BART operations. You can either use the `enterprisedb` (for Advanced Server) or `postgres` (for PostgreSQL) database user account or you can create a new BART user account. This user must be an operating system user who owns the BART backup catalog directory.
- Use the **Installation path** field to specify the directory path where BART is installed on the host or BART server.
- Use the **Backup path** field to specify the file system parent directory where all BART backups and archived WAL files will be stored.
- Use the **pg_basebackup_path** field to specify the path to the `pg_basebackup` utility.
- Use the **Xlog/WAL** method field to specify how the transaction log should be collected during the execution of `pg_basebackup`. The default option is `fetch`; it specifies that the transaction log files will be collected after the backup has completed. Set the **Xlog** method to `stream` to stream the transaction log in parallel with the full base backup creation. If streaming is used, the `max_wal_senders` configuration parameter in the `postgresql.conf` file for affected database servers must account for an additional session for the streaming of the transaction log (the setting must be a minimum of 2).

For more information about Xlog method, see:

<https://www.postgresql.org/docs/current/app-pgbasebackup.html>

- Use the **Retention policy** field to specify the retention policy for the backup. This determines when an active backup should be marked as obsolete, and hence, be a candidate for deletion. You can specify the retention policy in terms of number of backup or in terms of duration (days, weeks, or months).
- Use the **Log file** field to specify the path to BART log file. This is an optional field.

The screenshot shows the 'Create - BART server' dialog box with the 'Misc' tab selected. The configuration includes:

- Scanner log file:** Path to the Xlog/WAL scanner log file.
- Socket dir path:** Path to the socket directory where all BART sockets will be stored. The default directory is /tmp. This parameter is added from BART version 2.5.2 onwards.
- Socket name:** Using this option overrides the default BART socket name generated using MD5 checksum. This parameter is added from BART version 2.5.6 onwards.
- WAL compression?**: A switch set to **Disabled**. Compress the archived Xlog/WAL files in gzip format (The gzip must be in the BART user account's PATH).
- Copy Xlog/WAL during restore?**: A switch set to **Disabled**. Copy the archived Xlog/WAL files from the BART backup catalog to the restore_path/archived_wals directory prior to the database server archive recovery.
- Thread count:** Number of threads used to copy blocks (set to 1).
- Batch size:** Number of blocks of memory used for copying the modified blocks, the default value is 49412.
- Scan interval:** Number of seconds before forcing a scan of the Xlog/WAL files, default value 0 means no brute-force scanning will be started (set to 1).
- MBM scan timeout:** Number of seconds to wait for MBM file before timing out, applicable only for incremental backup (set to 20).
- Workers:** Number of parallel worker processes required to stream the modified blocks of an incremental backups to the restore host (set to 1).

At the bottom are buttons for **i**, **?**, **Cancel**, **Reset**, and **Save**.

Use the fields on the **Misc** tab to describe the backup-related properties of the BART Server:

- Use the **Scanner log file** field to specify the path to the Xlog/WAL scanner log file. This is an optional field; BART does not create a WAL scanner log file if you do not specify the path.
- Use the **Socket dir path** field to specify the path to the socket directory where all BART sockets will be stored. The default directory is **/tmp**. This parameter is added from BART version 2.5.2 onwards.
- Use the **Socket name** field to specify a user-friendly BART socket file name. Using this option overrides the default BART socket name generated using MD5 checksum. This parameter is added from BART version 2.5.6 onwards.
- Use the **WAL compression?** switch to specify if you want to compress the archived Xlog/WAL files in Gzip format. To enable WAL compression, the gzip compression program must be present in the BART user account's PATH. The WAL compression setting must not be enabled for those database servers where you need to take incremental backups.
- Use the **Copy WALs during restore?** field to specify how the archived WAL files are collected when invoking the RESTORE operation. Set to **Enabled** to copy the archived WAL files from the BART backup catalog to the **restore_path/archived_wals** directory prior to the database server archive recovery. Set to **disabled** to retrieve the archived WAL files directly from the BART backup catalog during the database server archive recovery. Enabling this option helps you save time during the restore operation.
- Use the **Thread count** field to specify the number of worker threads for copying blocks or data files from the database server to the BART backup catalog. Specify a **thread count** of **1** if you want to take the backup using the **pg_basebackup** utility.

- Use the **Batch size** field to specify the number of blocks of memory used for copying modified blocks. This is applicable only for incremental backups.
- Use the **scan interval** field to specify the number of seconds after which the WAL scanner should scan the new WAL files.
- Use the **MBM scan timeout** field to specify the number of seconds to wait for MBM files before timing out. This is applicable only for incremental backups.
- Use the **Workers** field to specify the number of parallel worker processes required to stream the modified blocks of an incremental backups to the restore host.

7.6.3 Associating the BART Server with a Database Server

After configuring the BART server, you need to associate it with the database server whose backup you want to manage with BART. You can do one of the following:

- Use the PEM console to modify the properties of an existing monitored database server to map it to the newly configured BART server.
- Use the PEM console to create a new monitored database server, and map it to the newly configured BART server.

To map the BART server to a new PEM database server, right-click the **PEM Server Directory** node and select **Create > Server**. Enter the details on all the generic tabs and then enter the BART-specific details on the **BART** tab.



Use the fields on the **General** tab to describe the general properties of the BART Server that will map to the PEM server:

- Use the **BART server** field to select the BART server name. All the BART servers configured in the PEM console will be listed in this drop down list.
- Use the **Server name** field to specify a name for the database server that you want to backup using the BART server. This name gets stored in the BART configuration file.
- Use the **Description** field to specify the description of the database server.
- Use the **Backup name** field to specify a template for user-defined names to be assigned to the backups of the database server. If you do not specify a backup name template, then the backup can only be referenced in BART sub-commands by the BART assigned, integer backup identifier.
- Use the **Host address** field to specify the IP address of the database server that you want to configure for

backup.

- Use the **Port** field to specify the port to be used for the database that you want to backup.
- Use the **User** field to specify the user of the database that you want to backup using BART through PEM console. If you want to enable incremental backups for this database server, then the user must be a superuser.
- Use the **Password** field to specify the password for the user of the database that you want to backup.
- Use the **Cluster owner** field to specify the Linux operating system user account that owns the database cluster. This is typically **enterprisedb** for Advanced Server database clusters installed in the Oracle databases compatible mode, or **postgres** for PostgreSQL database clusters and for Advanced Server database clusters installed in the PostgreSQL databases compatible mode.
- Use the **Override archive command?** switch to specify if you want to override the archive command in the database server's **postgresql.conf** file. If you override the archive command, the database server will be restarted or database configurations will be reloaded after the server gets added.
- Use the **Archive command** field to specify the desired format of the archive command string. Ensure to bind a PEM agent and provide **Service ID** to reload the database configuration or restart the server.
- Use the **Archive path** field to store the archived WAL files. The default location is the BART backup catalog. This parameter is added from BART version 2.5.2 onwards.
- Use the **Allow incremental backup?** switch to specify if incremental backup should be enabled for this database server.
- Use the **Setup passwordless SSH?** switch to specify if you want to create SSH certificates to allow passwordless logins between the database server and the BART server. You must ensure that a PEM agent is bound to the server before configuring passwordless SSH authentication. Passwordless SSH will not work for a database server that is being remotely monitored by a PEM agent.



Use the fields on the **Misc** tab to describe the miscellaneous properties of the BART Server:

- Use the **Override default configuration?** Switch to specify if you want to override the BART server configurations with the specific database server configurations.
- Use the **Xlog** method to specify how the transaction log should be collected during the execution of **pg_basebackup**.
- Use the **Retention policy** field to specify the retention policy for the backup. This determines when an active backup should be marked as obsolete, and hence, be a candidate for deletion. You can specify the retention policy in terms of number of backup or in terms of duration (days, weeks, or months).
- Use the **WAL compression** switch to specify if you want to compress the archived Xlog/WAL files in Gzip format. To enable WAL compression, the gzip compression program must be present in the BART user account's PATH. The wal_compression setting must not be enabled for those database servers where you need to take incremental backups.
- Use the **Copy WALS during restore** field to specify how the archived WAL files are collected when invoking

the RESTORE operation. Set to enabled to copy the archived WAL files from the BART backup catalog to the <restore_path>/archived_wals directory prior to the database server archive recovery. Set to disabled to retrieve the archived WAL files directly from the BART backup catalog during the database server archive recovery.

- Use the **Thread count** field to specify the number of threads to copy the blocks. You must set **thread count** to **1** if you want to take a backup with the **pg_basebackup** utility.
- Use the **Batch size** field to specify the number of blocks of memory used for copying modified blocks, applicable only for incremental backups.
- Use the **Scan interval** field to specify the number of seconds after which the WAL scanner should scan the new WAL files.
- Use the **MBM scan timeout** field to specify the number of seconds to wait for MBM files before timing out, applicable only for incremental backups.
- Use the **Workers** field to specify the number of parallel worker processes required to stream the modified blocks of an incremental backups to the restore host.

7.6.4 Viewing the BART Server Details on a PEM Dashboard

Once the BART server is associated with the database server, you can see the entire backup and restore related details for that particular BART server on the PEM Dashboard. You can also perform operations such as restoration or deletion of a backup that is listed on the dashboard.



When you select a monitored BART server, details of all the associated database servers along with their activities are displayed as a chart on the Dashboard in the **BART Tool Activities** panel. You can select the activities on any criteria that you specify in the filter boxes (the database server, status, duration or date).

The **Managed Database servers** panel displays a list of all the database servers managed by that particular BART server along with their high-level details.

The **Initiated Server Backups** panel displays a list of all the backups of the database servers managed by that particular BART server. You can filter the list to display the details of a particular database server. You can also filter the list on any criteria that you specify in the filter box. Typically, this filter works with any kind of string value (excluding date, time, and size) listed under the columns. For example, you can type **tar** to filter the list and display only those backups that are in tar format.

Backup details displayed include the **Backup Name**, **Backup ID**, **Status**, **Server Name**, **Start Time**, **Type**, **Parent ID**, **Format**, **Duration**, and **Size**. The **Status** column shows the status of the backups which can be one of the following: **In Progress**, **Active**, **Keep**, or **Obsolete**.

The backups are marked as **Obsolete** after the backup retention period has passed or number of retained backups that you have specified as retention policy of the BART server is met. If you want to make an exception so that a particular backup does not get marked as **Obsolete** even after the expiry of the duration of retention policy, then you need to mark that particular backup as **Keep**. Similarly, if you mark a particular backup as **NoKeep**, the backup is re-evaluated to determine if its status should be changed back to obsolete based upon the current retention policy.

Please note that if any of the scheduled tasks for backup, restore, validate host, validate server or delete obsolete backup for any of the BART Server gets deleted, it will not display under the **BART Tool Activities** graph of BART Server's dashboard. However, it gets listed under the **Initiated Server Backups** list.

A pin in the first column under **Actions** indicates that a backup can be marked as **Keep** by clicking the pin; while an inverted pin indicates that the backup can be marked as **NoKeep**. The second column under **Actions** displays the **Restore** icon; you can perform the **Restore** operation by clicking on the icon.

You can delete all the **Obsolete** backups by clicking the **Delete Obsolete** button. You can also refresh the list of backups by clicking the **Refresh** button.

7.6.5 Scheduling BART Backups

To schedule a backup using BART, select **Schedule Backup** under **Tools** menu. You can see a list of scheduled backups with details such as **Logs**, **Last result**, **Database server**, **Last backup name**, **Started on**, **Type**, **Parent**, **Format**, **Verify checksum?**, and **Use pg_basebackup?**. Click the Add icon (+) to add a schedule for the backup. Enter the details in the schedule definition dialog:

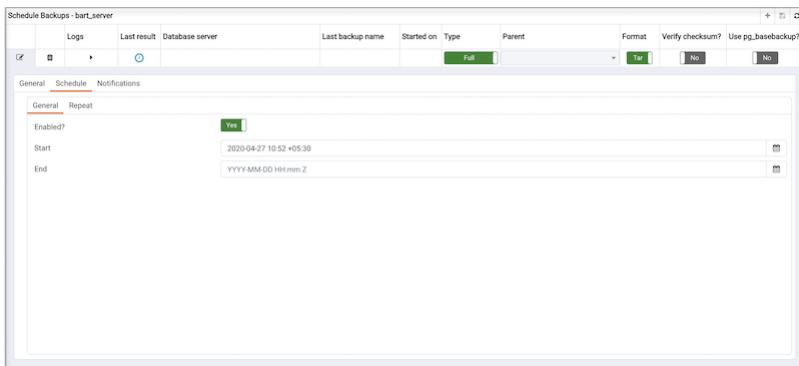


Use the fields on the **General** tab to describe the general properties of the backup:

- Use the **Database Server** field to specify the target database server that you want to back up.
- Use the **Backup name** to specify a user-defined name for the backup.
- Use the **Backup type** switch to specify the backup type i.e. full backup or incremental backup.
- Use the **Parent backup** field to select the ID of the parent backup for incremental backup. This parent backup

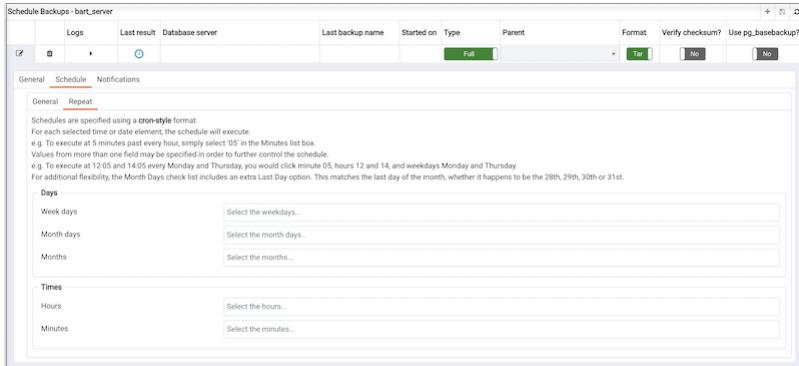
can either be a full or an incremental backup.

- Use the **Format switch** to specify the output format of the backup i.e plain text or tar. For incremental backup, you need to select plain text only.
- Use the **Gzip compression** switch to specify if gzip compression should be enabled for the backup. This option is applicable only for the tar format.
- Use the **Compression level** field to specify the gzip compression level on the tar file output.
- Use the **Thread count** field to specify the number of threads that will copy the blocks.
- Use the **MBM scan timeout** field to specify the number of seconds to wait for required MBM files before timing out.
- Use the **Checksum algorithm** field to specify checksum algorithm for MBM files of the backup.
- Use the **Verify checksum** field to specify if you want the application to verify the checksum of the backup.
- Use the **pg_basebackup** field to specify if the pg_basebackup utility should be used for the backup. Typically, pg_basebackup utility is used only for backing up the replica servers since it cannot be used for incremental backups.



Provide information on the **Schedule** tab to describe the scheduling details:

- Use the **Enabled?** switch to indicate if the schedule should be enabled (**Yes**) or disabled (**No**).
- Use the calendar selector in the **Start** field to specify the starting date and time for the schedule.
- Use the calendar selector in the **End** field to specify the ending date and time for the schedule.



Use the fields on the **Repeat** tab to specify the details about the schedule in a cron-style format. The schedule will execute on each date or time element selected on the **Repeat** tab. Click within a field to open a list of valid values for that field; click on a specific value to add that value to the list of selected values for the field. To clear the values from a field, click the **X** located at the right-side of the field.

Use the fields within the **Days** box to specify the days on which the schedule will execute:

- Use the **Week Days** field to select the days on which the schedule will execute.
- Use the **Month Days** field to select the numeric days on which the schedule will execute. Specify the Last Day to indicate that the schedule should be performed on the last day of the month, regardless of the date.
- Use the **Months** field to select the months in which the schedule will execute.

Use the fields within the **Times** box to specify the times at which the schedule will execute:

- Use the **Hours** field to select the hour at which the schedule will execute.
- Use the **Minutes** field to select the minute at which the schedule will execute.



Use the fields on the **Notifications** tab to specify the email notification settings for a scheduled backup:

- Use the **Send the notifications** field to specify when you want the email notifications to be sent.
- Use the **Email group** field to specify the email group that should receive the email notification.

7.6.6 Scheduling BART Obsolete Backups Deletion

Use the **Schedule Obsolete Backup Deletion** dialog to schedule or modify a BART obsolete backup deletion. Use context menu from database server where BART has been configured.



Provide information on the **General** tab to describe the scheduling details:

- Use the **Enabled?** switch to indicate if the schedule should be enabled (**Yes**) or disabled (**No**).
- Use the calendar selector in the **Start** field to specify the starting date and time for the schedule.
- Use the calendar selector in the **End** field to specify the ending date and time for the schedule.



Use the fields on the **Repeat** tab to specify the details about the schedule in a cron-style format. The schedule will execute on each date or time element selected on the **Repeat** tab. Click within a field to open a list of valid values for that field; click on a specific value to add that value to the list of selected values for the field. To clear the values from a field, click the **X** located at the right-side of the field.

Use the fields within the **Days** box to specify the days on which the schedule will execute:

- Use the **Week Days** field to select the days on which the schedule will execute.
- Use the **Month Days** field to select the numeric days on which the schedule will execute. Specify the Last Day to indicate that the schedule should be performed on the last day of the month, regardless of the date.
- Use the **Months** field to select the months in which the schedule will execute.

Use the fields within the **Times** box to specify the times at which the schedule will execute:

- Use the **Hours** field to select the hour at which the schedule will execute.
- Use the **Minutes** field to select the minute at which the schedule will execute.

7.6.7 BART Backup Dialog

Use the **BART Backup** dialog to take ad-hoc backups using BART. This dialog can be opened using **Backup...** context menu of BART server node and **Backup...** sub menu of **BART** context menu of Database server node.



Use the fields on the **General** tab to describe the general properties of the backup:

- Use the **Database Server** field to specify the target database server that you want to back up.
- Use the **Backup name** to specify a user-defined name for the backup.
- Use the **Backup type** switch to specify the backup type i.e. full backup or incremental backup.
- Use the **Parent backup** field to select the ID of the parent backup for incremental backup. This parent backup can either be a full or an incremental backup.
- Use the **Format switch** to specify the output format of the backup i.e plain text or tar. For incremental backup, you need to select plain text only.
- Use the **Gzip compression** switch to specify if gzip compression should be enabled for the backup. This option is applicable only for the tar format.
- Use the **Compression level** field to specify the gzip compression level on the tar file output.
- Use the **Thread count** field to specify the number of threads that will copy the blocks.
- Use the **Checksum algorithm** field to specify checksum algorithm for MBM files of the backup.
- Use the **Verify checksum** field to specify if you want the application to verify the checksum of the backup.
- Use the **pg_basebackup** field to specify if the pg_basebackup utility should be used for the backup. Typically, pg_basebackup utility is used only for backing up the replica servers since it cannot be used for incremental backups.



Use the fields on the **Notifications** tab to specify the email notification settings for a backup:

- Use the **Send the notifications** field to specify when you want the email notifications to be sent.
- Use the **Email group** field to specify the email group that should receive the email notification.

7.6.8 Restoring BART Backups

You can restore the backups that you have earlier created using BART server on a target remote host. When you select a particular BART server, all the associated backups are listed in the Dashboard under **Initiated Server Backups**.

To restore a backup, click the **Restore** icon next to the backup that you want to restore.



In the **Restore Backup** dialog, provide information in the fields on the **General** tab:

- Use the **Target agent** field name to specify the name of the agent where you want to restore the backup.
- Use the **Remote user** field to specify the use account on the remote database server host where you want to restore the backup.
- Use the **Remote host address** field to specify the IP address of the remote host where you want to restore the backup.
- Use the **SSH port** field to specify the SSH port to be used for restoring the backup.
- Use the **Restore path** field to specify the path where you want to restore the backup.
- Use the **Number of workers** field to specify processes to run in parallel to stream the modified blocks of an incremental backup to the restore location.
- Use the **Setup passwordless SSH?** switch to specify if you want to create SSH certificates to allow passwordless logins between the BART server and the target host for restore.
- Use the **Verify checksum** switch to specify if you want to verify checksum of a backup.



On the **Advanced** tab, specify your preferences for advanced options for restoring the backup:

- Use the **Copy WALs to restore path?** switch to specify if you want to copy WALs to the restore path.
- Use the **Point in time recovery** switch to specify if you want point in time recovery.
- Use the **Timeline ID** field to specify the timeline ID to be used for replaying the archived WAL files for point-in-time recovery.
- Use the **Transaction ID (XID)** field to specify the transaction ID for point-in-time recovery.
- Use the **Timestamp** field to the timestamp to be used for restore.

Note

You can specify either **Transaction ID** or **Timestamp** for the point-in-time recovery.



Use the fields on the **Notifications** tab to specify the email notification settings for restoring the backup.

- Use the **Send the notifications** field to specify when you want the email notifications to be sent.
- Use the **Email group** field to specify the email group that should receive the email notification.

7.7 SQL Profiler

SQL Profiler captures statistical information and query execution plans for SQL statements executed during a trace session. You can use the information stored by SQL Profiler to identify performance issues. Before using SQL Profiler, you must [install and configure SQL Profiler](#) on each database you intend to profile.

Contents:

7.7.1 Installing SQL Profiler

SQL Profiler allows a database superuser to locate and optimize poorly-running SQL code. Users of Microsoft SQL Server's Profiler will find PEM's SQL Profiler very similar in operation and capabilities. SQL Profiler is installed with each Advanced Server instance; if you are using PostgreSQL, you must download the SQL Profiler installer or packages and install the SQL Profiler product into each managed database instance you wish to profile.

SQL Profiler is officially supported only on the EnterpriseDB distributions of PostgreSQL version 9.4 or above and Advanced Server version 9.4 or above. The plugin is distributed via StackBuilder, or is available from the [EnterpriseDB website](#)

You can use the graphical installer to install any version of SQL Profiler on Windows platform.

On Linux, if you have installed your database server through graphical installer then you must use the graphical installer to install the SQL Profiler. If you have installed your database server using the RPM or DEB package, then you must use the RPM or DEB package to install the SQL Profiler.

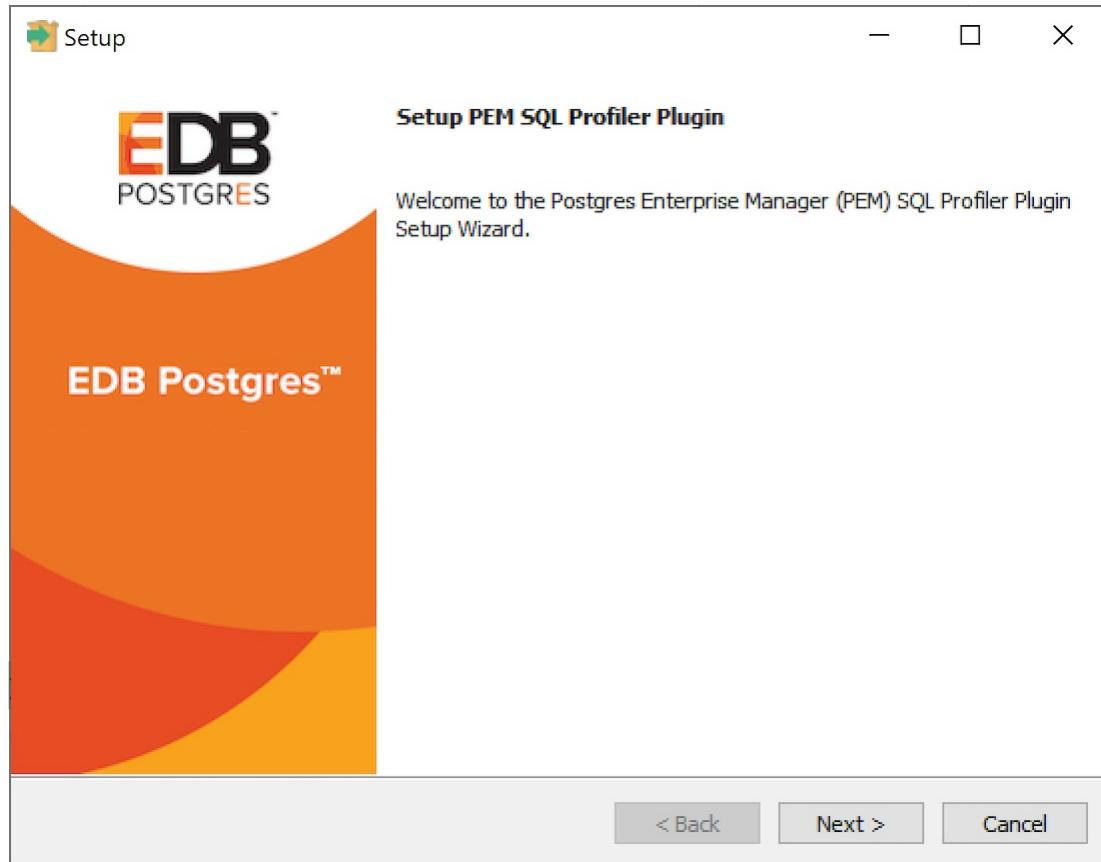
Installing SQL Profiler on Windows

To invoke the SQL Profiler graphical installer, assume superuser privileges (or **Administrator** privileges on Windows), navigate into the directory that contains the installer, and invoke the installer:

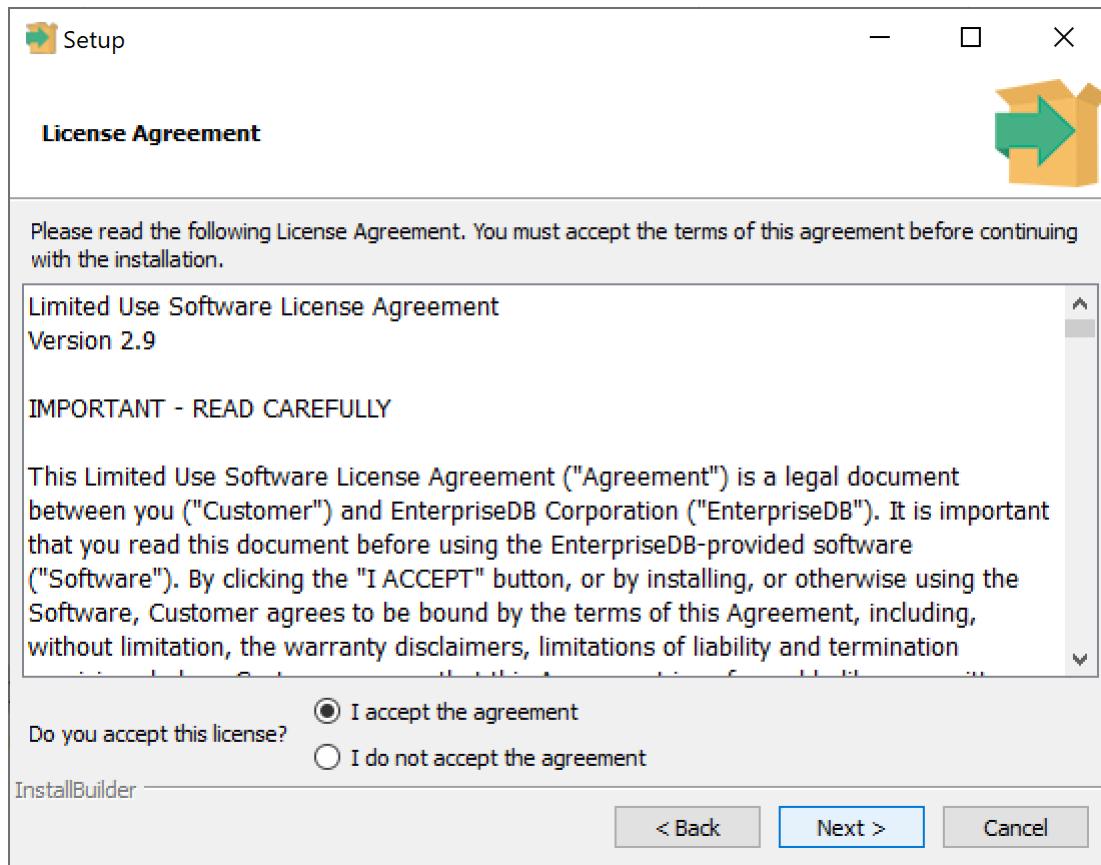
```
| sqlprofiler-pg-<pg_version>-<sql_profiler_version>-windows-x64.exe
```

Where, **pg_version** is the version of your PostgreSQL and **sql_profiler_version** is the version of SQL Profiler.

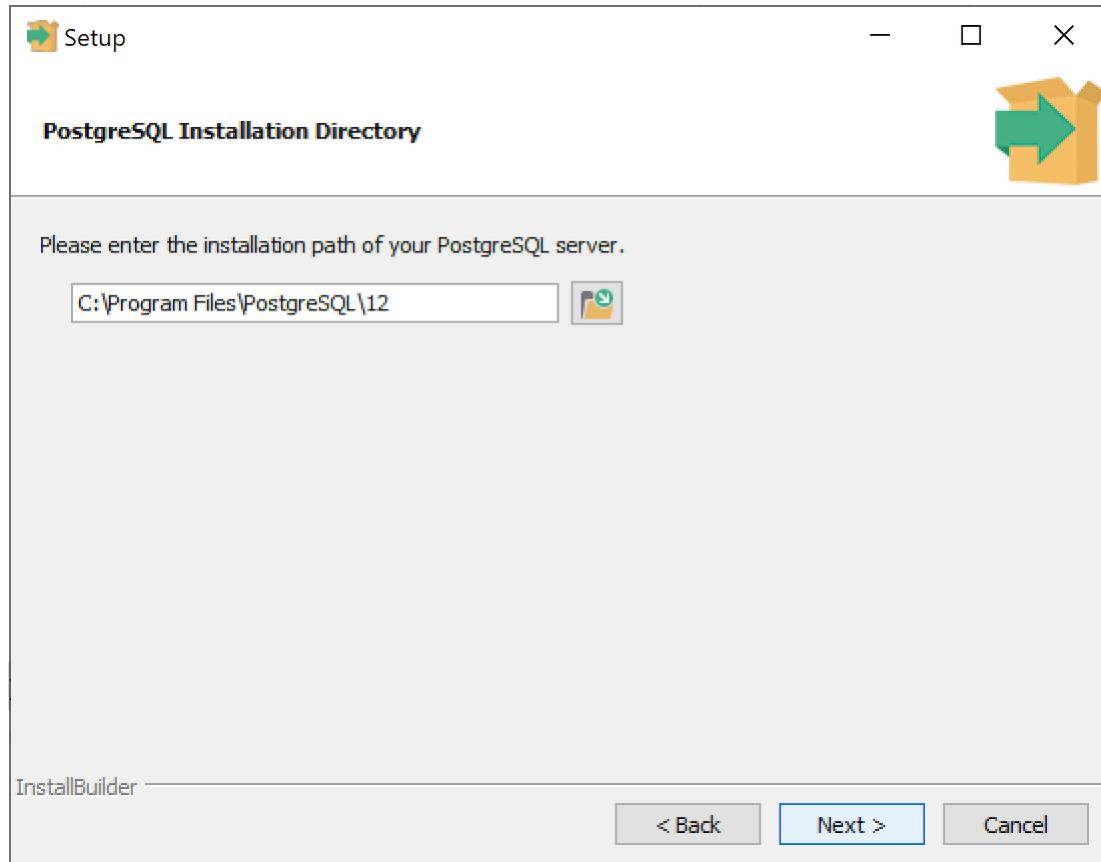
The SQL Profiler installer welcomes you to the Setup Wizard.



Click **Next** to continue to the **License Agreement**.



Carefully review the license agreement before highlighting the appropriate radio button and accepting the agreement; click **Next** to continue to the **Installation Directory** dialog.



Specify an alternate location for the installation directory, or accept the default location and click **Next** to continue.



The wizard is now ready to install the SQL Profiler plugin. Click **Next** to continue.



The SQL Profiler plugin installer displays progress bars as it copies files to your system.



When the installation is complete, the SQL Profiler plugin is ready to be configured.

Installing SQL Profiler on Linux using RPMs

Note

You may be required to add the `sslutils` package to your PostgreSQL database servers before installing SQL Profiler.

You can install SQL Profiler using rpm on RHEL or Centos 6 or 7, using yum command as root user:

```
yum install postgresql<pg_version>-sqlprofiler
```

Where, `pg_version` is the version of your PostgreSQL.

When the installation is complete, the SQL Profiler plugin is ready to be configured.

Installing SQL Profiler on Debian/Ubuntu using DEB

Note

You may be required to add the `sslutils` package to your PostgreSQL database servers before installing SQL Profiler.

You can install SQL Profiler using DEB on Debian 9.x or Ubuntu 18, using apt command as root user:

```
apt install postgresql-<pg_version>-sqlprofiler
```

Where, `pg_version` is the version of your PostgreSQL.

When the installation is complete, the SQL Profiler plugin is ready to be configured.

7.7.2 Configuring SQL Profiler

The SQL Profiler plugin is not automatically enabled when the installation process completes. This allows you to restart the server at a convenient time, and prevents the plugin from being loaded unnecessarily on systems where it is not required on a continual basis.

Use the following steps to enable the plugin for each database monitored by SQL Profiler:

1. Edit the `postgresql.conf` file on the server you wish to profile, modifying the `shared_preload_libraries` configuration parameter as shown below:

For Linux, `shared_preload_libraries = '$libdir/sql-profiler'`

For Windows, `shared_preload_libraries = '$libdir/sql-profiler.dll'`

1. Create the functions used by SQL Profiler in your database. The SQL Profiler installation program places a SQL script (named `sql-profiler.sql`) in the `share/contrib` subdirectory of the main PostgreSQL installation directory. You must invoke this script on the maintenance database (specified when registering the server with PEM).

You can also use the psql command line to invoke the configuration script. The following command uses psql to invoke the `sql-profiler.sql` script on PostgreSQL Server database on a Linux system:

```
$ /usr/pgsql-x/bin/psql -U postgres postgres -f /usr/pgsql-x/share/contrib/sql-profiler.sql
```

1. Stop and re-start the server for the changes to take effect.

Please note: if you have connected to the PEM server with the PEM client before configuring SQL Profiler, you must disconnect and reconnect with the server to enable SQL Profiler functionality.

7.7.3 Using SQL Profiler

You can use SQL Profiler to create and store up to 15 named traces; use menu options to create and manage traces.

Creating a Trace

You can use the Create trace... dialog to define a SQL Trace for any database on which SQL Profiler has been installed and configured. [installed and configured](#). To access the dialog, highlight the name of the database in the PEM client tree control; navigate through the **Management** menu to the **SQL Profiler** pull-aside menu, and select **Create trace....**



Use the fields on the **Trace options** tab to specify details about the new trace:

- Provide a name for the trace in the **Name** field.
- Click in the **User filter** field to specify the roles whose queries will be included the trace; optionally, check the box next to **Select All** to include queries from all roles.
- Click in the **Database filter** field to specify which databases to trace; optionally, check the box next to **Select All** to include queries against all databases.
- Specify a trace size in the **Maximum Trace File Size** field; SQL Profiler will terminate the trace when it reaches approximately the size specified.
- Specify **Yes** in the **Run Now** field to start the trace when you select the **Create** button; select **No** to enable fields on the **Schedule** tab.



Use the fields on the **Schedule** tab to specify scheduling details for the new trace:

- Use the **Start time** field to specify the starting time for the trace.
- Use the **End time** field to specify the ending time for the trace.
- Specify **Yes** in the **Repeat?** field to indicate that the trace should be repeated every day at the times specified; select **No** to enable fields on the **Periodic job options** tab.



Fields on the **Periodic job options** tab specify scheduling details for a recurring trace. Use fields in the **Days** section to specify the days on which the job will execute:

- Click in the **Week days** field to select the days of the week on which the trace will execute.
- Click in the **Month days** field to select the days of the month on which the trace will execute.
- Click in the **Months** field to select the months in which the trace will execute.

Use fields in the **Times** section to specify a time schedule for the trace execution:

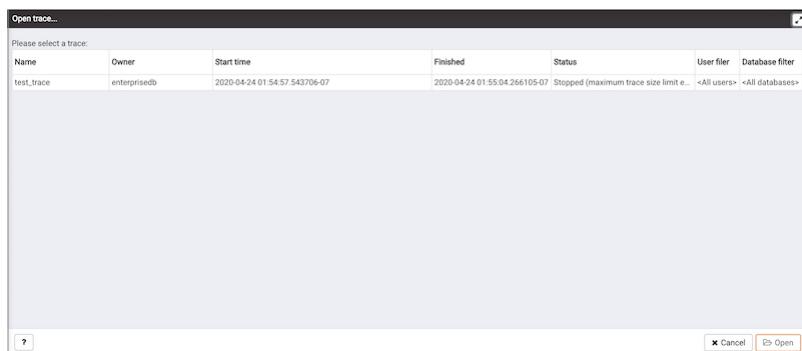
- Click in the **Hours** field to select the hours at which the trace will execute.
- Click in the **Minutes** field to select the hours at which the trace will execute.

When you've completed the **Create trace...** dialog, click **Create** to start the newly defined trace or to schedule the trace for a later time. If you elect to execute the trace immediately, the trace results will display in the PEM client.

#	Start Time	Duration (ms)	Query	Rows Affected	User	Database	PID	File System Read	File System Write	Page Fault
1	2020-04-24 01:54:57.593232-07	0.272112	SELECT pg_heartbeat_info..	1	agent1	pgem	91666	48	0	6
2	2020-04-24 01:54:57.627984-07	0.232971	/*pgsqlAdmin*/ SELECT ..	5	enterprisedb	postgres	33489	0	0	0
3	2020-04-24 01:54:57.644187-07	0.002485	SELECT 1 ..	1	enterprisedb	pgem	30543	0	0	0
4	2020-04-24 01:54:57.650644-07	0.004036	SELECT pg_has_role(p..	1	enterprisedb	pgem	30543	0	0	0
5	2020-04-24 01:54:57.660344-07	0.000893	SELECT 1 ..	1	enterprisedb	pgem	30543	0	0	0
6	2020-04-24 01:54:57.669933-07	0.957138	SET DateStyle=ISO_SE..	1	enterprisedb	postgres	33900	0	0	0
7	2020-04-24 01:54:57.696903-07	0.015561	SELECT oid AS did ..	1	enterprisedb	postgres	33900	0	0	0
8	2020-04-24 01:54:57.699242-07	0.027168	SELECT oid AS id_nohu ..	1	enterprisedb	postgres	33900	0	0	0
9	2020-04-24 01:54:58.63668-07	12.090508	/*pgsqlAdmin*/ SELECT ..	5	enterprisedb	postgres	33485	0	0	0
10	2020-04-24 01:54:58.64937-07	0.001994	SELECT 1 ..	1	enterprisedb	pgem	30543	0	0	0
11	2020-04-24 01:54:58.826576-07	0.050517	SELECT pg_has_reloge..	23	enterprisedb	pgem	30543	0	0	0
12	2020-04-24 01:54:59.627713-07	11.294604	/*pgsqlAdmin*/ SELECT ..	5	enterprisedb	postgres	33485	0	0	0
13	2020-04-24 01:56:00.137269-07	0.001693	SELECT 1 ..	1	enterprisedb	postgres	32578	0	0	0
14	2020-04-24 01:56:00.250787-07	11.271028	/*pgsqlAdmin*/ SELECT ..	5	enterprisedb	postgres	33485	0	0	0
15	2020-04-24 01:56:02.114481-07	0.627253	WITH agent_info AS (..	1	agent1	pgem	91666	680	0	52
16	2020-04-24 01:56:02.236089-07	12.07968	/*pgsqlAdmin*/ SELECT ..	5	enterprisedb	postgres	33485	0	0	0
17	2020-04-24 01:56:02.626516-07	13.478983	SELECT * FROM (SEL..	3	agent1	pgem	91666	1568	0	49
18	2020-04-24 01:56:03.049985-07	0.004995	SELECT * FROM pgem..	1	agent1	pgem	91666	0	0	0
19	2020-04-24 01:56:03.239984-07	0.017002	SELECT pg_has_reloge..	1	agent1	pgem	91666	0	0	0
20	2020-04-24 01:56:03.239984-07	18.527022	/*pgsqlAdmin*/ SELECT ..	5	enterprisedb	postgres	33485	0	0	0
21	2020-04-24 01:56:03.241196-07	0.128994	INSERT INTO pgm.jobs..	1	agent1	pgem	91666	82	0	1
22	2020-04-24 01:56:03.268871-07	0.03972	SELECT log_directory..	1	agent1	pgem	91666	0	0	0

Opening a Trace

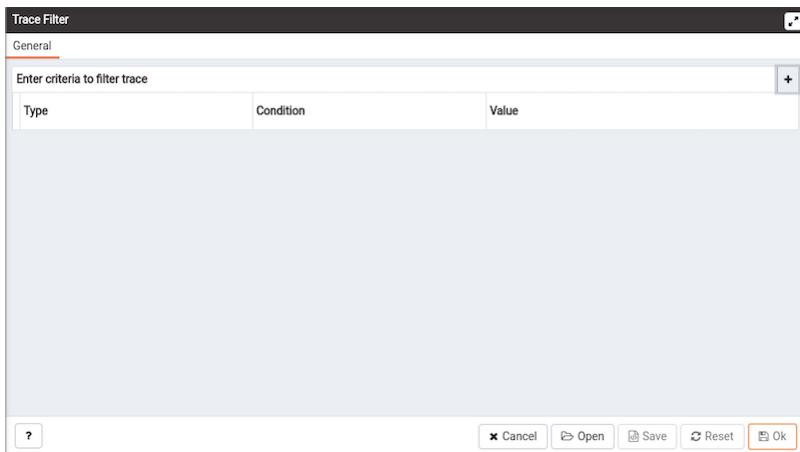
To view a previous trace, highlight the name of the profiled database in the PEM client tree control; navigate through the **Management** menu to the **SQL Profiler** pull-aside menu, and select **Open trace...**. You can also use the SQL Profiler toolbar menu to open a trace; select the **Open trace...** option. The **Open trace...** dialog opens.



Highlight an entry in the trace list and click **Open** to open the selected trace. The selected trace opens in the SQL Profiler tab.

Filtering a Trace

A filter is a named set of (one or more) rules, each of which can hide events from the trace view. When you apply a filter to a trace, the hidden events are not removed from the trace, but are merely excluded from the display. Click the **Filter** icon to open the **Trace Filter** dialog and create a rule (or set of rules) that define a filter. Each rule will screen the events within the current trace based on the identity of the role that invoked the event, or the query type invoked during the event.



To open an existing filter, select the **Open** button; to define a new filter, click the Add (+) button to add a row to the table displayed on the **General** tab and provide rule details:

- Use the **Type** drop-down listbox to specify the trace field that the filter rule will apply to.
- Use the **Condition** drop-down listbox to specify the type of operator that SQL Profiler will apply to the **Value** when it filters the trace:
 - Select **Matches** to filter events that contain the specified **Value**.
 - Select **Does not match** to filter events that do not contain the specified **Value**.
 - Select **Is equal to** to filter events that contain an exact match to the string specified in the **Value** field.
 - Select **Is not equal to** to filter events that do not contain an exact match to the string specified in the **Value** field.
 - Select **Starts with** to filter events that begin with the string specified in the **Value** field.
 - Select **Does not start with** to filter events that do not begin with the string specified in the **Value** field.
 - Select **Less than** to filter events that have a numeric value less than the number specified in the **Value** field.
 - Select **Greater than** to filter events that have a numeric value greater than the number specified in the **Value** field.
 - Select **Less than or equal to** to filter events that have a numeric value less than or equal to the number specified in the **Value** field.
 - Select **Greater than or equal to** to filter events that have a numeric value greater than or equal to the number specified in the **Value** field.
- Use the **Value** field to specify the string, number or regular expression that SQL Profiler will search for.

When you've finished defining a rule, click the Add (+) button to add another rule to the filter. To delete a rule from a filter, highlight the rule and click the **Delete** icon.

Click the **Save** button to save the filter definition to a file without applying the filter; to apply the filter, click **OK**. Select **Cancel** to exit the Trace Filter dialog and discard any changes to the filter.

Deleting a Trace

To delete a trace, highlight the name of the profiled database in the PEM client tree control; navigate through the **Management** menu to the **SQL Profiler** pull-aside menu, and select **Delete trace(s)...**. You can also use the SQL Profiler toolbar menu to delete a trace; select the **Delete trace(s)...** option. The **Delete traces...** dialog opens.



Click the icon to the left of a trace name to mark one or more traces for deletion and click **Delete**.



The PEM client will acknowledge that the selected traces have been deleted.

Viewing Scheduled Traces

To view a list of scheduled traces, highlight the name of the profiled database in the PEM client tree control; navigate through the **Management** menu to the **SQL Profiler** pull-aside menu, and select **Scheduled traces...**. You can also use the SQL Profiler toolbar menu to the list; select the **Scheduled traces...** option. The **Scheduled traces...** dialog opens.



The **Scheduled traces...** dialog displays a list of the traces that are awaiting execution. Click the edit button to the left of a trace name to access detailed information about the trace:



The **General** tab displays detailed information about the scheduled trace:

- The **Status** field lists the status of the current trace.
- The **Enabled?** switch displays **Yes** if the trace is enabled; **No** if the trace is disabled.
- The **Name** field displays the name of the trace.
- The **Agent** field displays the name of the agent responsible for executing the trace.
- The **Last run** field displays the date and time of the last execution of the trace.
- The **Next run** field displays the date and time of the next scheduled execution of the trace.
- The **Created** field displays the date and time that the trace was defined.

7.7.4 Using Index Advisor

Index Advisor helps you determine the application tables (and columns) on which you should create common B-tree type indexes. This can reduce the execution cost of queries you expect to use on your tables. Index Advisor comes pre-installed with EDB Postgres (R) Advanced Server. Index Advisor works with Advanced Server's query planner by creating "hypothetical indexes" for the query planner to use to calculate execution costs if such indexes were available.

Before using Index Advisor, you must:

1. Modify the `postgresql.conf` file on each Advanced Server host, adding the `index_advisor` library to the `shared_preload_libraries` parameter.
2. Install the Index Advisor contrib module. To install the module, use the `psql` client or PEM Query Tool to connect to the database, and invoke the following command:

```
<complete_path>/share/contrib/index_advisor.sql
```

3. Restart the server for your changes to take effect.

After installing Index Advisor, you can select one or more rows from within a trace, and select the Index Advisor icon to access Index Advisor functionality. For detailed installation and usage information about Index Advisor, please see the EDB Postgres Advanced Server Guide, available from the EnterpriseDB website at:

<http://www.enterprisedb.com>

Note

It is recommended that you disable the index advisor while using the pg_dump functionality.

7.7.5 The SQL Profiler Tab

Toolbar Options

Toolbar options on the SQL Profiler tab allow you to define new traces, start or stop existing traces, open and search through previous traces, and filter trace results.



Use the following options to manage your SQL Profiler traces:

Option	Action	Shortcut
Menu	Use options accessed through the drop_down menu icon to manage SQL Profiler traces.	Accesskey + O
Start Trace	Select the Start Trace icon to start a new trace, using the attributes (user names, database names, comments, etc) that were defined for the trace currently displayed in the SQL Profiler dialog.	Accesskey + S
Stop Trace	Select the Stop Trace icon to stop an executing trace.	Accesskey + Q
Refresh Trace	Select the Refresh Trace icon to update the display to include any recent changes to an active trace.	Accesskey + R
Clear Trace	Select the Clear Trace icon to delete the trace and close the SQL Profiler window.	Accesskey + C
Filter	Select the Filter icon to define a new filter, or apply an existing filter to the trace.	Accesskey + T
Information	Select the information icon to view the properties of the trace displayed in the SQL Profiler window.	Accesskey + P
Index Advisor	Select the Index Advisor icon to open the PEM Index Advisor .	Accesskey + I
Download Trace	Use options accessed through the Download Trace menu to download a CSV file that contains the trace events shown on the current page or the complete set of trace data.	Accesskey + X
Column Picker	Click the Column Picker icon to choose the columns to be displayed in below table.	Accesskey + W

Viewing Trace Data

The SQL Profiler tab is divided into three panes:

- The top of the tabbed browser window (the trace data pane) displays a list of the SQL commands executed during the trace.
- The lower-left panel displays the SQL query that was executed, or the metrics gathered during the execution.
- The lower-right panel displays the query execution plan; you can view the execution plan in a Text-based form, or as a

Graphical Plan.

The screenshot shows the SQL Profiler interface with the 'Graphical Plan' tab selected. The main pane displays a graphical representation of the query execution plan, showing nodes like 'agent1', 'pg_stat_activity', and 'agent', connected by arrows indicating data flow. Below this, a table lists individual SQL events with columns for Start Time, Duration (ms), Query, Rows Affected, User, Database, PID, File System Read, File System Write, and Page Fault. The bottom section of the window shows the 'Query/Metrics' pane with tabs for Graphical, Text-Based Plan, Analysis, and Statistics, and a table of metrics for the selected query.

The Trace Data Pane

The Trace Data pane displays the SQL commands executed during the trace. By default, the commands are displayed in the order that the command was executed.

Double-click a column heading to sort the trace by the column values; double-click the column heading a second time to toggle the data in the column to be in ascending or descending order.

Use the drop-down listbox next to the **Show queries per page** label to specify the number of events that SQL Profiler should display in the pane. Select from 20, 50, 100, 200, 500, 1000, or 2000. The default is 500.

If the number of events in the trace exceeds the count of events per page, use the page selector controls (located in the top left corner of the table) to navigate through pages of the trace.

To include or exclude events from the currently displayed trace, select the Filter icon from the SQL Profiler toolbar. The Trace Filter dialog will allow you to define and apply a filter that will screen the displayed trace.

The Query/Metrics Pane

The Query/Metrics pane is located in the lower-left corner of the SQL Profiler window. The tabs provide detailed information about the currently selected query:

- The **SQL Query** tab displays the text of the query that is currently highlighted in Trace Data pane.
- The **Metrics** tab displays detailed statistical information about the execution of the query. The table below describes the metrics that are displayed in the Metrics dialog; the percentages listed describe the percentage of the total quantity of the parameter that is attributed to the selected SQL command:

Property	Description
Executed (#)	The number of times that the selected SQL command executed.
Execution (%)	The percentage of the execution count that the SQL command represents. For example if the trace profiles 4 SQL commands, each command will represent 25% of the trace execution %.
Duration (%)	The percentage of the total trace time consumed by the highlighted SQL Command.
Rows updated (%)	The percentage of the rows updated during the trace that were updated by the selected SQL command.
Page faults (%)	The percentage of the page faults that occur during the trace that can be attributed to the selected SQL command.

Property	Description
Page reclaims (%)	The percentage of the pages reclaimed during the trace that can be attributed to the selected SQL command.
Swaps (%)	The percentage of swaps that occur during the trace that can be attributed to the selected SQL command.
File system in (%)	The percentage of bytes written to disk during the trace that can be attributed to the selected SQL command.
File system out (%)	The percentage of bytes read from disk during the trace that can be attributed to the selected SQL command.
Signals received (%)	Currently unused.
Messages received (%)	Currently unused.
Messages sent (%)	Currently unused.
Voluntary context switches	(%)Currently unused.
Involuntary context switches	(%)Currently unused.
Shared blocks read (%)	The percentage of the shared blocks read by the highlighted SQL command.
Shared blocks written (%)	The percentage of the shared blocks written by the highlighted SQL command.
Shared blocks hit (%)	The percentage of the shared blocks hit by the highlighted SQL command.
Local blocks read (%)	The percentage of local blocks read by the highlighted SQL command.
Local blocks written (%)	The percentage of local blocks written by the highlighted SQL command.
Local blocks hit (%)	The percentage of local blocks hit by the highlighted SQL commands.
Temporary blocks read (%)	The percentage of the temporary blocks read by the highlighted SQL commands.
Temporary blocks written (%)	The percentage of the temporary blocks written by the highlighted SQL commands.

The Explain Pane

The Graphical or Text-based explain pane displays one of two representations of the query execution plan for the selected query.

- Select the Text-based Plan tab to display the execution plan for the currently highlighted event in text format:
- Select the Graphical-based Plan tab to display a graphical interpretation of the execution plan of the highlighted query. For more information about interpreting the graphical query plan, see [Interpreting the Graphical Query Plan](#).

7.8 Developer Tools

The PEM client features powerful developer tools that you can use to execute and analyze complex SQL commands, manage data, and debug PL/SQL code.

Contents:

7.8.1 pgAdmin Debugger



The screenshot shows the pgAdmin Debugger window. At the top, there are several small icons. Below them, a status bar displays the command: `agent_down_status(p_agent_id integer, p_start_datetime timestamp with time zone, p_end_datetime timestamp with time zone, o_down_datetime timestamp with time zone, o_up_datetime timestamp with time zone)`. The main area contains a PL/pgSQL script:

```

1  DECLARE
2      v_alert_id integer;
3      v_curr_rec record;
4      v_prev_state pem.alert_state := NULL;
5
6  BEGIN
7      SELECT id INTO v_alert_id FROM pem.alert WHERE agent_id = p_agent_id AND template_id = (SELECT id FROM pem.alert_template WHERE display_name = 'Agent Down');
8      o_down_datetime := NULL;
9      o_up_datetime := NULL;
10
11     FOR v_curr_rec IN EXECUTE '
12     SELECT
13         state, generated as recorded_time
14     FROM
15         pem.alert_history
16     WHERE
17         alert_id = $1::integer AND generated >= $2::timestamptz AND generated <= $3::timestamptz
18     ORDER BY generated;1' USING v_alert_id, p_start_datetime, p_end_datetime

```

Below the script, there is a table titled "Parameters" with the following data:

Name	Type	Value
p_agent_id	integer	1
p_start_datetime	timestamp with time zone	NULL
p_end_datetime	timestamp with time zone	NULL
o_down_datetime	timestamp with time zone	NULL
o_up_datetime	timestamp with time zone	NULL

The debugger may be used to debug PL/pgSQL functions in PostgreSQL, as well as EDB-SPL functions, stored procedures and packages in Advanced Server. The Debugger is available as an extension for your PostgreSQL installation, and is distributed as part of Advanced Server. You must have superuser privileges to use the debugger.

Before using the debugger, you must modify the `postgresql.conf` file, adding the server-side debugger components to the value of the `shared_preload_libraries` parameter:

```
shared_preload_libraries = '$libdir/other_libraries/plugin_debugger'
```

After modifying the `shared_preload_libraries` parameter, restart the server to apply the changes.

The debugger may be used for either in-context debugging or direct debugging of a target function or procedure. When you use the debugger for in-context debugging, you set a breakpoint at the first line of a program; when a session invokes the target, control is transferred to the debugger. When using direct debugging, the debugger prompts you for any parameters required by the target, and then allows you to step through the code.

In-context Debugging

To set a breakpoint at the first line of a program, right-click the name of the object you would like to debug, and select `Set breakpoint` from the `Debugging` sub-menu. The debugger window will open, waiting for another session to invoke the program.



When another session invokes the target, the debugger will display the code, allowing you to add break points, or step through line-by-line. The other session is suspended until the debugging completes; then control is returned to the session.

The screenshot shows the same debugger interface as above, but now displaying actual code. The code is a PL/pgSQL block for handling agent status changes. The cursor is at line 17. The tabs at the bottom are Parameters, Local variables, Messages, Results, and Stack, with Stack being the active tab.

```

agent_down_status(p_agent_id integer,p_start_datetime timestamp with time zone,p_end_datetime timestamp with time zone,o_down_datetime timestamp with time zone,o_up_datetime timestamp with time zone)
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17

```

```

DECLARE
  v_alert_id integer;
  v_curr_rec record;
  v_prev_state pem.alert_state := NULL;
BEGIN
  SELECT id INTO v_alert_id FROM pem.alert WHERE agent_id = p_agent_id and template_id = (SELECT id FROM pem.alert_template WHERE display_name = 'Agent
  o_down_datetime := NULL;
  o_up_datetime := NULL;
FOR v_curr_rec IN EXECUTE '
SELECT
  state, generated as recorded_time
FROM
  pem.alert_history
WHERE
  alert_id = $1::integer AND generated >= $2::timestamptz AND generated <= $3::timestamptz

```

Name	Value	Line No.
pem.agent_down_status(integer,timestamp with time zone,timestamp with time zone)	p_agent_id=1,p_start_datetime=,p_end_datetime=	7

Direct Debugging

To use the debugger for direct debugging, right click on the name of the object that you wish to debug in the pgAdmin tree control and select **Debug** from the **Debugging** sub-menu. The debugger window will open, prompting you for any values required by the program:



Use the fields on the **Debugger** dialog to provide a value for each parameter:

- The **Name** field contains the formal parameter name.
- The **Type** field contains the parameter data type.
- Check the **Null?** checkbox to indicate that the parameter is a NULL value.
- Check the **Expression?** checkbox if the Value field contains an expression.
- Use the **Value** field to provide the parameter value that will be passed to the program. When entering

parameter values, type the value into the appropriate cell on the grid, or, leave the cell empty to represent NULL, enter " (two single quotes) to represent an empty string, or to enter a literal string consisting of just two single quotes, enter ". PostgreSQL 8.4 and above supports variadic function parameters. These may be entered as a comma-delimited list of values, quoted and/or cast as required.

- Check the **Use default?** checkbox to indicate that the program should use the value in the Default Value field.
- The **Default Value** field contains the default value of the parameter.

Provide values required by the program, and click the **Debug** button to start stepping through the program. The values of the arguments provided here are saved. The values will be pre-filled next time the dialog opens. To clear the values, use the **Clear All** button.



The screenshot shows the debugger interface with the following details:

- Toolbar:** Includes icons for Stop, Step Into, Step Over, Continue, Toggle Breakpoint, Clear All Breakpoints, and Stack.
- Code Panel:** Displays PostgreSQL code for `agent_down_status`. The code includes declarations, a BEGIN block, and a FOR loop. It uses parameters \$1, \$2, and \$3.
- Stack Panel:** Shows a table with columns Name, Value, and Line No. One entry is shown: `pem.agent_down_status(integer,timestamp with time zone,timestamp with time zone)` with Value `p_agent_id=1,p_start_datetime=,p_end_datetime=` and Line No. 7.

Using the Debugger

The main debugger window consists of two panels and a context-sensitive toolbar. Use toolbar icons to manage breakpoints and step into or through code; hover over an icon for a tooltip that identifies the option associated with the icon. The toolbar options are:



Option	Action
Step into	Click the Step into icon to execute the currently highlighted line of code.
Step over	Click the Step over icon to execute a line of code, stepping over any sub-functions invoked by the code. The sub-function executes, but is not debugged unless it contains a breakpoint.
Continue/Start	Click the Continue/Start icon to execute the highlighted code, and continue until the program encounters a breakpoint or completes.
Toggle breakpoint	Use the Toggle breakpoint icon to enable or disable a breakpoint (without removing the breakpoint).
Clear all breakpoints	Click the Clear all breakpoints icon to remove all breakpoints from the program.
Stop	Click the Stop icon to halt the execution of a program.

The top panel of the debugger window displays the program body; click in the grey margin next to a line number to add a breakpoint. The highlighted line in the top panel is the line that is about to execute.

```
agent_down_status(p_agent_id integer, p_start_datetime timestamp with time zone, p_end_datetime timestamp with time zone, o_down_datetime timestamp with time zone, o_up_datetime timestamp with time zone)

2  DECLARE
3      v_alert_id integer;
4      v_curr_rec record;
5      v_prev_state pem.alert_state := NULL;
6  BEGIN
7      SELECT id INTO v_alert_id FROM pem.alert WHERE agent_id = p_agent_id and template_id = (SELECT id FROM pem.alert_template WHERE display_name = 'Agent Down');
8      o_down_datetime := NULL;
9      o_up_datetime := NULL;
10
11     FOR v_curr_rec IN EXECUTE (
12         SELECT
13             state, generated as recorded_time
14         FROM
15             pem.alert_history
16         WHERE
17             alert_id = $1::integer AND generated >= $2::timestamptz AND generated <= $3::timestamptz
18         ORDER BY generated asc) USING v_alert_id, p_start_datetime, p_end_datetime;
```

Parameters	Local variables	Messages	Results	Stack
Name	Type			Value
p_agent_id	integer			1
p_start_datetime	timestamp with time zone			NULL
p_end_datetime	timestamp with time zone			NULL
o_down_datetime	timestamp with time zone			NULL
o_up_datetime	timestamp with time zone			NULL

The lower panel of the debugger window provides a set of tabs that allow you to review information about the program:

- The **Parameters** tab displays the value of each parameter.
- The **Local variables** tab displays the current value of the program variables.
- The **Messages** tab displays any messages returned by the server (errors, warnings and informational messages).
- The **Results** tab displays the server message when the program completes.
- The **Stack** tab displays the list of functions that have been invoked, but which have not yet completed.

As you step through a program, the **Local variables** tab displays the current value of each variable:

Parameters	Local variables	Messages	Results	Stack
Name	Type			Value
v_alert_id	integer			NULL
v_prev_state	pem.alert_state			NULL

When you step into a subroutine, the **Stack** tab displays the call stack, including the name of each caller, the parameter values for each caller (if any), and the line number within each caller:

Parameters	Local variables	Messages	Results	Stack
Name	Value			Line No.
pem.agent_down_status(integer,timestamp with time zone,timestamp with time zone)	p_agent_id=1,p_start_datetime=,p_end_datetime=			7

Select a caller to change focus to that stack frame and display the state of the caller in the upper panel.

When the program completes, the **Results** tab displays the message returned by the server. If the program encounters an error, the **Messages** tab displays details:

```

1  DECLARE
2      v_alert_id integer;
3      v_curr_rec record;
4      v_prev_state pem.alert_state := NULL;
5  BEGIN
6      SELECT id INTO v_alert_id FROM pem.alert WHERE agent_id = p_agent_id AND template_id = (SELECT id FROM pem.alert_template WHERE display_name = 'Agent
7      o_down_datetime := NULL;
8      o_up_datetime := NULL;
9
10     FOR v_curr_rec IN EXECUTE '
11     SELECT
12         state, generated as recorded_time
13     FROM
14         pem.alert_history_table
15     WHERE
16         alert_id = $1::integer AND generated >= $2::timestampz AND generated <= $3::timestampz
Parameters Local variables Messages Results Stack
ERROR: relation "pem.alert_history_table" does not exist
LINE 5:     pem.alert_history_table
          ^
QUERY:
SELECT
    state, generated as recorded_time
FROM
    pem.alert_history_table
WHERE
    alert_id = $1::integer AND generated >= $2::timestampz AND generated <= $3::timestampz
ORDER BY generated;
CONTEXT: PL/pgSQL function pem.agent_down_status(integer,timestamp with time zone,timestamp with time zone) line 10 at FOR over EXECUTE statement

```

7.8.2 Query Tool

The Query Tool is a powerful, feature-rich environment that allows you to execute arbitrary SQL commands and review the result set. You can access the Query Tool via the **Query Tool** menu option on the **Tools** menu, or through the context menu of select nodes of the Browser tree control. The Query Tool allows you to:

- Issue ad-hoc SQL queries.
- Execute arbitrary SQL commands.
- Edit the result set of a SELECT query if it is **updatable**.
- Displays current connection and transaction status as configured by the user.
- Save the data displayed in the output panel to a CSV file.
- Review the execution plan of a SQL statement in either a text, a graphical format or a table format (similar to <https://explain.depesz.com>).
- View analytical information about a SQL statement.

	attnum	attname	attnid	atttypid	attstattarget	attlen	attsmallint	attnum	attsmallint	attdims	attcacheoff	attcacheoff	atttypmod	attbyval	attstorage	attcharlen	attalign	attnotnull	attisdef
1	1255	oid	26	-1	4	1	0	-1	-1	-1	-1	-1	0	p	i	true	false		
2	1255	proname	19	-1	64	2	0	-1	-1	-1	-1	-1	0	p	c	true	false		
3	1255	pronamesp...	26	-1	4	3	0	-1	-1	-1	-1	-1	0	p	i	true	false		
4	1255	pronowne	26	-1	4	4	0	-1	-1	-1	-1	-1	0	p	i	true	false		
5	1255	prolong	26	-1	4	5	0	-1	-1	-1	-1	-1	0	p	i	true	false		
6	1255	procost	700	-1	4	6	0	-1	-1	-1	-1	-1	0	p	i	true	false		
7	1255	provows	700	-1	4	7	0	-1	-1	-1	-1	-1	0	p	i	true	false		
8	1255	pravaradic	26	-1	4	8	0	-1	-1	-1	-1	-1	0	p	i	true	false		
9	1255	prosport	24	-1	4	9	0	-1	-1	-1	-1	-1	0	p	i	true	false		
10	1255	prokind	18	-1	1	10	0	-1	-1	-1	-1	-1	0	p	c	true	false		
11	1255	prosedef	16	-1	1	11	0	-1	-1	-1	-1	-1	0	p	c	true	false		
12	1255	prolesproof	16	-1	1	12	0	-1	-1	-1	-1	-1	0	p	c	true	false		
13	1255	prosefunction	16	-1	1	13	n	-1	-1	-1	-1	-1	n	n	now	false			

You can open multiple copies of the Query tool in individual tabs simultaneously. To close a copy of the Query tool, click the **X** in the upper-right hand corner of the tab bar.

The Query Tool features two panels:

- The upper panel displays the **SQL Editor**. You can use the panel to enter, edit, or execute a query. It also shows the **History** tab which can be used to view the queries that have been executed in the session, and a **Scratch Pad** which can be used to hold text snippets during editing. If the Scratch Pad is closed, it can be re-opened (or additional ones opened) by right-clicking in the SQL Editor and other panels and adding a new panel.
- The lower panel displays the **Data Output** panel. The tabbed panel displays the result set returned by a query, information about a query's execution plan, server messages related to the query's execution and any asynchronous notifications received from the server.

The Query Tool Toolbar

The **Query Tool** toolbar uses context-sensitive icons that provide shortcuts to frequently performed tasks. If an icon is highlighted, the option is enabled; if the icon is grayed-out, the task is disabled. Please note that disabled icons may support functionality accessed via the [data editor](#).



Hover over an icon to display a tooltip that describes the icon's functionality:

Icon	Behavior	Shortcut
Open File	Click the Open File icon to display a previously saved query in the SQL Editor.	Accesskey + O
Save	Click the Save icon to perform a quick-save of a previously saved query, or to access the Save menu: - Select Save to save the selected content of the SQL Editor panel in a file. - Select Save As to open a new browser dialog and specify a new location to which to save the selected content of the SQL Editor panel.	Accesskey + S
Find	Use the Find menu to search, replace, or navigate the code displayed in the SQL Editor: - Select Find to provide a search target, and search the SQL Editor contents. - Select Find next to locate the next occurrence of the search target. - Select Find previous to move to the last occurrence of the search target. - Select Persistent find to identify all occurrences of the search target within the editor. - Select Replace to locate and replace (with prompting) individual occurrences of the target. - Select Replace all to locate and replace all occurrences of the target within the editor. - Select Jump to navigate to the next occurrence of the search target.	Cmd+F Cmd+G Cmd+Shift+G Cmd+Shift+F Alt+G
Copy	Click the Copy icon to copy the content that is currently highlighted in the Data Output panel. when in View/Edit data mode.	Accesskey + C
Paste	Click the Paste icon to paste a previously row into a new row when in View/Edit data mode.	Accesskey + P
Delete	Click the Delete icon to mark the selected rows for delete when in View/Edit data mode.	Accesskey + D

Icon	Behavior	Shortcut
Edit	<p>Use options on the Edit menu to access text editing tools; the options operate on the text displayed in the SQL Editor panel when in Query Tool mode:</p> <ul style="list-style-type: none"> - Select Indent Selection to indent the currently selected text. - Select Unindent Selection to remove indentation from the currently selected text. - Select Inline Comment Selection to enclose any lines that contain the selection in SQL style comment notation. - Select Inline Uncomment Selection to remove SQL style comment notation from the selected line. - Select Block Comment to enclose all lines that contain the selection in C style comment notation. This option acts as a toggle. 	Tab Shift+Tab Cmd+/ Cmd+. Shift+Cmd+/-
Filter	<p>Click the Filter icon to set filtering and sorting criteria for the data when in View/Edit data mode. Click the down arrow to access other filtering and sorting options:</p> <ul style="list-style-type: none"> - Click Sort/Filter to open the sorting and filtering dialogue. - Click Filter by Selection to show only the rows containing the values in the selected cells. - Click Exclude by Selection to show only the rows that do not contain the values in the selected cells. - Click Remove Sort/Filter to remove any previously selected sort or filtering options. 	Accesskey + F
Limit Selector	Select a value in the Limit Selector to limit the size of the dataset to a number of rows.	Accesskey + R
Stop	Click the Stop icon to cancel the execution of the currently running query.	Accesskey + Q
Execute/Refresh	<p>Click the Execute/Refresh icon to either execute or refresh the query highlighted in the SQL editor panel. Click the down arrow to access other execution options:</p> <ul style="list-style-type: none"> - Add a check next to Auto-Rollback to instruct the server to automatically roll back a transaction if an error occurs during the transaction. - Add a check next to Auto-Commit to instruct the server to automatically commit each transaction. Any changes made by the transaction will be visible to others, and durable in the event of a crash. - Click the Explain icon to view an explanation plan for the current query. <p>The result of the</p>	F5
Explain	EXPLAIN is displayed graphically on the Explain tab of the output panel, and in text form on the Data Output tab.	F7
Explain analyze	<p>Click the Explain analyze icon to invoke an EXPLAIN ANALYZE command on the current query.</p> <p>Navigate through the Explain Options menu to select options for the EXPLAIN command:</p> <ul style="list-style-type: none"> - Select Verbose to display additional information regarding the query plan. - Select Costs to include information on the estimated startup and total cost of each plan node, as well as the estimated number of rows and the estimated width of each row. - Select Buffers to include information on buffer usage. - Select Timing to include information about the startup time and the amount of time spent in each node of the query. 	Shift+F7
Commit	Click the Commit icon to commit the transaction.	Shift+CTRL+M

Icon	Behavior	Shortcut
Rollback	Click the Rollback icon to rollback the transaction.	Shift+CTRL+R
Clear	Use options on the Clear drop-down menu to erase display contents: - Select Clear Query Window to erase the content of the SQL Editor panel. - Select Clear History to erase the content of the History tab.	Accesskey + L
Download as CSV	Click the Download as CSV icon to download the result set of the current query to a comma-separated list. You can specify the CSV settings through Preferences -> SQL Editor -> CSV output dialogue.	F8
Macros	Click the <i>Macros</i> icon to manage the macros. You can create, edit or clear the macros through <i>Manage Macros</i> option.	

The SQL Editor Panel

The **SQL editor** panel is a workspace where you can manually provide a query, copy a query from another source, or read a query from a file. The SQL editor features syntax coloring and autocompletion.



To use autocomplete, begin typing your query; when you would like the Query editor to suggest object names or commands that might be next in your query, press the Control+Space key combination. For example, type "SELECT * FROM" (without quotes, but with a trailing space), and then press the Control+Space key combination to select from a popup menu of autocomplete options.



After entering a query, select the **Execute/Refresh** icon from the toolbar. The complete contents of the SQL editor panel will be sent to the database server for execution. To execute only a section of the code that is displayed in the SQL editor, highlight the text that you want the server to execute, and click the **Execute/Refresh** icon.

The screenshot shows the EDB Postgres Enterprise Manager interface. In the top bar, it says "pem/postgres@Postgres Enterprise Manager Server". Below that is the "Query Editor" tab, which is active. The query window contains the following SQL:

```
1 SELECT generate_series(1,1000) AS ID, 'JOHN' AS NAME
2 SELECT generate_series(1,1000) AS NO, 'JACK' AS NAME
```

Below the query window is a "Data Output" panel. It displays a table with two columns: "id" and "name". The "id" column has values from 1 to 15, and the "name" column has all entries set to "JOHN".

	id	name
1	1	JOHN
2	2	JOHN
3	3	JOHN
4	4	JOHN
5	5	JOHN
6	6	JOHN
7	7	JOHN
8	8	JOHN
9	9	JOHN
10	10	JOHN
11	11	JOHN
12	12	JOHN
13	13	JOHN
14	14	JOHN
15	15	JOHN

The message returned by the server when a command executes is displayed on the **Messages** tab. If the command is successful, the **Messages** tab displays execution details.

The screenshot shows the EDB Postgres Enterprise Manager interface. In the top bar, it says "edbstore/edbuser@PGSQL_12". Below that is the "Query Editor" tab, which is active. The query window contains the following SQL:

```
1 select empno, ename, deptno as department from edbuser.emp
```

Below the query window is a "Data Output" panel. It displays the message: "Successfully run. Total query runtime: 40 msec. 14 rows affected."

Options on the **Edit** menu offer functionality that helps with code formatting and commenting:

- The auto-indent feature will automatically indent text to the same depth as the previous line when you press the Return key.
- Block indent text by selecting two or more lines and pressing the Tab key.
- Implement or remove SQL style or toggle C style comment notation within your code.

You can also drag and drop certain objects from the treeview which can save time in typing long object names. Text containing the object name will be fully qualified with schema. Double quotes will be added if required. For functions and procedures, the function name along with parameter names will be pasted in the Query Tool.

The Data Output Panel

The **Data Output** panel displays data and statistics generated by the most recently executed query.

The screenshot shows the EDB Postgres Enterprise Manager interface. In the top bar, it says "edbstore/postgres@PGSQL_12". Below that is the "Query Editor" tab, which is active. The query window contains the following SQL:

```
1 SELECT * FROM pg_database
```

Below the query window is a "Data Output" panel. It displays a table with several columns: oid, datname, datdba, encoding, datcollate, datatype, datistemplate, datallowconn, datconnlimit, and datlastsysoid. The data is as follows:

	oid	datname	datdba	encoding	datcollate	datatype	datistemplate	datallowconn	datconnlimit	datlastsysoid
1	13432	postgres	10	6	en_IN	en_IN	false	true	-1	13431
2	16389	edbstore	16388	6	en_IN	en_IN	false	true	-1	13431
3	1	template1	10	6	en_IN	en_IN	true	true	-1	13431
4	13431	template0	10	6	en_IN	en_IN	true	false	-1	13431

The **Data Output** tab displays the result set of the query in a table format. You can:

- Select and copy from the displayed result set.
- Use the **Execute/Refresh** options to retrieve query execution information and set query execution options.
- Use the **Save results to file** icon to save the content of the **Data Output** tab as a comma-delimited file.
- Edit the data in the result set of a SELECT query if it is updatable.

A result set is updatable if:

- All columns are either selected directly from a single table, or are not table columns at all (e.g. concatenation of 2 columns). Only columns that are selected directly from the table are editable, other columns are read-only.
- All the primary key columns or OIDs of the table are selected in the result set.

Any columns that are renamed or selected more than once are also read-only.

Editable and read-only columns are identified using pencil and lock icons (respectively) in the column headers.

	empno	ename	department
1	7369	SMITH	20
2	7499	ALLEN	30
3	7521	WARD	30
4	7566	JONES	20
5	7654	MARTIN	30
6	7698	BLAKE	30
7	7782	CLARK	10
8	7788	SCOTT	20
9	7839	KING	10
10	7844	TURNER	30
11	7876	ADAMS	20
12	7900	JAMES	30
13	7902	FORD	20
14	7934	MILLER	10

The psycopg2 driver version should be equal to or above 2.8 for updatable query result sets to work.

An updatable result set is identical to the **Data Grid** in View/Edit Data mode, and can be modified in the same way.

If Auto-commit is off, the data changes are made as part of the ongoing transaction, if no transaction is ongoing a new one is initiated. The data changes are not committed to the database unless the transaction is committed.

If any errors occur during saving (for example, trying to save NULL into a column with NOT NULL constraint) the data changes are rolled back to an automatically created SAVEPOINT to ensure any previously executed queries in the ongoing transaction are not rolled back.

All rowsets from previous queries or commands that are displayed in the **Data Output** panel will be discarded when you invoke another query; open another query tool browser tab to keep your previous results available.

Explain Panel

To generate the **Explain** or **Explain Analyze** plan of a query, click on **Explain** or **Explain Analyze** button in the toolbar.

More options related to **Explain** and **Explain Analyze** can be selected from the drop down on the right side of **Explain Analyze** button in the toolbar.



Please note that pgAdmin generates the **Explain [Analyze]** plan in JSON format.

On successful generation of **Explain** plan, it will create three tabs/panels under the Explain panel.

- Graphical

Please note that **EXPLAIN VERBOSE** cannot be displayed graphically. Click on a node icon on the **Graphical** tab to review information about that item; a popup window will display on the right side with the information about the selected object. For information on JIT statistics, triggers and a summary, Click on the button top-right corner; a similar popup window will be displayed when appropriate.

Use the download button on top left corner of the **Explain** canvas to download the plan as an SVG file.

!!! Note Download as SVG is not supported on Internet Explorer.



Note that the query plan that accompanies the **Explain analyze** is available on the **Data Output** tab.

- Table

Table tab shows the plan details in table format, it generates table format similar to explain.depesz.com. Each row of the table represent the data for a **Explain Plan Node**. It may contain the node information, exclusive timing, inclusive timing, actual vs planned rows differences, actual rows, planned rows, loops.

background color of the exclusive, inclusive, and Rows X columns may vary based on the difference between actual vs planned.

If percentage of the exclusive/inclusive timings of the total query time is: > 90 - Red color > 50 - Orange (between red and yellow) color > 10 - Yellow color

If planner mis-estimated number of rows (actual vs planned) by 10 times - Yellow color 100 times - Orange (between Red and Yellow) color 1000 times - Red color

#	Node	Timings	Rows		
		Exclusive	Inclusive	Actual	Loops
1.	→ Unique (actual=56.576, 58.105 rows<3510 loops=1)	1.203 ms	58.105 ms	3510	1
2.	→ Sort (actual=56.574, 56.902 rows>5340 loops=1)	9.964 ms	56.902 ms	5340	1
3.	→ Nested Loop Left Join (actual=13.766, 46.939 rows>5340 loops=1)	5.186 ms	46.939 ms	5340	1
4.	→ Hash Inner Join (actual=13.751, 41.754 rows>5340 loops=1) Hash Cond: (dep.refclassid = pg.class.oid)	1.714 ms	41.754 ms	5340	1
5.	→ Hash Left Join (actual=13.599, 39.904 rows>7787 loops=1) Hash Cond: (pr.proptype = prtyp.oid)	2.009 ms	39.904 ms	7787	1
6.	→ Hash Left Join (actual=13.476, 37.783 rows>7787 loops=1) Hash Cond: (dep.refoid = fdw.oid)	1.703 ms	37.783 ms	7787	1
7.	→ Hash Left Join (actual=13.451, 36.079 rows>7787 loops=1) Hash Cond: (dep.refobjid = fs.oid)	1.733 ms	36.079 ms	7787	1
8.	→ Hash Left Join (actual=13.44, 34.345 rows>7787 loops=1) Hash Cond: ((cat.attrelid = ad.adrelid) AND (att.attnum = ad.adnum))	1.84 ms	34.345 ms	7787	1
9.	→ Merge Left Join (actual=13.42, 32.497 rows>7787 loops=1)	1.537 ms	32.497 ms	7787	1

- Statistics

Statistics tab shows two tables: 1. Statistics per Plan Node Type 2. Statistics per Table

Statistics per Node Type			
Node type	Count	Time spent	% of query
Hash	12	3.164 ms	5.45%
Hash Inner Join	1	1.714 ms	2.95%
Hash Left Join	10	12.938 ms	22.27%
Hash Right Join	1	0.062 ms	0.11%
Index Only Scan	1	0 ms	0%
Index Scan	3	0.065 ms	0.12%
Materialize	4	0.027 ms	0.05%
Merge Left Join	7	11.159 ms	19.21%
Nested Loop Left Join	4	12.214 ms	21.03%
Seq Scan	20	2.215 ms	3.82%
Sort	7	13.36 ms	23%
Unique	1	1.203 ms	2.08%

Statistics per Table			
Table name	Scan count	Total time	% of query
pg.attrelid	1	0.004 ms	0.01%
Seq Scan	1	0.004 ms	100%
pg.attribute	1	0.593 ms	1.03%
Seq Scan	1	0.593 ms	100%
pg.class	4	0.235 ms	0.41%
Index Scan	1	0.038 ms	16.18%
Seq Scan	3	0.197 ms	83.83%
pg.constraint	1	0.022 ms	0.04%
Index Scan	1	0.022 ms	100%
pg.depend	1	0.586 ms	1.01%
Seq Scan	1	0.586 ms	100%

Messages Panel

Use the **Messages** tab to view information about the most recently executed query:

```

1 SELECT * FROM pg.roles

```

ERROR: relation "pg.roles" does not exist
LINE 1: SELECT * FROM pg.roles

If the server returns an error, the error message will be displayed on the **Messages** tab, and the syntax that caused the

error will be underlined in the SQL editor. If a query succeeds, the **Messages** tab displays how long the query took to complete and how many rows were retrieved:

The screenshot shows the Query Editor interface with the following details:

- Toolbar:** Standard database management icons.
- Session:** edbstore/postgres@PGSQL_12
- Tab:** Query History (selected).
- Text Area:**

```

1 SELECT DISTINCT dep.deptype, dep.refclassid, cl.relkind, ad.adbin,
2 CASE WHEN cl.relkind IS NOT NULL THEN cl.relkind || COALESCE(dep.refobjsubid::character varying, '')
3 WHEN tg.oid IS NOT NULL THEN 't'::text
4 WHEN ty.oid IS NOT NULL AND ty.typbasetype = 0 THEN 'y'::text
5 WHEN ty.oid IS NOT NULL AND ty.typbasetype != 0 THEN 'd'::text
6 WHEN ns.oid IS NOT NULL THEN 'n'::text
7 WHEN pr.oid IS NOT NULL AND prtyp.typname = 'trigger' THEN 't'::text
8 WHEN pr.oid IS NOT NULL THEN 'P'::text
9 WHEN la.oid IS NOT NULL THEN 'l'::text
10 WHEN rw.oid IS NOT NULL THEN 'R'::text
11 WHEN co.oid IS NOT NULL THEN 'C'::text || contype
12 WHEN ad.oid IS NOT NULL THEN 'A'::text
13 WHEN fs.oid IS NOT NULL THEN 'F'::text
14 WHEN fdw.oid IS NOT NULL THEN 'f'::text
15 ELSE ''

```
- Bottom Bar:** Data Output, Explain, Messages (selected), Notifications.
- Messages:** Successfully run. Total query runtime: 61 msec.
1 rows affected.

Query tool output information

Query History Panel

Use the **Query History** tab to review activity for the current session:

The screenshot shows the Query History panel with the following details:

- Toolbar:** Standard database management icons.
- Session:** edbstore/postgres@PGSQL_12
- Tab:** Query History (selected).
- Text Area:**

Date	Rows Affected	Duration
11/17/2020 3:42:01 PM		61 msec

```

SELECT DISTINCT dep.deptype, dep.refclassid, cl.r...
15:42:01
SELECT DISTINCT dep.deptype, dep.refclassid, cl.r...
15:34:04
▶ SELECT * FROM pg.roles
15:23:25
▶ SELECT * FROM pg_roles
15:23:11
▶ SELECT * FROM pg_database
15:21:09
▶ select empno, ename, deptno as department from edbu...

```
- Bottom Bar:** Copy, Copy to Query Editor.
- Messages:** Successfully run. Total query runtime: 61 msec.
1 rows affected.

The Query History tab displays information about recent commands:

- The date and time that a query was invoked.
- The text of the query.
- The number of rows returned by the query.
- The amount of time it took the server to process the query and return a result set.
- Messages returned by the server (not noted on the **Messages** tab).
- The source of the query (indicated by icons corresponding to the toolbar).

You can show or hide the queries generated internally by pgAdmin (during 'View/Edit Data' or 'Save Data' operations).

To erase the content of the **Query History** tab, select **Clear history** from the **Clear** drop-down menu.

Query History is maintained across sessions for each database on a per-user basis when running in Query Tool mode. In View/Edit Data mode, history is not retained. By default, the last 20 queries are stored for each database. This can be adjusted in config_local.py by overriding the MAX_QUERY_HIST_STORED value.

Connection Status

Use the **Connection status** feature to view the current connection and transaction status by clicking on the status

icon in query tool:



Change connection +*****

User can connect to another server or database from existing open session of query tool.

- Click on the connection link next to connection status.
- Now click on the <New Connection> option from the dropdown.



- Now select server, database, user, and role to connect and click OK.



- A newly created connection will now get listed in the options.
- To connect, select the newly created connection from the dropdown list.

Macros

Query Tool Macros enable you to execute pre-defined SQL queries with a single key press. Pre-defined queries can contain the placeholder \$SELECTION\$. Upon macro execution, the placeholder will be replaced with the currently selected text in the Query Editor pane of the Query Tool.



To create a macro, select the *Manage Macros* option from the *Macros* menu on the *Query Tool*. Select the key you wish to use, enter the name of the macro, and the query, optionally including the selection placeholder, and then click the *Save* button to store the macro.



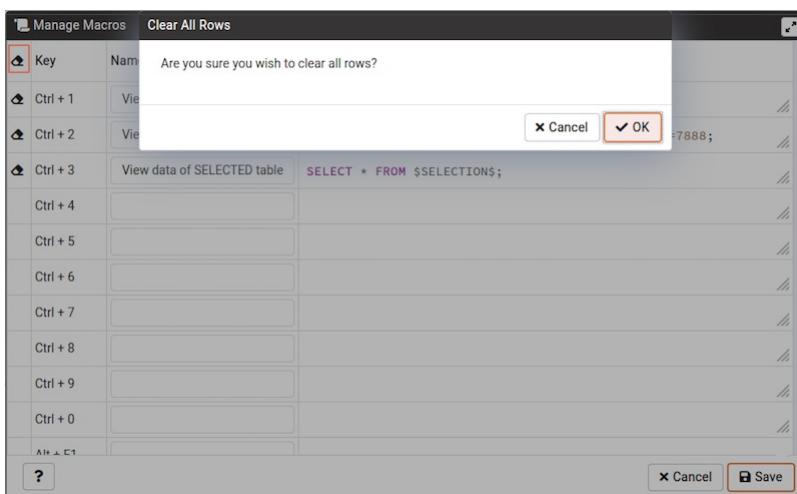
To clear a macro, select the macro on the *Manage Macros* dialogue, and then click the *Clear* button.



The server will prompt you for confirmation to clear the macro.



To clear all macros, click on the *Clear* button on left side of the key. The server will prompt you for confirmation to clear all the rows.



To execute a macro, simply select the appropriate shortcut keys, or select it from the *Macros* menu.

The screenshot shows the main interface of EDB Postgres Enterprise Manager. At the top is a navigation bar with links like Dashboard, Properties, SQL, Statistics, Dependencies, Dependents, Monitoring, and a connection dropdown. Below is a toolbar with various icons. The central area has tabs for Query Editor and Query History, with the Query Editor active. A scratch pad tab is also visible. The bottom section shows a table output with columns deptno, dname, and loc. The data is:

deptno	dname	loc
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

A message at the bottom right indicates: 'Successfully run. Total query runtime: 30 msec. 4 rows affected.'

7.8.3 Interpreting Graphical Query Plans

The graphical explain plan provides clues that can help you identify the aspects of the selected query that consume the most resources; within the diagram, thicker lines indicate the portions of the query that are expected to take the most processing time.

To view a graphical interpretation of an executed query, select **Explain** or **Explain Analyze** from the **Execute/Refresh** drop-down menu. Please note that you can use the **Explain Options** pull-aside menu to specify the level of detail displayed for each node.



Hover over an icon within the plan to view details for the selected node:



Each query operator (within the selected query) is represented in the graphical display by an icon. The table below describes the Advanced Server query operators:

Icon	Represents	Description
	Result Set	The Result Set icon represents a simple result set in the query plan. Typically, a Result Set operator is used to evaluate a query that does not fetch data from a table.
	Aggregate	The server creates an Aggregate operator whenever the query invokes an aggregate function (a function that returns a value computed from multiple rows). Aggregate functions include: AVG(), COUNT(), MAX(), MIN(), STDDEV(), SUM(), and VARIANCE().
	Window Aggregate	The server may use a Window aggregate operator to implement windowed aggregate functions; a windowed aggregate function is a function that returns a value computed from a set of rows within the input.

Icon	Represents	Description
	Seek	The server may use a Seek operator in any plan that includes an Index Scan operator. The Seek operator represents a probe into the heap to fetch the tuple that corresponds to an index entry (found by the Index Scan operator).
	Seq Scan	The server may use a Seq scan (sequential scan) to read through a table from beginning to end.
	Index Scan	The server may use an Index scan operator to read through a table in the order specified in the index.
	CTE Scan	The server may use a CTE Scan operator if the query performs a scan of a common table expression.
	Tuple ID Scan	The server uses a Tuple ID scan if the query uses the Tuple ID (ctid) as a constraint in a WHERE clause.
	Group	The server may use a Group operator when the query includes a GROUP BY clause. The operator requires a single input set ordered by the target column(s).
	Sort	The server may use a Sort operator when a query includes an ORDER BY clause to impose an order on a result set.
	Limit	The server may use the Limit operator to limit the size of a result set (when a query includes the LIMIT or OFFSET clause).
	Sub Plan	The server may use a Subplan operator for queries that include subselects.
	Unique	The server may use the Unique operator to remove duplicate values from a result set; the diagram will include a Unique operator if the query includes a DISTINCT clause.
	Hash	The server may use a Hash operator when joining two input sets that are not ordered by the column that controls the join.
	Hash Semi-Join	The server may use a Hash Semi-Join operator to evaluate a query that joins two tables, but returns data from only one of those tables.
	Hash Anti-Join	The server may use a Hash Anti-Join operator to evaluate a query that includes a NOT IN clause.
	Anti-Join	The server may use an Anti-Join operator to evaluate a query that includes a NOT IN clause.
	Join	The server may use a Join operator when joining two input sets that are ordered by the column that controls the join.
	Recursive Union	The server may use a Recursive Union operator if the query includes a WITH RECURSIVE clause.
	Set Operator	The server may use a Set operator if the query includes an INTERSECT, INTERSECT ALL, EXCEPT or EXCEPT ALL clause.
	Hash Set Operator (Setop) Intersect	The server may use an Intersect Set operator if the query includes an INTERSECT clause.
	Hash Set Operator (Setop) Intersect All	The server may use an Intersect Set operator if the query includes an INTERSECT ALL.
	Hash Set Operator (Setop) Except	The server may use an Except Set operator if the query includes an EXCEPT.

Icon	Represents	Description
	Hash Set Operator (Setop) Except All	The server may use an Except All Set operator if the query includes an EXCEPT ALL clause.
	Materialize	The server may choose to use a Materialize operator for a subselect operation (a nested query).
	Append	The server may use an Append operator to implement queries that contain a UNION clause.
	Nested Loop	The server may use a Nested Loop operator to perform a join between two tables. When implementing a nested loop, the server searches for rows from the inner table that match the corresponding row in the outer table.
	Merge Join	The server may use a Merge Join operator to join two tables. A Merge Join requires two sets of inputs, where each set is ordered by the column used for the comparison.
	Merge Semi-Join	The server may use a Merge Semi-Join operator to evaluate a query that joins two tables, but returns data from only one of those tables.
	Merge Anti-Join	The server may use a Merge Anti-Join operator to evaluate a query that includes a NOT IN clause.
	Nested Loop Semi-Join	The server may use a Nested Semi-Join operator to evaluate a query that joins two tables, but returns data from only one of those tables.
	Nested Loop Anti-Join	The server may use a Nested Anti-Join operator to evaluate a query that includes a NOT IN clause.
	Bitmap Index	The server may use a Bitmap Index operator when locating a subset of rows in an indexed table.
	Bitmap Heap	The server may use a Bitmap Heap operator when the query returns a subset of rows from an indexed table.

While you cannot directly specify the execution plan of a query, you can use indexes, configuration parameters and optimizer hints to direct Advanced Server as it selects from the query plans presented by the server. See the following resources for more information about query optimization:

- For more information about interpreting and understanding a query plan, see Using EXPLAIN, in the PostgreSQL documentation.
- For information about using PEM's Index Advisor, see the EDB Postgres Advanced Server Guide, available from the EnterpriseDB website at www.enterprisedb.com.
- For information about using configuration parameters to influence query plans, see Query Planning, in the PostgreSQL documentation.
- For more information about using Oracle-compatible optimizer hints, see the EDB Postgres Advanced Server Oracle Compatibility Developer's Guide, available from the EnterpriseDB website at www.enterprisedb.com.

7.8.4 Reviewing and Editing Data

To review or modify data, right click on a table or view name in the **Browser** tree control. When the context menu opens, use the **View/Edit Data** menu to specify the number of rows you would like to display in the editor panel.

The screenshot shows the EDB Postgres Enterprise Manager interface. At the top, there's a toolbar with various icons for file operations like save, open, and print, along with search and filter options. Below the toolbar is a header bar with the connection information "pem.probe/pem/enterprisedb@Postgres Enterprise Manager Server". Underneath the header, there are tabs for "Query Editor" (which is selected) and "Query History". The main area contains a SQL query editor with the following code:

```
1  SELECT * FROM pem.probe
2
```

Below the query editor is a "Data Output" panel containing a table with 12 rows of data. The columns are:

	id [PK] integer	display_name text	internal_name text	collection_method character (1)	target_type_id integer	applies_to_id integer	agent_capability text
1		Object Catalog: Tabl...	oc_tablespace	s	200	200	[null]
2		Object Catalog: Dat...	oc_database	s	200	300	[null]
3		Object Catalog: Sch...	oc_schema	s	300	400	[null]
4		Object Catalog: Table	oc_table	s	400	500	[null]
5		Object Catalog: Index	oc_index	s	400	600	[null]
6		Object Catalog: For...	oc_foreign_key	s	400	400	[null]
7		Object Catalog: Seq...	oc_sequence	s	400	700	[null]
8		Object Catalog: Fun...	oc_function	s	400	800	[null]
9		Server Information	server_info	s	200	200	[null]
10		Database Statistics	database_statistics	s	200	300	[null]
11		Tablespace Size	tablespace_size	s	200	200	[null]
12		Database Size	database_size	s	200	300	[null]

To modify the content of a table, each row in the table must be uniquely identifiable. If the table definition does not include an OID or a primary key, the displayed data is read only. Note that views cannot be edited; updatable views (using rules) are not supported.

The editor features a toolbar that allows quick access to frequently used options, and a work environment divided into two panels:

- The upper panel displays the SQL command that was used to select the content displayed in the lower panel.
- The lower panel (the Data Grid) displays the data selected from the table or view.

The View/Edit Data Toolbar

The toolbar includes context-sensitive icons that provide shortcuts to frequently performed tasks.



Hover over an icon to display a tooltip that describes the icon's functionality.

Icon	Behavior	Shortcut
Save	Use the Save icon to save your changes to the currently displayed table contents.	
Find	Use options on the Find menu to access Search and Replace functionality or to Jump to another line.	Ctrl/Cmd +F
Copy	Click the Copy icon to copy the currently selected data.	Ctrl+C
Paste Row	Click the Paste Row icon to paste the content that is currently on the clipboard.	
Delete Row	Use the Delete Row icon to delete all the selected rows from the output panel.	

Icon	Behavior	Shortcut
	Click the Filter icon to open a dialog that allows you to write and apply a filter for the content currently displayed in the output panel. Click the down arrow to open the Filter drop-down menu and select from pre-defined options:	
Filter	Use options on the Filter menu to quick-sort or quick-filter the data set: - Filter: This option opens a dialog that allows you to define a filter. A filter is a condition that is supplied to an arbitrary WHERE clause that restricts the result set. - Remove Filter: This option removes all selection / exclusion filter conditions. - By Selection: This option refreshes the data set and displays only those rows whose column value matches the value in the cell currently selected. - Exclude Selection: This option refreshes the data set and excludes those rows whose column value matches the value in the cell currently selected.	
No limit	Use the No limit drop-down listbox to specify how many rows to display in the output panel. Select from: No limit (the default), 1000 rows , 500 rows , or 100 rows .	
Execute/Refresh	Click the Execute/Refresh icon to execute the SQL command that is displayed in the top panel. If you have not saved modifications to the content displayed in the data grid, you will be prompted to confirm the execution. To preserve your changes before refreshing the content, click the Save toolbar button before executing the refresh.	F5
Stop	Click the Stop icon to cancel the execution of the currently running query.	
Clear History	Use the Clear History drop-down menu to erase the contents of the History tab.	
Download as CSV	Click the Download as CSV icon to download the result set of the current query to a comma-separated list. You can control the CSV settings through Preferences -> SQL Editor -> CSV output dialogue.	F8

The Data Grid

The top row of the data grid displays the name of each column, the data type, and if applicable, the number of characters allowed. A column that is part of the primary key will additionally be marked with [PK].

To modify the displayed data:

- To change a numeric value within the grid, double-click the value to select the field. Modify the content in the square in which it is displayed.
- To change a non-numeric value within the grid, double-click the content to access the edit bubble. After modifying the content of the edit bubble, click the **Save** button to display your changes in the data grid, or **Cancel** to exit the edit bubble without saving.

To enter a newline character, click Ctrl-Enter or Shift-Enter. Newline formatting is only displayed when the field content is accessed via an edit bubble.

To add a new row to the table, enter data into the last (unnumbered) row of the table. As soon as you store the data, the row is assigned a row number, and a fresh empty line is added to the data grid.

To write a SQL NULL to the table, simply leave the field empty. When you store the new row, the will server fill in the default value for that column. If you store a change to an existing row, the value NULL will explicitly be written.

To write an empty string to the table, enter the special string " (two single quotes) in the field. If you want to write a string containing solely two single quotes to the table, you need to escape these quotes, by typing "

To delete a row, press the **Delete** toolbar button. A popup will open, asking you to confirm the deletion.

To commit the changes to the server, select the **Save** toolbar button. Modifications to a row are written to the server automatically when you select a different row.

Geometry Data Viewer

If PostGIS is installed, you can view GIS objects in a map by selecting row(s) and clicking the 'View Geometry' button in the column. If no rows are selected, the entire data set will be rendered:

	id	name	address	city	state	zip
1	1	1369 Coffee House	1369 Cambridge St	Cambridge	MA	02105
2	2	1369 Coffee House	757 Massachusetts Ave	Cambridge	MA	02105
3	3	Aciutina Cafe	605 W Kendall St	Cambridge	MA	02105
4	4	Al's Deli-Cafe-Cambridge	1354 Massachusetts Ave	Cambridge	MA	02105
5	5	Algiers Coffee	40 Brattle St # 3	Cambridge	MA	02105
6	6	Area Four	500 Technology Sq	Cambridge	MA	02105
7	7	B & B Snack Bar	55 Broadway	Cambridge	MA	02105
8	8	Basha Cafe	26 New St	Cambridge	MA	02105
9	9	Beantowne Coffee House	1 Kendall Sq # B2105	Cambridge	MA	02105
10	10	Boston Tea Stop	54 JFK St # 1	Cambridge	MA	02105
11	11	Broadway Marketplace LLC	468 Broadway	Cambridge	MA	02105
12	12	Cafe Avec	2269 Massachusetts Ave # 1	Cambridge	MA	02105
13	13	Cafe Pamplona	12 Bow St	Cambridge	MA	02105
14	14	Cafe Zing	25 White St	Cambridge	MA	02105
15	15	Cambridge Coffee Shop	847 Cambridge St	Cambridge	MA	02105
16	16	Central Brew Cafe & Espresso	350 Massachusetts Ave	Cambridge	MA	02105
17	17	Conroy Commons	13 Appian Way	Cambridge	MA	02105
18	18	Crema Cafe	27 Brattle St	Cambridge	MA	02105

You can adjust the layout by dragging the title of the panel. To view the properties of the geometries directly in map, just click the specific geometry:

id	25
name	Dunkin' Donuts
address	1 White St
city	Cambridge
state	MA
zip	02140
lat	42.38863

Notes:

- Supported data types:** The Geometry Viewer supports 2D and 3DM geometries in EWKB format including Point, LineString, Polygon, MultiPoint, MultiLineString, MultiPolygon and GeometryCollection.
- SRIDs:** If there are geometries with different SRIDs in the same column, the viewer will render geometries with the

same SRID in the map. If SRID=4326 the OSM tile layer will be added into the map.

- **Data size:** For performance reasons, the viewer will render no more than 100000 geometries, totaling up to 20MB.
- **Internet access:** An internet connection is required for the Geometry Viewer to function correctly.

Sort/Filter options dialog

You can access **Sort/Filter options dialog** by clicking on Sort/Filter button. This allows you to specify an SQL Filter to limit the data displayed and data sorting options in the edit grid window:



- Use **SQL Filter** to provide SQL filtering criteria. These will be added to the "WHERE" clause of the query used to retrieve the data. For example, you might enter:

```
id > 25 AND created > '2018-01-01'
```

- Use **Data Sorting** to sort the data in the output grid

To add new column(s) in data sorting grid, click on the [+] icon.

- Use the drop-down **Column** to select the column you want to sort.
- Use the drop-down **Order** to select the sort order for the column.

To delete a row from the grid, click the trash icon.

- Click the **Help** button (?) to access online help.
- Click the **Ok** button to save work.
- Click the **Close** button to discard current changes and close the dialog.

7.8.4.1 View/Edit Data Filter

You can access **Data Filter dialog** by clicking on **Filtered Rows** toolbar button visible on the Browser panel or by selecting *View/Edit Data -> Filtered Rows* context menu option.

This allows you to specify an SQL Filter to limit the data displayed in the edit grid window:



Note

Use SHIFT + ENTER keys to apply filter.

7.8.5 Schema Diff

Schema Diff is a feature that allows you to compare objects between two database or two schemas. Use the **Tools** menu to access Schema Diff.

The Schema Diff feature allows you to:

- Compare and synchronize the database objects (from source to target).
- Visualize the differences between database objects.
- List the differences in SQL statement for target database objects.
- Generate synchronization scripts.

!!! Note > - The source and target database servers must be of the same major version. > - If you compare two schemas then dependencies won't be resolved.

Click on *Schema Diff* under the **Tools** menu to open a selection panel. To compare databases choose the source and target servers, and databases. To compare schemas choose the source and target servers, databases, and schemas. After selecting the objects, click on the *Compare* button.

You can open multiple copies of **Schema Diff** in individual tabs simultaneously. To close a copy of Schema Diff, click the *X* in the upper-right hand corner of the tab bar. You can rename the panel title by right-clicking and select the "Rename Panel" option.

The screenshot shows the Schema Diff (Beta) interface. In the top navigation bar, the 'Schema Diff (Beta)' tab is selected. Below it, 'Select Source' and 'Select Target' dropdowns are set to 'PostgreSQL 11' and 'postgres' respectively, with 'source_sc' and 'target_sc' selected as the schemas. A 'Compare' button is highlighted in orange. To the right are 'Generate Script' and 'Filter' buttons. The main area is divided into two panels: 'Schema Objects' on the left and 'Comparison Result' on the right. Under 'Schema Objects', there's a tree view with nodes like 'abstract', 'asset', and numerous 'col1' through 'col106' entries. The 'Comparison Result' panel shows a table where most objects are marked as 'Different'. At the bottom, a 'DDL Comparison' section displays the SQL scripts for creating and altering collations on both the Source and Target sides.

Use the [Preferences](#) dialog to specify following:

- *Schema Diff* should open in a new browser tab. Set *Open in new browser tab* option to true.
- *Schema Diff* should ignore the whitespaces while comparing string objects. Set *Ignore whitespaces* option to true.
- *Schema Diff* should ignore the owner while comparing objects. Set *Ignore owner* option to true.

The **Schema Diff** panel is divided into two panels; an Object Comparison panel and a DDL Comparison panel.

The Schema Diff Object Comparison Panel

In the object comparison panel, you can select the source and target servers of the same major version, and databases to be compared. You can select any server listed under the browser tree whether it is connected or disconnected. If you select a server that is not connected then it will prompt you for the password before using the server.

Next, select the databases that will be compared. The databases can be the same or different (and within the same server or from different servers).

The screenshot shows the Schema Diff (Beta) interface in its initial state. The top navigation bar has the 'Schema Diff (Beta)' tab selected. Below it, 'Select Source' and 'Select Target' dropdowns are both set to 'PostgreSQL 11' and 'postgres'. The 'source_sc' and 'target_sc' schemas are also selected. The 'Compare' button is not yet highlighted. On the right side, there's a note: 'Select the server, database and schema for the source and target and click Compare to compare them.'

After you select servers and databases, click on the **Compare** button to obtain the **Comparison Result**.

Object Type	Object Name	Status
Collations	abstract	Source Only
Collations	asset	Target Only
Collations	col1	Different
Collations	col10	Different
Collations	col100	Different
Collations	col1000	Different
Collations	col101	Different
Collations	col102	Different
Collations	col103	Different
Collations	col104	Different
Collations	col105	Different
Domains	fts1	Identical
Domains	fts10	Identical
Domains	fts100	Identical
Domains	fts1000	Identical
Domains	fts101	Identical
Domains	fts102	Identical
Domains	fts103	Identical
Foreign Tables	fts1	Identical
Foreign Tables	fts10	Identical
Foreign Tables	fts100	Identical
Foreign Tables	fts1000	Identical
Foreign Tables	fts101	Identical
Foreign Tables	fts102	Identical
Foreign Tables	fts103	Identical
FTS Configurations	fts1	Identical
FTS Configurations	fts10	Identical
FTS Configurations	fts100	Identical
FTS Configurations	fts1000	Identical
FTS Configurations	fts101	Identical
FTS Configurations	fts102	Identical
FTS Configurations	fts103	Identical
FTS Dictionaries	fts1	Identical
FTS Dictionaries	fts10	Identical
FTS Dictionaries	fts100	Identical
FTS Dictionaries	fts1000	Identical
FTS Dictionaries	fts101	Identical
FTS Dictionaries	fts102	Identical
FTS Dictionaries	fts103	Identical

Use the drop-down lists of Database Objects to view the DDL statements.

In the upper-right hand corner of the object comparison panel is a *Filter* option that you can use to filter the database objects based on the following comparison criteria:

- Identical – If the object is found in both databases with the same SQL statement, then the comparison result is identical.
- Different – If the object is found in both databases but have different SQL statements, then the comparison result is different.
- Source Only – If the object is found in source database only and not in target database, then the comparison result is source only.
- Target Only – If the object is found in target database only and not in source database, then the comparison result is target only.

Object Type	Object Name	Status
Collations	abstract	Source Only
Collations	asset	Target Only
Collations	col1	Different
Collations	col10	Different
Collations	col100	Different
Collations	col1000	Different
Collations	col101	Different
Collations	col102	Different
Collations	col103	Different
Collations	col104	Different
Collations	col105	Different

Click on any of the database objects in the object comparison panel to display the DDL Statements of that object in the DDL Comparison panel.

Schema Diff DDL Comparison Panel

The **DDL Comparison** panel displays three columns:

- The first column displays the DDL statement of the object from the source database.
- The second column displays the DDL statement of the object from the target database.
- The third column displays the difference in the SQL statement of the target database object.

The screenshot shows the Schema Diff (Beta) interface. In the 'Schema Objects' tree, under 'Collations', there are 1000 differences for 'coll1'. The 'Comparison Result' table shows the status for each object: 'Source Only', 'Target Only', or 'Different'. The 'DDL Comparison' panel displays the SQL code for both 'Source' and 'Target' databases, highlighting the differences in the 'Difference' section.

```

Source
1 -- Collation: coll;
2
3 -- DROP COLLATION source_sc.coll;
4
5 CREATE COLLATION source_sc.coll
6   (LC_COLLATE = 'POSIX', LC_CTYPE = 'POSIX');
7
8 ALTER COLLATION source_sc.coll
9   OWNER TO postgres;

Target
1 -- Collation: coll;
2
3 -- DROP COLLATION target_sc.coll;
4
5 CREATE COLLATION target_sc.coll
6   (LC_COLLATE = 'C', LC_CTYPE = 'C');
7
8 ALTER COLLATION target_sc.coll
9   OWNER TO postgres;

```

Difference:

```

1 -- WARNING:
2 -- We have found the difference in either of LC,
3 -- so we need to drop the existing collation file
4 DROP COLLATION target_sc.coll;
5
6 CREATE COLLATION target_sc.coll
7   (LC_COLLATE = 'POSIX', LC_CTYPE = 'POSIX');

```

You can review the DDL statements of all the database objects to check for the differences in the SQL statements.

Also, you can generate the SQL script of the differences found in the target database object based on the SQL statement of the source database object. To generate the script, select the checkboxes of the database objects in the object comparison panel and then click on the *Generate Script* button in the upper-right hand corner of the object comparison panel.

The screenshot shows the Schema Diff (Beta) interface. In the 'Schema Objects' tree, under 'Collations', there are 1000 differences for 'coll1'. The 'Comparison Result' table shows the status for each object: 'Source Only', 'Target Only', or 'Different'. The 'DDL Comparison' panel displays the SQL code for both 'Source' and 'Target' databases, highlighting the differences in the 'Difference' section. A 'Copy' button is visible in the 'Difference' panel.

```

Source
1 -- Collation: coll;
2
3 -- DROP COLLATION source_sc.coll;
4
5 CREATE COLLATION source_sc.coll
6   (LC_COLLATE = 'POSIX', LC_CTYPE = 'POSIX');
7
8 ALTER COLLATION source_sc.coll
9   OWNER TO postgres;

Target
1 -- Collation: coll;
2
3 -- DROP COLLATION target_sc.coll;
4
5 CREATE COLLATION target_sc.coll
6   (LC_COLLATE = 'C', LC_CTYPE = 'C');
7
8 ALTER COLLATION target_sc.coll
9   OWNER TO postgres;

```

Difference:

```

1 -- WARNING:
2 -- We have found the difference in either of LC,
3 -- so we need to drop the existing collation file
4 DROP COLLATION target_sc.coll;
5
6 CREATE COLLATION target_sc.coll
7   (LC_COLLATE = 'POSIX', LC_CTYPE = 'POSIX');

```

Select the database objects and click on the *Generate Script* button to open the **Query Tool** in a new tab, with the difference in the SQL statement displayed in the **Query Editor**.

If you have clicked on the database object to check the difference generated in the **DDL Comparison** Panel, and you have not selected the checkbox of the database object, PEM will open the **Query Tool** in a new tab, with the differences in the SQL statements displayed in the **Query Editor**.

You can also use the **Copy** button to copy the difference generated in the **DDL Comparison** panel.



```

1 -- This script was generated by a beta version of the Schema Diff utility in Postgres Enterprise Manager.
2 -- This version does not include dependency resolution, and may require manual changes
3 -- to the script to ensure changes are applied in the correct order.
4 -- Please report an issue for any failure with the reproduction steps.
5 v BEGIN;
6 -- WARNING:
7 -- We have found the difference in either of LC_COLLATE or LC_CTYPE or LOCALE,
8 -- so we need to drop the existing collation first and re-create it.
9 DROP COLLATION target_sc.col100;
10
11 CREATE COLLATION target_sc.col100
12   (LC_COLLATE = 'POSIX', LC_CTYPE = 'POSIX');
13 END;

```

Apply the SQL Statement in the target database to synchronize the databases.

7.9 Configuring pgBouncer for use with PEM Agents

pgBouncer is a lightweight connection pooler for Postgres. You can use pgBouncer to limit the number of connections from the PEM Agent towards the Postgres Enterprise Manager (PEM) server on a non-Windows machine.

Contents:

7.9.1 Connecting PEM to pgBouncer

Each PEM agent connects to the PEM database server using the SSL certificates for each individual user. For example, an agent with ID#1 connects to the PEM database server using the agent1 user.



Prior to PEM version 7.5, the following limitations did not allow use of the connection pooler between the PEM server and PEM agent:

- * The PEM agent uses an SSL Certificate to connect the PEM database server.
- * It uses an individual user identifier when connecting to the PEM database server. EnterpriseDB has modified the PEM agent to allow the agent to use a common database user (instead of the dedicated agent users) to connect the PEM database server.



We recommend using pgBouncer versions equal to or later than version 1.9.0 as the connection pooler. Since versions 1.9.0 or later support cert authentication; PEM Agents can connect to pgBouncer using SSL certificates.

7.9.2 Preparing the PEM Server for pgBouncer Connections

You must configure the PEM database server to accept connections from pgBouncer; the following example demonstrates the steps required to prepare the PEM database server.

1. Create a dedicated user named pgbouncer on the PEM database server. For example:

```
pem=# CREATE USER pgbouncer PASSWORD 'ANY_PASSWORD' LOGIN;
CREATE ROLE
```

2. Create a user named pem_admin1 (a non-super user) with pem_admin and pem_agent_pool role membership on the PEM database server. For example:

```
pem=# CREATE USER pem_admin1 PASSWORD 'ANY_PASSWORD' LOGIN CREATEROLE;
CREATE ROLE
pem=# GRANT pem_admin, pem_agent_pool TO pem_admin1;
GRANT ROLE
```

3. Grant CONNECT privilege to the pgbouncer user on the pem database. For example:

```
pem=# GRANT CONNECT ON DATABASE pem TO pgbouncer ;GRANT USAGE ON SCHEMA pem TO
pgbouncer;
GRANT
```

4. Grant USAGE privilege to the pgbouncer user for the pem schema on the pem database. For example:

```
pem=# GRANT USAGE ON SCHEMA pem TO pgbouncer;
GRANT
```

5. Grant EXECUTE privilege to the pgbouncer user on the pem.get_agent_pool_auth(text) function in the pem database. For example:

```
pem=# GRANT EXECUTE ON FUNCTION pem.get_agent_pool_auth(text) TO pgbouncer;
GRANT
```

6. Use the pem.create_proxy_agent_user(varchar) function to create a user named pem_agent_user1 on the PEM database server. The function will create a user with the same name with a random password, and grant **pem_agent** and **pem_agent_pool** roles to the user. This allows pgBouncer to use a proxy user on behalf of the agent. For example:

```
pem=# SELECT pem.create_proxy_agent_user('pem_agent_user1');
create_proxy_agent_user
-----
(1 row)
```

7. Add the following entries to the start of the **pg_hba.conf** file of the PEM database server; this will allow pgBouncer user to connect to the pem database using the md5 authentication method. For example:

```
# Allow the PEM agent proxy user (used by
# pgbouncer) to connect to PEM server using
# md5

local pem pgbouncer,pem_admin1 md5
```

After configuring the PEM server, you should configure pgBouncer.

7.9.3 Configuring pgBouncer

You must configure pgBouncer to work with the PEM database server. In our example, we will run pgBouncer as the enterpriseDB system user. The following steps outline the process of configuring pgBouncer (version >= 1.9).

1. Open a terminal window and navigate into the pgBouncer directory.
2. Change the owner of the etc directory for pgBouncer (where pgbouncer.ini resides) to enterpriseDB, and change the directory permissions to 0700. For example:

```
$ chown enterpriseDB:enterpriseDB /etc/edb/pgbouncer1.9
$ chmod 0700 /etc/edb/pgbouncer1.9
```

3. Change the contents of the pgbouncer.ini or edb-pgbouncer.ini file as follows:

```
[databases]
;; Change the pool_size according to maximum connections allowed
;; to the PEM database server as required.
;; 'auth_user' will be used for authenticate the db user (proxy
;; agent user in our case)
pem = port=5444 host=/tmp dbname=pem auth_user=pgbouncer pool_size=80
pool_mode=transaction
* = port=5444 host=/tmp dbname=pem auth_user=pgbouncer pool_size=10
[pgbouncer]
logfile = /var/log/edb/pgbouncer1.9/edb-pgbouncer-1.9.log
pidfile = /var/run/edb/pgbouncer1.9/edb-pgbouncer-1.9.pid
listen_addr = *
;; Agent needs to use this port to connect the pem database now
listen_port = 6432
;; Require to support for the SSL Certificate authentications
;; for PEM Agents
client_tls_sslmode = require
;; These are the root.crt, server.key, server.crt files present
;; in the present under the data directory of the PEM database
;; server, used by the PEM Agents for connections.
client_tls_ca_file = /var/lib/edb/as11/data/root.crt
client_tls_key_file = /var/lib/edb/as11/data/server.key
client_tls_cert_file = /var/lib/edb/as11/data/server.crt
;; Use hba file for client connections
auth_type = hba
;; Authentication file, Reference:
;; https://pgbouncer.github.io/config.html#auth_file
auth_file = /etc/edb/pgbouncer1.9/userlist.txt
;; HBA file
auth_hba_file = /etc/edb/pgbouncer1.9/hba_file
;; Use pem.get_agent_pool_auth(TEXT) function to authenticate
;; the db user (used as a proxy agent user).
auth_query = SELECT * FROM pem.get_agent_pool_auth($1)
;; DB User for administration of the pgbouncer
admin_users = pem_admin1
;; DB User for collecting the statistics of pgbouncer
stats_users = pem_admin1
server_reset_query = DISCARD ALL
;; Change based on the number of agents installed/required
max_client_conn = 500
;; Close server connection if its not been used in this time.
;; Allows to clean unnecessary connections from pool after peak.
server_idle_timeout = 60
```

4. Use the following command to create and update the /etc/edb/pgbouncer1.9/userlist.txt authentication file for pgBouncer:

```
pem=# COPY (
    SELECT 'pgbouncer'::TEXT, 'pgbouncer_password'
    UNION ALL
    SELECT 'pem_admin1'::TEXT, 'pem_admin1_password'
) TO '/etc/edb/pgbouncer1.9/userlist.txt'
    WITH (FORMAT CSV, DELIMITER ' ', FORCE_QUOTE *);
COPY 2
```

NOTE: A super user cannot invoke the PEM authentication query function pem.get_proxy_auth(text). If the pem_admin user is a super user, you must add the password to the authentication file, which is enterprisedb in the above example.

1. Create an HBA file (/etc/edb/pgbouncer1.9/hba_file) for pgBouncer that contains the following content:

```
# Use authentication method md5 for the local connections to
# connect pem database & pgbouncer (virtual) database.
local pgbouncer all md5

# Use authentication method md5 for the remote connections to
# connect to pgbouncer (virtual database) using enterprisedb
# user.
host pgbouncer,pem pem_admin1 0.0.0.0/0 md5
# Use authentication method cert for the TCP/IP connections to
# connect the pem database using pem_agent_user1
hostssl pem pem_agent_user1 0.0.0.0/0 cert
```

2. Change the owner of the HBA file (/etc/edb/pgbouncer1.9/hba_file) to enterprisedb, and change the directory permissions to 0600. For example:

```
$ chown enterprisedb:enterprisedb /etc/edb/pgbouncer1.9/hba_file
$ chmod 0600 /etc/edb/pgbouncer1.9/hba_file
```

3. Enable the pgBouncer service, and start the service. For example:

```
$ systemctl enable edb-pgbouncer-1.9
Created symlink from /etc/systemd/system/multi-user.target.wants/edb-pgbouncer-1.9.service to /usr/lib/systemd/system/edb-pgbouncer-1.9.service.
$ systemctl start edb-pgbouncer-1.9
```

7.9.4 Configuring the PEM Agent to use pgBouncer

You can use an RPM package to install a PEM Agent; for detailed installation information, please see the PEM Installation Guide, available from the [EnterpriseDB website](#)

Please note that PEM Agent which is responsible for sending SNMP notifications should not be configured with pgBouncer. For Example - If default PEM Agent installed along with PEM Server is used for SNMP notifications, then it should not be configured with pgBouncer.

Configuring a New PEM Agent (installed using an RPM)

After using an RPM package to install the PEM agent, you will need to configure it to work it against a particular PEM database server. Use the following command:

```
$ PGSSLMODE=require PEM_SERVER_PASSWORD=pem_admin1_password
/usr/edb/pem/agent/bin/pemworker --register-agent --pem-server 172.16.254.22 --pem-port 6432 --pem-user pem_admin1 --pem-agent-user pem_agent_user1 --display-name *Agent_Name*
```

Postgres Enterprise Manager Agent registered successfully!

In above command, the command line argument --pem-agent-user instructs the agent to create an SSL certificate and key pair for the pem_agent_user1 database user in /root/.pem directory. For example:

```
/root/.pem/pem_agent_user1.crt  
/root/.pem/pem_agent_user1.key
```

They will be used by the PEM agent to connect to the PEM database server as pem_agent_user1. It will also create /usr/edb/pem/agent/etc/agent.cfg.

You will find a line mentioning the agent-user to be used in the agent.cfg configuration file. For example:

```
$ cat /usr/edb/pem/agent/etc/agent.cfg
[PEM/agent]
pem_host=172.16.254.22
pem_port=6432
agent_id=12
agent_user=pem_agent_user1
agent_ssl_key=/root/.pem/pem_agent_user1.key
agent_ssl_crt=/root/.pem/pem_agent_user1.crt
log_level=warning
log_location=/var/log/pem/worker.log
agent_log_location=/var/log/pem/agent.log
long_wait=30
short_wait=10
alert_threads=0
enable_smtp=false
enable_snmp=false
enable_webhook=false
max_webhook_retries=3
allow_server_restart=true
max_connections=0
connect_timeout=-1
connection_lifetime=0
allow_batch_probes=false
heartbeat_connection=false
```

Configuring an Existing PEM Agent (installed using an RPM)

If you are using an existing PEM agent, you can copy the SSL certificate and key files to the target machine, and reuse the files. You will need to modify the files, adding a new parameter and replacing some parameters in the existing `agent.cfg` file.

Add a line for `agent_user` to be used for the agent. For example:

```
agent_user=pem_agent_user1
```

Update the port to specify the pgBouncer port. For example:

```
pem_port=6432
```

Update the certificate and key path locations. For example:

```
agent_ssl_key=/root/.pem/pem_agent_user1.key
agent_ssl_crt=/root/.pem/pem_agent_user1.crt
```

Please note: as an alternative, you can run the agent self registration, but that will create a new agent id. If you do run the agent self-registration, you must replace the new agent id with existing id, and disable the entry for the new agent id in the pem.agent table. For example:

```
pem=# UPDATE pem.agent SET active = false WHERE id = *new_agent_id*;
UPDATE 1
```

Please keep a backup of the existing SSL certificate, key file, and agent configuration file.

7.10 pgAgent

pgAgent is a job scheduling agent for Postgres databases, capable of running multi-step batch or shell scripts and SQL tasks on complex schedules.

pgAgent is distributed independently. You can download pgAgent from the [download area](#) of the pgAdmin website.

Contents:

7.10.1 Using pgAgent

pgAgent is a scheduling agent that runs and manages jobs; each job consists of one or more steps and schedules. If two or more jobs are scheduled to execute concurrently, pgAgent will execute the jobs in parallel (each with its own thread).

A step may be a series of SQL statements or an operating system batch/shell script. Each step in a given job is executed when the previous step completes, in alphanumeric order by name. Switches on the **pgAgent Job** dialog (accessed through the **Properties** context menu) allow you to modify a job, enabling or disabling individual steps as needed.

Each job is executed according to one or more schedules. Each time the job or any of its schedules are altered, the next runtime of the job is re-calculated. Each instance of pgAgent periodically polls the database for jobs with the next runtime value in the past. By polling at least once every minute, all jobs will normally start within one minute of the specified start time. If no pgAgent instance is running at the next runtime of a job, it will run as soon as pgAgent is next started, following which it will return to the normal schedule.

When you highlight the name of a defined job in the browser tree control, the **Properties** tab of the main PEM window will display details about the job, and the **Statistics** tab will display details about the job's execution.

Security concerns

pgAgent is a very powerful tool, but does have some security considerations that you should be aware of:

Database password - **DO NOT** be tempted to include a password in the pgAgent connection string - on Unix systems it

may be visible to all users in 'ps' output, and on Windows systems it will be stored in the registry in plain text. Instead, use a libpq `~/ .pgpass` file to store the passwords for every database that pgAgent must access. Details of this technique may be found in the [PostgreSQL documentation on .pgpass file](#).

System/database access - all jobs run by pgAgent will run with the security privileges of the pgAgent user. SQL steps will run as the user that pgAgent connects to the database as, and batch/shell scripts will run as the operating system user that the pgAgent service or daemon is running under. Because of this, it is essential to maintain control over the users that are able to create and modify jobs. By default, only the user that created the pgAgent database objects will be able to do this - this will normally be the PostgreSQL superuser.

7.10.2 Installing pgAgent

pgAgent runs as a daemon on Unix systems, and a service on Windows systems. In most cases it will run on the database server itself - for this reason, pgAgent is not automatically configured when PEM is installed. In some cases however, it may be preferable to run pgAgent on multiple systems, against the same database; individual jobs may be targeted at a particular host, or left for execution by any host. Locking prevents execution of the same instance of a job by multiple hosts.

Database setup

Before using PEM to manage pgAgent, you must create the pgAgent extension in the maintenance database registered with PEM. To install pgAgent on a PostgreSQL host, connect to the `postgres` database, and navigate through the [Tools](#) menu to open the Query tool. For server versions 9.1 or later, and pgAgent 3.4.0 or later, enter the following command in the query window, and click the [Execute](#) icon:

```
CREATE EXTENSION pgagent;
```

This command will create a number of tables and other objects in a schema called 'pgagent'.

The database must also have the pl/pgsql procedural language installed - use the PostgreSQL `CREATE LANGUAGE` command to install pl/pgsql if necessary. To install pl/pgsql, enter the following command in the query window, and click the [Execute](#) icon:

```
CREATE LANGUAGE plpgsql;
```

Daemon installation on Unix

Note

pgAgent is available in Debian/Ubuntu (DEB) and Redhat/Fedora (RPM) packages for Linux users, as well as source code. See the [pgAdmin Website](#). for more information.

To install the pgAgent daemon on a Unix system, you will normally need to have root privileges to modify the system startup scripts. Modifying system startup scripts is quite system-specific so you should consult your system documentation for further information.

The program itself takes few command line options, most of which are only needed for debugging or specialised

configurations:

Usage:

```
/path/to/pgagent [options] <connect-string>
```

options:

- f run in the foreground (do not detach from the terminal)
- t <poll time interval in seconds (default 10)>
- r <retry period after connection abort in seconds (>=10, default 30)>
- s <log file (messages are logged to STDOUT if not specified)>
- l <logging verbosity (ERROR=0, WARNING=1, DEBUG=2, default 0)>

The connection string is a standard PostgreSQL libpq connection string (see the [PostgreSQL documentation on the connection string](#) for further details). For example, the following command line will run pgAgent against a server listening on the localhost, using a database called 'postgres', connecting as the user 'postgres':

```
/path/to/pgagent hostaddr=127.0.0.1 dbname=postgres user=postgres
```

Service installation on Windows

Note

pgAgent is available in a pre-built installer if you use [EnterpriseDB's PostgreSQL Installers](#). Use the StackBuilder application to download and install it. If installed in this way, the service will automatically be created and the instructions below can be ignored.

pgAgent can install itself as a service on Windows systems. The command line options available are similar to those on Unix systems, but include an additional parameter to tell the service what to do:

Usage:

```
pgAgent REMOVE <serviceName>
pgAgent INSTALL <serviceName> [options] <connect-string>
pgAgent DEBUG [options] <connect-string>
```

options:

- u <user or DOMAIN\user>
- p <password>
- d <displayname>
- t <poll time interval in seconds (default 10)>
- r <retry period after connection abort in seconds (>=10, default 30)>
- l <logging verbosity (ERROR=0, WARNING=1, DEBUG=2, default 0)>

The service may be quite simply installed from the command line as follows (adjust the path as required):

```
"C:\Program Files\pgAgent\bin\pgAgent" INSTALL pgAgent -u postgres -p secret
hostaddr=127.0.0.1 dbname=postgres user=postgres
```

You can then start the service at the command line using `net start pgAgent`, or from the `Services` control panel applet. Any logging output or errors will be reported in the Application event log. The DEBUG mode may be used to run pgAgent from a command prompt. When run this way, log messages will output to the command window.

7.10.3 Creating a pgAgent Job

pgAgent is a scheduling agent that runs and manages jobs; each job consists of steps and schedules.

To create or manage a job, use the Browser tree control to browse to the server on which the pgAgent database objects were created. The tree control will display a **pgAgent Jobs** node, under which currently defined jobs are displayed. To add a new job, right click on the **pgAgent Jobs** node, and select **Create pgAgent Job...** from the context menu.

When the pgAgent dialog opens, use the tabs on the **pgAgent Job** dialog to define the steps and schedule that make up a pgAgent job.



Use the fields on the **General** tab to provide general information about a job:

- Provide a name for the job in the **Name** field.
- Move the **Enabled** switch to the **Yes** position to enable a job, or **No** to disable a job.
- Use the **Job Class** drop-down to select a class (for job categorization).
- Use the **Host Agent** field to specify the name of a machine that is running pgAgent to indicate that only that machine may execute the job. Leave the field blank to specify that any machine may perform the job.

!!! Note It is not always obvious what value to specify for the Host Agent in order to target a job step to a specific machine. With pgAgent running on the required machines and connected to the scheduler database, you can use the following query to view the hostnames as reported by each agent:

```
SELECT jagstation FROM pgagent.pga_jobagent
```

Use the hostname exactly as reported by the query in the Host Agent field.

- Use the **Comment** field to store notes about the job.



Use the **Steps** tab to define and manage the steps that the job will perform. Click the Add icon (+) to add a new step; then click the compose icon (located at the left side of the header) to open the step definition dialog:



Use fields on the step definition dialog to define the step:

- Provide a name for the step in the **Name** field; please note that steps will be performed in alphanumeric order by name.
- Use the **Enabled** switch to include the step when executing the job (**True**) or to disable the step (**False**).
- Use the **Kind** switch to indicate if the job step invokes SQL code (**SQL**) or a batch script (**Batch**).
 - If you select **SQL**, use the **Code** tab to provide SQL code for the step.
 - If you select **Batch**, use the **Code** tab to provide the batch script that will be executed during the step.

Note

The fields **Connection type**, **Database** and **Connection string** are only applicable when **SQL** is selected because **Batch** cannot be run on remote servers.

- Use the **Connection type** switch to indicate if the step is performed on a local server (**Local**) or on a remote host (**Remote**). If you specify a remote connection should be used for the step, the **Connection string** field will be enabled, and you must provide a libpq-style connection string.
- Use the **Database** drop-down to select the database on which the job step will be performed.
- Use the **Connection string** field to specify a libpq-style connection string to the remote server on which the

step will be performed. For more information about writing a connection string, please see the [PostgreSQL documentation](#).

- Use the **On error** drop-down to specify the behavior of pgAgent if it encounters an error while executing the step.

Select from:

- **Fail** - Stop the job if you encounter an error while processing this step.
- **Success** - Mark the step as completing successfully, and continue.
- **Ignore** - Ignore the error, and continue.

- Use the **Comment** field to provide a comment about the step.



Use the context-sensitive field on the step definition dialog's **Code** tab to provide the SQL code or batch script that will be executed during the step:

- If the step invokes SQL code, provide one or more SQL statements in the **SQL query** field.
- If the step performs a batch script, provide the script in the **Script** field. If you are running on a Windows server, standard batch file syntax must be used. When running on a Linux server, any shell script may be used, provided that a suitable interpreter is specified on the first line (e.g. `#!/bin/sh`).

When you've provided all of the information required by the step, click the compose icon to close the step definition dialog. Click the add icon (+) to add each additional step, or select the **Schedules** tab to define the job schedule.



Click the Add icon (+) to add a schedule for the job; then click the compose icon (located at the left side of the header) to open the schedule definition dialog:



Use the fields on the schedule definition tab to specify the days and times at which the job will execute.

- Provide a name for the schedule in the **Name** field.
- Use the **Enabled** switch to indicate that pgAgent should use the schedule (**Yes**) or to disable the schedule (**No**).
- Use the calendar selector in the **Start** field to specify the starting date and time for the schedule.
- Use the calendar selector in the **End** field to specify the ending date and time for the schedule.
- Use the **Comment** field to provide a comment about the schedule.

Select the **Repeat** tab to define the days on which the schedule will execute.



Use the fields on the **Repeat** tab to specify the details about the schedule in a cron-style format. The job will execute on each date or time element selected on the **Repeat** tab.

Click within a field to open a list of valid values for that field; click on a specific value to add that value to the list of selected values for the field. To clear the values from a field, click the X located at the right-side of the field.

Use the fields within the **Days** box to specify the days on which the job will execute:

- Use the **Week Days** field to select the days on which the job will execute.
- Use the **Month Days** field to select the numeric days on which the job will execute. Specify the **Last Day** to indicate that the job should be performed on the last day of the month, regardless of the date.

- Use the **Months** field to select the months in which the job will execute.

Use the fields within the **Times** box to specify the times at which the job will execute:

- Use the **Hours** field to select the hour at which the job will execute.
- Use the **Minutes** field to select the minute at which the job will execute.

Select the **Exceptions** tab to specify any days on which the schedule will **not** execute.



Use the fields on the **Exceptions** tab to specify days on which you wish the job to not execute; for example, you may wish for jobs to not execute on national holidays.

Click the Add icon (+) to add a row to the exception table, then:

- Click within the **Date** column to open a calendar selector, and select a date on which the job will not execute. Specify **<Any>** in the **Date** column to indicate that the job should not execute on any day at the time selected.
- Click within the **Time** column to open a time selector, and specify a time on which the job will not execute. Specify **<Any>** in the **Time** column to indicate that the job should not execute at any time on the day selected.

When you've finished defining the schedule, you can use the **SQL** tab to review the code that will create or modify your job.

```

DO $$
DECLARE
    jid integer;
    scid integer;
BEGIN
    -- Creating a new job
    INSERT INTO pgagent.pga_job(
        jobclid, jobname, jobdesc, jobhostagent, jobenabled
    ) VALUES (
        1::integer, 'eom_batch_process'::text, ''::text, ''::text, true
    ) RETURNING jobid INTO jid;
    -- Steps
    -- Inserting a step (jobid: NULL)
    INSERT INTO pgagent.pga_jobstep (
        jstjobid, jstname, jstenabled, jstkind,
        jstconnstr, jstdbname, jstonerror,
        jstcode, jstdesc
    ) VALUES (
        jid, 'ID'::text, true, 's'::character(1),
        ''::text, 'postgres'::name, 'f'::character(1),
        'BEGIN
        END'::text, ''::text
    );
    -- Schedules
    -- Inserting a schedule
    INSERT INTO pgagent.pga_schedule(
        jscjobid, jscname, jscdesc, jscenabled,
        jscstart, jscend, jscminutes, jschours, jscweekdays, jscmonthdays, jscmonths
    ) VALUES (
        jid, 'test_schedule'::text, ''::text, true,
    );

```

Click the **Save** button to save the job definition, or **Cancel** to exit the job without saving. Use the **Reset** button to remove your unsaved entries from the dialog.

After saving a job, the job will be listed under the **pgAgent Jobs** node of the browser tree control of the server on which it was defined. The **Properties** tab in the main PEM window will display a high-level overview of the selected job, and the **Statistics** tab will show the details of each run of the job.

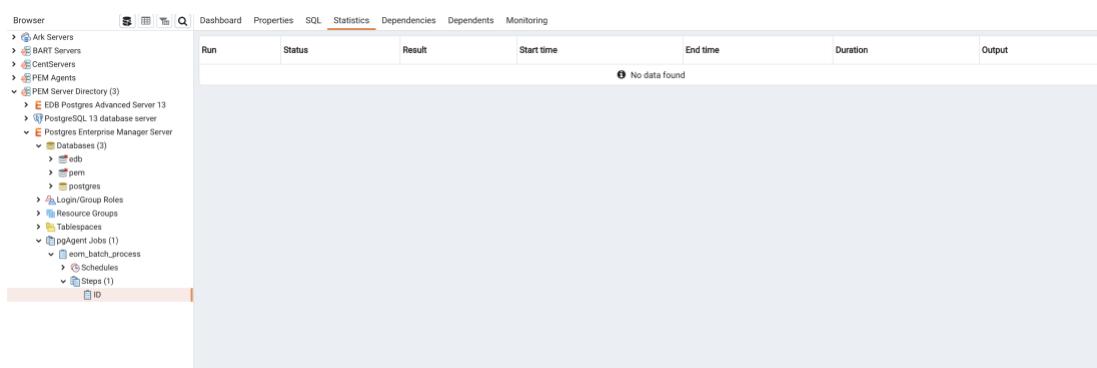


To modify an existing job or to review detailed information about a job, right-click on a job name, and select **Properties** from the context menu.

7.10.4 pgAgent Steps

Each Job consists of a number of steps, each of which may be an SQL script, or an operating system batch/shell script. Each step in a given job is run in turn, in alphanumeric name order.

Steps may be added to a job through the job properties dialogue, or added as a sub-object. The **Properties** tab of the main PEM client window will display details of the selected step, and the **Statistics** tab will display details of each run of the step, including and output or errors from the script.



Each step consists of the details shown on the screenshot below, most of which are self-explanatory. If **Kind** is set to **SQL**, then it goes without saying that a database against which to run the script must be selected. If set to **Batch**, the database/connection string should be left blank. The **On Error** option controls how failure of this step will affect the status of the overall job.

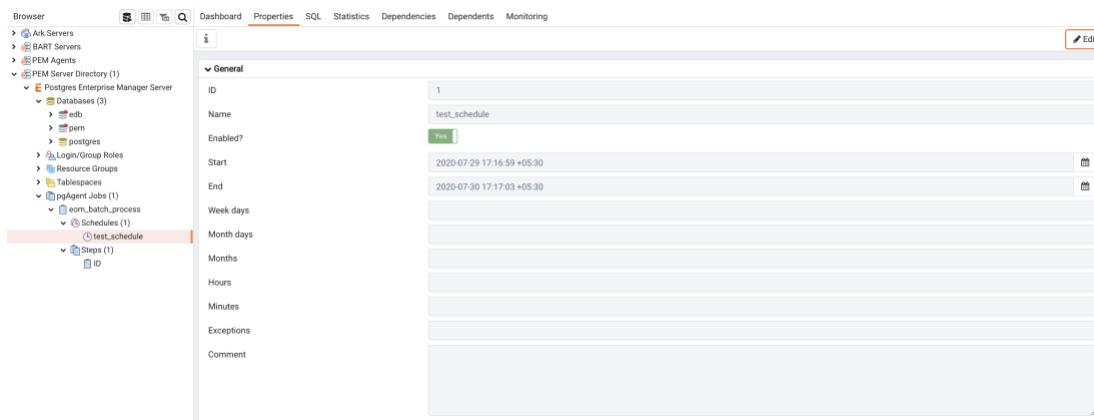


The **Definition** tab contains a single text box into which the step script should be entered. For SQL steps, this should be a series of one or more SQL statements. For batch jobs, when running on a Windows server, standard batch file syntax must be used, and when running on a *nix server, any shell script may be used, provided that a suitable interpreter is specified on the first line (e.g.#!/bin/sh*)*.

7.10.5 pgAgent Schedules

Each Job is executed according to one or more schedules. Each time the job or any of its schedules are altered, the next runtime of the job is re-calculated. Each instance of pgAgent periodically polls the database for jobs with the next runtime value in the past. By polling at least once every minute, all jobs will normally start within one minute of the specified start time. If no pgAgent instance is running at the next runtime of a job, it will run as soon as pgAgent is next started, following which it will return to the normal schedule.

Schedules may be added to a job through the job properties dialogue, or added as a sub-object. The **Properties** tab of the main PEM client window will display details of the selected schedule.



Each schedule consists of the basic details such as a name, whether or not it is enable and a comment. In addition, a start date and time is specified (before which the schedule has no effect), and optionally an end date and time (after which the schedule has no effect).

test_schedule

General Repeat Exceptions SQL

Name	test_schedule
Enabled?	Yes
Start	2020-04-27 12:08:06 +05:30
End	2020-04-27 12:08:08 +05:30
Comment	

i ? Cancel Reset Save

Schedules are specified using a cron-style format. For each selected time or date element, the schedule will execute. For example, to execute at 5 minutes past every hour, simply tick '5' in the **Minutes** list box. Values from more than one field may be specified in order to further control the schedule. For example, to execute at 12:05 and 14:05 every Monday and Thursday, you would tick minute 5, hours 12 and 14, and weekdays Monday and Thursday. For additional flexibility, the **Month Days** check list includes an extra **Last Day** option. This matches the last day of the month, whether it happens to be the 28th, 29th, 30th or 31st.

test_schedule

General Repeat Exceptions SQL

Schedules are specified using a **cron-style** format.
 For each selected time or date element, the schedule will execute.
 e.g. To execute at 5 minutes past every hour, simply select '05' in the Minutes list box.
 Values from more than one field may be specified in order to further control the schedule.
 e.g. To execute at 12:05 and 14:05 every Monday and Thursday, you would click minute 05, hours 12 and 14, and weekdays Monday and Thursday.
 For additional flexibility, the Month Days check list includes an extra Last Day option. This matches the last day of the month, whether it happens to be the 28th, 29th, 30th or 31st.

Days

Week Days	Select the weekdays...
Month Days	Select the month days..
Months	Select the months...

Times

i ? Cancel Reset Save

On occasion it may be desirable to specify an exception for a schedule - for example, you may not want a schedule to fire on a particular national holiday. To achieve this, each schedule may have a list of date and/or time exceptions attached to it. If a schedule lands on an exception, that instance will be skipped, and the following occurrence will become the next runtime.

test_schedule

General Repeat Exceptions SQL

Date	Time
2020-04-27	12:09

i ? Cancel Reset Save

7.11 Appendices

Contents:

| ----- | ----- | ----- |

7.11.1 Licence

pgAdmin is released under the [PostgreSQL Licence](#), which is a liberal Open Source licence similar to BSD or MIT, and approved by the Open Source Initiative. The copyright for the project source code, website and documentation is attributed to the [pgAdmin Development Team](#).

7.11.2 The MIT Kerberos Licence

PostgreSQL Enterprise Manager uses PostgreSQL's libpq library which may be linked with MIT Kerberos Libraries on some distributions. The MIT Kerberos licence is included below:

Kerberos Copyright

This software is being provided to you, the LICENSEE, by the Massachusetts Institute of Technology (M.I.T.) under the following license. By obtaining, using and/or copying this software, you agree that you have read, understood, and will comply with these terms and conditions:

Permission to use, copy, modify and distribute this software and its documentation for any purpose and without fee or royalty is hereby granted, provided that you agree to comply with the following copyright notice and statements, including the disclaimer, and that the same appear on ALL copies of the software and documentation, including modifications that you make for internal use or for distribution:

Copyright 1992-2004 by the Massachusetts Institute of Technology. All rights reserved.

THIS SOFTWARE IS PROVIDED "AS IS", AND M.I.T. MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED. By way of example, but not limitation, M.I.T. MAKES NO REPRESENTATIONS OR WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF THE LICENSED SOFTWARE OR DOCUMENTATION WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.

The name of the Massachusetts Institute of Technology or M.I.T. may NOT be used in advertising or publicity pertaining to distribution of the software. Title to copyright in this software and any associated documentation shall at all times remain with M.I.T., and USER agrees to preserve same.

Project Athena, Athena, Athena MUSE, Discuss, Hesiod, Kerberos, Moira, OLC, X Window System, and Zephyr are trademarks of the Massachusetts Institute of Technology (MIT). No commercial use of these trademarks may be made

without prior written permission of MIT.

7.11.3 The OpenSSL Licence

Postgres Enterprise Manager uses code from the OpenSSL project to provide support for SSL encrypted connections. The OpenSSL licence is included below:

Copyright (c) 1998-2011 The OpenSSL Project. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- All advertising materials mentioning features or use of this software must display the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit. (<http://www.openssl.org/>)"
- The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact openssl-core@openssl.org.
- Products derived from this software may not be called "OpenSSL" nor may "OpenSSL" appear in their names without prior written permission of the OpenSSL Project.
- Redistributions of any form whatsoever must retain the following acknowledgment: "This product includes software developed by the OpenSSL Project for use in the OpenSSL Toolkit (<http://www.openssl.org/>)"

THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT "AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

This product includes cryptographic software written by Eric Young (eay@cryptsoft.com). This product includes software written by Tim Hudson (tjh@cryptsoft.com).

7.11.4 The SNMP++ Licence

Postgres Enterprise Manager uses code from the SNMP++ project to send snmp v1/v2 notifications. The SNMP++ licence is included below:

Copyright (c) 2001-2010 Jochen Katz, Frank Fock

This software is based on SNMP++2.6 from Hewlett Packard:

Copyright (c) 1996 Hewlett-Packard Company

ATTENTION: USE OF THIS SOFTWARE IS SUBJECT TO THE FOLLOWING TERMS. Permission to use, copy, modify, distribute and/or sell this software and/or its documentation is hereby granted without fee. User agrees to display the above copyright notice and this license notice in all copies of the software and any documentation of the software. User agrees to assume all liability for the use of the software; Hewlett-Packard and Jochen Katz make no representations about the suitability of this software for any purpose. It is provided "AS-IS" without warranty of any kind, either express or implied. User hereby grants a royalty-free license to any and all derivatives based upon this software code base.

7.11.5 The jquery table sort Licence

TABLESORT.JS Copyright, Andy Croxall (mitya@mitya.co.uk) For documentation and demo see
<http://mitya.co.uk/scripts/Animated-table-sort-REGEXP-friendly-111>

USAGE This script may be used, distributed and modified freely but this header must remain in tact. For usage info and demo, including info on args and params, see www.mitya.co.uk/scripts

7.12 Release Notes

EDB Postgres Enterprise Manager release notes provide the information on the features and improvements in each release. This page includes release notes for major release and minor (bugfix) releases. Select your version from the list below to see the release notes for it.

7.12.1 PEM v8.0.1

Release date: 2021-03-03

Features

[Issue #5091](#) - Make Statistics, Dependencies, Dependents tabs closable and the user can add them back using the 'Add panel' option.

Housekeeping

[Issue #5338](#) - Improve code coverage and API test cases for pgAgent.

[Issue #5343](#) - Improve code coverage and API test cases for Debugger.

[Issue #6079](#) - Updated mimetype from 'text/javascript' to 'application/javascript' as 'text/javascript' is obsolete.

Bug fixes

PEM-3549 - Fixed the issue where latest BART backups were not displaying on BART dashboard refresh. [Support ticket # 1046037]

PEM-3577 - Allow superuser name containing "-" provided during configuring configure-pem-server.sh script. [Support ticket # 1055435]

PEM-3881 - Close the operating system resources properly during batch probe and command execution to avoid 'too many open files' error. [Support ticket # 1048713]

PEM-3845 - Bundle pgaevent dll in agent windows installer to suppress event message log error. [Support ticket # 1106021]

PEM-3901 - Fixed the unicode string handling support in pemAgent. [Support ticket # 1153153]

PEM-3903 - Fixed the issue where BART restore was failing when agent was not bound with BART server. [Support ticket # 1129454]

[Issue #4892](#) - Fixed an issue where pressing the back button will show another instance of the main page inside of the Query Tool tab.

[Issue #5282](#) - Added 'Count Rows' option to the partition sub tables.

[Issue #5488](#) - Improve the explain plan details by showing popup instead of tooltip on clicking of the specified node.

[Issue #5571](#) - Added support for expression in exclusion constraints.

[Issue #5809](#) - Fixed an issue where the focus is not properly set on the filter text editor after closing the error dialog.

[Issue #5871](#) - Ensure that username should be visible in the 'Connect to Server' popup when service and user name both specified.

[Issue #5875](#) - Ensure that the 'template1' database should not be visible after pg_upgrade.

[Issue #5886](#) - Fixed false error is shown while adding a new foreign key from the table dialog when a foreign key already exists with Auto FK Index set to true.

[Issue #5905](#) - Fixed an issue where the Save button is enabled by default in Macro.

[Issue #5906](#) - Remove extra line after Manage Macros menu while clearing all macros.

[Issue #5907](#) - Ensure that 'Clear All Rows' should not work if there is no existing macro available and the user does not specify any value.

[Issue #5929](#) - Fixed an issue where the server is disconnected error message displayed if the user creates Macro with invalid SQL.

[Issue #5965](#) - Ensure that the macro query result should be download properly.

[Issue #5973](#) - Added appropriate help message and a placeholder for letting users know about the account password expiry for Login/Group Role.

[Issue #5991](#) - Ensure that dirty indicator (*) should not be visible when renaming the tabs.

[Issue #5992](#) - Fixed an issue where escape character is shown when the server/database name has some special characters.

[Issue #5997](#) - Updated Flask-BabelEx to the latest.

[Issue #5998](#) - Fixed an issue where schema diff doesn't show the result of compare if source schema has tables with RLS.

[Issue #6003](#) - Fixed an issue where an illegal argument is showing for trigger SQL when a trigger is created for View.

[Issue #6045](#) - Fixed autocomplete issue where it is not showing any suggestions if the schema name contains escape characters.

[Issue #6046](#) - Fixed an issue where the state of the Save File icon does not match the dirty editor indicator.

[Issue #6047](#) - Fixed an issue where the dirty indicator stays active even if all changes were undone.

[Issue #6058](#) - Ensure that the rename panel should be disabled when the SQL file opened in the query tool.

[Issue #6061](#) - Fixed extra parentheses issue around joins for Views.

[Issue #6065](#) - Fixed accessibility issues in schema diff module.

[Issue #6069](#) - Fixed an issue on refreshing files in Query Tool.

[Issue #6077](#) - Fixed accessibility issues in various dialogs.

[Issue #6084](#) - Fixed TypeError exception in schema diff when selected any identical object.

[Issue #6087](#) - Fixed an issue where the dependencies tab showing multiple owners for the objects having shared dependencies.

[Issue #6098](#) - Fixed an issue of deleting records when the user tries to delete multiple records.

[Issue #6120](#) - Ensure that the user should be able to specify an older date for the account expiration of the role/user.

[Issue #6121](#) - Fixed an issue where the database list in the new connection window is not visible.

- [Issue #6122](#) - Added informative message when there is no difference found for schema diff.
 - [Issue #6128](#) - Fixed an issue where sequences are not created.
 - [Issue #6140](#) - Ensure that verbose logs should be visible for Utility(Backup, Maintenance) jobs.
 - [Issue #6144](#) - Ensure that the current value of the sequence should be ignored while comparing using schema diff.
 - [Issue #6157](#) - Fixed an issue where strike-through is not visible for rows selected for deletion after scrolling.
 - [Issue #6178](#) - Fixed an issue where the user unable to change the background color for a server.
 - [Issue #6187](#) - Limit the upgrade check to run once per day.
 - [Issue #6208](#) - Fixed an issue where utility(Backup, Maintenance, ...) jobs are failing when the log level is set to DEBUG.
-

7.12.2 PEM v8.0

Release date: 2020-12-09

Features

- PEM-3669 - Support new configurations of BART ('bart_socket_name') in PEM
- PEM-3612 - Enhanced the placeholders for alert script execution. [Support Ticket # 1051538]
- PEM-3613 - Added documentation for "How to replace alert placeholders inside script". [Support Ticket # 1051538]
- PEM-3786 - Added support for --checksum-algorithm and --disable-checksum parameters introduced in BART 2.6.0.
- PEM-2501 - Added alert details sql for template 'A user expires in N days'. [Support Ticket # 891377]
- PEM-3684, PEM-3665 - Added alert details sql for Alert Errors, Table bloat, Dead/live tuples and few other. [Support Ticket # 1065250]
- PEM-3802, PEM-3805 - Documented the best security practices for PEM to avoid security vulnerabilities.
- PEM-3808 - Improved installation and upgrade guide as per customer feedback. [Support Ticket # 1101462]
- PEM-3819 - Add webhooks for event-based alerting
- [Issue #1402](#) - Added Macro support.
- [Issue #2519](#) - Added support to view trigger function under the respective trigger node.
- [Issue #3318](#) - Added support to download utility files at the client-side.
- [Issue #3794](#) - Allow user to change the database connection from an open query tool tab.
- [Issue #4230](#) - Added support to rename query tool and debugger tabs title.
- [Issue #4231](#) - Added support for dynamic tab size.
- [Issue #4232](#) - Added tab title placeholder for Query Tool, View/Edit Data, and Debugger.
- [Issue #5200](#) - Added support to ignore the owner while comparing objects in the Schema Diff tool.
- [Issue #5857](#) - Added documentation for Macro support.

Housekeeping

- [Issue #5328](#) - Improve code coverage and API test cases for Foreign Tables.
- [Issue #5330](#) - Improve code coverage and API test cases for Functions.
- [Issue #5337](#) - Improve code coverage and API test cases for Views and Materialized Views.
- [Issue #5395](#) - Added RESQL/MSQL test cases for Functions.
- [Issue #5497](#) - Merged the latest code of 'pgcli' used for the autocomplete feature.
- [Issue #5938](#) - Documentation of Storage Manager.

Bug fixes

PEM-672 - Documented that while configuring the pem server, certificates must be present in data directory of backend database server in the installation guides. [Support ticket # 729238]

PEM-3184 - Fixed an issue where connections were not getting released when user disconnects the database server. [Support Ticket # 969833]

PEM-3737 - Fixed an issue where BART integration isn't working, when BART is installed at custom location. [Support Ticket # 1088574]

PEM-3791 - Do not include detail alert information while sending SNMP traps. [Support Ticket # 1069206, # 1115277]

PEM-3816 - Fixed the issue of making two database connections for failed login attempt which result in locking the user profile. [Support Ticket # 1103288]

PEM-3795 - Added python3 as prerequisite for PEM in Upgrade and migration guide.

PEM-3799 - Package deployment and streaming replication deprecation warning is added in Upgrade and migration guide. [Support ticket # 1021617]

[Issue #4639](#) - Ensure that some fields should be disabled for the trigger in edit mode.

[Issue #4806](#) - Added useful message when the explain plan is not used and empty.

[Issue #4855](#) - Fixed an issue where file extension is stripped on renaming a file.

[Issue #5131](#) - Ensure that 'ctrl + a' shortcut does not move the cursor in SQL editor.

[Issue #5826](#) - Fixed an issue where schema diff is showing identical table as different due to default vacuum settings.

[Issue #5830](#) - Fixed reverse engineering SQL where parenthesis is not properly arranged for View/MView definition.

[Issue #5835](#) - Fixed 'can't execute an empty query' message if the user change the option of Auto FK Index.

[Issue #5841](#) - Fixed an issue where the server is not able to connect using the service.

[Issue #5842](#) - Ensure that query history should be listed by date/time in descending order.

[Issue #5843](#) - Fixed an issue where the 'PARALLEL UNSAFE' option is missing from reverse engineering SQL of function/procedure.

[Issue #5853](#) - Fixed an issue where 'Rows X' column values were not visible properly for Explain Analyze in Dark theme.

[Issue #5855](#) - Ensure that the user should be able to change the start value of the existing sequence.

[Issue #5858](#) - Ensure that search object functionality works with case insensitive string.

[Issue #5882](#) - Fixed invalid literal issue when fetching dependencies for Materialized View.

[Issue #5885](#) - Fixed an issue where the user is unable to change the macro name.

[Issue #5895](#) - Fixed an issue where the suffix for Toast table size is not visible in the Statistics tab.

[Issue #5911](#) - Ensure that macros should be run on the older version of Safari and Chrome.

[Issue #5914](#) - Fixed an issue where a mismatch in the value of 'Estimated row' for functions.

[Issue #5923](#) - Fixed an issue where non-closeable tabs are getting closed.

[Issue #5943](#) - Ensure that folder rename should work properly in Storage Manager.

[Issue #5950](#) - Fixed an issue where a long file name is not visible on the process watcher dialog.

[Issue #5953](#) - Fixed an issue where connection to the server is on wait state if a different user is provided.

[Issue #5959](#) - Ensure that Grant Wizard should include foreign tables.

[Issue #5974](#) - Fixed an issue where the debugger's custom tab title not applied when opened in the new browser tab.

[Issue #5978](#) - Fixed an issue where dynamic tab title has not applied the first time for debugger panel.

[Issue #5983](#) - Added the appropriate server icon based on the server type in the new connection dialog.

[Issue #5985](#) - Fixed an issue where the process watcher dialog throws an error for the database server which is already removed.

7.12.3 PEM v7.16

Release date: 2020-09-30

Features

PEM-3157 - Documentation of Defining and Monitoring Postgres instances on AWS EC2 and RDS is added. [Support

Ticker #1060981]

PEM-322 - Use the same agent-id on agent registration using '--force-registration', and regenerate the certificates.

[Support Ticket #695978]

PEM-688 - Added capability to monitor parameters set by EFM (missingnodes, minimumstandbys and membershipcoordinator).

PEM-2200 - Auto discover database servers installed using debian packaging.

PEM-2578 - Allow user to schedule bulk alert blackout from PEM console. [Support Ticket #901007]

PEM-2651 - List all the events and display error message on tooltip in BART activities graph. [Support Ticket #950623]

PEM-3053 - Automate BART Obsolete backup Cleanup process in PEM. [Support Ticket #950518]

PEM-3054 - Manage bart-scanner from PEM console. [Support Ticket #951184, #1027171]

PEM-3421 - Perform EFM cluster switchover from PEM console. [Support Ticket #1059731]

PEM-3528 - Make the current state and related information of the alerts available through REST API

PEM-3529 - Make the state change history of the alerts available at Agent, Server & Database through REST API

PEM-3614 - Added details of directory creation and permissions for backups taken using BART in PEM - BART Management Features Guide [Support Ticket #1038375]

PEM-3615 - Allow incremental backup from a parent backup which was taken in tar.gz format. [Support Ticket #1059420]

[Issue #2042](#) - Added SQL Formatter support in Query Tool.

[Issue #3904](#) - Replace charting library Flotr2 with ChartJS using React on the Dashboard panel.

[Issue #4059](#) - Added a new button to the query tool toolbar to open a new query tool window.

[Issue #5126](#) - Modified schema diff tool to compare two databases instead of two schemas.

[Issue #5653](#) - Added High Contrast (Beta) theme support.

[Issue #5772](#) - Warn the user when connecting to a server that is older than PEM supports.

Housekeeping

[Issue #5323](#) - Improve code coverage and API test cases for Foreign Data Wrapper.

[Issue #5324](#) - Improve code coverage and API test cases for Foreign Servers and User Mappings.

[Issue #5327](#) - Improve code coverage and API test cases for Schemas.

[Issue #5332](#) - Improve code coverage and API test cases for Columns and Constraints (Index, Foreign Key, Check, Exclusion).

[Issue #5336](#) - Improve code coverage and API test cases for Types.

[Issue #5344](#) - Improve code coverage and API test cases for Grant Wizard.

[Issue #5731](#) - Upgrade font awesome from v4 to v5.

[Issue #5774](#) - Improve code coverage and API test cases for Tables.

Bug fixes

PEM-812/ Issue #5830 <<https://redmine.postgresql.org/issues/5830>> - Display parenthesis around AND/OR conditions in the same order within PEM and dba_views. Fixed reverse engineering SQL where parenthesis is not properly arranged for View/MView definition. [Support Ticket #315838]

PEM-2579 - Fixed issues in SNMP's MIB file reported by <https://www.simpleweb.org/ietf/mibs/validate/> [Support Ticket #900679]

PEM-3532 - Fixed the issue in monitoring dashboard line charts where data points were not showing correct local time information. [Support Ticket #1023887]

PEM-3542 - Fixed potential data type migration problem when upgrading PEM Server to PG v.12 [Support Ticket #1044052]

PEM-3487 - Fixed configure script issue for google cloud instance running RHEL7

PEM-3666 - Fixed security issue related to secure flag for session cookies. [Support Ticket #1035935]

PEM-3667 - Fixed security issue related to information exposed in Server response header. [Support Ticket #1035935]

PEM-3677 - Fixed security issue related to content Security Policy (CSP) configuration. [Support Ticket #1035935]

PEM-3678 - Fixed security issue related to HTTP Strict Transport Security (HSTS) in server response header. [Support

Ticket #1035935]

PEM-3679 - Fixed security issue related to session fixation. [Support Ticket #1035935]

PEM-3682 - Fixed security issue related to directory listing. [Support Ticket #1035935]

PEM-3693 - Fixed security issue in the Capacity manager report. [Support Ticket #1035935]

PEM-3695 - Transaction ID or timestamp must be provided during BART restore point-in time recovery operation [Support Ticket #1067393]

PEM-3696 - Fixed security issue in the pem configure script where it should not log sensitive informations. [Support Ticket #1035935]

PEM-3663 - Fixed pemworker segfault error in dmesg and messages file. [Support Ticket #999681]

PEM-3698 - Fixed current and previous alert state related issue during alert script execution when alert state gets clear. [Support Ticket #1078828]

PEM-2015 - Fixed vulnerability issues related to webserver. [Support Ticket # 856609]

[Issue #3767](#) - Ensure that the original file format should be retained when saving the same file in SQL editor.

[Issue #3791](#) - Added missing comments in reverse engineering SQL for each column of a View.

[Issue #4123](#) - Fixed an issue where debugger doesn't work if the search path is set other than 'public'.

[Issue #4216](#) - Ensure that schema names starting with 'pg' should be visible in browser tree when standard_conforming_strings is set to off.

[Issue #4361](#) - Fixed ssh tunnel hang issue when the user tries to disconnect the server.

[Issue #4387](#) - Fixed an issue where the user is not able to insert the data if the table and columns name contains special characters.

[Issue #4810](#) - Fixed an issue where the user is not able to save the new row if the table is empty.

[Issue #5137](#) - Fixed save button enable issue when focusing in and out of numeric input field.

[Issue #5417](#) - Fixed and improve API test cases for the schema diff tool.

[Issue #5426](#) - Adjusted the height of jobstep code block to use maximum space.

[Issue #5429](#) - Ensure that the Dictionaries drop-down shows all the dictionaries in the FTS configuration dialog.

[Issue #5526](#) - Fixed an issue where copying and pasting a cell with multiple line data will result in multiple rows.

[Issue #5530](#) - Ensure that the referenced table should be displayed on foreign key constraints.

[Issue #5567](#) - Fixed an issue where conversion of bytea to the binary string results in an error.

[Issue #5569](#) - Fixed reverse engineered SQL for partitions when storage parameters are specified.

[Issue #5604](#) - Fixed an issue where the entire logs is in red text when the user runs backup and restore.

[Issue #5632](#) - Ensure that the user will be able to modify the start value of the Identity column.

[Issue #5646](#) - Ensure that RLS Policy node should be searchable using search object.

[Issue #5652](#) - Modified the 'Commit' and 'Rollback' query tool button icons.

[Issue #5666](#) - Added missing dependencies/dependent and corrected some wrongly identified.

[Issue #5670](#) - Fixed an issue where the error message does not have a close button on utility dialogs.

[Issue #5675](#) - Fixed CSRF errors when PEM opened in an iframe on safari browser.

[Issue #5677](#) - Fixed text color issue in explain analyze for the Dark theme.

[Issue #5686](#) - Fixed issue where the user was not able to update policy if the policy is created with space.

[Issue #5689](#) - Added the 'ORDER BY' clause for the privileges type to fix schema diff issue.

[Issue #5710](#) - Fixed an issue when comparing the table with a trigger throwing error in schema diff.

[Issue #5713](#) - Corrected DROP SQL syntax for catalog.

[Issue #5716](#) - Fixed an issue where ajax call continues to fire even after disconnect the database server.

[Issue #5722](#) - Ensure that the user should be able to drop the database even if it is connected.

[Issue #5724](#) - Clarify some of the differences when running in server mode in the docs.

[Issue #5730](#) - Resolve schema diff dependencies by selecting the appropriate node automatically and maintain the order in the generated script.

[Issue #5732](#) - Fixed some accessibility issues.

[Issue #5734](#) - Update the description of GIN and GiST indexes in the documentation.

[Issue #5739](#) - Ensure that the import/export feature should work with SSH Tunnel.

[Issue #5746](#) - Fixed an issue where --load-server does not allow loading connections that use pg_services.

[Issue #5748](#) - Fixed incorrect reverse engineering SQL for Foreign key when creating a table.

[Issue #5754](#) - Fixed an issue where schema diff is not working when providing the options to Foreign Data Wrapper, Foreign Server, and User Mapping.

[Issue #5764](#) - Fixed SQL for Row Level Security which is incorrectly generated.

[Issue #5765](#) - Fixed an issue in the query tool when columns are having the same name as javascript object internal functions.

- [Issue #5766](#) - Fixed string indices must be integers issue for PostgreSQL < 9.3.
- [Issue #5779](#) - Remove illegal argument from trigger function in trigger DDL statement.
- [Issue #5794](#) - Fixed excessive CPU usage by stopping the indefinite growth of the graph dataset.
- [Issue #5802](#) - Remove maximum length on the password field in the server dialog.
- [Issue #5807](#) - Fixed an issue where a column is renamed and then removed, then the drop SQL query takes the wrong column name.
- [Issue #5815](#) - Fixed an issue where clicking on the 'Generate script' button shows a forever spinner due to pop up blocker.
- [Issue #5816](#) - Ensure that the 'CREATE SCHEMA' statement should be present in the generated script if the schema is not present in the target database.
- [Issue #5820](#) - Fixed an issue while refreshing Resource Group.
- [Issue #5833](#) - Fixed an issue where custom sequences are not visible when show system objects are set to false.
- [Issue #5834](#) - Ensure that the 'Remove Server Group' option is available in the context menu.
- [Issue #5839](#) - Ensure that multiple extensions can be dropped from the properties tab.
- [Issue #5845](#) - Fixed an issue where the query tool is not fetching more than 1000 rows for the table does not have any primary key.

7.12.4 PEM v7.15

Release date: 2020-07-22

Features

- PEM-2457 - Added support for schema level restriction. [Support Ticket #883404]
- PEM-2540 - Added capability to monitor the xLogReceive parameter set by EFM
- PEM-3432 - Added support to monitor PG/EPAS 13
- PEM-3445 - Support new configurations of BART ('archive_path', 'bart_socket_directory') in PEM [Support Ticket #1015972]
- PEM-3446 - Document SNMP trap oid detailed information used by PEM [Support Ticket #900679]
- PEM-3482 - Improve the performance diagnostics tool to show CPU usage for the active sessions along with the wait events for better performance analysis
- [Issue #5452](#) - Added connected PEM user and connection name in the log file.
- [Issue #5468](#) - Added option to ignore the whitespaces while comparing objects in schema diff.
- [Issue #5500](#) - Added server group name while selecting servers in schema diff.
- [Issue #5516](#) - Added support of Row Security Policies.
- [Issue #5576](#) - Improve error messaging if the storage and log directories cannot be created.
- [Issue #5601](#) - Added RLS Policy support in Schema Diff.
- [Issue #5622](#) - Added support for permissive/restricted policy type while creating RLS Policy.

Housekeeping

- [Issue #5325](#) - Improve code coverage and API test cases for Collations.
- [Issue #5326](#) - Improve code coverage and API test cases for Domain and Domain Constraints.
- [Issue #5329](#) - Improve code coverage and API test cases for FTS Configuration, FTS Parser, FTS Dictionaries, and FTS Template.
- [Issue #5333](#) - Improve code coverage and API test cases for Indexes.
- [Issue #5334](#) - Improve code coverage and API test cases for the Rules module.

- [Issue #5335](#) - Improve code coverage and API test cases for Triggers and Compound Triggers.
- [Issue #5455](#) - Refactor PEM entrypoint python file so it can be imported and is a lot more readable.
- [Issue #5493](#) - Search object UI improvements.
- [Issue #5581](#) - Documentation of Row Level Security Policies.

Bug fixes

- PEM-699 - "Long running queries" alert should not log autovacuum queries. [Support Ticket #679920]
- PEM-3349 - pemworker register server help option should show the agent configuration directory path. [Support Ticket #1001212]
- PEM-1508 - Documented about calculation of shared system memory and removed it from capacity manager metrics as is always constant. [809378]
- PEM-3322 - Documented about system jobs and their default schedules. [990553]
- PEM-3436 - Updated all the alert templates queries which show negative values and improve the performance while fetching the data from history tables. [992418]
- PEM-2490 - Supported Platforms and Versions link is added to Software prerequisites section in Installation guides. [Support Ticket #885334]
- PEM-3490 - Removed send_email from POST/PUT payload and added validation for all_low_alert_enable/high_low_alert_enable/med_low_alert_enable/low_alert_enable for their respective email group id. [Support Ticket #1005781]
- [Issue #3591](#) - Ensure that the query tool should display the proper error message while terminating the active session.
- [Issue #3669](#) - Ensure that proper error should be displayed for the deleted node.
- [Issue #3694](#) - Gracefully informed the user that the database is already connected when they click on "Connect Database...".
- [Issue #3787](#) - Disabled the Stop process button after clicking it and added a message 'Terminating the process...' to notify the user.
- [Issue #3814](#) - Fixed issue of error message not getting displayed when filename is empty for backup, restore, and import/export.
- [Issue #3851](#) - Add proper indentation to the code while generating functions, procedures, and trigger functions.
- [Issue #4033](#) - Fixed an issue where clicking on the cross button of the alert box on the login page is not working.
- [Issue #4099](#) - Fixed the SQL help issue for EDB Postgres Advanced Server.
- [Issue #4223](#) - Ensure that maintenance jobs should be worked properly for indexes under a materialized view.
- [Issue #4226](#) - Fixed an issue where select all checkbox only selects the first 50 tables.
- [Issue #4235](#) - Fixed tab indent issue on a selection of lines is deleting the content when 'use spaces == true' in the preferences.
- [Issue #4840](#) - Ensure that 'With OID' option should be disabled while taking backup of database server version 12 and above.
- [Issue #5001](#) - Fixed invalid literal issue when removing the connection limit for the existing role.
- [Issue #5287](#) - Fixed dark theme-related CSS and modify the color codes.
- [Issue #5398](#) - Fixed generated SQL issue for auto vacuum options.
- [Issue #5416](#) - Ensure that the query tool panel gets closed when clicking on the 'Don't Save' button.
- [Issue #5422](#) - Ensure that the dependencies tab shows correct information for Synonyms.
- [Issue #5434](#) - Fixed an issue where the newly added table is not alphabetically added to the tree.
- [Issue #5440](#) - Fixed list sorting issue in the schema diff tool.
- [Issue #5449](#) - Fixed an issue while comparing the two identical schemas using the schema diff tool.
- [Issue #5450](#) - Fixed an issue when renaming the column not added in the proper order.
- [Issue #5463](#) - Fixed an issue where CSV download quotes numeric columns.
- [Issue #5465](#) - Fixed an issue where the Edge browser version is showing wrong and warning message gets displayed.
- [Issue #5470](#) - Fixed backgrid row hover issue where on hover background color is set for edit and delete cell only.
- [Issue #5481](#) - Fixed data truncation issue when updating the data of type character with length.
- [Issue #5492](#) - Fixed an issue where the search object is unable to locate inherited tables and constraint filters are not working.
- [Issue #5496](#) - Fixed an issue where clicking on Select All button, not selecting all the options in pgAgent job scheduler.
- [Issue #5507](#) - Fixed connection and version number detection issue when the database server is upgraded.

- [Issue #5539](#) - Fixed typo in exception keyword.
 - [Issue #5584](#) - Fixed an issue where two identical tables showing different by schema diff tool.
 - [Issue #5620](#) - Fixed an issue while creating RLS Policy with the name having space.
 - [Issue #5621](#) - Remove extra brackets from reverse engineering SQL of RLS Policy.
 - [Issue #5629](#) - Fixed an issue where the user is able to edit properties when some of the collection nodes are selected.
 - [Issue #5631](#) - Fixed 'cant execute empty query' issue when remove the value of 'USING' or 'WITH CHECK' option of RLS Policy.
 - [Issue #5633](#) - Ensure that create RLS Policy menu should not be visible for catalog objects.
 - [Issue #5647](#) - Fixed an issue where difference DDL is showing the wrong SQL when changing the policy owner.
 - [Issue #5673](#) - Fixed an issue where fetching the schema throws an error if the database is not connected in Schema Diff.
-

7.12.5 PEM v7.14

Release date: 2020-05-13

Features

- PEM-799 - Allow multiple agents to send emails & SNMP traps by enabling the configurations 'enable_smtp' & 'enable_snmp' in agent configuration (agent.cfg) file.
- PEM-1593 - Introduced a new state 'Unmanaged' for the database servers for which are not being monitored by PEM, but registered with it.
- PEM-3283 - Modified the configuration script, installer & PEMAgent to generate certificates using 4096 bit key, and modified SSLUtils to sign certificates using SHA256 Hash [Support Ticket #989497]
- PEM-3283 - PEMAgent service will not get disabled after upgrading [Support Ticket #994563]
- PEM-3308 - Introduced new v3 version of REST API, which includes SNMP v3 support.
- [Issue #2172](#) - Added search object functionality.
- [Issue #5154](#) - Added accessibility support in AlertifyJS.
- [Issue #5179](#) - Added Python 3.8 support.
- [Issue #5184](#) - Added support for parameter toast_tuple_target and parallel_workers of the table.
- [Issue #5261](#) - Added support of Collation, FTS Configuration, FTS Dictionary, FTS Parser, and FTS Template to the Schema Diff.
- [Issue #5262](#) - Added support of Domain, Domain Constraints and Types to the Schema Diff.
- [Issue #5263](#) - Added support of Foreign Tables to the Schema Diff.
- [Issue #5264](#) - Added support of Packages, Sequences and Synonyms to the Schema Diff.
- [Issue #5399](#) - Warn the user if an unsupported, deprecated or unknown browser is detected.

Housekeeping

- [Issue #5133](#) - Improvements in the UI for both default and dark themes.
- [Issue #5271](#) - Enhance the color of switch control for both light and dark theme.
- [Issue #4620](#) - Add Reverse Engineered and Modified SQL tests for procedures.

Bug fixes

- PEM-965 - Fixed hanging issue after 'Enable Remote Monitoring?' prompted to user. [Support Ticket #771096]

PEM-2500 - Corrected the version 3 of the certificate generated by sslutils
 PEM-3129 - Update 'pem.bart_backups' table with primary key as server id and backup id. [Support Ticket #972254]
 PEM-3183 - SQL profiler plugin was not able to load the profiler traces on ppcle machine [Support Ticket #991929]
 PEM-3202 - Allow user to open the job step logs in the new browser window as well as download it from schedule tasks.
 PEM-3248 - Check for proper PEM schema version before running BART specific functions for backward compatibility [Support Ticket #981208]
 PEM-3272 - Fixed issue where SQL profiler filter dialog not able to display applied filter properly [Support Ticket #987042]
 PEM-3316 - Fixed PEMAgent service should not get disabled after upgrading [Support Ticket #994563]
 PEM-3323 - Fixed an issue where user was not able to change the email group from Agent dialog [Support Ticket #995734]
[Issue #1257](#) - Ensure all object types have a "System XXX?" property.
[Issue #2813](#) - Ensure that the password prompt should not be visible if the database server is in trust authentication mode.
[Issue #3495](#) - Fixed an issue where the query tool unable to load the file which contains the BOM marker.
[Issue #3523](#) - Fixed an issue where right-clicking a browser object does not apply to the object on which right-click was fired.
[Issue #3645](#) - Ensure that the start and end date should be deleted when clear the selection for pgAgent Job.
[Issue #3900](#) - Added multiple drop/delete functionality for the table constraints.
[Issue #3947](#) - Fixed copy-paste row issues in View/Edit Data.
[Issue #3972](#) - Modified keyboard shortcuts in Query Tool for OSX native support.
[Issue #3988](#) - Fixed cursor disappeared issue in the query editor for some of the characters when zoomed out.
[Issue #4180](#) - Fixed mouse click issue where it does not select an object in Browser unless the pointer is over the object.
[Issue #4237](#) - Fix an issue where the user can not change the value of DateTime picker control using keyboard.
[Issue #4440](#) - Ensure the DROP statements in reverse engineered SQL are properly quoted for all objects.
[Issue #4445](#) - Ensure all object names in the title line of the reverse-engineered SQL are not quoted.
[Issue #4504](#) - Fixed an issue where like options should be disabled if the relation is not selected while creating a table.
[Issue #4512](#) - Fixed calendar opening issue on the exception tab inside the schedules tab of pgAgent.
[Issue #4573](#) - Ensure that if the delimiter is set other than comma then download the file as '.txt' file.
[Issue #4608](#) - Fixed some accessibility issues in the dialogs.
[Issue #4684](#) - Fixed encoding issue while saving data in encoded charset other than 'utf-8'.
[Issue #4709](#) - Added schema-qualified dictionary names in FTS configuration to avoid confusion of duplicate names.
[Issue #4856](#) - Enable the save button by default when a query tool is opened with CREATE or other scripts.
[Issue #4858](#) - Fixed python exception error when user tries to download the CSV and there is a connection issue.
[Issue #4873](#) - Fixed an issue when changing the comments of the procedure with arguments gives error in case of overloading.
[Issue #4946](#) - Fixed an issue when the user creates a temporary table with 'on commit drop as' clause.
[Issue #4955](#) - Changed the color of selected and hovered item for Select2 dropdown.
[Issue #4957](#) - Ensure that Constraint Trigger, Deferrable, Deferred option should be disabled when the user selects EDB-SPL function for the trigger.
[Issue #4969](#) - Fixed an issue where changing the values of columns with JSONB or JSON types to NULL.
[Issue #4996](#) - Improve the style of the highlighted code after query execution for Dark mode.
[Issue #5007](#) - Ensure index dropdown should have existing indexes while creating unique constraints.
[Issue #5043](#) - Fixed an issue where columns names should be visible in the order of their creation in the browser tree.
[Issue #5053](#) - Fixed an issue where changing the columns in the existing view throws an error.
[Issue #5058](#) - Ensure that AlertifyJS should not be visible as a title for alert dialog.
[Issue #5077](#) - Changed background pattern for geometry viewer to use #fff for all themes.
[Issue #5101](#) - Fix an issue where debugger not showing all arguments anymore after hitting SQL error while debugging.
[Issue #5115](#) - Fix an issue where command and statements were parsed incorrectly for Rules.
[Issue #5142](#) - Ensure that all the transactions should be canceled before closing the connections when a server is disconnected using PEM.
[Issue #5143](#) - Fix accessibility issue for the maximize button of the Alertify dialog.
[Issue #5157](#) - Ensure that default sort order should be using the primary key in View/Edit data.
[Issue #5180](#) - Fixed an issue where the autovacuum_enabled parameter is added automatically in the RE-SQL when the table has been created using the WITH clause.
[Issue #5184](#) - Fixed Firefox monospaced issue by updating the font to the latest version.

[Issue #5213](#) - Fixed an issue when the user performs refresh on a large size materialized view.

[Issue #5214](#) - Update Flask-SQLAlchemy and SQLAlchemy package which is not working on Windows with Python 3.8.

[Issue #5215](#) - Fix syntax error when changing the event type for the existing rule.

[Issue #5221](#) - Improve logic to get the DDL statements as a part of the comparison.

[Issue #5227](#) - Fixed an issue where user cannot be added if many users are already exists.

[Issue #5241](#) - Fixed tab key navigation issue for Grant Wizard.

[Issue #5268](#) - Fixed generated SQL when any token in FTS Configuration or any option in FTS Dictionary is changed.

[Issue #5270](#) - Ensure that OID should be shown in properties for Synonyms.

[Issue #5275](#) - Fixed tab key navigation issue for parameters in table dialog.

[Issue #5279](#) - Fixed Unicode character issue causing error on Python2 environment.

[Issue #5302](#) - Fixed an issue where difference SQL is not seen in the schema diff tool for Types.

[Issue #5314](#) - Ensure that switch cell is in sync with switch control for accessibility.

[Issue #5315](#) - Fixed an issue where schema diff showing changes in the identical domain constraints.

[Issue #5350](#) - Fixed an issue where schema diff marks an identical table as different.

[Issue #5352](#) - Fixed the rightmost and bottom tooltip crop issues in the explain query plan.

[Issue #5356](#) - Fixed modified SQL issue while adding an exception in pgAgent job schedule.

[Issue #5361](#) - Fixes an issue where PEM GUI does not display properly in IE 11.

[Issue #5362](#) - Fixed an issue where the identical packages and sequences visible as different in the schema diff tool.

[Issue #5366](#) - Added alert message to Reset Layout if any of the panels from Query Tool failed to load.

[Issue #5371](#) - Fixed tab key navigation for some dialogs.

[Issue #5375](#) - Fixed an issue where the Mode cell of argument grid does not appear completely in the Functions dialog.

[Issue #5383](#) - Fixed syntax error while refreshing the existing synonyms.

[Issue #5387](#) - Fixed an issue where the mode is not shown in the properties dialog of functions/procedures if all the arguments are "IN" arguments.

[Issue #5396](#) - Fixed an issue where the search object module unable to locate the object in the browser tree.

[Issue #5400](#) - Fixed internal server error when the database server is logged in with non-super user.

[Issue #5401](#) - Fixed search object issue when the object name contains special characters.

[Issue #5410](#) - Fixed an issue while removing the package body showing wrong modified SQL.

[Issue #5441](#) - Fixed an issue where the search object not able to locate pg_toast* tables in the pg_toast schema.

[Issue #5415](#) - Ensure that the query tool context menu should work on the collection nodes.

[Issue #5447](#) - Fixed failed to fetch utility error when click on refresh(any option) materialized view.

7.12.6 PEM v7.13

Release date: 2020-02-20

Features

PEM-594 - Logout the user session when there is no user activity.

PEM-3107 - Core usage report.

PEM-2794 - Added capability in pemAgent to send SNMP trap with SNMP v3 support.

[Issue #2554](#) - Added support for a multi-level partitioned table.

[Issue #3452](#) - Added a Schema Diff tool to compare two schemas and generate the diff script.

[Issue #4762](#) - Allow screen-reader to read label & description of non-textable elements.

[Issue #4763](#) - Allow screen-reader to identify the alert errors.

[Issue #4764](#) - Allow screen-reader to read relationship attributes in nested elements.

[Issue #4770](#) - Added labels and titles after parsing and validating all the web pages for accessibility.

[Issue #4993](#) - Set input controls as read-only instead of disabled will allow tab navigation in the properties tab and also allow screen readers to read it.

Housekeeping

- [Issue #4988](#) - Refactored SQL of Table's and it's child nodes.
- [Issue #5023](#) - Refactored SQL of Views and Materialized Views.
- [Issue #5024](#) - Refactored SQL of Functions and Procedures.
- [Issue #5038](#) - Added support for on-demand loading of items in Select2.
- [Issue #5049](#) - Improve code coverage and API test cases for the CAST module.
- [Issue #5050](#) - Improve code coverage and API test cases for the LANGUAGE module.
- [Issue #5072](#) - Updated wcDocker package which includes aria-label accessibility improvements.
- [Issue #5088](#) - Improve code coverage and API test cases for the Event Trigger module.
- [Issue #5096](#) - Replace node-sass with sass for SCSS compilation.
- [Issue #5176](#) - Enhance logging by tracking stdout and stderr of subprocess when log level set to DEBUG.
- [Issue #5185](#) - Added option to override the class name of a label tag for select2 control.

Bug fixes

- PEM-383 - Provide the column picker toolbar button in SQL profiler trace window.
- PEM-723 - Fixed a SQL Syntax error when PEM Agent queries tablespaces. [Support Ticket #570926]
- PEM-789 - Fixed historical span issue with memory usage graph. [Support Ticket #665140]
- PEM-933 - Fixed sorting of table chart data when rendering on custom dashboard. [Support Ticket #764647]
- PEM-1406 - Fixed the Swap Consumption alert template to reflect the actual swap consumption instead of total swap space. [Support Ticket #806008]
- PEM-2255 - Fixed in PEM RestAPI for server black out option. [Support Ticket #882589]
- PEM-2545/[Issue #4511](#) <<https://redmine.postgresql.org/issues/4511>> - Grant wizard is not able to handle multiple objects because all objects are passed as query string URL [Support Ticket #897419]
- PEM-2573 - Fixed for shared_buffers dashboard's historical span issue. [Support Ticket #894867]
- PEM-2614 - Fixed idle in transaction (aborted) state check in the probe code for session info. [Support Ticket #908442]
- PEM-2647 - Fixed storage pie chart issue to display the correct value. [Support Ticket #907753]
- PEM-2686 - Fixed save password functionality is not working when server was registered by PEM worker. [Support Ticket #912948]
- PEM-2828 - Fixed "Alert Errors" alert template to count the errors for which alerts are enabled. [Support Ticket #903272]
- PEM-3041 - Fixed not able to see the agent status in the PEM Monitoring dashboard due to numeric overflow error. [Support Ticket #934778]
- PEM-3139 - RLS policy was not updated properly for "pem.job" table during the PEM upgrade. [Support Ticket #937823]
- [Issue #3812](#) - Ensure that path file name should not disappear when changing ext from the dropdown in file explorer dialog.
- [Issue #4198](#) - Fix syntax highlighting in code mirror for backslash and escape constant.
- [Issue #4410](#) - Fixed an issue while editing char[] or character varying[] column from View/Edit data throwing an error.
- [Issue #4506](#) - Fix an issue where clicking on an empty textbox like fill factor or comments, considers it as change and enabled the save button.
- [Issue #4601](#) - Added tab navigation on close buttons for all the panels and create/properties dialog.
- [Issue #4633](#) - Added support to view multilevel partitioned tables.
- [Issue #4724](#) - Fix network disconnect issue while establishing the connection via SSH Tunnel and it impossible to expand the Servers node.
- [Issue #4818](#) - Fix server connection drops out issue in query tool.
- [Issue #4827](#) - Fix column resizable issue in the file explorer dialog.
- [Issue #4842](#) - Ensure that constraints, indexes, rules, triggers, and compound triggers should be created on partitions.
- [Issue #4926](#) - Fix VPN network disconnect issue where it hangs on expanding the Servers node.
- [Issue #4933](#) - Ensure that the Servers collection node should expand independently of server connections.
- [Issue #4943](#) - Added more information to the 'Database connected/disconnected' message.
- [Issue #5000](#) - Logout the session when no user activity of mouse move, click or keypress.
- [Issue #5008](#) - Ensure that the error message should not be displayed if Tablespace is not selected while creating the

index.

[Issue #5009](#) - Fix an issue where operator, access method and operator class is not visible for exclusion constraints.

[Issue #5025](#) - Fix an issue where setting STORAGE_DIR to empty should show all the volumes on Windows in server mode.

[Issue #5047](#) - Added tab navigation for tabs under explain panel in query tool.

[Issue #5065](#) - Updated the incorrect icon used for the cast node on refresh.

[Issue #5066](#) - Fix an issue where refreshing a package results in the change in the object completely.

[Issue #5074](#) - Fix an issue where select, insert and update scripts on tables throwing an error.

[Issue #5107](#) - Set proper focus on tab navigation for file manager dialog.

[Issue #5116](#) - Fixed an issue where Save Password control disappears after clicking on it while creating a server.

7.12.7 PEM v7.12

Release date: 2019-12-02

Features

[PEM-2477](#) - Allow the pemAgent to create a password-less SSH authentication between two linux systems. User can now choose to create it from the database server properties dialog, and backup restore dialog.

[PEM-2848](#) - Allow the pemAgent to set/override the 'archive_command' of the database server configuration using 'BART INIT' command. User can choose to set/override the 'archive_command' from the database server properties dialog.

[Issue #1974](#) - Added encrypted password in reverse engineered SQL for roles.

[Issue #3741](#) - Added Dark (Beta) UI Theme option.

[Issue #4006](#) - Support Enable Always and Enable Replica on triggers.

[Issue #4351](#) - Add an option to request confirmation before cancelling changes on a Properties dialog.

[Issue #4396](#) - Warn the user on changing the definition of Materialized View about the loss of data and its dependent objects.

[Issue #4435](#) - Allow drag and drop functionality for all the nodes under the database node, excluding collection nodes.

[Issue #4711](#) - Use a 'play' icon for the Execute Query button in the Query Tool for greater consistency with other applications.

[Issue #4772](#) - Added aria-label to provide an invisible label where a visible label cannot be used.

[Issue #4773](#) - Added role="status" attribute to all the status messages for accessibility.

[Issue #4990](#) - Changed the open query tool and data filter icons.

Housekeeping

[Issue #4696](#) - Add Reverse Engineered and Modified SQL tests for Materialized Views.

[Issue #4701](#) - Optimize Webpack to improve overall performance.

[Issue #4807](#) - Refactored code of table and it's child nodes.

[Issue #4938](#) - Refactored code of columns node.

Bug fixes

[Issue #3130](#) - Ensure create new object dialog should be opened when alt+shift+n key is pressed on the collection node.

- [Issue #3279](#) - Fixed issue where Drop and Disconnect connection menu points are too close to each other.
- [Issue #3538](#) - Fix issue where the Reset button does not get enabled till all the mandatory fields are provided in the dialog.
- [Issue #3789](#) - Ensure context menus never get hidden below the menu bar.
- [Issue #3859](#) - Rename the context menu from 'Drop Server' to 'Remove Server'.
- [Issue #3913](#) - Ensure the correct "running at" agent is shown when a pgAgent job is executing.
- [Issue #3915](#) - Fix an issue in the Query Tool where shortcut keys could be ignored following a query error.
- [Issue #3999](#) - Fix the toggle case shortcut key combination.
- [Issue #4191](#) - Ensure comments are shown in reverse engineered SQL for table partitions.
- [Issue #4242](#) - Handle NULL values appropriately when sorting backgrid tables.
- [Issue #4341](#) - Give appropriate error messages when the user tries to use an blank master password.
- [Issue #4451](#) - Remove arbitrary (and incorrect) requirement that composite types must have at least two members.
- [Issue #4459](#) - Don't quote bigints when copying them from the Query Tool results grid.
- [Issue #4482](#) - Ensure compression level is passed to pg_dump when backing up in directory format.
- [Issue #4483](#) - Ensure the number of jobs can be specified when backing up in directory format.
- [Issue #4516](#) - Remove the sorting of table headers with no labels.
- [Issue #4564](#) - Ensure Javascript errors during Query Tool execution are reported as such and not as Ajax errors.
- [Issue #4610](#) - Suppress Enter key presses in Alertify dialogues when the come from Select2 controls to allow item selection with Enter.
- [Issue #4647](#) - Ensure that units are respected when sorting by file size in the File dialog.
- [Issue #4659](#) - Updated documentation for default privileges to clarify more on the grantor.
- [Issue #4674](#) - Fix query tool launch error if user name contains HTML characters. It's a regression.
- [Issue #4730](#) - Ensure all messages are retained in the Query Tool from long running queries.
- [Issue #4734](#) - Updated documentation for the delete row button that only strikeout the row instead of deleting it.
- [Issue #4761](#) - Fix an issue where the wrong type is displayed when changing the datatype from timestamp with time zone to timestamp without time zone.
- [Issue #4779](#) - Updated documentation for the query tool toolbar buttons.
- [Issue #4792](#) - Ensure that the superuser should be able to create database, as the superuser overrides all the access restrictions.
- [Issue #4878](#) - Ensure that the superuser should be able to create role, as the superuser overrides all the access restrictions.
- [Issue #4893](#) - Fix reverse engineering SQL issue for partitions when specifying digits as comments.
- [Issue #4896](#) - Fixed an issue where escape key not working to close the open/save file dialog.
- [Issue #4906](#) - Fixed an issue where keyboard shortcut for context menu is not working when using Firefox on CentOS7.
- [Issue #4923](#) - Enhance the logic to change the label from 'Delete/Drop' to 'Remove' for the server and server group node.
- [Issue #4925](#) - Shown some text on process watcher till the initial logs are loaded.
- [Issue #4930](#) - Fix main window tab navigation accessibility issue.
- [Issue #4935](#) - Fix accessibility issues.
- [Issue #4947](#) - Fix XSS issue in explain and explain analyze for table and type which contain HTML.
- [Issue #4952](#) - Fix an issue of retrieving properties for Compound Triggers. It's a regression of #4006.
- [Issue #4953](#) - Fix an issue where PEM unable to retrieve table node if the trigger is already disabled and the user clicks on Enable All.
- [Issue #4958](#) - Fix reverse engineering SQL issue for triggers when passed a single argument to trigger function.
- [Issue #4964](#) - Fix an issue where length and precision are not removed from table/column dialog.
- [Issue #4965](#) - Fix an issue where the Interval data type is not displayed in the properties dialog of table/column.
- [Issue #4966](#) - Fix 'Could not find the object on the server.' error while refreshing the check constraint.
- [Issue #4975](#) - Fix issue where the user can not switch the UI language. It's a regression of #4348.
- [Issue #4976](#) - Fix reverse engineering SQL issue where when clause is not visible for PG/EPAS 12.
- [Issue #4982](#) - Added statistics and storage information in reverse engineering SQL of table/column.
- [Issue #4985](#) - Fix an issue where the inherited table name with quotes did not escape correctly.
- [Issue #4991](#) - Fix an issue where context menu is open along with submenu and the focus is not on context menu or submenu.

7.12.8 PEM v7.11

Release date: 2019-12-18

Features

[Issue #4778](#) - Implemented the Query Plan Analyser.

Housekeeping

Bug fixes

PEM-2793 - User with 'super_pem_admin' role should be able to see all servers and agents. (922716)

PEM-2827 - Fixed a security risk - PEMAgent used to run the SHELL/BATCH script(s), defined as the step(s) of user job(s), alert script(s) and custom batch probes, as root user on non-windows system and as an Administrator on Windows.

[Issue #3386](#) - Ensure backup a partition table should not backup the whole database.

[Issue #4590](#) - Fix issue where backup fails for schema name that needs quoting.

[Issue #4728](#) - Highlighted the color of closing or opening parenthesis when user select them in CodeMirror.

[Issue #4751](#) - Fix issue where export job fails when deselecting all the columns.

[Issue #4753](#) - Fix an error where 'false' string is displayed when we add a new parameter in the Parameters tab, also clear the old value when the user changes the parameter name.

[Issue #4760](#) - Ensure the search path should not be quoted for Database.

[Issue #4780](#) - Ensure the search path should not be quoted for Function, Procedure and Trigger Function.

[Issue #4791](#) - Fix issue where VALID foreign keys show as NOT VALID in the SQL tab for tables.

[Issue #4845](#) - Fixed potential error in the properties dialog for the Code tab.

[Issue #4850](#) - Fixed an issue where Datetimepicker control opens when clicking on the label.

7.12.9 PEM v7.10

Release date: 2019-10-09

Features

[Issue #PEM-952](#) - Allow to configure and manage the BART server through PEM (limited functionalities)

[Issue #PEM-2088](#) - Allow a banner to be displayed on the login and other related pages showing custom text. (869690)

[Issue #PEM-2152](#) - Allow to manage and monitor PostgreSQL 12 and EDB Postgres Advanced Server 12.

[Issue #3009](#) - Added Copy with headers functionality when copy data from Query Tool/View Data.

[Issue #4144](#) - Add support of Compound Triggers for EPAS 12+.

[Issue #4139](#) - Allow some objects to be dragged/dropped into the Query Tool to insert their signature into the query text.

[Issue #4333](#) - Add support for planner support functions in PostgreSQL 12+ functions.

[Issue #4334](#) - Add support for generated columns in Postgres 12+.

[Issue #4540](#) - Use the full tab space for CodeMirror instances on dialogues where appropriate.

[Issue #4566](#) - Allow enhanced cookie protection to be disabled for compatibility with dynamically addressed hosting environments.

[Issue #4570](#) - Add an optimisation to the internal code responsible for searching for treeview nodes.

[Issue #4574](#) - Display the row count in the popup message when counting table rows, not just in the properties list.

[Issue #4612](#) - Add support in query history to show internal queries generated by pgAdmin during save data operations.

[Issue #4667](#) - Ensure editable and read-only columns in Query Tool should be identified by icons and tooltips in the column header.

Housekeeping

[Issue #4472](#) - Add Reverse Engineered and Modified SQL tests for Synonyms.

[Issue #4546](#) - Add Reverse Engineered SQL tests for Columns.

[Issue #4554](#) - Add Reverse Engineered SQL tests for Trigger Functions.

[Issue #4555](#) - Add Reverse Engineered SQL tests for Exclusion Constraint.

[Issue #4560](#) - Add a --modules option to the RE-SQL test suite to allow testing of specific object types.

[Issue #4575](#) - Add Reverse Engineered SQL tests for Schemas.

[Issue #4576](#) - Add Reverse Engineered SQL tests for Views.

[Issue #4600](#) - Add Reverse Engineered SQL tests for Rules.

[Issue #4616](#) - Add Reverse Engineered and Modified SQL tests for Foreign Keys.

[Issue #4617](#) - Add Reverse Engineered and Modified SQL tests for Foreign Servers.

[Issue #4618](#) - Add Reverse Engineered and Modified SQL tests for Foreign Tables.

[Issue #4619](#) - Add Reverse Engineered and Modified SQL tests for FTS Templates.

[Issue #4621](#) - Add Reverse Engineered and Modified SQL tests for Indexes.

[Issue #4624](#) - Add Reverse Engineered and Modified SQL tests for Primary Keys.

[Issue #4627](#) - Add Reverse Engineered and Modified SQL tests for User Mappings.

[Issue #4628](#) - Add Reverse Engineered and Modified SQL tests for Unique Constraints.

[Issue #4690](#) - Add Modified SQL tests for Resource Group.

Bug fixes

[Issue #PEM-706](#) - Changed the label from "Blocked Users" to "Blocked Sessions" on User Activity chart. (661653)

[Issue #PEM-2492](#) - Ensure parameter values are quoted when needed when editing roles. (876762)

[Issue #PEM-2581](#) - Error when changing kind(SQL/BATCH) for pgAgent job step. (893794)

[Issue #PEM-2727](#) - Upgrading SNMP to fix the agent crash issue (913881)

[Issue #2706](#) - Added ProjectSet icon for explain module.

[Issue #2828](#) - Added Gather Merge, Named Tuple Store Scan and Table Function Scan icon for explain module.

[Issue #3605](#) - Fix issue where Deleting N number of rows makes first N number of rows disable.

[Issue #3778](#) - Ensure Boolean columns should be editable using keyboard keys.

[Issue #3936](#) - Further code refactoring to stabilise the Feature Tests.

[Issue #4179](#) - Fix generation of reverse engineered SQL for tables with Greenplum 5.x.

[Issue #4199](#) - Ensure that 'ENTER' key in the data filter should not run the query.

[Issue #4229](#) - Update wcDocker to allow the browser's context menu to be used except in tab strips and panel headers.

[Issue #4381](#) - Fix an issue where oid column should not be pasted when copy/paste row is used on query output containing the oid column.

[Issue #4401](#) - Ensure type names are properly encoded in the results grid.

[Issue #4408](#) - Fix display of validation error message in SlickGrid cells.

[Issue #4412](#) - Fix issue where Validated switch option is inverted for the Foreign Key.

[Issue #4414](#) - Fix generation of reverse engineered SQL for partition table, partitions were shown as a child of indexes.

[Issue #4419](#) - Fix a debugger error when using Python 2.7.

[Issue #4429](#) - Ensure drag/drop from the treeview works as expected on Firefox.

- [Issue #4461](#) - Fix error while importing data to a table using Import/Export dialog and providing "Not null columns" option.
- [Issue #4486](#) - Ensure View should be created with special characters.
- [Issue #4487](#) - Ensure Boolean columns should be editable in View/Edit data and Query Tool.
- [Issue #4489](#) - Update wcDocker to prevent window state loading creating blank dialogues.
- [Issue #4490](#) - Fix accessibility issue for checkbox in IE11.
- [Issue #4492](#) - Ensure the Query Tool doesn't throw an error when viewing the contents of a table with no columns.
- [Issue #4496](#) - Ensure columns can be created when they are IDENTITY fields with the CYCLE option enabled.
- [Issue #4497](#) - Ensure purely numeric comments can be saved on new columns.
- [Issue #4508](#) - Fix accessibility issue for Datetime cell in backgrid.
- [Issue #4520](#) - Ensure the query tool will work with older versions of psycopg2 than we officially support, albeit without updatable resultsets.
- [Issue #4525](#) - Ensure command tags are shown in the messages tab of the Query Tool.
- [Issue #4536](#) - Fix load on demand in View/Edit data mode.
- [Issue #4552](#) - Fix some errors thrown on the JS console when dragging text in the Query Tool.
- [Issue #4559](#) - Ensure triggers should be updated properly for EPAS server.
- [Issue #4565](#) - Fix the reverse engineered SQL for trigger functions with the WINDOW option selected.
- [Issue #4577](#) - Fix an error that could be seen when click on any system column of a table.
- [Issue #4578](#) - Ensure enable trigger menu should be visible when trigger is disabled.
- [Issue #4581](#) - Ensure the comment on a Primary Key constraint can be edited under the Table node.
- [Issue #4582](#) - Fix console error when changing kind(SQL/BATCH) for pgAgent job step.
- [Issue #4584](#) - Unescape HTML entities in database names in the Query Tool title bar.
- [Issue #4585](#) - Fix double click issue to expand the contents of a cell if the resultset was not editable.
- [Issue #4586](#) - Fix generation of reverse engineered SQL for Rules.
- [Issue #4631](#) - Add editor options for plain text mode and to disable block folding to workaround rendering speed issues in CodeMirror with very large scripts.
- [Issue #4635](#) - Ensure compound triggers for event should be updated properly.
- [Issue #4638](#) - Ensure compound triggers should be displayed under Views.
- [Issue #4641](#) - Ensure Truncate option should be available for Compound Triggers.
- [Issue #4643](#) - Fix Truncate option deselect issue for compound triggers.
- [Issue #4644](#) - Fix length and precision enable/disable issue when changing the data type for Domain node.
- [Issue #4650](#) - Fix SQL tab issue for Views. It's a regression of compound triggers.
- [Issue #4663](#) - Fix exception in query history for python 2.7.
- [Issue #4674](#) - Fix query tool launch error if user name contain html characters.
- [Issue #4681](#) - Increase cache control max age for static files to improve performance over longer run.
- [Issue #4698](#) - Fix SQL issue of length and precision when changing the data type of Column.
- [Issue #4702](#) - Fix modified SQL for Index when reset the value of Fill factor and Clustered?.
- [Issue #4703](#) - Fix reversed engineered SQL for btree Index when provided sort order and NULLs.
- [Issue #4726](#) - Ensure sequence with negative value should be created.
- [Issue #4727](#) - Fix issue where EXEC script doesn't write the complete script for Procedures.
- [Issue #4742](#) - Ensure Primary Key should be created with Index.
- [Issue #4750](#) - Fix query history exception for Python 3.6.
- [Issue #4756](#) - Fix issue where PEM/pgAdmin does not load completely if loaded in an iframe.
- [Issue #4777](#) - Fix issue where query history is not visible in the query history tab.

7.12.10 PEM v7.9

Release date: 2019-07-31

Features

- [Issue #PEM-2525](#) - Send SMTP notification on the completion of a scheduled task
- [Issue #PEM-2526](#) - Schedule a SHELL/BATCH script, SQL based jobs at agent level
- [Issue #PEM-2048](#) - Allow to configure pem-server without disabling the selinux on RHLE/CentOS (883342)
- [Issue #1760](#) - Add support for editing of resultsets in the Query Tool, if the data can be identified as updatable.
- [Issue #2653](#) - Allow the UI layout to be fully locked or to prevent docking changes.
- [Issue #3174](#) - Visually distinguish simple tables from tables that are inherited and from which other tables are inherited.
- [Issue #4283](#) - Initial support for PostgreSQL 12.
- [Issue #4288](#) - Initial support for PostgreSQL 12.
- [Issue #4290](#) - Initial support for PostgreSQL 12.
- [Issue #4318](#) - Set the mouse cursor appropriately based on the layout lock state.
- [Issue #4335](#) - Add EXPLAIN options for SETTINGS and SUMMARY.

Housekeeping

- [Issue #4202](#) - Add a framework for testing reversed engineered SQL and CRUD API endpoints.
- [Issue #4415](#) - Add Reverse Engineered SQL tests for Roles and Resource Groups.
- [Issue #4441](#) - Add Reverse Engineered SQL tests for FDWs.
- [Issue #4450](#) - Fix reverse engineered sql for Foreign Data Wrapper created on EPAS server in redwood mode.
- [Issue #4452](#) - Add Reverse Engineered SQL tests for Languages.
- [Issue #4453](#) - Add Reverse Engineered SQL tests for Extensions.
- [Issue #4454](#) - Add Reverse Engineered SQL tests for FTS Configurations.
- [Issue #4456](#) - Add Reverse Engineered SQL tests for Packages.
- [Issue #4460](#) - Add Reverse Engineered SQL tests for FTS Dictionaries.
- [Issue #4463](#) - Add Reverse Engineered SQL tests for Domains.
- [Issue #4464](#) - Add Reverse Engineered SQL tests for Collations.
- [Issue #4468](#) - Add Reverse Engineered SQL tests for Types.
- [Issue #4469](#) - Add Reverse Engineered SQL tests for Sequences.
- [Issue #4471](#) - Add Reverse Engineered SQL tests for FTS Parsers.

Bug fixes

- [Issue #PEM-2459](#) - Tuning wizard shows wrong original value for 'max_wal_size' parameter [Support Ticket #880186]
- [Issue #PEM-2499](#) - Index Advisor is not suggesting index on PEM UI. [Support Ticket #891318]
- [Issue #PEM-2503](#) - Issue while scheduling the pgAgent job on PEM UI. [Support Ticket #891220]
- [Issue #3919](#) - Allow keyboard navigation of all controls on subnode grids.
- [Issue #3994](#) - Fix issue where the dependencies tab for inherited tables/foreign keys shows partial text.
- [Issue #3996](#) - Fix dropping of pgAgent schedules through the Job properties.
- [Issue #4036](#) - Allow editing of data where a primary key column includes a % sign in the value.
- [Issue #4162](#) - Fix syntax error when adding more than one column to the existing table.
- [Issue #4169](#) - Omit the geometry viewer in the Query Tool from layout saving.
- [Issue #4171](#) - Fix issue where reverse engineered SQL was failing for foreign tables, if it had "=" in the options.
- [Issue #4195](#) - Fix keyboard navigation in "inner" tabs such as the Query Tool and Debugger.
- [Issue #4224](#) - Prevent flickering of large tooltips on the Graphical EXPLAIN canvas.
- [Issue #4228](#) - Ensure the correct label is used in panel headers when viewing filtered rows.
- [Issue #4253](#) - Fix issue where new column should be created with Default value.
- [Issue #4255](#) - Prevent the geometry viewer grabbing key presses when not in focus under Firefox, IE and Edge.
- [Issue #4284](#) - Fix syntax error when creating a table with a serial column.
- [Issue #4320](#) - Fix issue where SSH tunnel connection using password is failing, it's regression of Master Password.
- [Issue #4329](#) - Fix an initialisation error when two functions with parameters are debugged in parallel.
- [Issue #4343](#) - Fix issue where property dialog of column should open properly for EPAS v12.

[Issue #4345](#) - Capitalize the word 'export' used in Import/Export module.

[Issue #4349](#) - Ensure strings are properly encoded in the Query History.

[Issue #4350](#) - Ensure we include the CSRF token when uploading files.

[Issue #4360](#) - Ensure the debugger control buttons are only enabled once initialisation is complete.

[Issue #4362](#) - Remove additional "SETOF" included when generating CREATE scripts for trigger functions.

[Issue #4365](#) - Fix help links for backup globals and backup server.

[Issue #4367](#) - Fix an XSS issue seen in View/Edit data mode if a column name includes HTML.

[Issue #4378](#) - Ensure Python escaping matched JS escaping and fix a minor XSS issue in the Query Tool that required superuser access to trigger.

[Issue #4380](#) - Ensure that both columns and partitions can be edited at the same time in the table dialog.

[Issue #4386](#) - Fix an XSS issue when username contains XSS vulnerable text.

[Issue #4389](#) - Fix an error that could be seen when editing column privileges.

[Issue #4393](#) - Ensure parameter values are quoted when needed when editing roles.

[Issue #4403](#) - Ensure the browser close confirmation is only shown when closing a Query Tool which is running in a separate browser tab.

[Issue #4404](#) - Prevent an error that may occur when editing data with an integer primary key.

[Issue #4407](#) - Fix a quoting issue that caused a blank UI to be displayed when running in French.

[Issue #4427](#) - Fix an error while retrieving json data from the table.

[Issue #4428](#) - Fix 'malformed array literal' error when updating a pgAgent job.

[Issue #4437](#) - Fix table icon issue when updating any existing field.

[Issue #4446](#) - Use ROLE consistently when generating RE-SQL for roles, not USER.

[Issue #4448](#) - Fix an error seen when updating a connection string in a pgAgent job step.

[Issue #4462](#) - Fix some minor UI issues on IE11.

[Issue #4470](#) - Fix sequence reverse engineered SQL generation with quoted names on PG/EPAS 10+.

[Issue #4484](#) - Fix an issue where Explain and Explain Analyze are not working, it's regression of #1760.

[Issue #4485](#) - Fix an issue where Filter toolbar button is not working in view/edit data, it's regression of keyboard navigation.

7.12.11 PEM v7.8

Release date: 2019-06-05

Features

[Feature #PEM-614](#) - Added --enable-heartbeat-connection parameter to use dedicated heartbeat connection while self registering pemagent. [Support Ticket #742120]

[Feature #4017](#) - Make the Query Tool history persistent across sessions.

[Feature #4018](#) - Remove the large and unnecessary dependency on React and 87 other related libraries.

[Feature #4030](#) - Add support for IDENTITY columns.

Bug fixes

[Bug #PEM-986](#) - Fetch the pem schema version again in pemagent when PEM server restarts. [Support Ticket #769569]

[Bug #PEM-1449](#) - Copy alert REST API document is missing. [Support Ticket #804272]

[Bug #PEM-2046](#) - Use the 'g' flag to replace all occurrence of placeholder in the alert script code, alert email template, alert email subject. [Support Ticket #856746]

[Bug #PEM-2087](#) - Line chart on Dashboard is not honouring the user's selected start & end timespan. [Support Ticket

#859558]

Bug #PEM-2135 - No take over is happening after PEM agent and server upgrade from version 7.4 to 7.7. [Support Ticket #859340]

Bug #PEM-2166 - Table chart created using custom probe should render properly. [Support Ticket #872498]

[Bug #4217](#)/PEM-2204 - Fixed CSRF security vulnerability issue.

[Bug #1269](#) - Fix naming inconsistency for the column and FTS parser modules.

[Bug #2392](#) - Ensure that on clicking Delete button should not delete rows immediately from the database server, it should be deleted when Save button will be clicked.

[Bug #2627](#) - Include inherited column comments and defaults in reverse engineered table SQL.

[Bug #3582](#) - Ensure that JSON strings as comments should be added properly for all the objects.

[Bug #3605](#) - Fix an issue where Deleting N number of rows makes first N number of rows disable.

[Bug #3933](#) - Ignore exceptions in the logger.

[Bug #3938](#) - Added support for Default Partition.

[Bug #4019](#) - Update all Python and JavaScript dependencies.

[Bug #4037](#) - Include comment SQL for inherited columns in reverse engineered table SQL.

[Bug #4050](#) - Make the WHEN field a CodeMirror control on the Event Trigger dialogue.

[Bug #4052](#) - Fix the online help button on the resource group dialogue.

[Bug #4053](#) - Enable the online help button on the index dialogue.

[Bug #4062](#) - Fix handling of numeric arrays in View/Edit Data.

[Bug #4063](#) - Enlarge the grab handles for resizing dialogs etc.

[Bug #4069](#) - Append the file suffix to filenames when needed in the File Create dialogue.

[Bug #4073](#) - Change the CodeMirror active line background colour to \$color-danger-lighter so it doesn't conflict with the selection colour.

[Bug #4081](#) - Fix the RE-SQL syntax for roles with a VALID UNTIL clause.

[Bug #4082](#) - Prevent an empty error message being shown when "downloading" a CREATE script using the CSV download.

[Bug #4084](#) - Overhaul the layout saving code so it includes the Query Tool and Debugger, and stores the layout when change events are detected rather than (unreliably) on exit.

[Bug #4085](#) - Display errors during CSV download from the Query Tool in the UI rather than putting them in the CSV file.

[Bug #4087](#) - Fix an issue where 'GRANT UPDATE' sql should be displayed for default sequence privileges.

[Bug #4096](#) - Ensure the toolbar buttons are properly reset following a CSV download in the Query Tool.

[Bug #4099](#) - Fix SQL help for EPAS 10+, and refactor the URL generation code into a testable function.

[Bug #4100](#) - Ensure sequences can be created with increment, start, minimum and maximum options set.

[Bug #4101](#) - Ensure that confirmation dialog should be popped up before reload of query tool or debugger if it is opened in a new browser tab.

[Bug #4104](#) - Ensure that record should be add/edited for root partition table with primary keys.

[Bug #4105](#) - Fix an issue where JSON data would not be rendered in the Query Tool.

[Bug #4109](#) - Ensure View/Materialized View node should be visible after updating any property.

[Bug #4110](#) - Fix custom autovacuum configuration for Materialized Views.

[Bug #4121](#) - Fixed alignment issue of columns in definition section of Index node.

[Bug #4131](#) - Relabel the Save button on the datagrid text editor to avoid confusion with the actual Save button that updates the database.

[Bug #4134](#) - Fixed 'Location cannot be empty' error when open Tablespace properties.

[Bug #4138](#) - Fix an issue where the dropdown becomes misaligned/displaced.

[Bug #4142](#) - Added recommended ESLinter checks.

[Bug #4143](#) - Ensure that pgAdmin4 should work properly with psycopg2 v2.8

[Bug #4154](#) - Ensure the treeview shows all sequences except those used to implement IDENTITY columns (which can be edited as part of the column). Show all if Show System Objects is enabled.

[Bug #4160](#) - Fixed 'Increment value cannot be empty' error for existing tables.

[Bug #4161](#) - Ensure that parameters of procedures for EPAS server 10 and below should be set/reset properly.

[Bug #4163](#) - Prevent duplicate columns being included in reverse engineered SQL for tables.

[Bug #4164](#) - Fix file browser path issue which occurs when client is on Windows and server is on Mac/Linux.

[Bug #4182](#) - Ensure sanity of the permissions on the storage and session directories and the config database.

[Bug #4218](#) - Properly assign dropdownParent in Select2 controls.

[Bug #4219](#) - Ensure popper.js is installed when needed.

[Bug #4246](#) - Fixed console error when subnode control is used in panels.

- Bug #4194 - Fix accessibility issue for menu navigation.
 - Bug #4227 - Fixed Tab key navigation for Maintenance dialog.
 - Bug #4244 - Fix Tab key issue for Toggle switch controls and button on the dialog footer in Safari browser.
 - Bug #4245 - Ensure that element should get highlighted when they get focus on using Tab key.
 - Bug #4261 - Stop using application/x-javascript as a mime type and use the RFC-compliant application/javascript instead.
 - Bug #4262 - Fixed error on displaying table properties of a table partitioned by list having a default partition.
 - Bug #4263 - Fix handling of JSON in the Query Tool with NULL elements.
 - Bug #4269 - Fix navigation of switch cells in grids.
 - Bug #4276 - Relax the permission check on the directory containing the config database, as it may fail in some environments such as OpenShift.
 - Bug #4278 - Prevent Backgrid Password cells from losing focus if the browser opens an autocomplete list.
 - Bug #4308 - Fix the issue of accessing the SQL for Views and Materialized Views. Regression of pluralisation of folder names.
-

7.12.12 PEM v7.7.1

Release date: 2019-04-10

This release contains bug fixes.

Bug fixes

- Bug #PEM-2092 - Remove the obsolete packages while upgrading the edb-pem-server (v7.5 and earlier) using the RPM
 - Bug #PEM-2089 - Zooming on a line chart does not work when line charts are linked.
-

7.12.13 PEM v7.7

Release date: 2019-03-13

This release contains a new PEM UI update and some of the features and bug fixes.

Features

- Feature #2233 - Add a "scratch pad" to the Query Tool to hold text snippets whilst editing.
- Feature #2418 - Add Commit and Rollback buttons to the Query Tool.
- Feature #3439 - Allow X-FRAME-OPTIONS to be set for security. Default to SAMEORIGIN.
- Feature #3559 - Automatically expand child nodes as well as the selected node on the treeview if there is only one.
- Feature #4034 - Support double-click on Query Tool result grid column resize handles to auto-size to the content.

Bug fixes

Bug #PEM-732 - Allow user to download explain plan from sql profiler [Support Ticket #484049]
 Bug #PEM-1544 - Give proper privileges and explicit type cast to hstore extension [Support Ticket #820063]
 Bug #PEM-1596 - PEM Agent registration creates cert files with additional .crt extension [Support Ticket #853579]
 Bug #PEM-1684 - Handle proper exception during alert processing. [Support Ticket #830175]
 Bug #PEM-1743 - Delete all the agent entry except for the lastest one sorted by login time [Support Ticket #829037]
 Bug #PEM-1855 - Server info probe shows numeric overflow error if shared memory size is too high [Support Ticket #833535]
 Bug #PEM-1943 - Honour the pause of the auto-refresh on the 'Alert Status' table whilst showing the alert details [Support Ticket #853200]
 Bug #PEM-1964 - RPM libboost dependencies issues on IBM powerbox [Support Ticket #849462]
 Bug #PEM-1976 - Fetch the session information for the same server when fetching the lock information [Support Ticket #853549]
 Bug #3051 - Replace Bootstrap switch with Bootstrap4 toggle to improve the performance.
 Bug #3096 - Ensure size stats are prettified on the statistics tab when the UI language is not English.
 Bug #3272 - Replace the PyCrypto module with the cryptography module.
 Bug #3352 - Handle display of roles with expiration set to infinity correctly.
 Bug #3418 - Allow editing of values in columns with the oid datatype which are not an actual row OID.
 Bug #3453 - Fixed SQL for foreign table options.
 Bug #3475 - Fixed execution time to show Hours part for long running queries in Query Tool.
 Bug #3505 - Fix SQL generated for tables with inherited columns.
 Bug #3544 - Make the Query Tool tab titles more concise and useful.
 Bug #3549 - Display event trigger functions correctly on EPAS.
 Bug #3575 - Ensure the context menu works after a server is renamed.
 Bug #3583 - Update CodeMirror to 5.43.0 to resolve issues with auto-indent.
 Bug #3587 - Fix support for bigint's in JSONB data.
 Bug #3600 - Ensure JSON data isn't modified in-flight by psycopg2 when using View/Edit data.
 Bug #3608 - Messages tab of query tool should be clear on subsequent execution of table/view using View/Edit Data.
 Bug #3609 - Clear drop-down menu should be disabled for View/Edit Data.
 Bug #3664 - Fixed Statistics panel hang issue for 1000+ tables.
 Bug #3673 - Modify the Download as CSV option to use the same connection as the Query Tool its running in so temporary tables etc. can be used.
 Bug #3679 - Fix a webpack issue that could cause the Query Tool to fail to render.
 Bug #3693 - Proper error should be thrown when server group is created with existing name.
 Bug #3695 - Ensure long string should be wrap in alertify dialogs.
 Bug #3697 - Ensure that output of the query should be displayed even if Data Output window is detached from the Query Tool.
 Bug #3702 - Ensure we display the relation name (and not the OID) in the locks table wherever possible.
 Bug #3740 - Inline edbspl trigger functions should not be visible in Grant Wizard.
 Bug #3774 - Proper SQL should be generated when create function with return type as custom type argument.
 Bug #3780 - Ensure that null values handled properly in CSV download.
 Bug #3800 - Ensure that database restriction of server dialog should work with special characters.
 Bug #3809 - Ensure auto complete should works when first identifier in the FROM clause needs quoting.
 Bug #3810 - Ensure auto complete should works for columns from a schema-qualified table.
 Bug #3811 - Ensure that Backup/Restore button should work on single click.
 Bug #3836 - Fix ordering of VACUUM options which changed in PG11.
 Bug #3837 - Fixed SQL for when clause while creating Trigger.
 Bug #3838 - Proper SQL should be generated when creating/changing column with custom type argument.
 Bug #3840 - Ensure that file format combo box value should be retained when hidden files checkbox is toggled.
 Bug #3842 - Don't show system catalogs in the schemas property list unless show system objects is enabled.
 Bug #3846 - Proper SQL should be generated when create procedure with custom type arguments.
 Bug #3849 - Ensure that browser should warn before close or refresh.
 Bug #3850 - Fixed EXEC script for procedures.
 Bug #3853 - Proper SQL should be generated when create domain of type interval with precision.
 Bug #3856 - Fixed an issue while creating export job.
 Bug #3858 - Drop-down should be closed when click on any other toolbar button.
 Bug #3861 - Fix help for the backup/restore dialogues.

Bug #3862 - Fixed keyboard navigation for dialog tabs.
 Bug #3865 - Increase frames splitter mouse hover area to make it easier to resize.
 Bug #3866 - Ensure that last row of table data should be visible and user will be able to add new row.
 Bug #3871 - Fixed alignment of tree arrow icons for Internet Explorer.
 Bug #3872 - Ensure object names in external process dialogues are properly escaped.
 Bug #3873 - Fix context sub-menu alignment on Safari.
 Bug #3877 - Make the browser more robust in the face of multibyte characters in SQL_ASCII databases.
 Bug #3891 - Correct order of Save and Cancel button for json/jsonb editing.
 Bug #3897 - Data should be updated properly for FTS Configurations, FTS Dictionaries, FTS Parsers and FTS Templates.
 Bug #3899 - Fixed unable to drop multiple Rules and Foreign Tables from properties tab.
 Bug #3903 - Fixed Query Tool Initialization Error.
 Bug #3906 - Fix alignment of Close and Maximize button of Grant Wizard.
 Bug #3908 - Fixed keyboard navigation for Select2 and Privilege cell in Backgrid.
 Bug #3911 - Add full support and tests for all PG server side encodings.
 Bug #3912 - Fix editing of table data with a JSON primary key.
 Bug #3916 - Correct schema should be displayed in Materialized View dialog.
 Bug #3927 - Fixed debugger issue for procedure inside package for EPAS servers.
 Bug #3929 - Fix alignment of help messages in properties panels.
 Bug #3932 - Fix alignment of submenu for Internet Explorer.
 Bug #3935 - Ensure that grant wizard should list down functions for EPAS server running with no-redwood-compat mode. [Support Ticket #833292]
 Bug #3941 - Dashboard graph optimization.
 Bug #3946 - Fix alignment of checkbox to drop multiple schedules of pgAgent job.
 Bug #3948 - Set the background colour for backform notes, and add an icon.
 Bug #3954 - Remove Python 2.6 code that's now obsolete.
 Bug #3958 - Don't exclude SELECT statements from transaction management in the Query Tool in case they call data-modifying functions.
 Bug #3959 - Optimise display of Dependencies and Dependents, and use on-demand loading of rows in batches of 100.
 Bug #3961 - Exclude HTTPExceptions from the all_exception_handler as they should be returned as-is.
 Bug #3963 - Fix alignment of import/export toggle switch.
 Bug #3968 - Update wcDocker to fix the issue where the Scratch Pad grows in size if the results panel is resized.
 Bug #3970 - Prevent an error when closing the Sort/Filter dialogue with an empty filter string.
 Bug #3973 - Use 'set_config(...)' function to update the 'bytea_output' settings instead of 'UPDATE' statement, which is not allowed in the the read-only instances.
 Bug #3974 - Fix alignment of Connection type toggle switch of pgagent.
 Bug #3982 - Add full support and tests for all PG server side encodings.
 Bug #3992 - Add full support and tests for all PG server side encodings.
 Bug #3995 - Avoid 'bogus varno' message from Postgres when viewing the SQL for a table with triggers.
 Bug #3998 - Custom-encode forward slashes in URL parameters as Apache HTTPD doesn't allow them in some cases.
 Bug #4000 - Update CodeMirror to 5.43.0 to resolve issues with tab indent with use spaces enabled.
 Bug #4013 - Ensure long queries don't cause errors when downloading CSV in the Query Tool.
 Bug #4021 - Disable the editor and execute functions whilst queries are executing.
 Bug #4054 - Handle resultsets with zero columns correctly in the Query Tool.
 Bug #4071 - Ensure that Firefox prompts for a filename/location when downloading query results as a CSV file.

7.12.14 PEM v7.6

Release date: 2019-01-09

This release contains a number of features and fixes reported since the release of PEM v7.6

Features

Feature #PEM-545 - Allow to delete the server group [Support Ticket #737596]
 Feature #PEM-945 - Allow to use the performance diagnostic (edb_wait_stat) plugin for EDB Postgres Advance Server
 Feature #PEM-1522 - Adding support in the sslutils to work with OpenSSL 1.1+
[Feature #1513](#) - Add support for dropping multiple objects at once from the collection Properties panel.
[Feature #3562](#) - Migrate from Bootstrap 3 to Bootstrap 4.
 Feature #PEM-1528/[Feature #3589](#) - Allow query plans to be downloaded as an SVG file [Support Ticket #816621]

Bug fixes

Bug #PEM-1368/[Bug #3676](#) - Fix CREATE Script functionality for EDB-Wrapped functions. [Support Ticket #796491]
 Bug #PEM-1530 - Fix the server level charts [Support Ticket #815806]
 Bug #PEM-1532 - Disable TRACE option from the configuration file for the apache server [Support Ticket #784345, #818383]
 Bug #PEM-1610 - EFM Status in streaming replication dashborad on PEM not shown [Support Ticket #826397]
 Bug #PEM-1611 - PEM not able to monitor EFM with error: Failed to parse EFM json file [Support Ticket #826197]
[Bug #3016](#) - Ensure previous notices are not removed from the Messages tab in the Query Tool if an error occurs during query execution.
[Bug #3029](#) - Allow the selection order to be preserved in the Select2 control to fix column ordering in data Import/Export.
[Bug #3083](#) - Increase the size of the resize handle of the edit grid text pop-out.
[Bug #3232](#) - Ensure that Utilities(Backup/Restore/Maintenence/Import-Export) should not be started if binary path is wrong and also added 'Stop Process' button to cancel the process.
[Bug #3354](#) - Fix handling of array types as inputs to the debugger.
[Bug #3433](#) - Fix an issue that could cause the Query Tool to fail to render.
[Bug #3559](#) - Further improvements to treeview restoration.
[Bug #3619](#) - Add titles to the code areas of the Query Tool and Debugger to ensure that panels can be re-docked within them.
[Bug #3629](#) - Allow use of 0 (integer) and empty strings as parameters in the debugger.
[Bug #3638](#) - Fix syntax error when creating new pgAgent schedules with a start date/time and exception.
[Bug #3711](#) - Fix an encoding issue in the query tool.
[Bug #3722](#) - Ensure that utility existence check should work for schema and other child objects while taking Backup/Restore.
[Bug #3723](#) - Properly report errors when debugging cannot be started.
[Bug #3726](#) - Include the WHERE clause on EXCLUDE constraints in RE-SQL.
[Bug #3734](#) - Prevent the debugger controls being pressed again before previous processing is complete.
[Bug #3736](#) - Fix toggle breakpoints buttons in the debugger.
[Bug #3742](#) - Fix changes to the NOT NULL and default value options in the Table Dialogue.
[Bug #3746](#) - Fix dropping of multiple functions/procedures at once.
[Bug #3753](#) - Fix an issue when user define Cast from smallint->text is created.
[Bug #3757](#) - Hide Radio buttons that should not be shown on the maintenance dialogue.
[Bug #3797](#) - Prevent attempts to bulk-drop schema objects.
[Bug #3798](#) - Ensure the browser toolbar buttons work in languages other than English.
[Bug #3805](#) - Allow horizontal sizing of the edit grid text pop-out.
[Bug #3821](#) - Ensure identifiers are properly displayed in the plan viewer.
[Bug #3823](#) - Delete/Drop and drop cascade option under properties section is disabled for multiple objects.
[Bug #3824](#) - Ensure the dashboard tabs are styles correctly.

7.12.15 PEM v7.5

Release date: 2018-10-24

This release contains a number of features and fixes reported since the release of PEM v7.5

Features

Feature #PEM-796 - As a user I should be able to exclude particular mount point from the disk-space probe
 Feature #PEM-1020 - As a user I would like to monitor the advanced server V11
 Feature #PEM-1021 - As a user I would like to monitor the postgres V11
 Feature #PEM-1022 - As a user I would like to use PG/EPAS v11 as backend server for the PEM
 Feature #PEM-1295 - Limit the number of connections to the PEM DB Server (using pgbouncer - connection pooler)
 [Support Ticket #784672]
[Feature #3564](#) - Add shortcuts for View Data and the Query tool to the Browser header bar.
[Feature #3514](#) - Add optional data point markers and mouse-over tooltips to display values on graphs.
[Feature #1407](#) - Add a geometry viewer that can render PostGIS data on a blank canvas or various map sources.
[Feature #1253](#) - Save the treeview state periodically, and restore it automatically when reconnecting.

Bug fixes

Bug #PEM-724 - Can't debug function and procedure in package(EnterpriseDB) use non-superuser role [Support Ticket #569062]
 Bug #PEM-918/[#3596](#) - Fix support for the CLOB datatype in EPAS [Support Ticket #761853]
 Bug #PEM-1004 - Allow to enter floating values for threshold fields in Alerts configuration window [Support Ticket #775364]
 Bug #PEM-1330 - User can not save password when connecting to server [Support Ticket #792298]
 Bug #PEM-1350 - Allow to install on PostgreSQL/EDB Postgres Advanced Server 9.4 [Support Ticket #796149]
 Bug #PEM-1431 - Release the connections for the connected servers on logout [Support Ticket #807009]
[Bug #3191](#) - Fixed debugger execution issues.
[Bug #3420](#) - Merge pgcli code with version 1.10.3, which is used for auto complete feature.
[Bug #3525](#) - Ensure that refresh button on dashboard should refresh the table.
[Bug #3551](#) - Fix handling of backslashes in the edit grid.
[Bug #3554](#) - Fix auto scrolling issue in debugger on step in and step out. [Support Ticket #779956]
[Bug #3576](#) - Ensure queries are no longer executed when dashboards are closed.
[Bug #3604](#) - Correct the documentation of View/Edit data.
[Bug #3607](#) - Fix logic around validation and highlighting of Sort/Filter in the Query Tool.
[Bug #3630](#) - Ensure auto-complete works for objects in schemas other than public and pg_catalog.
[Bug #3657](#) - Ensure changes to Query Tool settings from the Preferences dialogue are applied before executing queries.
[Bug #3658](#) - Swap the Schema and Schemas icons and Catalog and Catalogs icons that had been used the wrong way around.
[Bug #3660](#) - Rename the 'SQL Editor' section of the Preferences to 'Query Tool' as it applies to the whole tool, not just the editor.
[Bug #3674](#) - Cleanup session files periodically.
[Bug #3700](#) - Fix connection garbage collector.
[Bug #3703](#) - Purge connections from the cache on logout. [Support Ticket #807009]

8 EDB Postgres Enterprise Manager Configuring pgBouncer for Use with PEM Agents

This document provides detailed information about using pgBouncer as a connection pooler for limiting the number of connections from the PEM Agent towards the Postgres Enterprise Manager (PEM) server on non-Windows machine:

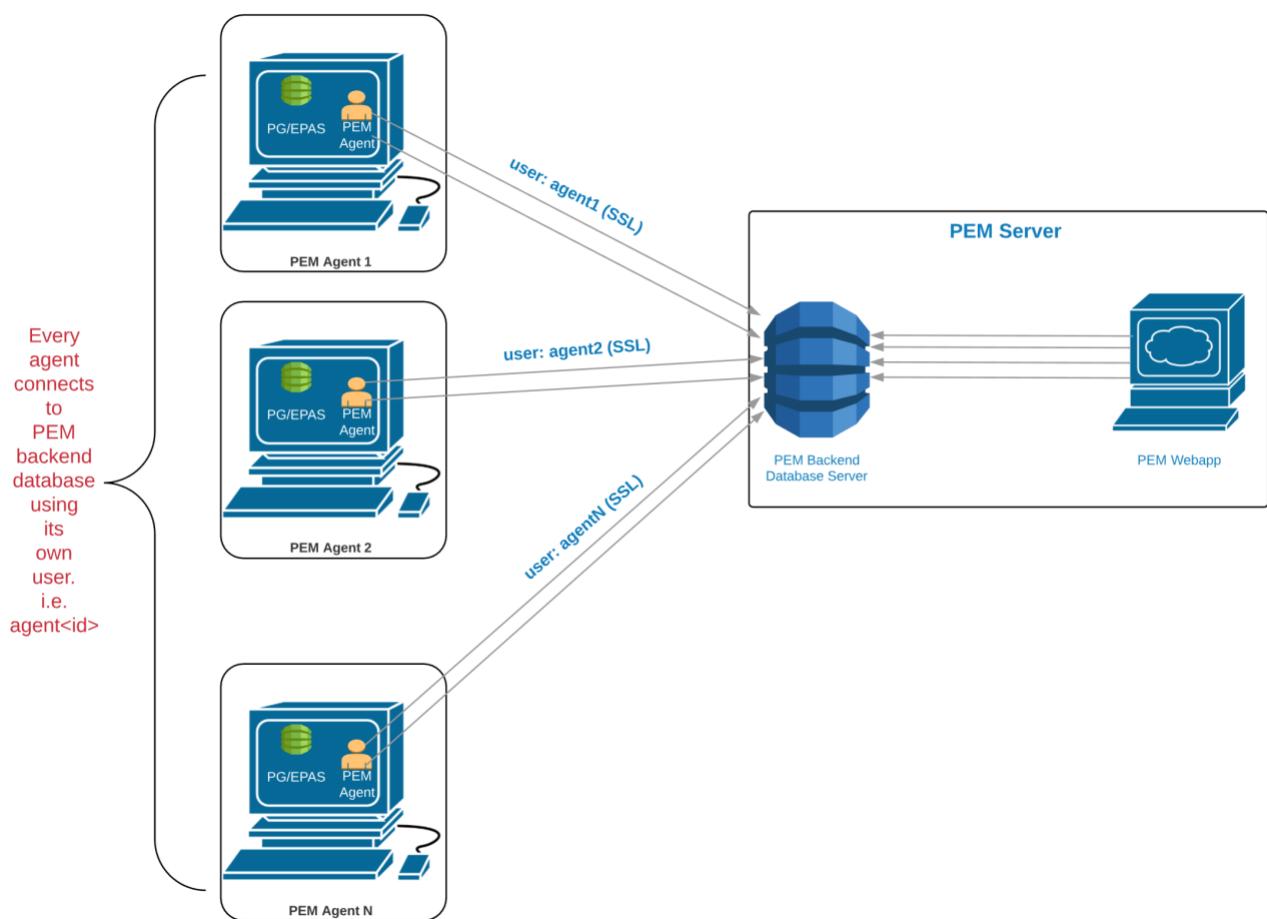
- Preparing the PEM Database Server – provides information about preparing the PEM database server to be used with pgBouncer.
- Configuring pgBouncer – provides detailed information about configuring pgBouncer to make it work with the PEM database server.
- Configuring the PEM Agent – provides detailed information about configuring a PEM Agent to connect to pgBouncer.

For detailed information about using the PEM web interface, please see the [PEM Administrator's Guide](#).

This document uses the term **Postgres** to mean either the PostgreSQL or the Advanced Server database.

8.1 The PEM Server - PEM Agent Connection Management Mechanism

Each PEM Agent connects to the PEM database server using the SSL certificates for each individual user. For example, an Agent with **ID#1** connects to the PEM database server using the **agent1** user.



Prior to PEM version 7.5, the following limitations disallowed the use of the connection pooler between the PEM Server and PEM Agent:

- The PEM Agent uses an SSL Certificate to connect the PEM database server.
- It uses an individual user identifier when connecting to the PEM database server.

EDB has modified the PEM Agent to allow the agent to use a common database user (instead of the dedicated agent users) to connect the PEM database server.



We recommend using PgBouncer versions equal to or later than version 1.9.0 as the connection pooler. Versions 1.9.0 or later support `cert` authentication; PEM Agents can connect to pgBouncer using SSL certificates.

8.2 Preparing the PEM Database Server

You must configure the PEM database server to work with PgBouncer; the following example demonstrates the steps required to configure the PEM database server.

1. Create a dedicated user named `pgbouncer` on the PEM database server. For example:

```
pem=# CREATE USER pgbouncer PASSWORD 'ANY_PASSWORD' LOGIN;
CREATE ROLE
```

2. Create a user named `pem_admin1` (a non-super user) with `pem_admin` and `pem_agent_pool role`

membership on the PEM database server. For example:

```
pem=# CREATE USER pem_admin1 PASSWORD 'ANY_PASSWORD' LOGIN
CREATEROLE;
CREATE ROLE
pem=# GRANT pem_admin, pem_agent_pool TO pem_admin1;
GRANT ROLE
```

- Grant **CONNECT** privileges to the **pgbouncer** user on the **pem** database. For example:

```
pem=# GRANT CONNECT ON DATABASE pem TO pgbouncer ;GRANT USAGE ON
SCHEMA pem TO pgbouncer;
GRANT
```

- Grant **USAGE** privileges to the **pgbouncer** user for the **pem** schema on the **pem** database. For example:

```
pem=# GRANT USAGE ON SCHEMA pem TO pgbouncer;
GRANT
```

- Grant **EXECUTE** privileges to the **pgbouncer** user on the **pem.get_agent_pool_auth(text)** function in the **pem** database. For example:

```
pem=# GRANT EXECUTE ON FUNCTION pem.get_agent_pool_auth(text) TO
pgbouncer;
GRANT
```

- Use the **pem.create_proxy_agent_user(varchar)** function to create a user named **pem_agent_user1** on the PEM database server. For example:

```
pem=# SELECT pem.create_proxy_agent_user('pem_agent_user1');
create_proxy_agent_user
-----
(1 row)
```

The function will create a user with the same name with a random password, and grant **pem_agent** and **pem_agent_pool** roles to the user. This allows pgBouncer to use a proxy user on behalf of the agent.

- Add the following entries to the start of the **pg_hba.conf** file of the PEM database server; this will allow pgBouncer user to connect to the **pem** database using the md5 authentication method. For example:

```
# Allow the PEM agent proxy user (used by
# pgbouncer) to connect to PEM server using
# md5

local pem pgbouncer,pem_admin1 md5
```

8.3 Configuring PgBouncer

You must configure PgBouncer to work with the PEM database server. In our example, we will run PgBouncer as the **enterprisedb** system user. The following steps outline the process of configuring pgBouncer (version >= 1.9).

1. Open a terminal window and navigate into the pgBouncer directory.
2. Change the owner of the `etc` directory for pgBouncer (where `pgbouncer.ini` resides) to `enterprisedb`, and change the directory permissions to `0700`. For example:

```
$ chown enterprisedb:enterprisedb /etc/edb/pgbouncer1.9
$ chmod 0700 /etc/edb/pgbouncer1.9
```

3. Change the contents of the `pgbouncer.ini` or `edb-pgbouncer.ini` file as follows:

```
[databases]
;; Change the pool_size according to maximum connections allowed
;; to the PEM database server as required.
;; 'auth_user' will be used for authenticate the db user (proxy
;; agent user in our case)

pem = port=5444 host=/tmp dbname=pem auth_user=pgbouncer
pool_size=80 pool_mode=transaction
* = port=5444 host=/tmp dbname=pem auth_user=pgbouncer
pool_size=10

[pgbouncer]
logfile = /var/log/edb/pgbouncer1.9/edb-pgbouncer-1.9.log
pidfile = /var/run/edb/pgbouncer1.9/edb-pgbouncer-1.9.pid
listen_addr = *
;; Agent needs to use this port to connect the pem database now
listen_port = 6432
;; Require to support for the SSL Certificate authentications
;; for PEM Agents
client_tls_sslmode = require
;; These are the root.crt, server.key, server.crt files present
;; in the present under the data directory of the PEM database
;; server, used by the PEM Agents for connections.
client_tls_ca_file = /var/lib/edb/as11/data/root.crt
client_tls_key_file = /var/lib/edb/as11/data/server.key
client_tls_cert_file = /var/lib/edb/as11/data/server.crt
;; Use hba file for client connections
auth_type = hba
;; Authentication file, Reference:
;; https://pgbouncer.github.io/config.html#auth\_file
auth_file = /etc/edb/pgbouncer1.9/userlist.txt
;; HBA file
auth_hba_file = /etc/edb/pgbouncer1.9/hba_file
;; Use pem.get_agent_pool_auth(TEXT) function to authenticate
;; the db user (used as a proxy agent user).
auth_query = SELECT * FROM pem.get_agent_pool_auth($1)
;; DB User for administration of the pgbouncer
admin_users = pem_admin1
;; DB User for collecting the statistics of pgbouncer
stats_users = pem_admin1
server_reset_query = DISCARD ALL
;; Change based on the number of agents installed/required
max_client_conn = 500
;; Close server connection if its not been used in this time.
;; Allows to clean unnecessary connections from pool after peak.
```

```
server_idle_timeout = 60
```

4. Use the following command to create and update the `/etc/edb/pgbouncer1.9/userlist.txt` authentication file for PgBouncer.

```
pem=# COPY (
  SELECT 'pgbouncer'::TEXT, 'pgbouncer_password'
  UNION ALL
  SELECT 'pem_admin1'::TEXT, 'pem_admin1_password'
  TO '/etc/edb/pgbouncer1.9/userlist.txt'
  WITH (FORMAT CSV, DELIMITER ' ', FORCE_QUOTE *);

COPY 2
```

!!! Note A super user cannot invoke the PEM authentication query function `pem.get_proxy_auth(text)`. If the `pem_admin` user is a super user, you must add the password to the authentication file (`enterprisedb` in the above example).

5. Create an HBA file (`/etc/edb/pgbouncer1.9/hba_file`) for PgBouncer that contains the following content:

```
# Use authentication method md5 for the local connections to
# connect pem database & pgbouncer (virtual) database.
local pgbouncer all md5
# Use authentication method md5 for the remote connections to
# connect to pgbouncer (virtual database) using enterprisedb
# user.

host pgbouncer,pem pem_admin1 0.0.0.0/0 md5

# Use authentication method cert for the TCP/IP connections to
# connect the pem database using pem_agent_user1

hostssl pem pem_agent_user1 0.0.0.0/0 cert
```

6. Change the owner of the HBA file (`/etc/edb/pgbouncer1.9/hba_file`) to `enterprisedb`, and change the directory permissions to `0600`. For example:

```
$ chown enterprisedb:enterprisedb /etc/edb/pgbouncer1.9/hba_file
$ chmod 0600 /etc/edb/pgbouncer1.9/hba_file
```

7. Enable the PgBouncer service, and start the service. For example:

```
$ systemctl enable edb-pgbouncer-1.9

Created symlink from
/etc/systemd/system/multi-user.target.wants/edb-pgbouncer-1.9.service
to /usr/lib/systemd/system/edb-pgbouncer-1.9.service.

$ systemctl start edb-pgbouncer-1.9
```

8.4 Configuring the PEM Agent

You can use an RPM package to install a PEM Agent; for detailed installation information, please see the [PEM Linux Installation Guide](#).

Please note that PEM Agent which is responsible for sending SNMP notifications should not be configured with pgBouncer. For example, if the default PEM Agent installed along with PEM Server is used for SNMP notifications, then it should not be configured with pgBouncer.

Configuring a New PEM Agent (Installed via RPM)

After using an RPM package to install the PEM agent, you will need to configure it to work it against a particular PEM database server. Use the following command:

```
$ PGSSLMODE=require PEM_SERVER_PASSWORD=pem_admin1_password
/usr/edb/pem/agent/bin/pemworker --register-agent --pem-server
pem_agent_user1 --display-name "Agent Name"
```

Postgres Enterprise Manager Agent registered successfully!

In above command, the `--pem-agent-user` argument instructs the agent to create an SSL certificate and key pair for the `pem_agent_user1` database user in `/root/.pem` directory.

For example:

```
/root/.pem/pem_agent_user1.crt
/root/.pem/pem_agent_user1.key
```

The keys will be used by the PEM agent to connect to the PEM database server as `pem_agent_user1`. It will also create an agent configuration file named `/usr/edb/pem/agent/etc/agent.cfg`.

You will find a line mentioning the agent-user to be used in the `agent.cfg` configuration file.

For example:

```
$ cat /usr/edb/pem/agent/etc/agent.cfg
[PEM/agent]
pem_host=172.16.254.22
pem_port=6432
agent_id=12
agent_user=pem_agent_user1
agent_ssl_key=/root/.pem/pem_agent_user1.key
agent_ssl_crt=/root/.pem/pem_agent_user1.crt
log_level=warning
log_location=/var/log/pem/worker.log
agent_log_location=/var/log/pem/agent.log
long_wait=30
short_wait=10
alert_threads=0
enable_smtp=false
enable_snmp=false
enable_webhook=false
max_webhook_retries=3
allow_server_restart=true
allow_package_management=false
```

```
allow_streaming_replication=false
max_connections=0
connect_timeout=-1
connection_lifetime=0
allow_batch_probes=false
heartbeat_connection=false
```

Configuring an Existing PEM Agent (Installed via RPM)

If you are using an existing PEM agent, you can copy the SSL certificate and key files to the target machine, and reuse the files. You will need to modify the files, adding a new parameter and replacing some parameters in the existing `agent.cfg` file:

Add a line for `agent_user` to be used for the agent. For example:

```
agent_user=pem_agent_user1
```

Update the port to specify the pgBouncer port. For example:

```
pem_port=6432
```

Update the certificate and key path locations. For example:

```
agent_ssl_key=/root/.pem/pem_agent_user1.key
agent_ssl_crt=/root/.pem/pem_agent_user1.crt
```

Please note: as an alternative, you can run the agent self registration script, but that will create a new agent id. If you do run the agent self-registration script, you must replace the new agent id with existing id, and disable the entry for the new agent id in the `pem.agent` table. For example:

```
pem=# UPDATE pem.agent SET active = false WHERE id = <new_agent_id>;
```

```
UPDATE 1
```

!!! Note Keep a backup of the existing SSL certificate, key file, and agent configuration file.

9 PEM Security Guide

This document provides information about security practices you should consider when configuring PEM. PEM functionality does not require you to enforce these practices; however, EDB recommends these practices to enhance the overall system's security.

PEM is dependent on third-party components from the vendor repository, including the Apache web server, OpenSSL, snmp++, libcurl, etc. To ensure these components are up to date, you should update your operating system and regularly apply security updates to avoid any security vulnerability. Without the most recent security patches, your system is potentially vulnerable to cyberattacks. Security patches protect your devices and their data by applying the latest updates to guard against the latest threats.

Some of the benefits of regularly applying security patches include:

- Reduced exposure to cyberattacks
 - Avoiding lost productivity
 - Data protection from malware (like ransomware)
 - Avoid worm infections that use security loopholes to spread over the network
-

9.1 Apache HTTPD Security Configurations

On a Windows system, Apache HTTPD is named PEM HTTPD, and the Apache httpd configuration file (`pme.conf`) is located in the `<Apache_Installation_Path>/conf/addons` directory. The ssl configuration file (`httpd-ssl-pem.conf`) is located in the `<Apache_Installation_Path>/conf/addons` directory.

On a Linux system, the apache httpd configuration file (`edb-pem.conf`) is located in the `<Apache_Installation_Path>/conf.d` directory and the SSL configuration file (`edb-ssl-pem.conf`) is located in the `<Apache_Installation_Path>/conf.d` directory.

Disable SSLv2 and SSLv3

You should disable SSL versions SSLv2, SSLv3, TLS 1, and TLS 1.1 as these versions are the most vulnerable and are affected by cryptographic concerns.

To disable the versions, please add the following command to apache httpd configuration file:

```
SSLProtocol -ALL +TLSv1.2
```

You need to restart the Web Server to apply the changes to the configuration file.

By default, PEM adds the following lines to the SSL configuration file to allow use of TLS 1.2 for security:

```
SSLProtocol -All TLSv1.2
```

```
SSLProxyProtocol -All TLSv1.2
```

Secure httpd with SSL Certificates

EDB recommends having an additional layer of SSL security for the web application.

By default, during PEM installation, PEM will generate and use self-signed certificates for Apache/HTTPD server. If you want to use your own SSL certificate for PEM, you must update the Apache configuration file.

On a Linux system, you need to open the Apache httpd configuration file (`edb-ssl-pem.conf`) in a text editor as a user with write permission on the file. You must also change the server name and file names in the configuration file to match your certificate files.

There are two SSL Directives within the PEM VirtualHost section which you need to update:

- `SSLCertificateFile` is your DigiCert certificate file (e.g.,`your_domain_name.crt`).
- `SSLCertificateKeyFile` is the `.key` file generated when you created the CSR (e.g., `your_private.key`).

For example, update:

```
SSLEngine on
SSLCertificateFile /path/to/your_domain_name.crt
SSLCertificateKeyFile /path/to/your_private.key
```

You can also replace the httpd self-signed SSL certificates with trusted CA-signed certificates in Postgres Enterprise Manager. Follow the link shown below for the steps:

<https://www.enterprisedb.com/postgres-tutorials/how-replacing-httpd-self-signed-ssl-certificates-trusted-ca-signed-certificates>

Disable Web Server Information Exposure

EDB recommends disabling all web server signatures as part of web server security. The web server will expose a software signature; to disable the signature, add the following parameters to the Apache httpd configuration file. By default, PEM disables exposure of the information by adding the below parameters to the Apache httpd configuration file:

```
ServerTokens Prod
```

```
ServerSignature Off
```

The `ServerTokens` directive controls the server response header field, which is returned to the client. We recommend hiding the Apache server version by adding this parameter in the Apache httpd configuration file.

The `ServerSignature` directive includes a footer for server-produced documents. The footer contains information regarding the Apache configuration, like the Apache and operating system version. To limit the display of such information, we recommend disabling this directive in the Apache httpd configuration file.

You need to restart the web server to apply any changes to the Apache httpd configuration file.

Disable Directory Listing

The directory listing can allow an attacker to view complete directory contents. By default, the web server enables this option, and an attacker can discover and view any file. This listing could lead to the attacker reverse engineering an application to obtain the source code, analyze it for possible security flaws, and discover more information about an application.

To avoid this, you should disable the directory listing by setting the `Options` directive in the Apache httpd configuration file. By default, PEM disables the directory listing by setting the option below in the web server configuration file:

```
<Directory /application/directory> Options -Indexes </Directory>
```

You need to restart the web server to apply the changes made to the configuration file.

Restrict the Access to a Network or IP Address

Apache provides access control based on the client hostname or IP address. To view the application by specific IP address or network, a user can modify the Apache configuration file as shown below to provide your network address within the `Allow` directive:

```
<Directory /application/hostname>
  Options None
  AllowOverride None
  Order deny,allow
  Deny from all
  Allow from 192.168.0.0/24
</Directory>
```

PEM uses the `ALLOWED_HOSTS` configuration parameter in the application configuration file to provide the allowed hosts' list of IP addresses. The application configuration `config_local.py` file is located in `<PEM_INSTALLATION_PATH>/web`.

By default, PEM allows all the hosts to connect with the application.

For example:

You can set the range of IP addresses in the configuration file as below:

```
ALLOWED_HOSTS = ['225.0.0.0/8', '226.0.0.0/7', '228.0.0.0/6']
```

You can set the IP addresses to allow a host on a subnet level in the configuration file as below:

```
ALLOWED_HOSTS = ['192.0.2.0/28', '::192.0.2.0/124']
```

You can set a specific individual host address (based on the IP address) in the configuration file as below:

```
ALLOWED_HOSTS = ['127.0.0.1', '192.168.0.1']
```

You need to restart the web server to apply the changes to the application configuration file.

Cross-site Tracing

There are two HTTP methods to debug the web server connections - TRACE and TRACK. When an HTTP TRACE request is sent to a web server that supports it, that server will respond, echoing the data passed to it, including any HTTP headers. We recommend disabling these methods within the Apache Configuration.

To disable the TRACE method for all virtual hosts, add the following line to the Apache httpd configuration file:

```
TraceEnable off
```

To disable these methods for a specific virtual host, add the following lines for each virtual host in the Apache configuration file. PEM does add the following lines to the Apache httpd configuration file:

```
RewriteEngine on
RewriteCond %{REQUEST_METHOD} ^(TRACE|TRACK|OPTIONS)
RewriteRule .\* - [F]
```

Run Web Server from a Non-privileged User Account

Running the Apache web server as a root user creates a security issue. We always recommend running the web server as a unique non-privileged user. This helps to secure other services running in the event of any security breaches.

PEM runs as a WSGI application. To delegate the WSGI applications that are running, you can create distinct daemon processes using the `WSGIDaemonProcess` directive.

On Linux, the Apache web server starts as the root user, but the daemon processes which PEM application runs as `pem` user. On Windows, the `WSGIDaemonProcess` directive and its features are not available. PEM HTTPD is installed as a service during the installation, and the user needs to be a member of the `Administrators` group for the service installation to succeed.

By default the Apache services are registered to run as the system user (the `LocalSystem` account).

Customize Security HTTP Headers in PEM WebServer

PEM contains its own configuration file to fix the following security issues. We recommend overriding the configuration only of `config_local.py` and not of `config.py`. The `config_local.py` is not present on the systems in most of the cases; hence users need to create it to override the application-level configurations. Please note that during a PEM upgrade, `config_local.py` will not be overwritten, but changes in `config.py` and `config_distro.py` will be overridden. Users need to remove `config_local.py` after uninstalling the PEM.

By default, `config_local.py` is located in `/usr/edb/pem/web` on Linux, and at `C:\ProgramFiles\edb\pem\server\share\web` on Windows.

Host Header Injection Attacks

HTTP host header attacks exploit vulnerable websites that handle the host header value in an unsafe way. If the server implicitly trusts the host header and fails to validate or escape it properly, an attacker may be able to use this input to inject harmful payloads that manipulate server-side behavior. The web applications typically don't know what domain they are deployed on unless specified in a configuration file during setup. When they need to know the current domain, for example, they may resort to retrieving the domain from the host header to generate an absolute URL. The host header is a potential vector for exploiting a range of other vulnerabilities, most notably web cache poisoning & SQL injections.

X-Frame-Options

X-Frame-Options can be used to indicate if a browser should be allowed to render a page in an `<iframe>` tag. It was designed specifically to protect against clickjacking. PEM has a host validation `X_FRAME_OPTIONS` option to prevent such kind of attacks, which you can configure in the `config_local.py` file. The default is:

```
X_FRAME_OPTIONS = "SAMEORIGIN"
```

Content-Security-Policy

Content-Security-Policy is part of the HTML5 standard and provides a broader range of protection than the X-Frame-Options header (which it replaces). It is designed in such a way that website authors can whitelist individual domains from which resources (like scripts, stylesheets, and fonts) can be loaded, and also domains that are permitted to embed a page.

PEM has a host validation `CONTENT_SECURITY_POLICY` option to prevent such kinds of attacks, which you can configure in the `config_local.py` file. The default is:

```
CONTENT_SECURITY_POLICY = "default-src https: data: blob: 'unsafe-inline' 'unsafe-eval';"
```

Strict-Transport-Security

The Strict-Transport-Security response header (often abbreviated as HSTS) allows a web site or web application to tell browsers that it should only be accessed using HTTPS instead of HTTP. This option allows you to prevent a man-in-the-middle attack. The default is:

```
STRICT_TRANSPORT_SECURITY = "max-age=31536000;includeSubDomains"
```

!!! Note Adding this parameter may cause problems if config is changed. Hence it is recommended to add this only after PEM installation is complete and tested.

X-Content-Type-Options

The X-Content-Type-Options response HTTP header is a marker used by the server to indicate that the MIME types advertised in Content-Type headers should not be changed and followed. This is a way to opt out of MIME type sniffing, or, in other words, to say that the MIME types are deliberately configured. The default is:

```
X_CONTENT_TYPE_OPTIONS = "nosniff"
```

X-XSS-Protection

Cross-site scripting (XSS) is one of the most common application-layer vulnerabilities in the web servers. XSS enables attackers to inject client-side script into web pages viewed by other users. The HTTP X-XSS-Protection response to the header is a feature of Internet Explorer, Chrome, and Safari that stops pages from loading when they detect reflected cross-site scripting (XSS) attacks. Although these protections are largely unnecessary in modern browsers when sites implement a strong Content-Security-Policy that disables the use of inline JavaScript ('unsafe-inline') can still provide protections for users of older web browsers that don't yet support CSP. The default is:

```
X_XSS_PROTECTION = "1; mode=block"
```

To avoid this, you need to add the following options to the Apache configuration file:

```
<IfModule mod_headers.c>
  Header set X-XSS-Protection "1; mode=block"
</IfModule>
```

A web server restart is required for configuration file changes to be applied.

By default, PEM sets `X-XSS-Protection` to "1; mode=block" in the application configuration file which is located at `/usr/edb/pem/web/config.py`.

This changes requires an Apache service restart in order to take effect

For more detailed information on `config.py` file you can see [PEM Online Help](#).

Cookie Security

Cookies are small packets of data that a server can send to your browser to store configuration data. The browser automatically sends them along with all requests to the same server, so it's important to know how to secure cookies. There are multiple configuration options provided by PEM to make cookies secure which you can refer to in `config.py` but the three that follow are most important.

1. `SESSION_COOKIE_SECURE` - Setting the secure flag prevents the cookie from ever being sent over an unencrypted connection. It basically tells the browser to never add the cookie to any request to the server that does not use an encrypted channel. The browser will only add cookies to connections such as HTTPS. The default is:

```
SESSION_COOKIE_SECURE = True
```

2. `SESSION_COOKIE_HTTPONLY` - By default the content of cookies can be read via JavaScript. The `HTTPOnly` flag prevents scripts from reading the cookie. As the name `HTTPOnly` implies, the browser will only use the cookie in HTTP(S) requests. This prevents hackers from using XSS vulnerabilities to learn the contents of the cookie. For example, for the `sessionId` cookie it is never necessary to read the cookie with a client-side script, so for `sessionId` cookies, you can always set the `HTTPOnly` flag. The default is:

```
SESSION_COOKIE_HTTPONLY = True
```

3. `ENHANCED_COOKIE_PROTECTION` - When you set this option to `True` then a token will be generated according to the IP address and user agent. In all subsequent requests, the token will be recalculated and checked against the one computed for the first request. If the session cookie is stolen and the attacker tries to use it from another location, the generated token will be different, and in that case, the extension will clear the session and block the request. The default is:

```
ENHANCED_COOKIE_PROTECTION = True
```

!!! Note This option can cause problems when the server is deployed in dynamic IP address hosting environments, such as Kubernetes or behind load balancers. In such cases, this option should be set to `False`.

This changes requires an Apache service restart in order to take effect.

For more detailed information on `config.py` file you can see [PEM Online Help](#).

9.2 PEM application Security Configurations

Session Timeout

Insufficient session expiration by the web application increases the exposure of other session-based attacks, as it allows time for the attacker to be able to reuse a valid session ID and hijack the associated session. The shorter the session interval is, the lesser the time an attacker has to use the valid session ID. We recommend setting the inactivity timeout for the web application to a low value to avoid this security issue.

Postgres Enterprise Manager provides a way to set the timeout value for a user session. When there is no user activity for a specified duration on the web console, PEM will log out the user from the web console. A PEM Administrator can set the length of time for inactivity. This value is application-wise, rather than for an individual user. To configure the timeout duration, modify the `USER_INACTIVITY_TIMEOUT` parameter in `config_local.py` file, located in the `<PEM_INSTALLATION_PATH>/web` directory. By default this functionality is disabled.

For example, to specify that an application should log out a user after 15 minutes of inactivity, set:

```
USER_INACTIVITY_TIMEOUT = 900
```

!!! Note The timeout value is specified in seconds.

This changes requires an Apache service restart in order to take effect.

For more detailed information on `config.py` file you can see [PEM Online Help](#).

RestAPI Header Customization

You can customize the RestAPI token headers as per your requirements. The default values are not exposed by the `config.py` file; customize the following headers in the `config_local.py` file:

`PEM_HEADER_SUBJECT_TOKEN_KEY`

This configuration option will allow you to change the HTTP header name to get the generated token. By default, when the user sends a request to create a token, the server response will have an ‘X-Subject-Token’ header, which will contain the value of a newly generated token. If you want to customize the header name, then you can update the `config_local.py` file as shown:

```
PEM_HEADER_SUBJECT_TOKEN_KEY = 'Pem-RestAPI-Generate-Token'
```

Which will give you the following output:

```
curl -ik -X POST -d '{"username":"enterprisedb","password":"edb"}' -H "Content-Type: application/json" https://localhost:8443/pem/api/token/
```

```
HTTP/1.1 201 CREATED
```

```
Date: Thu, 29 Oct 2020 11:03:48 GMT
```

```
Server: Apache
```

```
Content-Length: 326
```

```
Pem-RestAPI-Generate-Token: 997aef95-d46d-4d84-932a-a80146eaf84f
```

`PEM_HEADER_TOKEN_KEY`

This configuration option will allow you to change the HTTP request header name through which you will send the token to the PEM server. By default, when the user sends a request to generate a token, the token header will be `X-Auth-Token`. If you want to customize the RestAPI request header name then you can update the `config_local.py` file as follows:

```
PEM_HEADER_TOKEN_KEY = 'Pem-Token'
```

Which will allow you to send the token:

```
$ curl -Lk -X GET -H "Pem-Token: gw5rzaloxyp91ttd1c97w24b5sv60clic24sxy9"
https://localhost:8443/pem/api/v4/agent
```

PEM_TOKEN_EXPIRY

This configuration option will allow you to change the PEM RestAPI token expiry time after it is generated. By default, the token expiry time is set to 20 minutes (1200 seconds). If you want to change the token expiry time to 10 minutes then you can update the `config_local.py` file as shown:

```
PEM_TOKEN_EXPIRY = 600
```

This changes requires an Apache service restart in order to take effect.

Role-based Access Control in PEM

Role-based access control (RBAC) restricts application access based on a user's role within an organization and is one of the primary methods for access control. The roles in RBAC refer to the levels of access that users have to the application. Users are only allowed to access the information necessary to effectively perform their job duties. Roles in PEM are inheritable and additive rather than subscriptive. In simple terms, as a PEM admin you need to grant the lowest level role to the user and then grant the roles which are required for the user to perform their respective tasks. For example, to give access only to SQL profiler:

```
CREATE ROLE user_sql_profiler WITH LOGIN NOSUPERUSER NOCREATEDB NOCREATEROLE INHERIT
NOREPLICATION CONNECTION LIMIT -1 PASSWORD 'xxxxxx';
```

```
GRANT pem_user, pem_comp_sqlprofiler TO user_sql_profiler;
```

For more detailed information on roles, you can see [PEM Roles](#).

SQL/Protect Plugin

Preventing a SQL injection attack is usually the responsibility of the application developer. The database administrator typically has little or no control over the potential threat. The difficulty for database administrators is that the application must have access to the data to function properly.

`SQL/Protect` is a module that allows a database administrator to protect a database from SQL injection attacks. `SQL/Protect` provides a layer of security in addition to the standard database security policies by examining incoming queries for typical SQL injection profiles.

There are several different techniques used to perpetrate SQL injection attacks. A specific signature characterizes each technique. `SQL/Protect` examines queries for Unauthorized Relations, Utility Commands, SQL Tautology, Unbounded DML Statements. `SQL/Protect` gives the control back to the database administrator by alerting the administrator to

potentially dangerous queries and blocking them.

!!! Note This plugin works only on the EPAS server, so this is useful only when your PEM database is hosted on the EPAS server.

For more detailed information about the SQL Profiler plugin, see the [PEM Online Help - SQL Profiler](#).

Password Management

One security tip for PEM administrative users is to change your PEM login passwords to something new regularly. Changing your password avoids a number of dangers including:

- breaches of multiple accounts
- prevents constant access
- prevents the use of saved passwords on a physically unsecured system
- limits access gained by keystroke loggers

Run pemAgent Jobs with a Non-root User

In most cases, `pemAgent` is installed as a root user, and runs as a daemon process with root privileges. By default, PEM disables running the scheduled jobs/task. PEM provides support for running scheduled jobs as a non-root user by changing the `pemAgent` configuration file.

To run scheduled jobs as a non-root user, modify the entry for the `batch_script_user` parameter in the `agent.cfg` file and specify the user that should be used to run the script. You can either specify a non-root user or root user identity. If you do not specify a user, or the specified user does not exist, then the script will not execute. Restart the agent after modifying the file. If a non-root user is running `pemagent`, then the value of `batch_script_user` will be ignored, and the same non-root user used for running the `pemagent` will execute the script.

To invoke a script on a Windows system, set the registry entry for `AllowBatchJobSteps` to `true` and restart the PEM agent. PEM registry entries are located in:

`HKEY_LOCAL_MACHINE\Software\Wow6432Node\EnterpriseDB\PEM\agent`

Changing the pemAgent and PEM Backend Database Server Certificates

By default, when you install PEM, the installer will generate and use self signed certificates for the pemAgent and PEM database server. PemAgent uses these certificates when connecting to the PEM database server. To use your own SSL certificate for the pemAgent and PEM database server, follow the steps mentioned in the [Managing Certificates](#) section of the [PEM Administrators's Guide](#):

!!! Note PEM does not support placing the SSL CA certificates at a custom location; you should not change the location of `ca_certificate.crt` and `ca_key.key`.

10 SQL Profiler

Postgres Enterprise Manager (PEM) is designed to assist database administrators, system architects, and performance analysts when administering, monitoring, and tuning PostgreSQL and Advanced Server database servers. PEM has been designed to manage and monitor a single server or multiple servers from a single console, allowing complete control over monitored databases.

The SQL Profiler Plugin works with PEM to allow you to profile a server's workload. The SQL Profiler plugin may be installed on servers with or without a PEM Agent, however traces can only be run in ad-hoc mode on unmanaged servers, and may only be scheduled on managed servers.

This document provides step-by-step instructions to guide you through the installation and use of SQL Profiler.

SQL Profiler is officially supported only on the EDB distributions of PostgreSQL and Advanced Server supported versions. The plugin is distributed via StackBuilder, or as operating system dependent packages in EDB's yum repositories. The plugin is also distributed and installed with the Advanced Server installations.

Throughout this guide, the term **Postgres** refers to either a PostgreSQL or an Advanced Server installation, where either is appropriate.

10.1 Installing the SQL Profiler Plugin

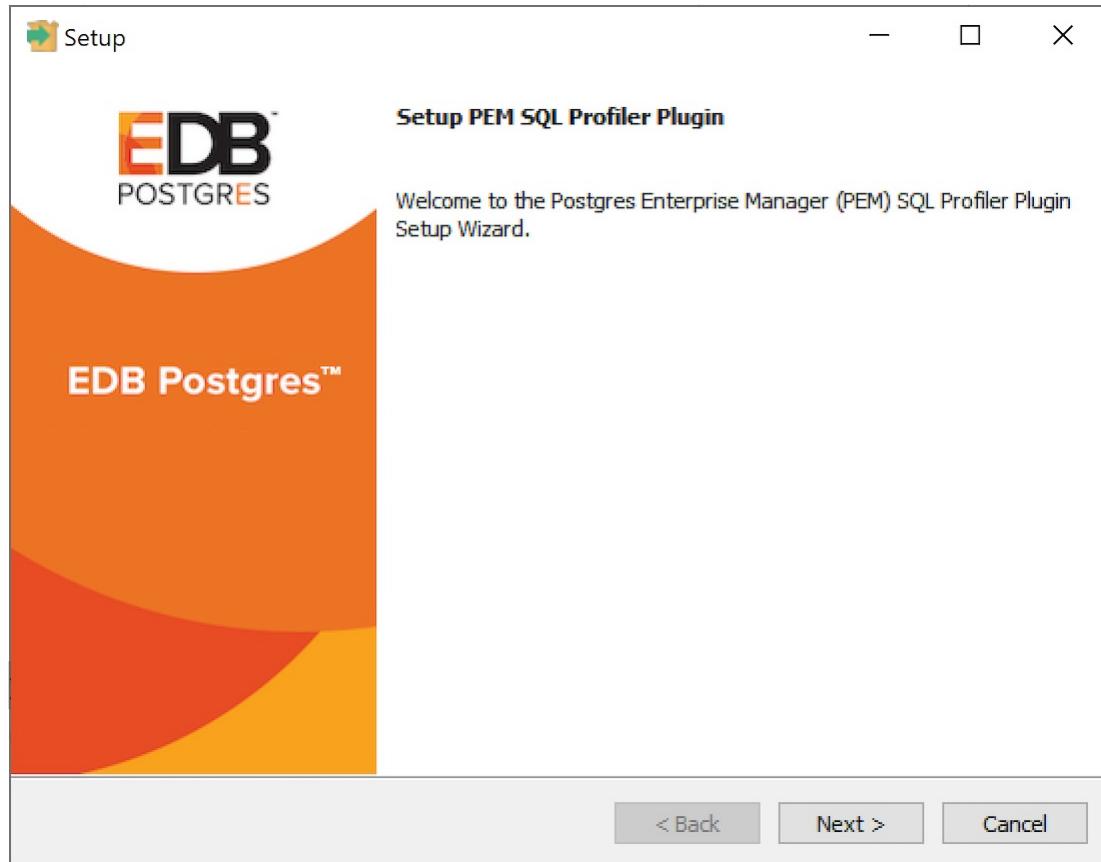
You must install the plugin on each server on which you wish to use SQL Profiler. For example, if you have a host running PostgreSQL 9.6 and PostgreSQL 10, you must install two versions of the plugin, one for each server.

Follow the installation steps listed below to install the plugin for PostgreSQL before continuing to the **Configuration** section. If you are using Advanced Server, you can skip installation and move ahead to the **Configuration** section.

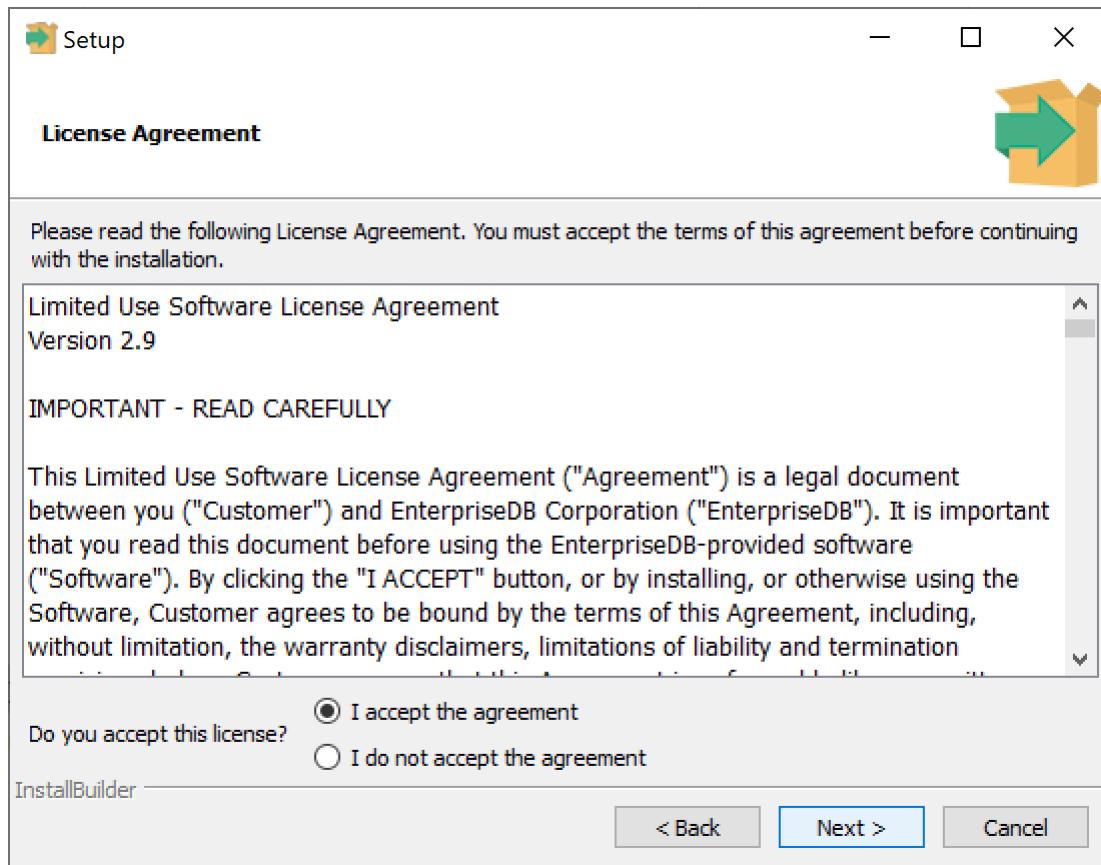
You can use the graphical installer to install any version of SQL Profiler on the Windows platform. On Linux, use an RPM package to install the SQL Profiler. For detailed information about configuring the EDB repository for your host platform, see the [PEM Linux Installation Guide](#).

Installing SQL Profiler on Windows

To invoke the SQL Profiler graphical installer, assume **Administrator** privileges, navigate into the directory that contains the installer, and double-click the installer icon. The SQL Profiler installer welcomes you to the Setup Wizard.



Click **Next** to continue to the **License Agreement**.



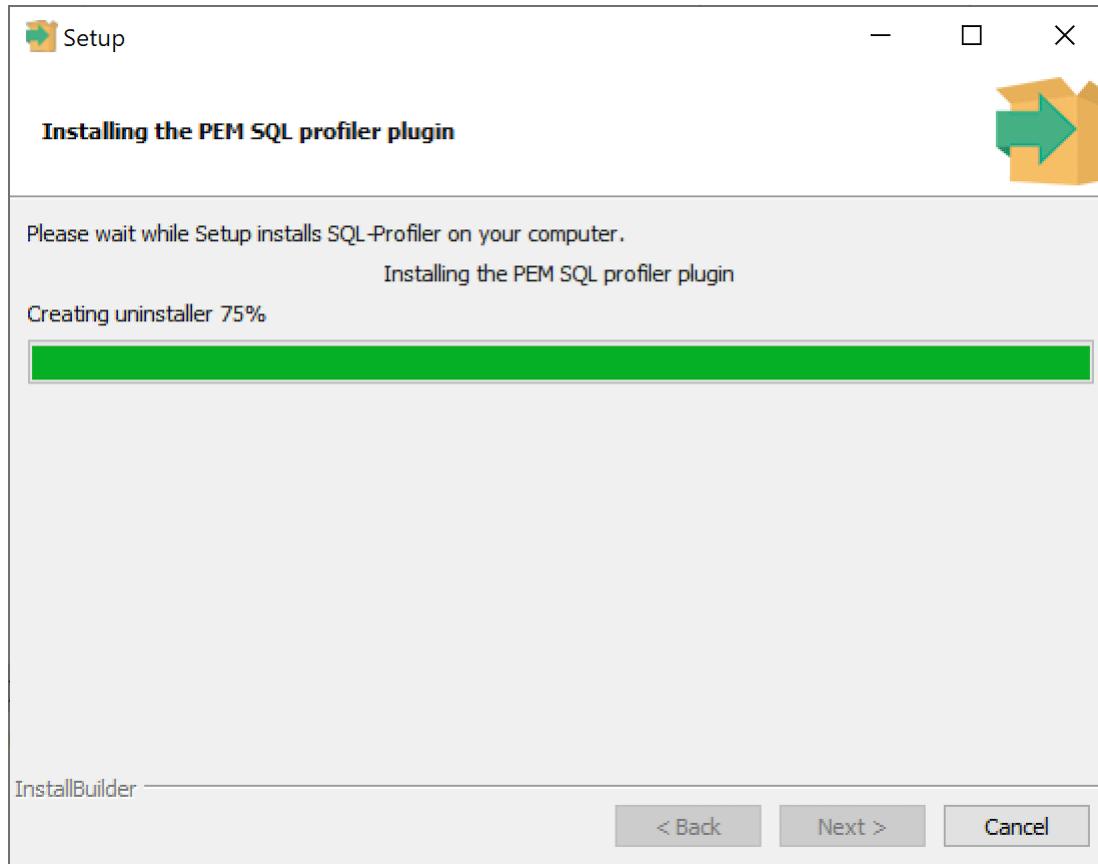
Carefully review the license agreement before highlighting the appropriate radio button and accepting the agreement; click **Next** to continue to the **Installation Directory** dialog.



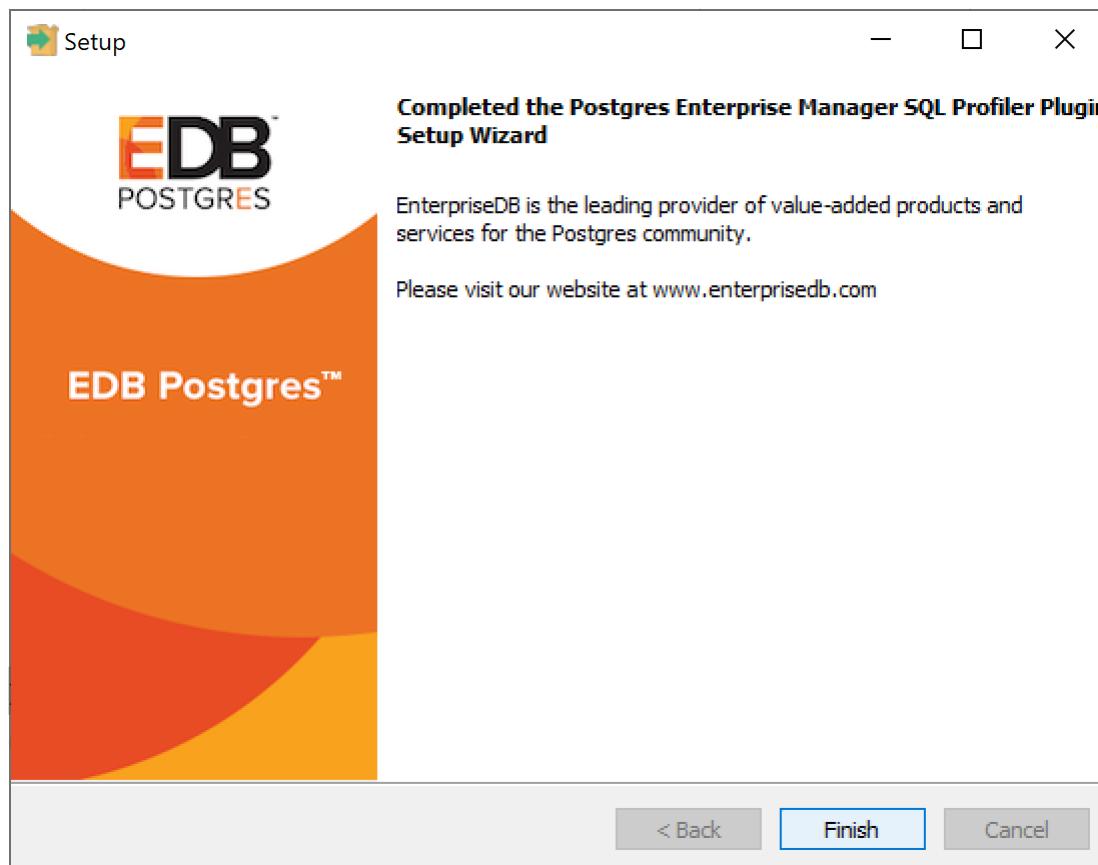
Specify an alternate location for the installation directory, or accept the default location and click **Next** to continue.



The wizard is now ready to install the SQL Profiler plugin. Click **Next** to continue.



The SQL Profiler plugin installer displays progress bars as it copies files to your system.



When the installation is complete, the SQL Profiler plugin is ready to be configured.

Using an RPM Package to Install SQL Profiler

!!! Note You may be required to add the `sslutils` package to your PostgreSQL database servers before installing SQL Profiler.

If you have already configured the EDB repository on your system, you can use `yum` or `dnf` to install SQL Profiler:

```
yum install postgresql<X>-sqlprofiler
```

or

```
dnf install postgresql<X>-sqlprofiler
```

Where, `<X>` is the version of your Postgres installation.

For detailed information about configuring the EDB repository, please see the [PEM Linux Installation Guide](#).

Installing SQL Profiler on Debian/Ubuntu

!!! Note You may be required to add the `sslutils` package to your PostgreSQL database servers before installing SQL Profiler.

You can use an `apt` command to install SQL Profiler using DEB on Debian 9.x or Ubuntu 18; assume root privileges and enter:

```
apt install postgresql-<X>-sqlprofiler
```

Where, `<X>` is the version of your Postgres installation.

When the installation is complete, the SQL Profiler plugin is ready to be configured.

Configuring SQL Profiler

The SQL Profiler plugin is not automatically enabled when the installation process completes. This allows you to restart the server at a convenient time, and prevents the plugin from being loaded unnecessarily on systems where it is not required on a continual basis.

Use the following steps to enable the plugin:

1. Edit the `postgresql.conf` file on the server you wish to profile, modifying the `shared_preload_libraries` parameter as shown below:

```
shared_preload_libraries = '$libdir/sql-profiler'
```

2. Restart the Postgres server.

3. Using the `Query Tool` or the `psql` command line interface, run the `sql-profiler.sql` script in the database specified as the `Maintenance Database` on the server you wish to profile. If you are using:

- PostgreSQL, the default maintenance database is `postgres`.
- Advanced Server, the default maintenance database is `edb`.

To use the PEM Query Tool to run the script, highlight the name of the maintenance database in the **Browser** tree control, and navigate through the **Tools** menu to select **Query tool**. When the Query Tool opens, use the **Open** option on the **Files** menu to open a web browser and navigate to the **sql-profiler.sql** script. By default, the **sql-profiler.sql** script is located in the **contrib** folder, under your Postgres installation.

When the script opens in the **SQL Editor** panel of the Query Tool, highlight the content of the script in the SQL Editor and select the **Execute** option from the **Query** menu (or click the **Execute** icon) to invoke the script and configure SQL Profiler.

You can also use the psql command line to invoke the configuration script. The following command uses psql to invoke the **sql-profiler.sql** script on an Advanced Server database on a Linux system:

```
$ /usr/edb/as<x>/bin/psql -U postgres postgres <
/usr/edb/as<x>/share/contrib/sql-profiler.sql
```

where <x> is the version of the Advanced Server.

After configuring SQL Profiler, it is ready to use with all databases that reside on the server.



To access SQL Profiler functionality, highlight the name of the database in the PEM **Browser** tree control; navigate through **Server** option under **Tools** menu to the **SQL Profiler** pull-aside menu. Menu options allow you to manage your SQL traces:

- Select **Create trace...** to define a new trace.
- Select **Open trace...** to open an existing trace.
- Select **Delete trace(s) ...** to delete one or more traces.
- Select **View scheduled trace(s) ...** to review a list of scheduled traces.

10.2 SQL Profiling and Analysis

Most RDBMS experts agree that inefficient SQL code is the leading cause of most database performance problems. The challenge for DBAs and developers is to locate the poorly-running SQL code in large and complex systems, and then optimize that code for better performance.

The SQL Profiler component allows a database superuser to locate and optimize poorly-running SQL code. Users of Microsoft SQL Server's Profiler will find PEM's SQL Profiler very similar in operation and capabilities. SQL Profiler is installed with each Advanced Server instance; if you are using PostgreSQL, you must download the SQL Profiler installer, and install the SQL Profiler product into each managed database instance you wish to profile.

For each database monitored by SQL Profiler, you must:

1. Edit the `postgresql.conf` file; you must include the SQL Profiler library in the `shared_preload_libraries` configuration parameter.

For Linux installations, the parameter value should include:

```
$libdir/sql-profiler
```

on Windows, the parameter value should include:

```
$libdir/sql-profiler.dll
```

2. Create the functions used by SQL Profiler in your database. The SQL Profiler installation program places a SQL script (named `sql-profiler.sql`) in the `share/postgresql/contrib` subdirectory of the main PostgreSQL installation directory on Linux systems. On Windows systems, this script is located in the `share` subdirectory. You must invoke this script on the maintenance database specified when registering the server with PEM.
3. Stop and re-start the server for the changes to take effect.

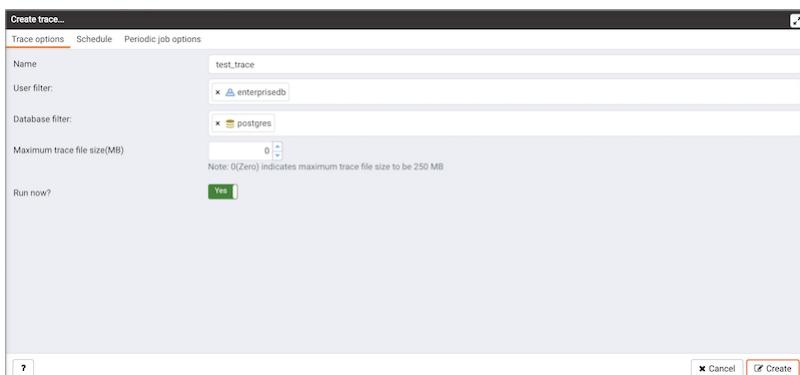
Please note: if you have connected to the PEM server with the PEM client before configuring SQL Profiler, you must disconnect and reconnect with the server to enable SQL Profiler functionality. For more detailed information about installing and configuring the SQL Profiler plugin, please refer to the [PEM Installation Guides](#).

Creating a New SQL Trace

SQL Profiler captures and displays a specific SQL workload for analysis in a SQL trace. You can start and review captured SQL traces immediately, or save captured traces for review at a later time. You can use SQL Profiler to create and store up to 15 named traces; use menu options to create and manage traces.

Creating a Trace

You can use the `Create trace...` dialog to define a SQL Trace for any database on which SQL Profiler has been installed and configured. To access the dialog, highlight the name of the database in the PEM client tree control; navigate through the Management menu to the SQL Profiler pull-aside menu, and select Create trace....



Use the fields on the `Trace options` tab to specify details about the new trace:

- Provide a name for the trace in the Name field.
- Click in the `User filter` field to specify the roles whose queries will be included the trace; optionally, check the

box next to Select All to include queries from all roles.

- Click in the **Database filter** field to specify which databases to trace; optionally, check the box next to Select All to include queries against all databases.
- Specify a **trace size in the Maximum Trace File Size** field; SQL Profiler will terminate the trace when it reaches approximately the size specified.
- Specify Yes in the **Run Now** field to start the trace when you select the Create button; select No to enable fields on the Schedule tab.



Use the fields on the **Schedule** tab to specify scheduling details for the new trace:

- Use the **Start time** field to specify the starting time for the trace.
- Use the **End time** field to specify the ending time for the trace.
- Specify Yes in the **Repeat?** field to indicate that the trace should be repeated every day at the times specified; select No to enable fields on the Periodic job options tab.



Fields on the **Periodic job options** tab specify scheduling details about a recurring trace. Use fields in the Days section to specify the days on which the job will execute:

- Click in the **Week days** field to select the days of the week on which the trace will execute.
- Click in the **Month days** field to select the days of the month on which the trace will execute.
- Click in the **Months** field to select the months in which the trace will execute.

Use fields in the **Times** section to specify a time schedule for the trace execution:

- Click in the **Hours** field to select the hours at which the trace will execute.
- Click in the **Minutes** field to select the hours at which the trace will execute.

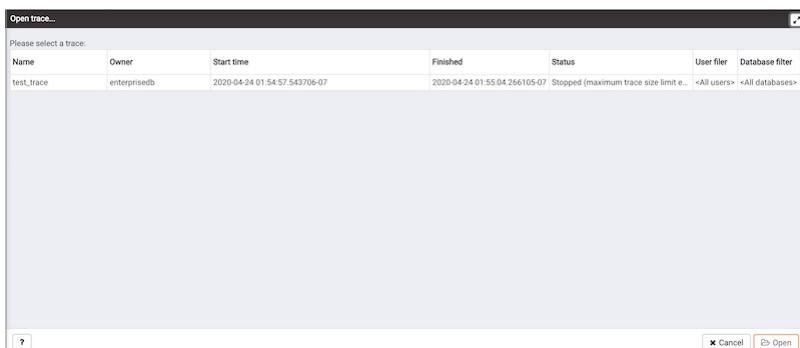
When you've completed the **Create trace...** dialog, click **Create** to start the newly defined trace or to schedule the trace for a later time.

The screenshot shows the SQL Profiler interface for a trace named 'test_trace'. The main pane displays a table of events with the following columns: Start Time, Duration (ms), Query, Rows Affected, User, Database, PID, File System Read, File System Write, and Page Fault. The 'Query' column contains various PostgreSQL commands like SELECT, INSERT, and pg_stat_activity. The 'User' column shows 'agent1' and 'enterprisedb'. The 'Database' column shows 'postgres'. The 'PID' column lists process IDs such as 91666, 33489, 30543, and 33485. The 'File System Read' and 'File System Write' columns are mostly zero. The 'Page Fault' column shows values like 6, 0, 0, and 0.

If you elect to execute the trace immediately, the trace results will display in the PEM client.

Opening an Existing Trace

To view a previous trace, highlight the name of the profiled database in the PEM client tree control; navigate through the **Management** menu to the SQL Profiler pull-aside menu, and select **Open trace....**. You can also use the **SQL Profiler toolbar** menu to open a trace; select the **Open trace...** option. The Open trace... dialog opens.



Highlight an entry in the trace list and click Open to open the selected trace. The selected trace opens in the SQL Profiler tab.

Filtering a Trace

A filter is a named set of (one or more) rules, each of which can hide events from the trace view. When you apply a filter to a trace, the hidden events are not removed from the trace, but are merely excluded from the display.

Click the Filter icon to open the **Trace Filter** dialog and create a rule (or set of rules) that define a filter. Each rule will screen the events within the current trace based on the identity of the role that invoked the event, or the query type invoked during the event.

To open an existing filter, select the **Open** button; to define a new filter, click the **Add (+)** icon to add a row to the table displayed on the General tab and provide rule details:

- Use the **Type** drop-down listbox to specify the trace field that the filter rule will apply to.
- Use the **Condition** drop-down listbox to specify the type of operator that SQL Profiler will apply to the Value when it filters the trace:

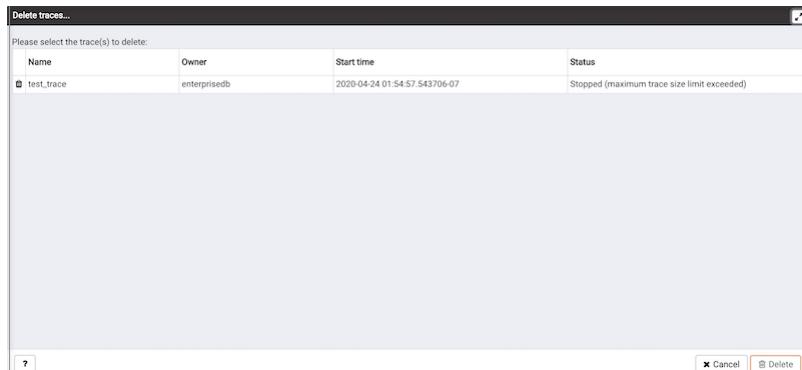
- Select **Matches** to filter events that contain the specified Value.
 - Select **Does not match** to filter events that do not contain the specified Value.
 - Select **Is equal** to to filter events that contain an exact match to the string specified in the Value field.
 - Select **Is not equal** to to filter events that do not contain an exact match to the string specified in the Value field.
 - Select **Starts with** to filter events that begin with the string specified in the Value field.
 - Select **Does not start with** to filter events that do not begin with the string specified in the Value field.
 - Select **Less than** to filter events that have a numeric value less than the number specified in the Value field.
 - Select **Greater than** to filter events that have a numeric value greater than the number specified in the Value field.
 - Select **Less than or equal to** to filter events that have a numeric value less than or equal to the number specified in the Value field.
 - Select **Greater than or equal to** to filter events that have a numeric value greater than or equal to the number specified in the Value field.
- Use the **Value** field to specify the string, number or regular expression that SQL Profiler will search for.

When you've finished defining a rule, click the Add (+) icon to add another rule to the filter. To delete a rule from a filter, highlight the rule and click the Delete icon.

Click the **Save** button to save the filter definition to a file without applying the filter; to apply the filter, click **OK**. Select **Cancel** to exit the dialog and discard any changes to the filter.

Deleting a Trace

To delete a trace, highlight the name of the profiled database in the PEM client tree control; navigate through the **Management** menu to the SQL Profiler pull-aside menu, and select **Delete trace(s)...**. You can also use the SQL Profiler toolbar menu to delete a trace; select the **Delete trace(s)...** option. The **Delete traces** dialog opens.



Click the icon to the left of a trace name to mark one or more traces for deletion and click **Delete**. The PEM client will acknowledge that the selected traces have been deleted.

Viewing Scheduled Traces

To view a list of scheduled traces, highlight the name of the profiled database in the PEM client tree control; navigate through the **Management** menu to the SQL Profiler pull-aside menu, and select **Scheduled traces...**. You can also use the SQL Profiler toolbar menu to the list; select the **Scheduled traces...** option.



The **Scheduled traces...** dialog displays a list of the traces that are awaiting execution. Click the edit button to the left of a trace name to access detailed information about the trace:

- The **Status** field lists the status of the current trace.
- The **Enabled?** switch displays Yes if the trace is enabled; No if it is disabled.
- The **Name** field displays the name of the trace.
- The **Agent** field displays the name of the agent responsible for executing the trace.
- The **Last run** field displays the date and time of the last execution of the trace.
- The **Next run** field displays the date and time of the next scheduled trace.
- The **Created** field displays the date and time that the trace was defined.

Using the Index Advisor

Index Advisor is distributed with Advanced Server 9.0 and above. Index Advisor works with SQL Profiler by examining collected SQL statements and making indexing recommendations for any underlying tables to improve SQL response time. The Index Advisor works on all DML (INSERT, UPDATE, DELETE) and SELECT statements that are invoked by a superuser.

Diagnostic output from the Index Advisor includes:

- Forecasted performance benefits from any recommended indexes
- The predicted size of any recommended indexes
- DDL statements you can use to create the recommended indexes

Before using Index Advisor, you must:

1. Modify the **postgresql.conf** file on each Advanced Server host, adding the `index_advisor` library to the `shared_preload_libraries` parameter.
2. Install the **Index Advisor contrib** module. To install the module, use the `psql` client or PEM Query Tool to connect to the database, and invoke the following command:

```
\i <complete_path>/share/contrib/index_advisor.sql
```

3. Restart the server for your changes to take effect.

Index Advisor can make indexing recommendations based on trace data captured by SQL Profiler. Simply highlight one or more queries in the **SQL Profiler Trace Data** pane, and click the **Index Advisor** toolbar button (or select Index Advisor from the View menu). For detailed usage information about Index Advisor, please see the EDB Postgres Advanced Server Guide.

Please note: Index Advisor cannot analyze statements invoked by a non-superuser. If you attempt to analyze statements

invoked by a non-superuser, the server log will include the following error:

```
ERROR: access to library "index_advisor" is not allowed
```

!!! Note It is recommended that you disable the index advisor while using the pg_dump functionality.

For more information about configuring and using Index Advisor, please see the EDB Postgres Advanced Server Guide, available from EDB at:

<https://www.enterprisedb.com/docs>

10.3 Uninstalling SQL Profiler

The process of uninstalling SQL Profiler is platform-specific.

Uninstalling SQL Profiler from Windows Hosts

If you are using SQL Profiler on a Windows host, Windows will lock any files that have been executed or loaded into memory. To release any locked files, you must stop the Postgres server before performing an uninstall.

On Windows, you can use the **Services** dialog to control the service. To open the **Services** dialog, navigate through the **Control Panel** to the **System and Security menu**. Select **Administrative Tools**, and then double-click the **Services** icon. When the **Services** dialog opens, highlight the service name in the list, and use the option provided on the dialog to **Stop** the service.

After stopping the Postgres Server:

Delete the existing SQL Profiler query set on each node by invoking the **uninstall-sql-profiler.sql** script.

By default, the script resides in the **share\contrib** directory under your Advanced Server or PostgreSQL installation.

Uninstalling SQL Profiler from CentOS or RHEL Hosts

To uninstall a SQL Profiler installation that resides on a Linux host:

Delete the existing SQL Profiler query set on each node by invoking the **uninstall-sql-profiler.sql** script.

By default, if you are using Advanced Server on a Linux host, the script resides in the **share/contrib** directory under the Advanced Server installation.

If you are using a PostgreSQL installation on a Linux host, the script resides in the **share/contrib** directory under the PostgreSQL installation.

11 PEM Upgrade and Migration

This guide provides detailed information about upgrading the Postgres Enterprise Manager (PEM) Components:

- Upgrading a PEM Installation - This section provides information about upgrading your PEM Server, PEM Agent and SQL Profiler from one major version to another (i.e. from 6.0 to 7.14).
- Upgrading the Backend Database - This section provides detailed information about upgrading the backend database, while maintaining the same version of the PEM Server.
- Moving a PEM Server -This section provides detailed information about moving the PEM Server from one host to another host.
- Troubleshooting -This section provides detailed information about troubleshooting the errors that you may encounter during PEM upgrade.

This document uses the term **Postgres** to mean either the PostgreSQL or the Advanced Server database.

11.1 Upgrading a PEM Installation

The process of upgrading a PEM installation is platform-specific. You can update a PEM Agent or Server on a Windows host by using the PEM graphical installer available for Windows. Prior to PEM 7.8 release, PEM Agent or Server could be installed on Linux either by using the graphical installer or by using the RPMs. From PEM version 7.8 onwards, PEM graphical installers for Linux are discontinued. To update a PEM Agent or Server on a Linux host from any lower version to PEM 7.9 or higher versions, you must use native packages.

Links to PEM and SQL Profiler installers and RPMs are available at the [EDB website](#).

11.1.1 Upgrading a PEM Installation on Windows Host

To upgrade PEM component software on Windows hosts, simply invoke a newer version of the PEM component installers in the following order:

1. Invoke the PEM agent installer on each monitored node **except** the PEM Server host.
2. Invoke the PEM Server installer; this installer will upgrade **both** the PEM Server and the PEM Agent that resides on the PEM Server host.

During an installation, the component installer will automatically detect an existing installation, and perform an upgrade. After upgrading the PEM Agent and Server, you can upgrade SQL Profiler if required; this step is platform-specific.

The following sections will walk you through the upgrade process on a Windows host, step-by-step.

Upgrading a PEM Agent on a Windows Host

To upgrade a system that is currently monitored by a PEM agent to a more-recent PEM agent, simply download and

invoke a newer version of the PEM Agent installer on the system that the agent is monitoring.

You can invoke the installer by right-clicking on the downloaded installer's icon, and selecting **Run as Administrator**. The **PEM Agent Setup Wizard** opens, welcoming you.



Read and accept the **License Agreement** before clicking **Next** to continue.



The setup wizard will automatically detect an existing agent, and upgrade the installed version. Click **Next** to continue.



The **pemAgent service account** dialog may prompt you for the password of the account under which the PEM Agent service runs.



If prompted, provide the password, and click **Next** to continue.

When the **Ready to Install** dialog informs you that the installation is about to begin, click **Next** to continue. It will upgrade your PEM Agent to the latest version.



The setup wizard displays progress bars to inform you of each component that is being installed.

The **PEM Agent Setup Wizard** will inform you when the installation completes. Click **Finish** to exit the wizard and close the window.



After the installation completes, a window pops-up to restart the machine. Click **Yes** to restart the machine and the PEM Agent.



Upgrading the PEM Server on a Windows Host

The PEM Server installer facilitates upgrading directly between major versions of the PEM Server; you can upgrade directly from version 5.0 to version 7.16 without first upgrading to version 6.0.

You can invoke the installer by right-clicking on the downloaded installer's icon, and selecting **Run as Administrator**.



The **PEM Server Setup Wizard** welcomes you, as shown in the image. Click **Next** to continue to the **License Agreement**.

The **PEM Server setup wizard** will prompt you to accept the **License Agreement**. After reviewing the license agreement, check the radio button next to **I accept the agreement**, and click **Next** to continue to the **Existing installation dialog**.



The wizard will check the PEM Server host for an existing PEM Server installation; if the wizard locates an installation, it will perform an upgrade. Click **Next** to continue.



Before upgrading the PEM Server, the wizard will confirm that the requirements of the new PEM Server are present. If any supporting components are missing, or are a version that will not support the new PEM installation, the PEM installation wizard will inform you that it must upgrade the dependencies, and will invoke the required installers.



When the installation wizards complete the dependency upgrades, then a window pops-up asking whether you want to restart the machine or not.



Click on **No** to continue the upgrade process.

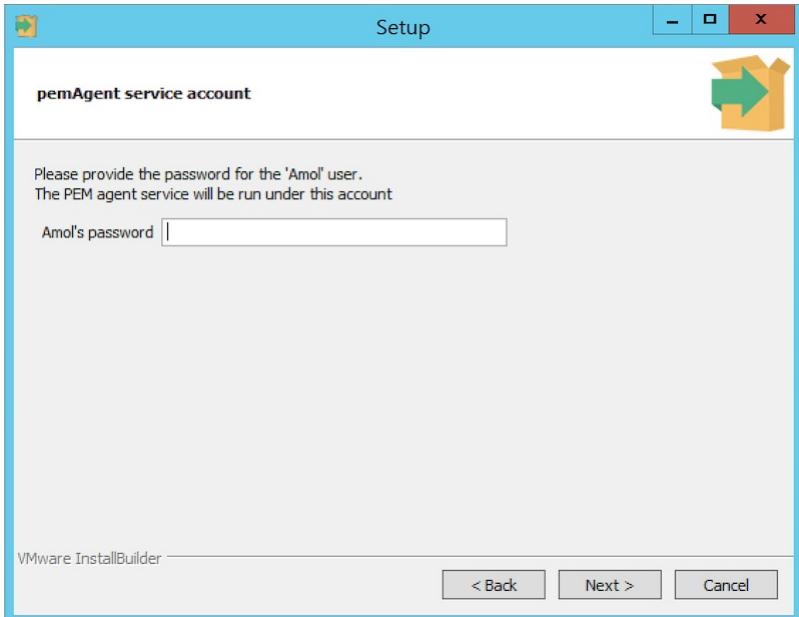
The wizard then opens the **Database Server Installation Details** dialog, prompting you for connection credentials for the database superuser of the PEM backend database. Provide:

- The name of the database superuser in the **User** field.
- The password associated with the database superuser in the **Password** field.

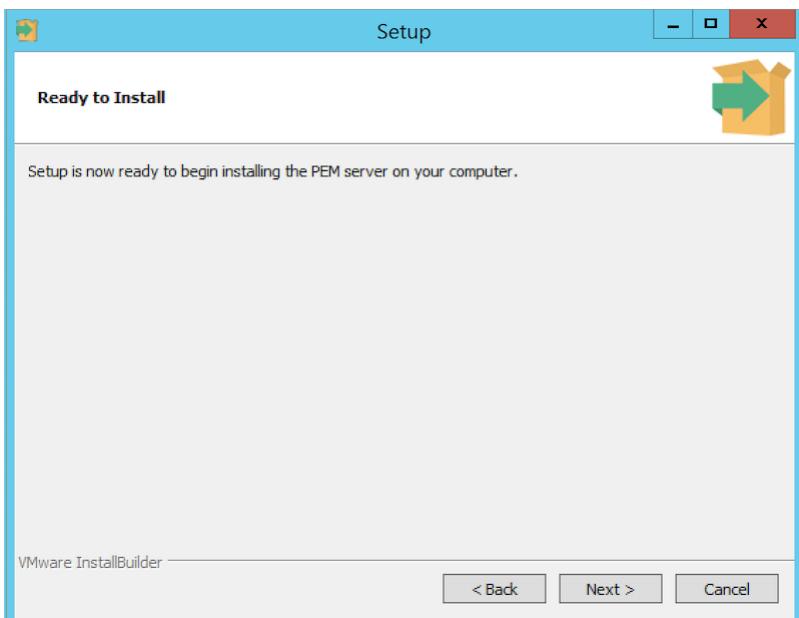
Click **Next** to continue.



The **pemAgent service account** dialog may prompt you for the password of the account under which the PEM agent service runs.



If prompted, provide the password, and click **Next** to continue.



The **Ready to Install** dialog will inform you that the setup wizard is ready to perform the installation. Click **Next** to start the installation.

During the installation, progress bars will keep you informed of the progress of the update.



After upgrading the PEM Server (and the agent that resides on the same host as the PEM server) and configuring the web service, the PEM setup wizard notifies you of the port on which the service is listening. Use this port number when connecting to the PEM Server with the PEM client.



Click **OK** to close the **Info** popup. The PEM server setup wizard informs you that the installation is complete.



If the window pops-up asking to restart the machine, then click on **Yes** to restart the machine and hence the **httpd** service.

If you have installed the PEM backend database server and PEM-HTTPD on different hosts, then you must run the PEM Server installer twice – once on each host. Extract the language pack installer, and install it on the host of PEM-HTTPD before invoking the PEM installer. Include the following keywords when invoking the installer to extract the language pack:

```
--extract-languagepack <path>
```

Where **<path>** specifies an existing path for extracting the language pack installer.

!!! Note By default EDB Language Pack is installed in **C:\edb\languagepack\v1**.

If you are upgrading the PEM Server via StackBuilder Plus then you might face the error shown below; after displaying the error, PEM will say that installation is completed. Please note that the installation is not done and you will need to do the installation by invoking the installer file from the location where it is downloaded.



After upgrading the PEM Server, you may wish to upgrade the backend database to a more recent version; for information about upgrading the backend database, see [Upgrading the Backend Postgres Database](#).

Upgrading SQL Profiler on a Windows Host

If you are using SQL Profiler on a Windows host, Windows will lock any files that have been executed or loaded into memory. To release any locked files, you must stop the Postgres server before performing an upgrade.

On Windows, you can use the `Services` dialog to control the service. To open the `Services` dialog, navigate through the `Control Panel` to the `System and Security menu`. Select `Administrative Tools`, and then double-click the `Services` icon. When the `Services` dialog opens, highlight the service name in the list, and use the option provided on the dialog to `Stop` the service.

After stopping the Postgres Server:

1. Delete the existing SQL Profiler query set on each node by invoking the `uninstall-sql-profiler.sql` script. By default, on a Windows host the script resides in the `share\contrib` directory under your Advanced Server or PostgreSQL installation.

You can use the following server-specific commands.

For PostgreSQL:

```
psql -f C:\Program Files\PostgreSQL\<x>\share\contrib\uninstall-sql-profiler.sql
-d postgres -U postgres
```

Where, `<x>` is the version of PostgreSQL and `-d` specifies the name of the maintenance database.

For Advanced Server:

```
psql -f C:\Program Files\edb\as<x>\share\contrib\uninstall-sql-profiler.sql -d
```

```
edb -U enterpriseedb
```

Where `x` is the version of Advanced Server and `-d` specifies the name of the maintenance database.

2. Invoke the new SQL Profiler installer on each node you wish to profile; run the installer as `Administrator`.

For PostgreSQL:

```
sqlprofiler-pg-<x>-<y>-windows-x64.exe
```

Where, `x` is the version of the PostgreSQL and `y` is the version of SQL Profiler. For example: `sqlprofiler-pg-12-7.14.0-1-windows-x64.exe`.

For Advanced Server:

```
sqlprofiler-edb-as<x>-<y>-windows-x64.exe
```

Where `x` is the version of Advanced Server and `y` is the version of SQL Profiler. For example: `sqlprofiler-edb-as12-7.14.0-1-windows-x64.exe`.

The SQL Profiler installer will detect the existing `SQL Profiler` installation and upgrade it with the latest version of SQL Profiler.

3. Run the `sql-profiler.sql` script file in the maintenance database.

For PostgreSQL:

```
psql -f C:\Program Files\PostgreSQL\<x>\share\contrib\sql-profiler.sql -d
postgres -U postgres
```

Where `x` is the version of PostgreSQL and `-d` specifies the name of the maintenance database.

For Advanced Server:

```
psql -f C:\Program Files\edb\as<x>\share\contrib\sql-profiler.sql -d edb -U
enterpriseedb
```

Where, `x` is the version of Advanced Server and `-d` specifies the name of the maintenance database.

4. Then, restart the Postgres Server to resume profiling the node from a PEM client.

After updating the PEM components, you are ready to update the backend database.

11.1.2 Upgrading a PEM Native Package Installation on a Linux Host

To upgrade PEM component software on Linux hosts, simply install a newer version of the PEM component native packages in the following order:

1. Invoke the PEM agent native package installer on each monitored node `except` the PEM server host.
2. Invoke the PEM server native package installer; it will upgrade `both` the PEM server and the PEM agent that resides on the PEM server host.

During an installation, the component installation will automatically detect an existing installation, and perform an upgrade. After upgrading the PEM agent and server, you can upgrade SQL Profiler if required; this step is platform-specific.

!!! Note If you have already configured or are planning to configure any shell/batch script run by a Linux agent that is upgraded from any lower version to 7.11 or later version, the user for the `batch_script_user` parameter must be specified in the `agent.cfg` file. It is strongly recommended that a non-root user is used to run the scripts. Using the root user may result in compromising the data security and operating system security. However, if you want to restore the `pemagent` to its original settings using a root user to run the scripts, then the `batch_script_user` parameter value must be set to `root`.

The following sections will walk you through the upgrade process on a Linux host, step-by-step.

Prerequisites to Upgrade a PEM Installation on Linux Host

PEM is dependent on third-party components from the vendor repository, including the python3, libboost, openssl, snmp++, libcurl, etc. To ensure these components are up to date, you should update your operating system using following platform-specific commands. Minimum version require for openssl is 1.0.2k. If you are using a version of PostgreSQL or Advanced Server older than version 10, before the upgrade you must install the `libs` package for version 10 or above on the system where the PEM server is installed. Use the following platform-specific commands to install the `libs` version 10 or above on your host:

Prerequisites to Upgrade a PEM Installation on a CentOS or RHEL Host

To upgrade packages on a CentOS or RHEL 7.x host

```
yum upgrade
```

To upgrade packages on a CentOS or RHEL 8.x host

```
dnf upgrade
```

To upgrade Advanced Server libs:

```
yum install edb-as<X>-server-libs
```

To upgrade PostgreSQL libs:

```
yum install postgresql<X>-libs
```

Where `<X>` is the PostgreSQL or Advanced Server version whose `libs` package you want to install.

Prerequisites to Upgrade a PEM Installation on a Debian or Ubuntu Host

To upgrade packages on a Debian or Ubuntu host

```
apt-get update
```

To upgrade Advanced Server libs:

```
apt-get install edb-as<X>-server-libs
```

To upgrade PostgreSQL libs:

```
apt-get install postgresql<X>-libs
```

Where <X> is the PostgreSQL or Advanced Server version whose **libs** package you want to install.

Prerequisites to Upgrade a PEM Installation on a SLES Host

To upgrade packages on a SLES host

```
zypper update
```

To upgrade Advanced Server libs:

```
zypper install edb-as<x>-server-libs
```

To upgrade PostgreSQL libs:

```
zypper install postgresql<x>-libs
```

Where <X> is the PostgreSQL or Advanced Server version whose **libs** package you want to install.

Upgrading a PEM Agent Native Package Installation

You can use native packages to upgrade existing PEM Agents that were initially installed using native packages. The upgrade process does not update the PEM agent configuration file. After installing the new agent, you must manually copy the configuration file of the existing agent to the new installation location.

Upgrading the PEM Agent on a CentOS or RHEL Host

For CentOS or RHEL 7.x or 8.x:

To upgrade a PEM agent, use the following command:

```
yum upgrade edb-pem-agent
```

For CentOS or RHEL 8.x, you can also use the following command:

```
dnf upgrade edb-pem-agent
```

Upgrading a PEM Agent on a Debian or Ubuntu Host

To upgrade a PEM Agent, use the following command:

```
apt-get upgrade edb-pem-agent
```

Upgrading a PEM Agent on a SLES Host

To upgrade a PEM Agent, use the following command:

```
zypper update edb-pem-agent
```

Upgrading a PEM Server Native Package Installation

If you initially used native packages to install your PEM server, you can use native packages to upgrade your PEM server. The commands to upgrade are platform-specific; please refer to your platform's section below.

If you wish to upgrade a PEM server that is installed on a machine in an isolated network, you need to create a PEM repository on that machine before you upgrade the PEM server. For more information about creating a PEM repository on an isolated network, see [Creating a PEM Repository on an Isolated Network](#).

Upgrading a PEM Server on a CentOS or RHEL Host

To use an RPM package to upgrade an existing RPM installation you must:

You can use the `yum` package manager to upgrade the installed version of the PEM server on CentOS or RHEL 7.x or 8.x:

```
yum upgrade edb-pem
```

You can also use the `dnf` command on CentOS or RHEL 8.x:

```
dnf upgrade edb-pem
```

!!! Note If you are doing a fresh installation of the PEM Server on CentOS or RHEL 7.x host, the installer will install the `edb-python3-mod_wsgi` package along with the installation as the package is a requirement of the operating system. If you are *upgrading* the PEM Server on CentOS or RHEL 7.x host, the `mod_wsgi` package will be replaced by the `edb-python3-mod_wsgi` package to meet the requirements of the operating system.

After upgrading the PEM Server using `yum` or `dnf`, you must configure the PEM Server. For more detailed information see [Configuring the PEM Server on Linux Platforms](#).

Upgrading the PEM Server on a Debian or Ubuntu Host

You can use the `apt-get` package manager to upgrade the installed version of the PEM Server on supported versions of Debian or Ubuntu:

```
apt-get upgrade edb-pem
```

After upgrading the PEM Server with `apt-get`, you need to configure the PEM Server. For more detailed information see [Configuring the PEM Server on Linux Platforms](#).

Upgrading the PEM Server on a SLES Host

You can use the `zypper` package manager to upgrade the installed version of the PEM Server on supported versions of

SLES Host:

```
zypper update edb-pem
```

After upgrading the PEM Server using `zypper`, you need to configure the PEM Server. For more detailed information see [Configuring the PEM Server on Linux platforms](#).

Upgrading a SQL Profiler Native Package Installation

To upgrade a SQL Profiler installation that resides on a Linux host:

1. Delete the existing SQL Profiler query set on each node by invoking the `uninstall-sql-profiler.sql` script. By default, on a Linux host the script resides in the `share/contrib` directory under your Advanced Server or PostgreSQL installation.

You can use the following server-specific command:

For PostgreSQL:

```
/usr/pgsql-<x>/bin/psql -f /usr/pgsql-<x>/share/contrib/uninstall-sql-
profiler.sql -d postgres -U postgres
```

Where, `x` is the version of PostgreSQL and `-d` specifies the name of the maintenance database.

For Advanced Server:

```
/usr/edb/as-<x>/bin/psql -f /usr/edb/as-<x>/share/contrib/uninstall-sql-
profiler.sql -d edb -U enterprisedb
```

Where, `x` is the version of Advanced Server and `-d` specifies the name of the maintenance database.

2. Invoke the new SQL Profiler installer on each node you wish to profile.

For PostgreSQL:

```
yum upgrade postgresql-<x>-sqlprofiler
```

Where, `x` is the version of the PostgreSQL.

For Advanced Server:

```
yum upgrade edb-as-<x>-server-sqlprofiler
```

Where `x` is the version of Advanced Server.

The installer will detect the existing `SQL Profiler` installation and upgrade with the latest version of SQL Profiler.

Please see the following example of upgrading SQL Profiler for PostgreSQL:

```
[root@localhost Downloads]# yum install postgresql12-sqlprofiler
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirrors.piconecs.webwerks.in
 * epel: hkg.mirror.rackspace.com
 * extras: mirrors.piconecs.webwerks.in
 * updates: mirrors.piconecs.webwerks.in
Resolving Dependencies
--> Running transaction check
--> Package postgresql12-sqlprofiler.x86_64 0:7.12.0-1.rhel7 will be updated
--> Package postgresql12-sqlprofiler.x86_64 0:7.14.0-1.rhel7 will be an update
--> Finished Dependency Resolution

Dependencies Resolved

=====
| Package           | Arch | Version      | Repository | Size |
|:-----|:-----|:-----|:-----|:-----|
| Updating:        |       |             |            |       |
| postgresql12-sqlprofiler | x86_64 | 7.14.0-1.rhel7 | edb       | 88 k |
| Transaction Summary |
|:-----|:-----|:-----|:-----|:-----|
| Upgrade 1 Package |
| Total download size: 88 k
| Is this ok [y/d/N]: y
| Downloading packages:
| No Presto metadata available for edb
| postgresql12-sqlprofiler-7.14.0-1.rhel7.x86_64.rpm
| Running transaction check
| Running transaction test
| Transaction test succeeded
| Running transaction
|   Updating : postgresql12-sqlprofiler-7.14.0-1.rhel7.x86_64 1/2
|   Cleanup   : postgresql12-sqlprofiler-7.12.0-1.rhel7.x86_64 2/2
|   Verifying  : postgresql12-sqlprofiler-7.14.0-1.rhel7.x86_64 1/2
|   Verifying  : postgresql12-sqlprofiler-7.12.0-1.rhel7.x86_64 2/2
| Updated:
|   postgresql12-sqlprofiler.x86_64 0:7.14.0-1.rhel7
| Complete!
```

- Then, run the `sql-profiler.sql` script file in the maintenance database.

For PostgreSQL:

```
/usr/pgsql-<x>/bin/psql -f /usr/pgsql-<x>/share/contrib/sql-profiler.sql -d
postgres -U postgres
```

Where, `x` is the version of PostgreSQL and `-d` specifies the name of the maintenance database.

For Advanced Server:

```
/usr/edb/as<x>/bin/psql -f /usr/edb/as<x>/share/contrib/sql-profiler.sql -d edb -
-U enterpriseedb
```

Where, `x` is the version of Advanced Server and `-d` specifies the name of the maintenance database.

- Restart the PostgreSQL/Advanced Server to resume profiling the node from the PEM Client.

After updating the PEM components, you are ready to update the backend database.

11.1.3 Creating a PEM Repository on an Isolated Network

You can create a local repository to act as a host for the PEM RPM packages if the server on which you wish to upgrade PEM cannot directly access the EDB repository. Please note that this is a high-level overview of the steps required; you may need to modify the process for your individual network. To create and use a local repository, you must:

- Use the following commands on a system with Internet access to download the dependencies for PEM:

```
yum install yum-plugin-downloadonly
mkdir /<pem_dir>
```

```
yum install --downloadonly --downloaddir=/<pem_dir>/ edb-pem
mkdir /<epel_dir>
yum install --downloadonly --downloaddir=/<epel_dir>/ epel-release*
```

Where `<pem_dir>` and `<epel_dir>` are the local directories that you create for downloading the RPMs.

2. Copy the directories `/<pem_dir>` and `/<epel_dir>` to the machine that is in the isolated network.

3. Create the repositories:

```
createrepo /<pem_dir>
createrepo /<epel_dir>
```

4. Create a repository configuration file called `/etc/yum.repos.d/pem.repo` with connection information that specifies:

```
[pemrepo]
name=PEM Repository
baseurl=file:///pem7/
enabled=1
gpgcheck=0
```

5. Create a repository configuration file called `/etc/yum.repos.d/epel.repo` with connection information that specifies:

```
[epelrepo]
name=epel Repository
baseurl=file:///pem7/
enabled=1
gpgcheck=0
```

6. After specifying the location and connection information for your local repository, you can use yum commands to install or upgrade PEM server:

To install PEM server:

```
yum install edb-pem
```

To upgrade PEM server:

```
yum upgrade edb-pem
```

For more information about creating a local yum repository, visit: <https://wiki.centos.org/HowTos/CreateLocalRepos>

11.1.4 Upgrading a Graphical PEM Installation on a Linux Host

To upgrade PEM component software on Linux hosts, simply install a newer version of the PEM component native

packages in the following order:

1. Invoke the PEM Agent native package installation on each monitored node **except** the PEM server host.
2. Invoke the PEM server native package installation; it will upgrade **both** the PEM server and the PEM Agent that resides on the PEM server host.

During an installation, the component installation will automatically detect an existing installation, and perform an upgrade. After upgrading the PEM Agent and server, you can upgrade SQL Profiler if required; this step is platform-specific.

The following section walks you through the process of upgrading a PEM installation that was performed via a graphical installer on a Linux host.

Prerequisites to Upgrade a PEM Installation on a Linux Host

If you are using a version of Postgres or Advanced Server released older than version 10, before the upgrade you must install the **libs** package for version 10 or above on the system where the PEM server is installed.

For Advanced Server:

```
yum install edb-as<X>-server-libs
```

For PostgreSQL:

```
yum install postgresql<X>-libs
```

Where **<X>** is the Postgres or Advanced Server version whose libs package you want to install.

Upgrading a PEM Agent (Graphical Installation) on a CentOS or RHEL Host

The default installation location for the PEM Agent when installed by the graphical installer is **/opt/edb/pem**. In the example that follows, substitute your server installation location for **<PEM_installation_path>**.

1. Use the version specific command to stop the **pemagent** service.

On a CentOS or RHEL 7.x Or 8.x host:

```
systemctl stop pemagent
```

2. Install the supporting **epel-release** package on the host by running any one of the following commands:

```
yum install epel-release
```

```
yum -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-<X>.noarch.rpm
```

Where **X** is the OS version.

!!! Note You may need to enable the **[extras]** repository definition in the **CentOS-Base.repo** file (located in **/etc/yum.repos.d**).

If you are a Red Hat Network user,

- You must also enable the `rhel-<x>-server-optional-rpms` repository to use EPEL packages, where ** specifies the version of RHEL on the host. You can make the repository accessible by enabling the `RHEL optional subchannel` for `RHN-Classic`. If you have a certificate-based subscription, then you must also enable `rhel-<x>-server-eus-optional-rpms` repository to use EPEL packages or please see the `Red Hat Subscription Management Guide` for the required repository.
- You must also enable the `rhel-<x>-server-extras-rpms` repository, where `x` specifies the version of the RHEL on the host.

3. Install and configure the `edb.repo` file:

- a. You must also have credentials that allow access to the EDB repository. To request credentials, visit:

[EDB Repository Access Steps.](#)

- b. Create a repository configuration file; assume superuser privileges, and invoke the following command:

```
yum -y install https://yum.enterprisedb.com/edb-repo-rpms/edb-repo-latest.noarch.rpm
```

The repository configuration file is named `edb.repo`. The file resides in `/etc/yum.repos.d`.

- c. After creating the `edb.repo` file, to ensure that the value of the enabled parameter is 1, and the `USERNAME` and `PASSWORD` placeholders in the `baseurl` specification are replaced with the name and password of a registered EDB user, run the following command:

```
sed -i "s@<username>:<password>@USERNAME:PASSWORD@" /etc/yum.repos.d/edb.repo
```

If you want to install PEM Agent on a machine that is in isolated network, you must first create PEM repository on that machine. For more information about creating PEM repository on an isolated network, see [Creating a PEM repository in an Isolated Network](#).

4. The `yum makecache` command downloads the metadata for the currently enabled repositories; when the command completes, check the available packages to confirm that the list includes the latest PEM Agent:

On a CentOS or RHEL 7.x or 8.x:

```
yum makecache  
yum list edb-pem-agent
```

On a CentOS or RHEL 8.x:

```
dnf makecache  
dnf list edb-pem-agent
```

5. Install the PEM Agent RPM; when the installation completes, you can use the `yum info` command to confirm installation information for the PEM Agent:

On a CentOS or RHEL 7.x or 8.x:

```
yum install edb-pem-agent  
yum info edb-pem-agent
```

On a CentOS or RHEL 8.x:

```
dnf install edb-pem-agent
```

6. After installation, copy the PEM Agent configuration file (`agent.cfg`) from the previous location to the location required by the RPM installer:

```
cp /PEM_installation_path/agent/etc/agent.cfg /usr/edb/pem/agent/etc/agent.cfg
```

!!! Note The Package Management and Streaming Replication features are deprecated from PEM 7.16 version. Please remove these parameters from agent.cfg file while copying the file if existing - allow_package_management and allow_streaming_replication, to avoid any warning messages in the worker log file after PEM Agent service restart.

- a. Open the agent configuration file located at:

```
/usr/edb/pem/agent/etc/agent.cfg
```

Then set the value of the `ca_file` parameter:

```
ca_file=/usr/libexec/libcurl-pem/share/certs/ca-bundle.crt
```

- b. Take a backup of the service file and agent certificates:

On CentOS or RHEL 7.x or 8.x, use the following command to back up the service file:

```
cp /usr/lib/systemd/system/pemagent.service  
/usr/lib/systemd/system/pemagent.service_bkp
```

Then, copy the agent certificates; in the following commands, `<agent_id>` should specify the agent identifier (for example, agent2 or agent3):

```
mv /root/.pem/<agent_id>.key /root/.pem/<agent_id>.key.bkp  
mv /root/.pem/<agent_id>.crt /root/.pem/<agent_id>.crt.bkp
```

- c. Uninstall the PEM agent using bitrock uninstaller:

```
/PEM_installation_path/agent/uninstall-pemagent
```

- d. Use version specific commands to restore the service file backup and agent certificates to original location. For example:

On a RHEL or CentOS 7.x or 8.x host:

```
cp /usr/lib/systemd/system/pemagent.service_bkp  
/usr/lib/systemd/system/pemagent.service
```

Then, move the agent certificate files; in the following commands, `<agent_id>` should specify the agent identifier (for example, agent2 or agent3):

```
mv /root/.pem/<agent_id>.key.bkp /root/.pem/<agent_id>.key  
mv /root/.pem/<agent_id>.crt.bkp /root/.pem/<agent_id>.crt
```

7. Enable the `pemagent` service, and start `pemagent` and `httpd`.

On a RHEL or CentOS 7.x or 8.x host, use the commands:

```
systemctl enable pemagent
systemctl start pemagent
```

At this point, the PEM agent should be up and running; you can use the PEM web interface to check the agent version and status.

Upgrading a PEM Server that was Installed with a Graphical Installer

The default installation location for the PEM server when installed by the graphical installer is `/opt/edb/pem`. In the example that follows, substitute your server installation location for `<PEM_installation_path>`.

Upgrading a PEM Server that was Installed with a Graphical Installer on a CentOS or RHEL Host

1. Logout from PEM.
2. Stop the PEMHTTPD service on PEM server. In case PEM server and web server are on two different systems, run the command on the web server:

On a CentOS or RHEL 7.x Or 8.x host:

```
systemctl stop PEMHTTPD
systemctl stop pemagent
```

3. Install the supporting `epel-release` package on the host by running any one of the following commands:

```
yum install epel-release
```

```
yum -y install https://dl.fedoraproject.org/pub/epel/epel-release-latest-
<X>.noarch.rpm
```

Where `<X>` is the OS version.

!!! Note You may need to enable the `[extras]` repository definition in the `CentOS-Base.repo` file (located in `/etc/yum.repos.d`).

If you are a Red Hat Network user,

- You must also enable the `rhel-<x>-server-optional-rpms` repository to use EPEL packages, where `x` specifies the version of RHEL on the host. You can make the repository accessible by enabling the `RHEL optional subchannel` for `RHN-Classic`. If you have a certificate-based subscription, then you must also enable `rhel-<x>-server-eus-optional-rpms` repository to use EPEL packages or please see the `Red Hat Subscription Management Guide` for the required repository.
- You must also enable the `rhel-<x>-server-extras-rpms` repository, where `x` specifies the version of the RHEL on the host.

4. Install and configure `edb.repo` file:

- a. You must also have credentials that allow access to the EDB repository. To request credentials, visit:

[EDB Repository Access Steps](#).

- b. Create a repository configuration file; assume superuser privileges, and invoke the following command:

```
yum -y install https://yum.enterprisedb.com/edb-repo-rpms/edb-repo-latest.noarch.rpm
```

The repository configuration file is named `edb.repo`. The file resides in `/etc/yum.repos.d`.

- c. After creating the `edb.repo` file, to ensure that the value of the enabled parameter is `1`, and the `username` and `password` placeholders in the `baseurl` specification are replaced with the name and password of a registered EDB user, run the following command:

```
sed -i "s@<username>:<password>@USERNAME:PASSWORD@" /etc/yum.repos.d/edb.repo
```

If you want to install PEM Agent on a machine that is in isolated network, you must first create PEM repository on that machine. For more information about creating PEM repository on an isolated network, see [Creating a PEM repository in an Isolated Network](#).

5. The `yum makecache` command downloads the metadata for the currently enabled repositories; when the command completes, check the available packages to confirm that the list includes the latest PEM Server:

On a CentOS or RHEL 7.x or 8.x:

```
yum makecache
```

```
yum list edb-pem
```

On a CentOS or RHEL 8.x:

```
dnf makecache
```

```
dnf list edb-pem
```

6. Install the PEM Server RPM; when the installation completes, you can use the `yum info` command to confirm installation information for the PEM Server:

On a CentOS or RHEL 7.x or 8.x:

```
yum install edb-pem
```

```
yum info edb-pem
```

On a CentOS or RHEL 8.x:

```
dnf install edb-pem
```

7. After installation, copy the `agent.cfg` file from the current location (the location required by the graphical installer) to the location required by the RPM package:

```
cp /<PEM_installation_path>/agent/etc/agent.cfg /usr/edb/pem/agent/etc/agent.cfg
```

8. Open the agent configuration file located at:

```
/usr/edb/pem/agent/etc/agent.cfg
```

Then, set the value of the `ca_file` parameter:

```
ca_file=/usr/libexec/libcurl-pem/share/certs/ca-bundle.crt
```

9. Copy the `pem.db` file (and other required files) to the RPM installation location and change the file ownership. In case PEM server and web server are on two different systems, run the below commands on the web server:

```
cp -r /<PEM_installation_path>/server/share/pemhome/.pem/* /var/lib/pemhome/.pem/
chown -R pem:pem /var/lib/pemhome/.pem/
```

10. Change the home directory in the `passwd` file from the location identified by the graphical installer to the RPM location. In case PEM server and web server are on two different systems, run the commands on PEM server as well as web server:

```
usermod -m -d /var/lib/pemhome pem
```

```
cat /etc/passwd | grep pem
```

11. Take a backup of the PEM service file and agent certificates:

```
cp /usr/lib/systemd/system/pemagent.service
/usr/lib/systemd/system/pemagent.service_bkp
```

```
mv /root/.pem/agent1.key /root/.pem/agent1.key.bkp
```

```
mv /root/.pem/agent1.crt /root/.pem/agent1.crt.bkp
```

12. Uninstall the PEM server using the graphical uninstaller from PEM server and web server machines:

```
/<PEM_installation_path> /server/uninstall-pemserver
```

13. Restore the service file backup and agent certificates to original location on PEM server.

```
cp /usr/lib/systemd/system/pemagent.service_bkp
/usr/lib/systemd/system/pemagent.service
```

```
mv /root/.pem/agent1.crt.bkp /root/.pem/agent1.crt
```

```
mv /root/.pem/agent1.key.bkp /root/.pem/agent1.key
```

14. Install the `sslutils` through RPM or Installer (In case of Windows, you may want to stop the database server before installation as installer may not be able to overwrite the file otherwise it would require system restart)

Restart the database server (Restarting the database server is required as shared object of the extension has been upgraded)

Refer section [Install `sslutils`](#)

Use "ALTER EXTENSION" statement to upgrade the version

```
ALTER EXTENSION sslutils UPDATE [ TO new_version ]
```

15. Execute the PEM RPM configuration script on PEM server and web server; when prompted, provide the backend database details: the script should run without generating errors:

```
/usr/edb/pem/bin/configure-pem-server.sh
```

16. Enable the `pemagent` service on PEM server and web server. Start the `pemagent` and `httpd` services on the web server.

On a CentOS or RHEL 7.x or 8.x host:

```
systemctl enable pemagent
systemctl start pemagent
systemctl start httpd
```

17. Launch the PEM web interface. Check the server and agent to confirm the PEM version, server status, and schema version. At this point, everything should be up and running.

18. You can now uninstall the PEMHTTD service from your web server, as it is no longer in use.

```
/opt/edb/pem/httpd/uninstall-pemhttpd
```

Upgrading a SQL Profiler Installation with a Graphical Installer

SQL Profiler graphical installers are available for PostgreSQL and Advanced Server versions prior to 11.x (i.e. 10.x, 9.6).

To upgrade a SQL Profiler installation on a Linux host:

1. Delete the existing SQL Profiler query set on each node by invoking the `uninstall-sql-profiler.sql` script. By default, if you are using PostgreSQL on a Linux host that was installed with a graphical installer, the script resides in the `share/postgresql/contrib` directory under the PostgreSQL installation; for Advanced Server, the script resides in the `share/contrib` directory under the Advanced Server installation. You can use the following server-specific commands.

For PostgreSQL:

```
/opt/PostgreSQL/<x>/bin/psql -f
/opt/PostgreSQL/<x>/share/postgresql/contrib/uninstall-sql-profiler.sql -d
postgres -U postgres
```

Where, `x` is the version of PostgreSQL and `-d` specifies the name of the maintenance database.

For Advanced Server:

```
/opt/edb/as<x>/bin/psql -f /opt/edb/as<x>/share/contrib/uninstall-sql-
profiler.sql -d edb -U enterprisedb
```

Where, `x` is the version of Advanced Server and `-d` specifies the name of the maintenance database.

2. Then, invoke the new SQL Profiler installer on each node you wish to profile. Run the installer as Administrator:

For a PostgreSQL host:

```
sqlprofiler-pg-<x>-<y>-linux.exe
```

Where `x` is the version of the PostgreSQL and `y` is the version of SQL Profiler. For example: `sqlprofiler-pg-12-7.14.0-1-linux.exe`.

For Advanced Server:

```
sqlprofiler-edb-as<x>-<y>-linux.exe
```

Where `x` is the version of Advanced Server and `y` is the version of SQL Profiler. For example: `sqlprofiler-edb-as12-7.14.0-1-linux.exe`.

The SQL Profiler installer will detect the existing `SQL Profiler` installation and upgrade it with the latest version of SQL Profiler.

3. Run the `sql-profiler.sql` script file in the maintenance database.

For PostgreSQL:

```
/opt/PostgreSQL/<x>/bin/psql -f /opt/PostgreSQL/<x>/share/postgresql/contrib/sql-profiler.sql -d postgres -U postgres
```

Where, `x` is the version of PostgreSQL and `-d` specifies the name of the maintenance database.

For Advanced Server:

```
/opt/edb/as<x>/bin/psql -f /opt/edb/as<x>/share/contrib/sql-profiler.sql -d edb -U enterprisedb
```

Where, `x` is the version of Advanced Server and `-d` specifies the name of the maintenance database.

4. Restart the PostgreSQL/Advanced Server to resume profiling the node from the PEM Client.

After updating the PEM components, you are ready to update the backend database.

11.1.5 Configuring a PEM Server on a Linux Host

After upgrading the PEM Server you can use the following command to configure the PEM Server:

```
/usr/edb/pem/bin/configure-pem-server.sh
```

When invoking the configuration script, you can include command line options to specify configuration properties; the script will prompt you for values that you omit on the command line. The accepted options are:

Option Description

<code>-acp</code>	Defines PEM Agent certificate path. The default is <code>/root/.pem</code> .
<code>-ci</code>	CIDR formatted network address range that Agents will connect to the server from, to be added to the server's <code>pg_hba.conf</code> file. For example, <code>192.168.1.0/24</code> . The default is <code>0.0.0.0/0</code> .
<code>-dbi</code>	The directory for the database server installation. For example, <code>/usr/edb/as11</code> for Advanced Server or <code>/usr/pgsql-11</code> for PostgreSQL.
<code>-ds</code>	The unit file name of the PEM database server. For Advanced Server, the default file name is <code>edb-as-11</code> ; for PostgreSQL, it is <code>postgresql-11</code> .
<code>-ho</code>	The host address of the PEM database server.
<code>-p</code>	The port number of the PEM database server.

Option Description

-ps	The service name of the pemagent; the default value is pemagent.
-sp	The superuser password of the PEM database server. This value is required.
-su	The superuser name of the PEM database server.
-t	The installation type: Specify 1 if the configuration is for web services and backend database, 2 if you are configuring web services, or 3 if you are configuring the backend database. If you specify 3, please note that the database must reside on the local host.
-un	To unregister the PEM server.
-h	Displays help.

If you do not provide configuration properties on the command line, you will be prompted for values by the script. To view script-related help, use the command:

```
/usr/edb/pem/bin/configure-pem-server.sh --help
```

After executing the PEM server configuration file, use your version-specific service control command to restart the `httpd` service.

For detailed information about using an RPM package to install or configure the PEM server or PEM Agent, please see the [PEM Linux Installation Guide](#).

11.2 Upgrading the PEM Backend Postgres Database

If you are updating both PEM components and the PEM backend database, you should perform PEM component updates (the server and Agent) before updating the backend database. For more information about updating PEM component software, see [Upgrading a PEM Installation](#).

!!! Note From PEM 8.0 onwards, the PostgreSQL or EPAS version 11 or higher are only supported as backend database server. Hence if your backend database server is lower than version 11 then first you need to upgrade your backend database server and then upgrade the PEM components.

The update process described in this section uses the `pg_upgrade` utility to migrate from one version of the backend server to a more recent version. `pg_upgrade` facilitates migration between any supported version of Postgres, and any subsequent release of Postgres that is supported on the same platform.

!!! Note If the source PEM Server is lower than the 7.16 version, then you need to replace the following functions before you run `pg_upgrade`:

- The `abstime`, `reltime`, and `tinterval` datatypes are deprecated from Postgres version 12 or later, hence to replace those datatypes with `timestamptz` data type use below command:

```
```text
DO
$$
DECLARE
 rec record;
 cnt integer;
```

```

BEGIN
 -- Check for the deprecated type in our user info probe
 SELECT count(*) INTO cnt
 FROM pem.probe_column
 WHERE sql_data_type = 'abstime' AND internal_name = 'valuntil';
 IF cnt = 0 THEN
 RETURN;
 END IF;
 ALTER TABLE pemdata.user_info
 ALTER COLUMN valuntil SET DATA TYPE timestamptz;
 ALTER TABLE pemhistory.user_info
 ALTER COLUMN valuntil SET DATA TYPE timestamptz;
 -- Now update the pem.probe_column itself
 UPDATE pem.probe_column
 SET sql_data_type = 'timestamptz'
 WHERE sql_data_type = 'abstime' AND internal_name = 'valuntil';
END;
$$ LANGUAGE 'plpgsql';
```

```

- Replace the below function to avoid any alert errors:

```

```text
CREATE OR REPLACE FUNCTION pem.check_alert_params_array_size(
template_id pem.alert_template.id%type, params text[]
)
RETURNS bool AS $FUNC$
DECLARE
 res bool := TRUE;
BEGIN
 /*
 * During restoring the pem database, it does not maintain the order while
 * inserting data in the table, and uses the sort table based on the
 * names.
 * Hence - we need to check the foreign key constraint is present before
 * validating these values.
 */
 IF EXISTS(
 SELECT 1 FROM information_schema.table_constraints
 WHERE constraint_name='alert_template_id_fkey' AND
 table_name='alert' AND table_schema='pem'
) THEN
 /*
 * Need to use the IS TRUE construct outside the main query, because
 * otherwise if there's no template by that ID then the query would return
 * 0 rows and the result of the function would be undefined and CHECK
 * constraint would succeed.
 * Probably this is being over-cautious, because pem.alert.template_id
 * references pem.alert_template.id. But the SQL standard (probably) does
 * not define the order in which the CHECK or the FOREIGN KEY constraints
 * should be validated; in case CHECK is validated first, we want it to
 * fail.
 */
 EXECUTE SQL
 SELECT (

```

```

 SELECT pem.check_array_size_equal(t.param_names, $2)
 FROM pem.alert_template AS t
 WHERE id = $1
) IS TRUE
SQL INTO res USING template_id, params;
END IF;
RETURN res;
END
$FUNC$ LANGUAGE 'plpgsql';
```

```

`pg_upgrade` supports a transfer of data between servers of the same type. For example, you can use `pg_upgrade` to move data from a PostgreSQL 9.6 backend database to a PostgreSQL 11 backend database, but not to an Advanced Server 11 backend database. If you wish to migrate to a different type of backend database (i.e from a PostgreSQL server to Advanced Server), see [Moving the Postgres Enterprise Manager Server](#).

You can find more information about using `pg_upgrade` at:

<http://www.postgresql.org/docs/current/static/pgupgrade.html>

1. Download and invoke the updated installer; installers for PostgreSQL and Advanced Server are available through the EDB website:

<https://www.enterprisedb.com/software-downloads-postgres>

After downloading the installer for the server version to which you will be upgrading, invoke the installer on the host of the PEM server. Follow the onscreen instructions of the installation wizard to configure and install the Postgres server.

You can optionally use a custom-built PostgreSQL server as a host of the PEM backend database. Note that if you are upgrading from a PostgreSQL backend database listening on port `5432`, the new server must be configured to listen on a different port.

2. Configure SSL utilities on the new server. The new backend database must be running the same version of `sslutils` that the current backend database is running; you can download the SSL Utils package that is used in EDB installers at:

<https://www.enterprisedb.com/downloads/modified-gpl-source-code>

You are *not* required to manually add the `sslutils` extension when using the Advanced Server as the new backend database. The process of configuring `sslutils` is platform-specific.

On Linux

- On an Advanced Server backend database, the `sslutils` extension is installed by default.
- If you are using a PostgreSQL as PEM backend database, ensure you have access to the PostgreSQL community repository, and use the command:

```
yum install sslutils_<X>
```

Where `<X>` is the server version.

- If you are using a EDB one-click installer of PostgreSQL as PEM backend database

```
yum install gcc openssl-devel
```

Set the value of PATH so it can locate the pg_config program

```
export PATH=$PATH:/opt/postgres_inst_dir/<X>/bin/
```

Move into the sslutils folder, and enter

```
make USE_PGXS=1
make USE_PGXS=1 install
```

Use psql to create the sslutils extension

```
CREATE EXTENSION sslutils
```

Please note that Debian 10 and Ubuntu 20 has increased the requirements to accept the certificates due to security reason. If a user wants to install the PEM Agent on any of the machines, they must upgrade `sslutils` to 1.3 where 4096 bit RSA key and sha256 signature algorithm support has added. If the user does not upgrade `sslutils` to 1.3, then PEM Agent may fail to connect to the PEM backend database server, and it might log the error `ca_md too weak`.

On Windows

- `sslutils` must be compiled on the new backend database with the same compiler that was used to compile `sslutils` on the original backend database. If you are moving to a Postgres database that was installed using a PostgreSQL one-click installer (from EDB) or an Advanced Server installer, use Visual Studio to build `sslutils`. If you are upgrading to PostgreSQL 9.6 or later, use Visual Studio 2010.

For detailed information about building a specific version of Postgres on Windows, please consult the core documentation for that version. Core documentation is available at the PostgreSQL project website at:

<https://www.postgresql.org/docs/current/install-windows.html>

- While specific details of the process will vary by platform and compiler, the basic steps on each platform are the same. The example that follows demonstrates compiling OpenSSL support for PostgreSQL on a 32-bit Windows system.

Before compiling the OpenSSL extension, you must locate and install OpenSSL for your version of Windows. Before invoking the OpenSSL installer you may be required to download and install a pre-requisite redistributable (such as `vcredist_x86.exe`).

After installing OpenSSL, download and unpack the `sslutils` utility package available at:

<https://www.enterprisedb.com/downloads/modified-gpl-source-code>

- Copy the unpacked `sslutils` folder to the Postgres installation directory (i.e. `C:\ProgramFiles\PostgreSQL\<x.x>`)
- Open the Visual Studio command line, and navigate into the `sslutils` directory. Use the following commands to build `sslutils`:

```
SET USE_PGXS=1
SET GETTEXTPATH=\ <path_to_gettext>
SET OPENSSLSPATH=\ <path_to_openssl>
SET PGPATH=\ <path_to_pg_installation_dir>
```

```
SET ARCH=x86

msbuild sslutils.proj /p:Configuration=Release
```

Where:

- `path_to_gettext` specifies the location of the `GETTEXT` library and header files.
- `path_to_openssl` specifies the location of the `openssl` library and header files.
- `path_to_pg_installation_dir` specifies the location of the Postgres installation.
- For example, the following set of commands builds OpenSSL support into the PostgreSQL 11 server:

```
SET USE_PGXS=1

SET OPENSSLPATH=C:\OpenSSL-Win32

SET GETTEXTPATH="C:\Program Files\PostgreSQL\11"

SET PGPATH="C:\Program Files\PostgreSQL\11"

SET ARCH=x86

msbuild sslutils.proj /p:Configuration=Release
```

- When the build completes, the `sslutils` directory will contain the following files:

```
sslutils--1.3.sql
sslutils--unpackaged--1.3.sql
sslutils--pemagent.sql.in
sslutils.dll
```

- Copy the compiled `sslutils` files to the appropriate directory for your installation; for example:

```
COPY sslutils*.sql "%PGPATH%\share\extension\"
COPY sslutils.dll "%PGPATH%\lib\"
COPY sslutils.control "%PGPATH%\share\extension\"
```

3. Stop the services of both the old backend database and the new backend database.

On RHEL or CentOS 7.x or 8.x, open a command line and assume the identity of a superuser. Enter the command:

```
systemctl <service_name> stop
```

Where `<service_name>` specifies the name of the Postgres service.

On Windows, you can use the `Services` dialog to control the service. To open the `Services` dialog, navigate through the `Control Panel` to the `System and Security` menu. Select `Administrative Tools`, and then double-click the `Services` icon. When the `Services` dialog opens, highlight the service name in the list,

and use the option provided on the dialog to stop the service.

4. Use the `pg_upgrade` utility to perform an in-place transfer of existing data between the old backend database and the new backend database. If your server is configured to enforce `md5` authentication, you may need to add an entry to the `.pgpass` file that specifies the connection properties (and password) for the database superuser, or modify the `pg_hba.conf` file to allow trust connections before invoking `pg_upgrade`. For more information about creating an entry in the `.pgpass` file, please see the PostgreSQL core documentation, available at:

<http://www.postgresql.org/docs/current/static/libpq-pgpass.html>

During the upgrade process, `pg_upgrade` will write a series of log files. The cluster owner must invoke `pg_upgrade` from a directory in which they have write privileges. If the upgrade completes successfully, `pg_upgrade` will remove the log files when the upgrade completes. To instruct `pg_upgrade` to not delete the upgrade log files, include the `--retain` keyword when invoking `pg_upgrade`.

To invoke `pg_upgrade`, assume the identity of the cluster owner, navigate into a directory in which the cluster owner has write privileges, and execute the command:

```
<path_to_pg_upgrade> pg_upgrade
-d <old_data_dir_path>
-D <new_data_dir_path>
-b <old_bin_dir_path> -B <new_bin_dir_path>
-p <old_port> -P <new_port>
-u <user_name>
```

Where:

- `path_to_pg_upgrade` specifies the location of the `pg_upgrade` utility. By default, `pg_upgrade` is installed in the `bin` directory under your Postgres directory.
- `old_data_dir_path` specifies the complete path to the data directory of the old backend database.
- `new_data_dir_path` specifies the complete path to the data directory of the new backend database.
- `old_bin_dir_path` specifies the complete path to the bin directory of the old backend database.
- `new_bin_dir_path` specifies the complete path to the bin directory of the new backend database.
- `old_port` specifies the port on which the old server is listening.
- `new_port` specifies the port on which the new server is listening.
- `user_name` specifies the name of the cluster owner.

For example, the following command:

```
C:\>"C:\Program Files\PostgreSQL\11\bin\pg_upgrade.exe"
-d "C:\Program Files\PostgreSQL\9.6\data"
-D "C:\Program Files\PostgreSQL\11\data"
-b "C:\Program Files\PostgreSQL\9.6\bin"
-B "C:\Program Files\PostgreSQL\11\bin"
-p 5432 -P 5433
```

```
-U postgres
```

Instructs `pg_upgrade` to migrate the PEM database from PostgreSQL 9.6 to PostgreSQL 11 on a Windows system (if the backend databases are installed in their default locations).

Once invoked, `pg_upgrade` will perform consistency checks before moving the data to the new backend database. When the upgrade is finished, `pg_upgrade` will notify you that the upgrade is complete.

For detailed information about using `pg_upgrade` options, or troubleshooting the upgrade process, please see:

<http://www.postgresql.org/docs/current/static/pgupgrade.html>

5. Copy the following certificate files from the `data` directory of the old backend database to the `data` directory of the new backend database:

`ca_certificate.crt`

`ca_key.key`

`root.crt`

`root.crl`

`server.key`

`server.crt`

Once in place on the target server, the files should have the (platform-specific) permissions described below:

Permissions and Ownership on Linux

| File Name | Owner | Permissions |
|---------------------------------|-----------------------|-------------------------|
| <code>ca_certificate.crt</code> | <code>postgres</code> | <code>-rw-----</code> |
| <code>ca_key.key</code> | <code>postgres</code> | <code>-rw-----</code> |
| <code>root.crt</code> | <code>postgres</code> | <code>-rw-----</code> |
| <code>root.crl</code> | <code>postgres</code> | <code>-rw-----</code> |
| <code>server.key</code> | <code>postgres</code> | <code>-rw-----</code> |
| <code>server.crt</code> | <code>postgres</code> | <code>-rw-r--r--</code> |

On Linux, the certificate files must be owned by `postgres`. You can use the following command at the command line to modify the ownership of the files:

```
chown postgres <file_name>
```

Where `file_name` specifies the name of the certificate file.

The `server.crt` file may only be modified by the owner of the file, but may be read by any user. You can use the following command to set the file permissions for the `server.crt` file:

```
chmod 644 server.crt
```

The other certificate files may only be modified or read by the owner of the file. You can use the following command to set the file permissions:

```
chmod 600 <file_name>
```

Where `file_name` specifies the name of the file.

Permissions and Ownership on Windows

On Windows, the certificate files moved from the source host must be owned by the service account that performed the PEM server and backend database installation on the target host. If you invoked the PEM server and Postgres installer using the `Run as Administrator` option (selected from the context menu of the installer), the owner of the certificate files will be `Administrators`.

To review and modify file permissions on Windows, right-click on the file name, and select `Properties`.



**** The Security tab ****

Navigate to the `Security` tab and highlight a `Group or user name` to view the assigned permissions. Select `Edit` or `Advanced` to access dialogs that allow you to modify the permissions associated with the selected user.

6. The `postgresql.conf` file contains parameter settings that specify server behavior. You will need to modify the `postgresql.conf` file on the new server to match the configuration specified in the `postgresql.conf` file of the old server.

By default, the `postgresql.conf` file is located:

- For Postgres version lower than 10 on Linux, in `/opt/PostgreSQL/<X>/data`
- For Postgres version 10 or higher when installed with graphical installers on Linux, in `/opt/PostgreSQL/<X>/data`
- For Postgres version 10 or higher when installed with an RPM on Linux, in `/usr/pgsql/<X>/data`
- For any Postgres version on Windows, in `C:\Program Files\PostgreSQL\<X>\data`

Where, `<X>` is the version of Postgres on your system.

Use your choice of editor to update the `postgresql.conf` file of the new server. Modify the following parameters:

- The `port` parameter to listen on the port monitored by your original backend database (typically, `5432`).
- The `ssl` parameter should be set to `on`.

You must also ensure that the following parameters are enabled. If the parameters are commented out, remove the pound sign from in front of each `postgresql.conf` file entry:

- `ssl_cert_file = 'server.crt' # (change requires restart)`
- `ssl_key_file = 'server.key' # (change requires restart)`
- `ssl_ca_file = 'root.crt' # (change requires restart)`

- `ssl_crl_file = 'root.crl'`

Your installation may have other parameter settings that require modification to ensure that the new backend database behaves in a manner comparable to the old backend database. Review the `postgresql.conf` files carefully to ensure that the configuration of the new server matches the configuration of the old server.

7. The `pg_hba.conf` file contains parameter settings that specify how the server will enforce host-based authentication. When you install the PEM server, the installer modifies the `pg_hba.conf` file, adding entries to the top of the file:

```
# Adding entries for PEM Agens and admins to connect to PEM server

hostssl pem +pem_user 192.168.2.0/24 md5

hostssl pem +pem_agent 192.168.2.0/24 cert

# Adding entries (localhost) for PEM Agens and admins to connect to PEM server

hostssl pem +pem_user 127.0.0.1/32 md5

hostssl postgres +pem_user 127.0.0.1/32 md5

hostssl pem +pem_user 127.0.0.1/32 md5

hostssl pem +pem_agent 127.0.0.1/32 cert
```

By default, the `pg_hba.conf` file is located at the following location:

- For Postgres version lower than 10 on Linux, in `/opt/PostgreSQL/<X>/data`
- For Postgres version 10 or higher when installed with graphical installers on Linux, in `/Opt/PostgreSQL/<X>/data`
- For Postgres version 10 or higher when installed with RPMs on Linux, in `/var/lib/pgsql/<X>/data`
- For Advanced Server version 10 or higher when installed with RPMs on Linux, in `/var/lib/edb/as<X>/data`
- For any Postgres version on Windows, in `C:\Program Files\PostgreSQL\<X>\data`

Where, `<X>` is the version of Postgres on your system.

Using your editor of choice, copy the entries from the `pg_hba.conf` file of the old server to the `pg_hba.conf` file for the new server.

8. Restart the service of the new backend database.

On RHEL or CentOS 7.x or 8.x, open a command line and assume the identity of a superuser. Enter the command:

```
systemctl stop <service_name>
```

Where `service_name` is the name of the backend database server.

If you are using Windows, you can use the `Services` dialog to control the service. To open the `Services` dialog, navigate through the `Control Panel` to the `System and Security` menu. Select `Administrative Tools`, and then double-click the `Services` icon. When the `Services` dialog opens, highlight the service name in the list, and use the option provided on the dialog to start the service.

11.3 Moving the Postgres Enterprise Manager Server

The steps in this section describe how to move a PEM server from one host machine to a new host machine. The PEM server on the new host (the target) must be installed with the same version of the PEM server installer as the original host (the source). Please note that if you do not use the same installer version, you may encounter a schema-mismatch error.

The backend database of the target server (either PostgreSQL or Advanced Server) may be of the same type and version, or a different type and version than the backend database of the source PEM server. A PEM server that resides on a PostgreSQL host can be migrated to an Advanced Server host, or vice versa.

Before starting the server migration, you should ensure that the firewalls between the source host, the target host, and the host of any PEM Agent will allow connections between the services.

1. Prepare the Target Host

Invoke the installer for the PEM server on the target host. Please note that you must use the same version of the PEM server installer that you used when installing the source PEM server.

The backend database of the target server may be a different version or type than the backend database of the source. If the new PEM server does *not* reside on the same type of backend database as the original server, you must ensure that the same version of the `sslutils` extension is installed on the new server host. The version of `sslutils` that is distributed with the PEM installers is freely available for download from the EDB website at:

<https://www.enterprisedb.com/downloads/modified-gpl-source-code>

For information about installing the PEM server or the `sslutils` extension, please refer to the [PEM Installation Guides](#), available at:

<https://www.enterprisedb.com/docs/p/edb-postgres-enterprise-manager>

2. Drop Existing Schemas from the New PEM Server

The migration process re-creates the `pem`, `pemdata`, and `pemhistory` schemas from the source PEM server on the target PEM server. In preparation for the move, use the `psql` client to delete these schemas from the `pem` database on the target host. You can open the `psql` client at the command line, or by selecting [SQL Shell \(psql\)](#) from the [Postgres Enterprise Manager](#) menu.

When the `psql` client opens, connect to the `pem` backend database as the database superuser. After connecting to the `pem` database on the target host, use the following commands to drop the schemas:

```
DROP SCHEMA pem CASCADE;
DROP SCHEMA pemdata CASCADE;
DROP SCHEMA pemhistory CASCADE;
```

When dropping the schemas, you must include the `CASCADE` keyword, instructing the server to delete all dependent objects. When executing the command, the `psql` client displays a list of the dependent objects; the client confirms each the schema is removed by displaying `DROP SCHEMA`.

3. Prepare the PEM Agents on the New PEM Server

Before moving the PEM server, you must identify the number of Agents that are monitored by the source PEM server, and create identities for that number of Agents (less one) on the target server. To discover the total number of [PEM](#)

Agents monitored by the PEM server, connect to the pem database on the source host with the `psql` client, and query the `pem.agent` table.

```
SELECT id FROM pem.agent WHERE active = true;
```

You must manually create the number of Agents that reside on the original PEM server, less one; the PEM server installer has already created one Agent on the target host. For example, if the source server contains three Agents, you must manually create two additional Agents. Open a `psql` session with the `pem` database on the target server, and create the required Agents. Use the command:

```
CREATE USER agent<X>;
```

Where `<X>` specifies an Agent number. Remember, `agent1` is created on the target host by the PEM Server installer.

Then, use the `GRANT` command to assign each Agent that resides on the target PEM Server `pem_agent` permissions:

```
GRANT pem_agent TO agent<X>;
```

Where `<X>` specifies an agent number.

4. Generate a Backup Script of the Source PEM Server

You can use the `pg_dump` utility to generate a script that contains the commands required to recreate the `pem` database on the target host. By default, `pg_dump` is installed in the `bin` directory under your Postgres installation. To invoke `pg_dump`, open a command line, navigate to the `bin` directory, and enter:

```
pg_dump -U <user_name> <db_name> > <file_name>
```

Where:

- `<user_name>` specifies the name of the database superuser for the PEM backend database.
- `<db_name>` specifies the name of the PEM backend database.
- `<file_name>` specifies the name of the script generated by `pg_dump`.

When prompted, provide the password associated with the user specified.

The command shown instructs `pg_dump` to generate a script that (when executed) will re-create the `pem` database. The script will be named `backup.sql`, and will be created in the `tmp` directory. `pg_dump` is connecting to the server using the credentials of the user, `postgres`.

Note that invoking the `pg_dump` utility will not interrupt current database users.

!!! Note If the source PEM Server is lower than the 7.16 version, then you need to replace the following functions before you run `pg_dump` to take backup:

– The ```abstime```, ```reltime```, and ```tinterval``` datatypes are deprecated from Postgres version 12 or later, hence to replace those datatypes with ```timestampz``` data type use below command:

```
```text
DO
$$
```

```

DECLARE
 rec record;
 cnt integer;
BEGIN
 -- Check for the deprecated type in our user info probe
 SELECT count(*) INTO cnt
 FROM pem.probe_column
 WHERE sql_data_type = 'abstime' AND internal_name = 'valuntil';
 IF cnt = 0 THEN
 RETURN;
 END IF;
 ALTER TABLE pemdata.user_info
 ALTER COLUMN valuntil SET DATA TYPE timestamptz;
 ALTER TABLE pemhistory.user_info
 ALTER COLUMN valuntil SET DATA TYPE timestamptz;
 -- Now update the pem.probe_column itself
 UPDATE pem.probe_column
 SET sql_data_type = 'timestamptz'
 WHERE sql_data_type = 'abstime' AND internal_name = 'valuntil';
END;
$$ LANGUAGE 'plpgsql';
```

```

- Replace the below function to avoid any alert errors:

```

```text
CREATE OR REPLACE FUNCTION pem.check_alert_params_array_size(
template_id pem.alert_template.id%type, params text[]
)
RETURNS bool AS $FUNC$
DECLARE
 res bool := TRUE;
BEGIN
 /*
 * During restoring the pem database, it does not maintain the order while
 * inserting data in the table, and uses the sort table based on the
 * names.
 * Hence - we need to check the foreign key constraint is present before
 * validating these values.
 */
 IF EXISTS(
 SELECT 1 FROM information_schema.table_constraints
 WHERE constraint_name='alert_template_id_fkey' AND
 table_name='alert' AND table_schema='pem'
) THEN
 /*
 * Need to use the IS TRUE construct outside the main query, because
 * otherwise if there's no template by that ID then the query would return
 * 0 rows and the result of the function would be undefined and CHECK
 * constraint would succeed.
 * Probably this is being over-cautious, because pem.alert.template_id
 * references pem.alert_template.id. But the SQL standard (probably) does
 * not define the order in which the CHECK or the FOREIGN KEY constraints
 * should be validated; in case CHECK is validated first, we want it to
 * fail.
 */

```

```

*/
EXECUTE SQL
 SELECT (
 SELECT pem.check_array_size_equal(t.param_names, $2)
 FROM pem.alert_template AS t
 WHERE id = $1
) IS TRUE
SQL INTO res USING template_id, params;
END IF;
RETURN res;
END
$FUNC$ LANGUAGE 'plpgsql';
```

```

5. Move the Backup to the Target Host

Move the script generated by the `pg_dump` utility to the target host of the PEM server.

6. Restore the Backup on the Target Host

Open a command line on the target host and navigate into the `bin` directory (under the Postgres backend database installation directory). Start `psql`, executing the script generated by the `pg_dump` utility:

```
psql -U <user_name> -d pem -f <file_name>
```

Where:

- `<user_name>` specifies the name of the database superuser. The user specified must have connection privileges for the backend database.
- `<file_name>` specifies the complete path to the backup script generated by `pg_dump`.

When prompted, provide the password associated with the database superuser.

The example shown uses the `psql` client to invoke a script named `backup.sql` to recreate the `pem` database. The script is invoked using the privileges associated with the database superuser, `postgres`.

7. Stop the Database Server on the Target Host

To stop the PEM Server on CentOS or RHEL 7.x or 8.x, use the command:

```
systemctl stop <service_name>
```

Where, `<service_name>` specifies the name of the backend database server. For a PostgreSQL backend database, the service name is `postgresql-<x>`, and for an Advanced Server backend database, the service name is `edb-as-<x>`, where `<x>` specifies the version number.

If you are using Windows, you can use the `Services` dialog to control the service. To open the `Services` dialog, navigate through the `Control Panel` to the `System and Security` menu. Select `Administrative Tools`, and then double-click the `Services` icon. When the `Services` dialog opens, highlight the service name in the list, and use the option provided on the dialog to Stop the service.

8. Copy the Certificate Files to the Target Host

You must replace the certificate files that are created when the target host is installed with the certificate files of the source host. Copy the following files from the source PEM server to the target PEM server:

- `ca_certificate.crt`
- `ca_key.key`
- `root.crt`
- `root.crl`
- `server.key`
- `server.crt`

Copy the files to the `data` directory under the Postgres installation that provides the backend database for the target cluster.

On Linux, the files reside in:

```
/var/lib/pgsql/<X>/data/
```

On Windows, the files reside in:

```
C:\Program Files\PostgreSQL\<X>\data
```

Where:

`<X>` specifies the version of PostgreSQL on your system.

The files will already exist on the target cluster; delete the existing files before performing the copy, or overwrite the existing files with the files from the source server. Once in place on the target server, the files should have the (platform-specific) permissions described in the sections that follow.

Permissions and Ownership on Linux

| File Name | Owner | Permissions |
|---------------------------------|-----------------------|-------------------------|
| <code>ca_certificate.crt</code> | <code>postgres</code> | <code>-rw-----</code> |
| <code>ca_key.key</code> | <code>postgres</code> | <code>-rw-----</code> |
| <code>root.crt</code> | <code>postgres</code> | <code>-rw-----</code> |
| <code>root.crl</code> | <code>postgres</code> | <code>-rw-----</code> |
| <code>server.key</code> | <code>postgres</code> | <code>-rw-----</code> |
| <code>server.crt</code> | <code>postgres</code> | <code>-rw-r--r--</code> |

On Linux, the certificate files must be owned by `postgres`. You can use the following command at the command line to modify the ownership of the files:

```
chown postgres <file_name>
```

Where `file_name` specifies the name of the certificate file.

The `server.crt` file may only be modified by the owner of the file, but may be read by any user. You can use the following command to set the file permissions for the `server.crt` file:

```
chmod 644 server.crt
```

The other certificate files may only be modified or read by the owner of the file. You can use the following command to set the file permissions:

```
chmod 600 <file_name>
```

Where `file_name` specifies the name of the file.

Permissions and Ownership on Windows

On Windows, the certificate files moved from the source host must be owned by the service account that performed the PEM server and backend database installation on the target host. If you invoked the PEM server and Postgres installer using the `Run as Administrator` option (selected from the context menu of the installer), the owner of the certificate files will be `Administrators`.

To review and modify file permissions on Windows, right-click on the file name, and select `Properties`.



**** The Permissions tab ****

Navigate to the `Security` tab and highlight a `Group or user name` to view the assigned permissions. Select `Edit` or `Advanced` to access dialogs that allow you to modify the permissions associated with the selected user.

9. Move the PEM Agent Certificate Files to the PEM Server Host

You must move the certificate files used by the PEM Agent of the source PEM server to the target host. This step is platform-specific.

On Linux

Copy the `agent1.key` and `agent1.crt` files from the source host to the target host. By default, on Linux, the files are installed in `/root/.pem`; copy the files to the same directory on the target host.

File ownership and permissions of the files must be set to:

| File Name | Owner | Permissions |
|------------|-------|-------------|
| agent1.key | root | -rw----- |
| agent1.crt | root | -rw-r--r-- |

If necessary, navigate to `/root/.pem`, and use the following commands to modify the permissions and ownership of the `agent1.key` file:

```
chmod 600 agent1.key
```

```
chown root agent1.key
```

Use the following commands to modify the permissions and ownership of the `agent1.crt` file:

```
chmod 644 agent1.crt
```

```
chown root agent1.crt
```

On Windows

Copy the `agent1.key` and `agent1.crt` files from the source host to the target host. On Windows, the files are located in:

```
C:\Users\<user_name>\AppData\Roaming\pem
```

Where `user_name` is the name of the user that invoked the PEM installer.

The ownership and permissions associated with the certificate files on the target machine should match the ownership and permissions of the certificate files on the source machine. If you invoked the PEM server and Postgres installer using the `Run as Administrator` option (selected from the context menu of the installer), the owner of the Agent certificate files will be `Administrators`.

To review and modify file permissions on Windows, right-click on the file name, and select `Properties`. Navigate to the `Security` tab and highlight a `Group or user name` to view the assigned permissions. Select `Edit` or `Advanced` to access dialogs that allow you to modify the permissions associated with the selected user.

10. Update the `pg_hba.conf` Files on the Target Host

Modify the `pg_hba.conf` file on the target host to allow connections from each PEM Agent. By default, the `pg_hba.conf` file is located in the data directory under your Postgres installation.

11. Start the Server on the Target Host

After modifying the `pg_hba.conf` file, you must restart the server for the changes to take effect.

To restart the database server on Linux, use the command:

```
/etc/init.d/<service_name> start
```

Where `service_name` is the name of the backend database server.

If you are using Windows, you can use the `Services` dialog to control the service. To open the `Services` dialog, navigate through the `Control Panel` to the `System and Security` menu. Select `Administrative Tools`, and then double-click the `Services` icon. When the `Services` dialog opens, highlight the service name in the list, and use the option provided on the dialog to Start the service.

12. Connecting Monitored Agents to the New PEM Server Host

To instruct existing PEM Agents to connect to the new PEM server host, you must:

- Ensure that the PEM Agent host can connect to the new PEM server host.
- Modify the registry (on each Windows host with a PEM Agent) or the Agent configuration files (on each Linux host with a PEM Agent), specifying the IP address and port of the new PEM server.
- Restart the PEM Agent's service. These steps are platform-specific:
 - [On Linux](#)
 - [On Windows](#)

If the PEM Agent Resides on Linux

Use your choice of editor to modify the `agent.cfg` file, specifying the new IP address and port number of the PEM

server in the `pem_host` and `pem_port` parameters.

By default, the `agent.cfg` file is located in:

```
/usr/edb/pem/agent/etc/agent.cfg
```



After modifying the `agent.cfg` file, you must restart the PEM Agent service; you can use the `pemagent` service script on the Linux command line to restart the service:

```
/etc/init.d/pemagent restart
```

If the PEM Agent Resides on Windows

Before modifying the Windows registry on the monitored node, confirm that the firewall on the host of the PEM Agent will allow connections to the PEM server. After confirming that the PEM Agent host can connect to the PEM server host, you can use the Windows `Registry Editor` to review and edit the `PEM_HOST` and `PEM_PORT` entries to ensure that they correctly identify the host and port used by the PEM server. To open the `Registry Editor`, enter `regedit` in the Windows `Run` dialog or in the Windows start menu search box. Navigate through the registry tree control to view or modify registry entries.

On 64-bit Windows, the PEM Agent registry entries are located:

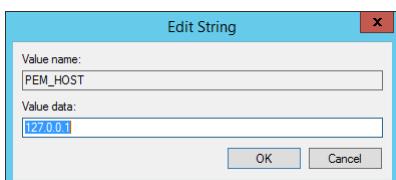
```
HKEY_LOCAL_MACHINE SOFTWARE wow6432Mode EnterpriseDB PEM Agent
```

On 32-bit Windows, the PEM Agent registry entries are located:

```
HKEY_LOCAL_MACHINE SOFTWARE EnterpriseDB PEM Agent
```

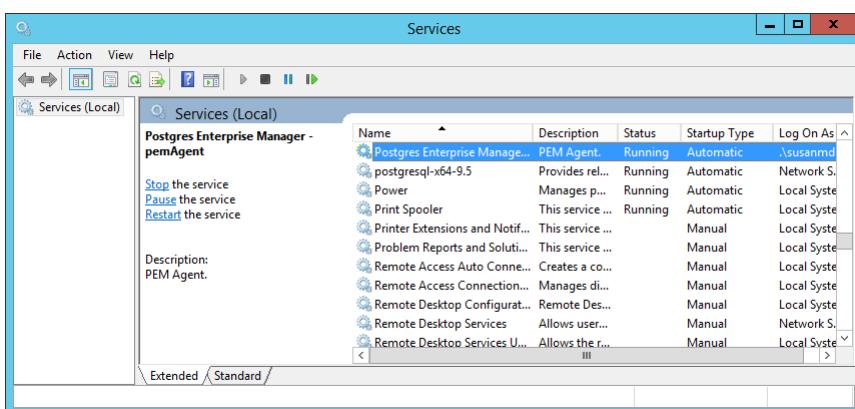


The **PEM_HOST** and **PEM_PORT** entries must specify the address and port number of the new PEM server on the target host. To modify a registry entry, right click on the entry **Name**, and select **Modify** from the context menu to open the **Edit String** dialog.



Use the **Edit String** dialog to make any changes to the value of the entry. When you're finished, click **OK** to save your changes, or **Cancel** to exit without saving.

After modifying the registry, you must restart the PEM Agent's service; you can use the **Services** dialog (accessed through the Windows **Control Panel**) to restart the **Postgres Enterprise Manager - pemagent** service .



** Restarting the PEM Agent service **

After moving the server, change the connection properties in any installed PEM clients to connect to the new host of the PEM server, Agents, and monitored servers.

11.4 Troubleshooting

Reconfiguring the PEM Server

In some situations you may need to uninstall the PEM server, install it again, and then reconfigure the server. Use the following commands in the given sequence:

1. Use the following command to remove the PEM server configuration and uninstall:

```
/usr/edb/pem/bin/configure-pem-server.sh -un
```

2. Use the following command to remove the PEM packages:

```
yum erase edb-pem-server
```

3. Use the following command to drop the `pem` database:

```
DROP DATABASE pem
```

4. Move the certificates from `/root/.pem/` to another location:

```
mv /root/.pem/* <new_location>
```

5. Move the `agent.cfg` file from `/usr/edb/pem/agent/etc/agent.cfg` to another location:

```
mv /usr/edb/pem/agent/etc/agent.cfg <new_location>
```

6. Then, use the following command to configure the PEM server again:

```
/usr/edb/pem/bin/configure-pem-server.sh
```