# pgAdmin 4 Documentation

Release 8.3

The pgAdmin Development Team

# Contents

1	Getti	ing Started	3
	1.1	Deployment	4
	1.2	Login Page	40
	1.3	Enabling two-factor authentication (2FA)	41
	1.4	User Management Dialog	42
	1.5	Change Ownership Dialog	47
	1.6	Change User Password Dialog	48
	1.7	Lock/Restore Account	49
	1.8	Enabling LDAP Authentication	50
	1.9	Enabling Kerberos Authentication	52
	1.10	Enabling OAUTH2 Authentication	54
	1.11	Enabling Webserver Authentication	56
	1.12	User Interface	58
	1.13	Menu Bar	59
	1.14	Toolbar	64
	1.15	Tabbed Browser	65
	1.16	Tree Control	73
	1.17	Preferences Dialog	76
	1.18	Keyboard Shortcuts:	100
	1.19	Search objects	103
2	E-4		105
2	2.1	rnal database for pgAdmin user settings  Use SQLite Database	105
	2.2	Use External Database	
	2.3	Use PostgreSQL Database	106
3	Conr	necting To A Server	107
	3.1	Server Group Dialog	107
	3.2	Server Dialog	
	3.3	PostgreSQL Cloud Deployment	
	3.4	Master Password	
	3.5	Connect to Server	
	3.6	Connection Error	
	3.7	Import/Export Servers	
4		- B - B - B - B - B - B - B - B - B - B	147
	4.1	Database Dialog	147

	4.2	Resource Group Dialog	3
	4.3	Login/Group Role Dialog	5
	4.4	Tablespace Dialog	2
	4.5	Role Reassign/Drop Own Dialog	7
5		aging Database Objects 17	
	5.1	Cast Dialog	
	5.2	Collation Dialog	
	5.3	Domain Dialog	
	5.4	Domain Constraints Dialog	
	5.5	Event Trigger Dialog	6
	5.6	Extension Dialog	0
	5.7	Foreign Data Wrapper Dialog	2
	5.8	Foreign Server Dialog	8
	5.9	Foreign Table Dialog	13
	5.10	FTS Configuration Dialog	
	5.11	FTS Dictionary Dialog	
	5.12	FTS Parser Dialog	
	5.13	FTS Template Dialog	
	5.14	Function Dialog	
	5.15	Language Dialog	
	5.16	Materialized View Dialog	
	5.17	Package Dialog	
	5.18		
		Procedure Dialog	
	5.19	Publication Dialog	
	5.20	Schema Dialog	
	5.21	Sequence Dialog	
	5.22	Subscription Dialog	
	5.23	Synonym Dialog	
	5.24	Trigger Function Dialog	
	5.25	Type Dialog	5
	5.26	User Mapping Dialog	
	5.27	View Dialog	7
	~		
6		ting or Modifying a Table 30	
	6.1	Check Dialog	
	6.2	Column Dialog	
	6.3	Compound Trigger Dialog	
	6.4	Exclusion Constraint Dialog	
	6.5	Foreign key Dialog	
	6.6	Index Dialog	0
	6.7	Primary key Dialog	4
	6.8	RLS Policy Dialog	7
	6.9	Rule Dialog	1
	6.10	Table Dialog	7
	6.11	Trigger Dialog	8
	6.12	Unique Constraint Dialog	
7	Mana	agement Basics 37	7
	7.1	Add Named Restore Point Dialog	7
	7.2	Change Password Dialog	8
	7.3	Grant Wizard	9
	7.4	Import/Export Data Dialog	1
	7.5	Maintenance Dialog	6

	7.6	Storage Manager         38	39
8	Backı	ip and Restore	)5
	8.1	Backup Dialog	<b>)</b> 5
	8.2	Backup Globals Dialog	
	8.3	Backup Server Dialog	
	8.4	Restore Dialog	
	0.1	Testore Blands	,,
9	Devel	oper Tools 41	
	9.1	Debugger	
	9.2	Query Tool	21
	9.3	View/Edit Data	16
	9.4	Schema Diff	52
	9.5	ERD Tool	57
	9.6	PSQL Tool	
10	Proce		
		Process Watcher	
	10.2	Watch the demo	71
4.4		, .	
11	pgAg		_
		Using pgAgent	
		Installing pgAgent	
	11.3	Creating a pgAgent Job	76
12	ngAd	min Project Contributions 48	₹7
14	12.1	Submitting Pull Requests	
		Code Overview	
		Coding Standards	
		Code Snippets	
		Code Review Notes	
	12.6	Translations	)/
13	Relea	se Notes	)9
		Version 8.4	)9
		Version 8.3	
		Version 8.2	
		Version 8.1	
		Version 8.0	
		Version 7.8	
		Version 7.7	
		Version 7.6	
			_
		Version 7.5	
		Version 7.4	
		Version 7.3	
		Version 7.2	
		Version 7.1	
		Version 7.0	
		Version 6.21	
		Version 6.20	_
	13.17	Version 6.19	29
	13.18	Version 6.18	30
	13.19	Version 6.17	30
	13.20	Version 6.16	31
	13.21	Version 6.15	33

13.22	Version	6.14		 	 	 		 		 	 		 	 	 	 	534
13.23	Version	6.13		 	 	 		 		 	 		 		 	 	535
13.24	Version	6.12		 	 	 		 		 	537						
13.25	Version	6.11		 	 	 		 		 	 		 	 	 	 	538
13.26	Version	6.10		 	 	 		 		 	 		 	 	 	 	538
13.27	Version	6.9	 	 	 	 		 		 	 	 	 		 	 	539
13.28	Version	6.8	 	 	 	 		 		 	 		 		 	 	541
	Version																
	Version																
	Version																
	Version																
	Version																
	Version																
	Version																
	Version																
	Version																
	Version																
	Version																
	Version																
	Version																
	Version																553
	Version																
	Version																
	Version			 	 	 	•	 		 	 		 		 	 	
	Version			 	 	 		 		 	 		 		 	 	
	Version				 	 		 		 	 		 		 	 	
13.48	Version	4.27			 	 		 		 	 		 		 	 	560
13.49	Version	4.26		 	 	 		 		 	 		 		 	 	561
13.50	Version	4.25		 	 	 		 		 	 		 		 	 	563
13.51	Version	4.24		 	 	 		 		 	 	 	 		 	 	564
13.52	Version	4.23		 	 	 		 		 	 		 	 	 	 	565
13.53	Version	4.22		 	 	 		 		 	 		 	 	 	 	566
13.54	Version	4.21		 	 	 		 		 	567						
13.55	Version	4.20		 	 	 		 		 	 	 	 		 	 	569
13.56	Version	4.19		 	 	 		 		 	 		 		 	 	570
13.57	Version	4.18			 	 		 		 	 		 		 	 	571
	Version																572
	Version																572
	Version																
	Version																
	Version																
	Version			 	 	 		 	 •	 	 		 		 	 	
	Version																
	Version																
	Version																
	Version																
	Version																
	Version																
10	Version																
	Version																
	Version																
	Version																
	Version																
13.75	Version	4.0	 	 	 	 		 		 	 		 	 	 	 	590

Inc	lex	613
14	Licence	611
	13.91 Version 1.0	610
	13.90 Version 1.1	609
	13.89 Version 1.2	607
	13.88 Version 1.3	606
	13.87 Version 1.4	605
	13.86 Version 1.5	604
	13.85 Version 1.6	602
	13.84 Version 2.0	600
	13.83 Version 2.1	598
	13.82 Version 3.0	595
	13.81 Version 3.1	
	13.80 Version 3.2	
	13.79 Version 3.3	
	13.78 Version 3.4	
	13.77 Version 3.5	
	13.76 Version 3.6	591



Welcome to pgAdmin 4. pgAdmin is the leading Open Source management tool for Postgres, the world's most advanced Open Source database. 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 1

2 Contents

# CHAPTER 1

# **Getting Started**

Pre-compiled and configured installation packages for pgAdmin 4 are available for a number of desktop environments; we recommend using an installer whenever possible.

In a *Server Deployment*, the pgAdmin application is deployed behind a webserver or with the WSGI interface. If you install pgAdmin in server mode, you will be prompted to provide a role name and pgAdmin password when you initially connect to pgAdmin. The first role registered with pgAdmin will be an administrative user; the administrative role can use the pgAdmin *User Management* dialog to create and manage additional pgAdmin user accounts. When a user authenticates with pgAdmin, the pgAdmin tree control displays the server definitions associated with that login role.

In a *Desktop Deployment*, the pgAdmin application is configured to use the desktop runtime environment to host the program on a supported platform. Typically, users will install a pre-built package to run pgAdmin in desktop mode, but a manual desktop deployment can be installed and though it is more difficult to setup, it may be useful for developers interested in understanding how pgAdmin works.

It is also possible to use a Container Deployment of pgAdmin, in which Server Mode is pre-configured for security.

# 1.1 Deployment

Pre-compiled and configured installation packages for pgAdmin 4 are available for a number of desktop environments; we recommend using an installer whenever possible. If you are interested in learning more about the project, or if a pgAdmin installer is not available for your environment, the pages listed below will provide detailed information about creating a custom deployment.

### 1.1.1 The config.py File

There are multiple configuration files that are read at startup by pgAdmin. These files are used for configuration options that:

- may be required to be set prior to startup of pgAdmin as they control how the application will operate.
- system administrators may wish to control across an organisation to enforce security policies.
- are so rarely or unlikely to be changed that it doesn't make sense to allow them to be changed through the user interface.

The configuration 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 pgAdmin 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 the owner of the installation 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.
- config\_system.py: This file is read after config\_local.py and is intended for system administrators to include settings that are configured system-wide from a secure location that users cannot normally modify and that is outside of the pgAdmin installation. The location for this file varies based on the platform, and only needs to be created if desired.

Platform	File Location
Linux	/etc/pgadmin/config_system.py
macOS	/Library/Preferences/pgadmin/config_system.py
Windows	$\% Common Program Files \% \ \ pgadmin \ \ \ config\_system.py$

**Note:** If the SERVER\_MODE or DATA\_DIR settings are changed in config\_distro.py, config\_local. py, or config\_system.py LOG\_FILE, SQLITE\_PATH, SESSION\_DB\_PATH, STORAGE\_DIR, KERBEROS\_CCACHE\_DIR, and AZURE\_CREDENTIAL\_CACHE\_DIR values will be set based on DATA\_DIR unless values are explicitly overridden for any of the variable in any of the above file.

The default config.py file is shown below for reference:

```
# pgAdmin 4 - PostgreSQL Tools
# Copyright (C) 2013 - 2024, The pgAdmin Development Team
# This software is released under the PostgreSQL Licence
# config.py - Core application configuration settings
import builtins
import logging
import os
import sys
from collections import OrderedDict
# We need to include the root directory in sys.path to ensure that we can
# find everything we need when running in the standalone runtime.
root = os.path.dirname(os.path.realpath(__file__))
if sys.path[0] != root:
   sys.path.insert(0, root)
# The config database connection pool size.
# Setting this to 0 will remove any limit.
CONFIG_DATABASE_CONNECTION_POOL_SIZE = 5
# The number of connections allowed to overflow beyond
# the connection pool size.
CONFIG_DATABASE_CONNECTION_MAX_OVERFLOW = 100
from pgadmin.utils import env, IS_WIN, fs_short_path
from version import APP_VERSION, APP_RELEASE, APP_REVISION, APP_SUFFIX, \
   APP_VERSION_INT
from branding import APP_NAME, APP_ICON, APP_COPYRIGHT, APP_PATH, \
   APP_WIN_PATH, APP_SHORT_NAME, APP_DEFAULT_EMAIL
# Misc stuff
# Path to the online help.
HELP_PATH = '../../docs/en_US/_build/html/'
# Languages we support in the UI
LANGUAGES = {
   'en': 'English',
   'zh': 'Chinese (Simplified)',
   'cs': 'Czech',
   'fr': 'French'
   'de': 'German',
   'id': 'Indonesian'.
   'it': 'Italian',
   'ja': 'Japanese',
   'ko': 'Korean',
```

(continues on next page)

```
'pl': 'Polish',
    'pt_BR': 'Portuguese (Brazilian)',
   'ru': 'Russian',
   'es': 'Spanish',
}
# DO NOT CHANGE UNLESS YOU KNOW WHAT YOU ARE DOING!
# List of modules to skip when dynamically loading
MODULE_BLACKLIST = ['test']
# DO NOT CHANGE UNLESS YOU KNOW WHAT YOU ARE DOING!
# List of treeview browser nodes to skip when dynamically loading
NODE_BLACKLIST = []
# Server settings
# The server mode determines whether or not we're running on a web server
# requiring user authentication, or desktop mode which uses an automatic
# default login.
# DO NOT DISABLE SERVER MODE IF RUNNING ON A WEBSERVER!!
# We only set SERVER_MODE if it's not already set. That's to allow the
# runtime to force it to False.
# NOTE: If you change the value of SERVER_MODE or DATA_DIR in an included
       config file, you may also need to redefine any values below that are
#
#
       derived from it, notably various paths such as LOG_FILE, SQLITE_PATH,
#
       SESSION_DB_PATH, STORAGE_DIR, KERBEROS_CCACHE_DIR, and
       AZURE_CREDENTIAL_CACHE_DIR
if (not hasattr(builtins, 'SERVER_MODE')) or builtins.SERVER_MODE is None:
   SERVER_MODE = True
else:
   SERVER_MODE = builtins.SERVER_MODE
# HTTP headers to search for CSRF token when it is not provided in the form.
# Default is ['X-CSRFToken', 'X-CSRF-Token']
WTF_CSRF_HEADERS = ['X-pgA-CSRFToken']
# User ID (email address) to use for the default user in desktop mode.
# The default should be fine here, as it's not exposed in the app.
DESKTOP_USER = APP_DEFAULT_EMAIL
# This option allows the user to host the application on a LAN
# Default hosting is on localhost (DEFAULT_SERVER='localhost').
# To host pgAdmin4 over LAN set DEFAULT_SERVER='0.0.0.0' (or a specific
# adaptor address.
# NOTE: This is NOT recommended for production use, only for debugging
```

```
# or testing. Production installations should be run as a WSGI application
# behind Apache HTTPD.
DEFAULT SERVER = '127.0.0.1'
# The default port on which the app server will listen if not set in the
# environment by the runtime
DEFAULT_SERVER_PORT = 5050
# This param is used to override the default web server information about
# the web technology and the frameworks being used in the application
# An attacker could use this information to fingerprint underlying operating
# system and research known exploits for the specific version of
# software in use
WEB_SERVER = 'Python'
# Enable X-Frame-Option protection.
# Set to one of "SAMEORIGIN", "ALLOW-FROM origin" or "" to disable.
# Note that "DENY" is NOT supported (and will be silently ignored).
# See https://tools.ietf.org/html/rfc7034 for more info.
X_FRAME_OPTIONS = "SAMEORIGIN"
# The Content-Security-Policy header allows you to restrict how resources
# such as JavaScript, CSS, or pretty much anything that the browser loads.
# see https://content-security-policy.com/#source_list for more info
# e.g. "default-src https: data: 'unsafe-inline' 'unsafe-eval';"
CONTENT_SECURITY_POLICY = "default-src ws: http: data: blob: 'unsafe-inline'" \
                          " 'unsafe-eval':"
# STRICT_TRANSPORT_SECURITY_ENABLED when set to True will set the
# Strict-Transport-Security header
STRICT_TRANSPORT_SECURITY_ENABLED = False
# The Strict-Transport-Security header tells the browser to convert all HTTP
# requests to HTTPS, preventing man-in-the-middle (MITM) attacks.
# e.g. 'max-age=31536000; includeSubDomains'
STRICT_TRANSPORT_SECURITY = "max-age=31536000; includeSubDomains"
# The X-Content-Type-Options header forces the browser to honor the response
# content type instead of trying to detect it, which can be abused to
# generate a cross-site scripting (XSS) attack.
# e.g. nosniff
X_CONTENT_TYPE_OPTIONS = "nosniff"
# The browser will try to prevent reflected XSS attacks by not loading the
# page if the request contains something that looks like JavaScript and the
# response contains the same data. e.g. '1; mode=block'
X_XSS_PROTECTION = "1; mode=block"
# This param is used to validate ALLOWED_HOSTS for the application
# This will be used to avoid Host Header Injection attack
# ALLOWED_HOSTS = ['225.0.0.0/8', '226.0.0.0/7', '228.0.0.0/6']
# ALLOWED_HOSTS = ['127.0.0.1', '192.168.0.1']
```

(continues on next page)

```
# if ALLOWED_HOSTS= [] then it will accept all ips (and application will be
# vulnerable to Host Header Injection attack)
ALLOWED_HOSTS = []
# Hashing algorithm used for password storage
SECURITY_PASSWORD_HASH = 'pbkdf2_sha512'
# Minimum password length
PASSWORD_LENGTH_MIN = 6
# Reverse Proxy parameters
# You must tell the middleware how many proxies set each header
# so it knows what values to trust.
# See https://tinyurl.com/yyg7r9av
# for more information.
# Number of values to trust for X-Forwarded-For
PROXY_X_FOR_COUNT = 1
# Number of values to trust for X-Forwarded-Proto.
PROXY X PROTO COUNT = 1
# Number of values to trust for X-Forwarded-Host.
PROXY X HOST COUNT = 0
# Number of values to trust for X-Forwarded-Port.
PROXY_X_PORT_COUNT = 1
# Number of values to trust for X-Forwarded-Prefix.
PROXY_X_PREFIX_COUNT = 0
# NOTE: CSRF_SESSION_KEY, SECRET_KEY and SECURITY_PASSWORD_SALT are no
        longer part of the main configuration, but are stored in the
        configuration databases 'keys' table and are auto-generated.
# COMPRESSION
COMPRESS_MIMETYPES = [
    'text/html', 'text/css', 'text/xml', 'text/javascript',
    'application/json', 'application/javascript'
COMPRESS\_LEVEL = 9
COMPRESS_MIN_SIZE = 500
# Set the cache control max age for static files in flask to 1 year
SEND_FILE_MAX_AGE_DEFAULT = 31556952
# This will be added to static urls as url parameter with value as
# APP_VERSION_INT for cache busting on version upgrade. If the value is set as
# None or empty string then it will not be added.
# eg - http:localhost:5050/pgadmin.css?intver=3.13
APP_VERSION_PARAM = 'ver'
```

```
# Add the internal version param to below extensions only
APP_VERSION_EXTN = ('.css', '.js', '.html', '.svg', '.png', '.gif', '.ico')
# Data directory for storage of config settings etc. This shouldn't normally
# need to be changed - it's here as various other settings depend on it.
# On Windows, we always store data in %APPDATA%\$(APP_WIN_PATH). On other
# platforms, if we're in server mode we use /var/lib/$(APP_PATH),
# otherwise ~/.$(APP_PATH)
if IS_WIN:
   # Use the short path on windows
   DATA_DIR = os.path.realpath(
       os.path.join(fs_short_path(env('APPDATA')), APP_WIN_PATH)
else:
   if SERVER_MODE:
      DATA_DIR = os.path.join('/var/lib/', APP_PATH)
   else:
      DATA_DIR = os.path.realpath(
          os.path.expanduser('~/' + '.' + APP_PATH + '/')
       )
# An optional login banner to show security warnings/disclaimers etc. at
# login and password recovery etc. HTML may be included for basic formatting,
# For example:
# LOGIN_BANNER = "<h4>Authorised Users Only!</h4>" \
               "Unauthorised use is strictly forbidden."
LOGIN_BANNER = ""
# Log settings
# Debug mode?
DEBUG = False
# Application log level - one of:
# CRITICAL 50
  ERROR 40
  WARNING 30
  SQL 25
  INFO
          20
  DEBUG
          10
  NOTSET
          0
CONSOLE_LOG_LEVEL = logging.WARNING
FILE_LOG_LEVEL = logging.WARNING
# Log format.
JSON_LOGGER = False
CONSOLE_LOG_FORMAT_JSON = OrderedDict([
   ("time", "asctime"),
   ("message", "message"),
   ("level", "levelname")
```

(continues on next page)

```
])
FILE_LOG_FORMAT_JSON = OrderedDict([
   ("time", "asctime"),
   ("message", "message"),
   ("level", "levelname")
])
CONSOLE_LOG_FORMAT = '%(asctime)s: %(levelname)s\t%(name)s:\t%(message)s'
FILE_LOG_FORMAT = '%(asctime)s: %(levelname)s\t%(name)s:\t%(message)s'
# Log file name. This goes in the data directory, except on non-Windows
# platforms in server mode.
if SERVER_MODE and not IS_WIN:
   LOG_FILE = os.path.join('/var/log', APP_PATH, APP_SHORT_NAME + '.log')
else:
   LOG_FILE = os.path.join(DATA_DIR, APP_SHORT_NAME + '.log')
# Log rotation setting
# Log file will be rotated considering values for LOG_ROTATION_SIZE
# & LOG_ROTATION_AGE. Rotated file will be named in format
# - LOG_FILE.Y-m-d_H-M-S
LOG_ROTATION_SIZE = 10 # In MBs
LOG_ROTATION_AGE = 1440 # In minutes
LOG_ROTATION_MAX_LOG_FILES = 90 # Maximum number of backups to retain
# Server Connection Driver Settings
# The default driver used for making connection with PostgreSQL
PG_DEFAULT_DRIVER = 'psycopg3'
# Maximum allowed idle time in minutes before which releasing the connection
# for the particular session. (in minutes)
MAX_SESSION_IDLE_TIME = 60
# External Database Settings
# All configuration settings are stored by default in the SQLite database.
# In order to use external databases like PostgreSQL sets the value of
# CONFIG_DATABASE_URI like below:
# dialect+driver://username:password@host:port/database
# PostgreSQL:
# postgresql://username:password@host:port/database
# Specify Schema Name
# postgresq1://username:password@host:port/database?options=-csearch_path=pgadmin
# Using PGPASS file
# postgresql://username@host:port?options=-csearch_path=pgadmin
```

```
CONFIG DATABASE URI = ''
# User account and settings storage
# The default path to the SQLite database used to store user accounts and
# settings. This default places the file in the same directory as this
# config file, but generates an absolute path for use througout the app.
SQLITE_PATH = env('SQLITE_PATH') or \
  os.path.join(DATA_DIR, APP_SHORT_NAME + '.db')
# SQLITE_TIMEOUT will define how long to wait before throwing the error -
# OperationError due to database lock. On slower system, you may need to change
# this to some higher value.
# (Default: 500 milliseconds)
SQLITE\_TIMEOUT = 500
# Allow database connection passwords to be saved if the user chooses.
# Set to False to disable password saving.
ALLOW SAVE PASSWORD = True
# Maximum number of history queries stored per user/server/database
MAX_QUERY_HIST_STORED = 20
# Server-side session storage path
# SESSION_DB_PATH (Default: $HOME/.pgadmin4/sessions)
# We use SQLite for server-side session storage. There will be one
# SQLite database object per session created.
# Specify the path used to store your session objects.
# If the specified directory does not exist, the setup script will create
# it with permission mode 700 to keep the session database secure.
# On certain systems, you can use shared memory (tmpfs) for maximum
# scalability, for example, on Ubuntu:
# SESSION_DB_PATH = '/run/shm/pgAdmin4_session'
SESSION_DB_PATH = os.path.join(DATA_DIR, 'sessions')
SESSION_COOKIE_NAME = 'pga4_session'
# Mail server settings
```

(continues on next page)

```
# These settings are used when running in web server mode for confirming
# and resetting passwords etc.
# See: http://pythonhosted.org/Flask-Mail/ for more info
MAIL_SERVER = 'localhost'
MAIL PORT = 25
MAIL_USE_SSL = False
MAIL_USE_TLS = False
MAIL_USERNAME = ''
MAIL_PASSWORD = ''
MAIL_DEBUG = False
# Flask-Security overrides Flask-Mail's MAIL_DEFAULT_SENDER setting, so
# that should be set as such:
SECURITY_EMAIL_SENDER = 'no-reply@localhost'
# Mail content settings
# These settings define the content of password reset emails
SECURITY_EMAIL_SUBJECT_PASSWORD_RESET = "Password reset instructions for %s" \
                              % APP_NAME
SECURITY_EMAIL_SUBJECT_PASSWORD_NOTICE = "Your %s password has been reset" \
                               % APP_NAME
SECURITY_EMAIL_SUBJECT_PASSWORD_CHANGE_NOTICE = \
   "Your password for %s has been changed" % APP_NAME
# Email address validation
CHECK_EMAIL_DELIVERABILITY = False
SECURITY_EMAIL_VALIDATOR_ARGS = \
   {"check_deliverability": CHECK_EMAIL_DELIVERABILITY}
# Upgrade checks
# Check for new versions of the application?
UPGRADE_CHECK_ENABLED = True
# Where should we get the data from?
UPGRADE_CHECK_URL = 'https://www.pgadmin.org/versions.json'
# What key should we look at in the upgrade data file?
UPGRADE_CHECK_KEY = 'pgadmin4'
# Which CA file should we use?
# Default to cacert.pem in the same directory as config.py et al.
CA_FILE = os.path.join(os.path.dirname(os.path.realpath(__file__)),
                 "cacert.pem")
```

```
# Check if the detected browser is supported
CHECK_SUPPORTED_BROWSER = True
# Storage Manager storage url config settings
# If user sets STORAGE_DIR to empty it will show all volumes if platform
# is Windows, '/' if it is Linux, Mac or any other unix type system.
# For example:
# 1. STORAGE_DIR = get_drive("C") or get_drive() # return C:/ by default
# where C can be any drive character such as "D", "E", "G" etc
# 2. Set path manually like
# STORAGE_DIR = "/path/to/directory/"
STORAGE_DIR = os.path.join(DATA_DIR, 'storage')
# Default locations for binary utilities (pg_dump, pg_restore etc)
# These are intentionally left empty in the main config file, but are
# expected to be overridden by packagers in config_distro.py.
# A default location can be specified for each database driver ID, in
# a dictionary. Either an absolute or relative path can be specified.
# Version-specific defaults can also be specified, which will take priority
# over un-versioned paths.
#
# In cases where it may be difficult to know what the working directory
# is, "$DIR" can be specified. This will be replaced with the path to the
# top-level pgAdmin4.py file. For example, on macOS we might use:
# $DIR/../../SharedSupport
DEFAULT_BINARY_PATHS = {
  "pg": "",
   "pg-12": ""
   "pg-13": ""
   "pg-14": ""
   "pg-15": "".
   "pg-16": ""
   "ppas": "".
   "ppas-12": ""
   "ppas-13": ""
   "ppas-14": ""
   "ppas-15": ""
   "ppas-16": ""
}
```

(continues on next page)

```
# Test settings - used primarily by the regression suite, not for users
# The default path for SQLite database for testing
TEST_SQLITE_PATH = os.path.join(DATA_DIR, 'test_pgadmin4.db')
# Allows flask application to response to the each request asynchronously
THREADED MODE = True
# Do not allow SQLALCHEMY to track modification as it is going to be
# deprecated in future
SQLALCHEMY_TRACK_MODIFICATIONS = False
# Number of records to fetch in one batch in query tool when query result
# set is large.
ON_DEMAND_RECORD_COUNT = 1000
# Allow users to display Gravatar image for their username in Server mode
SHOW GRAVATAR IMAGE = True
# Set cookie path and options
COOKIE_DEFAULT_PATH = '/'
COOKIE_DEFAULT_DOMAIN = None
SESSION_COOKIE_DOMAIN = None
SESSION_COOKIE_SAMESITE = 'Lax'
SESSION_COOKIE_SECURE = False
SESSION_COOKIE_HTTPONLY = True
# Skip storing session in files and cache for specific paths
SESSION_SKIP_PATHS = [
  '/misc/ping'
1
# Session expiration support
# SESSION_EXPIRATION_TIME is the interval in Days. Session will be
# expire after the specified number of *days*.
SESSION_EXPIRATION_TIME = 1
```

```
# Make SESSION_EXPIRATION_TIME to 1 week in DESKTOP mode
if not SERVER_MODE:
   SESSION EXPIRATION TIME = 7
# CHECK_SESSION_FILES_INTERVAL is interval in Hours. Application will check
# the session files for cleanup after specified number of *hours*.
CHECK_SESSION_FILES_INTERVAL = 24
# USER_INACTIVITY_TIMEOUT is interval in Seconds. If the pgAdmin screen is left
# unattended for <USER_INACTIVITY_TIMEOUT> seconds then the user will
# be logged out. When set to 0, the timeout will be disabled.
# If pgAdmin doesn't detect any activity in the time specified (in seconds),
# the user will be forcibly logged out from pgAdmin. Set to zero to disable
# the timeout.
# Note: This is applicable only for SERVER_MODE=True.
USER_INACTIVITY_TIMEOUT = 0
# OVERRIDE_USER_INACTIVITY_TIMEOUT when set to True will override
# USER_INACTIVITY_TIMEOUT when long running queries in the Query Tool
# or Debugger are running. When the queries complete, the inactivity timer
# will restart in this case. If set to False, user inactivity may cause
# transactions or in-process debugging sessions to be aborted.
OVERRIDE_USER_INACTIVITY_TIMEOUT = True
# SSH Tunneling supports only for Python 2.7 and 3.4+
SUPPORT_SSH_TUNNEL = True
# Allow SSH Tunnel passwords to be saved if the user chooses.
# Set to False to disable password saving.
ALLOW_SAVE_TUNNEL_PASSWORD = False
# Master password is used to encrypt/decrypt saved server passwords
# Applicable for desktop mode only
MASTER_PASSWORD_REQUIRED = True
# pgAdmin encrypts the database connection and ssh tunnel password using a
# master password or pgAdmin login password (for other authentication sources)
# before storing it in the pgAdmin configuration database.
# Below setting is used to allow the user to specify the path to a script
# or program that will return an encryption key which will be used to
# encrypt the passwords. This setting is used only in server mode when
# auth sources are oauth, Kerberos, and webserver.
# You can pass the current username as an argument to the external script
# by specifying %u in config value.
# E.g. - MASTER_PASSWORD_HOOK = '<PATH>/passwdgen_script.sh %u'
```

(continues on next page)

```
MASTER_PASSWORD_HOOK = None
# Allows pgAdmin4 to create session cookies based on IP address, so even
# if a cookie is stolen, the attacker will not be able to connect to the
# server using that stolen cookie.
# Note: This 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.
ENHANCED_COOKIE_PROTECTION = True
# External Authentication Sources
# Default setting is internal
# External Supported Sources: ldap, kerberos, oauth2
# Multiple authentication can be achieved by setting this parameter to
# ['ldap', 'internal'] or ['oauth2', 'internal'] or
# ['webserver', 'internal'] etc.
# pgAdmin will authenticate the user with ldap/oauth2 whatever first in the
# list, in case of failure the second authentication option will be considered.
AUTHENTICATION_SOURCES = ['internal']
# MAX_LOGIN_ATTEMPTS which sets the number of failed login attempts that
# are allowed. If this value is exceeded the account is locked and can be
# reset by an administrator. By setting the variable to the value zero
# this feature is deactivated.
MAX LOGIN ATTEMPTS = 3
# Only consider password to check the failed login attempts, email is
# excluded from this check
LOGIN_ATTEMPT_FIELDS = ['password']
# LDAP Configuration
# After ldap authentication, user will be added into the SQLite database
# automatically, if set to True.
# Set it to False, if user should not be added automatically,
# in this case Admin has to add the user manually in the SQLite database.
LDAP_AUTO_CREATE_USER = True
# Connection timeout
LDAP_CONNECTION_TIMEOUT = 10
```

```
# Server connection details (REQUIRED)
# example: ldap://<ip-address>:<port> or ldap://<hostname>:<port>
LDAP_SERVER_URI = 'ldap://<ip-address>:<port>'
# The LDAP attribute containing user names. In OpenLDAP, this may be 'uid'
# whilst in AD, 'sAMAccountName' might be appropriate. (REQUIRED)
LDAP_USERNAME_ATTRIBUTE = '<User-id>'
# 3 ways to configure LDAP as follows (Choose anyone):
# 1. Dedicated User binding
# LDAP Bind User DN Example: cn=username.dc=example.dc=com
# Set this parameter to allow the connection to bind using a dedicated user.
# After the connection is made, the pgadmin login user will be further
# authenticated by the username and password provided
# at the login screen.
LDAP_BIND_USER = None
# LDAP Bind User Password
LDAP_BIND_PASSWORD = None
# OR ################
# 2. Anonymous Binding
# Set this parameter to allow the anonymous bind.
# After the connection is made, the pgadmin login user will be further
# authenticated by the username and password provided
LDAP_ANONYMOUS_BIND = False
# OR #################
# 3. Bind as pgAdmin user
# BaseDN (REQUIRED)
# AD example:
# (&(objectClass=user) (memberof=CN=MYGROUP, CN=Users, dc=example, dc=com))
# OpenLDAP example: CN=Users, dc=example, dc=com
LDAP_BASE_DN = '<Base-DN>'
# Configure the bind format string
# Default: LDAP_BIND_FORMAT="
  {LDAP_USERNAME_ATTRIBUTE}={LDAP_USERNAME}, {LDAP_BASE_DN}"
# The current available options are:
# LDAP_USERNAME_ATTRIBUTE, LDAP_USERNAME, LDAP_BASE_DN
# Example: LDAP_BIND_FORMAT="myldapuser@sales.example.com"
          LDAP_BIND_FORMAT="NET\\myldapuser"
LDAP_BIND_FORMAT = '{LDAP_USERNAME_ATTRIBUTE}={LDAP_USERNAME}, {LDAP_BASE_DN}'
```

(continues on next page)

```
# Search ldap for further authentication (REQUIRED)
# It can be optional while bind as pgAdmin user
LDAP_SEARCH_BASE_DN = '<Search-Base-DN>'
# The LDAP attribute indicates whether the DN (Distinguished Names)
# are case sensitive or not
LDAP_DN_CASE_SENSITIVE = False
# Filter string for the user search.
# For OpenLDAP, '(cn=*)' may well be enough.
# For AD, you might use '(objectClass=user)' (REQUIRED)
LDAP_SEARCH_FILTER = '(objectclass=*)'
# Search scope for users (one of BASE, LEVEL or SUBTREE)
LDAP_SEARCH_SCOPE = 'SUBTREE'
# Use TLS? If the URI scheme is ldaps://, this is ignored.
LDAP_USE_STARTTLS = False
# TLS/SSL certificates. Specify if required, otherwise leave empty
LDAP_CA_CERT_FILE = ''
LDAP CERT FILE = ''
LDAP_KEY_FILE = ''
# Some flaky LDAP servers returns malformed schema. If True, no exception
# will be raised and schema is thrown away but authentication will be done.
# This parameter should remain False, as recommended.
LDAP_IGNORE_MALFORMED_SCHEMA = False
# Kerberos Configuration
KRB_APP_HOST_NAME = DEFAULT_SERVER
# If the default_keytab_name is not set in krb5.conf or
# the KRB_KTNAME environment variable is not set then, explicitly set
# the Keytab file
KRB_KTNAME = '<KRB5_KEYTAB_FILE>'
# After kerberos authentication, user will be added into the SQLite database
# automatically, if set to True.
# Set it to False, if user should not be added automatically,
# in this case Admin has to add the user manually in the SQLite database.
KRB_AUTO_CREATE_USER = True
KERBEROS_CCACHE_DIR = os.path.join(DATA_DIR, 'krbccache')
```

```
# Create local directory to store azure credential cache
AZURE_CREDENTIAL_CACHE_DIR = os.path.join(DATA_DIR, 'azurecredentialcache')
# OAuth2 Configuration
# Multiple OAUTH2 providers can be added in the list like [{...},{...}]
# All parameters are required
OAUTH2_CONFIG = [
   {
      # The name of the of the oauth provider, ex: github, google
      'OAUTH2_NAME': None,
      # The display name, ex: Google
      'OAUTH2_DISPLAY_NAME': '<Oauth2 Display Name>',
      # Oauth client id
      'OAUTH2_CLIENT_ID': None,
      # Oauth secret
      'OAUTH2_CLIENT_SECRET': None,
      # URL to generate a token,
      # Ex: https://github.com/login/oauth/access_token
      'OAUTH2_TOKEN_URL': None,
      # URL is used for authentication,
      # Ex: https://github.com/login/oauth/authorize
      'OAUTH2_AUTHORIZATION_URL': None,
      # server metadata url might optional for your provider
       'OAUTH2_SERVER_METADATA_URL': None.
      # Oauth base url, ex: https://api.github.com/
      'OAUTH2_API_BASE_URL': None,
      # Name of the Endpoint, ex: user
       'OAUTH2_USERINFO_ENDPOINT': None,
      # Oauth scope, ex: 'openid email profile'
      # Note that an 'email' claim is required in the resulting profile
      'OAUTH2_SCOPE': None,
      # The claim which is used for the username. If the value is empty the
      # email is used as username, but if a value is provided,
      # the claim has to exist.
      'OAUTH2_USERNAME_CLAIM': None.
      # Font-awesome icon, ex: fa-github
      'OAUTH2_ICON': None,
      # UI button colour, ex: #0000ff
       'OAUTH2_BUTTON_COLOR': None.
      # The additional claims to check on user ID Token or Userinfo response.
      # This is useful to provide additional authorization checks
      # before allowing access.
      # Example for GitLab: allowing all maintainers teams, and a specific
      # developers group to access pgadmin:
```

(continues on next page)

```
# 'OAUTH2_ADDITIONAL_CLAIMS': {
            'https://gitlab.org/claims/groups/maintainer': [
       #
                  'kuberheads/applications',
                  'kuberheads/dba'.
                  'kuberheads/support'
       #
             ],
            'https://gitlab.org/claims/groups/developer': [
                  'kuberheads/applications/team01'
       #
             7.
       # }
       # Example for AzureAD:
       # 'OAUTH2_ADDITIONAL_CLAIMS': {
            'groups': ["0760b6cf-170e-4a14-91b3-4b78e0739963"],
       #
            'wids': ["cf1c38e5-3621-4004-a7cb-879624dced7c"],
       # }
       'OAUTH2_ADDITIONAL_CLAIMS': None,
       # Set this variable to False to disable SSL certificate verification
       # for OAuth2 provider.
       # This may need to set False, in case of self-signed certificates.
       # Ref: https://github.com/psf/requests/issues/6071
       'OAUTH2_SSL_CERT_VERIFICATION': True,
       # set this variable to invalidate the session of the oauth2 provider
       # Example for keycloak:
       # 'OAUTH2_LOGOUT_URL':
       # 'https://example.com/realms/master/protocol/openid-connect/logout?post_logout_
→redirect_uri={redirect_uri}&id_token_hint={id_token}'
       'OAUTH2_LOGOUT_URL': None
   }
]
# After Oauth authentication, user will be added into the SQLite database
# automatically, if set to True.
# Set it to False, if user should not be added automatically,
# in this case Admin has to add the user manually in the SQLite database.
OAUTH2_AUTO_CREATE_USER = True
# Webserver Configuration
WEBSERVER_AUTO_CREATE_USER = True
# REMOTE_USER variable will be used to check the environment variable
# is set or not first, if not available,
# request header will be checked for the same.
# Possible values: REMOTE_USER, HTTP_X_FORWARDED_USER, X-Forwarded-User
WEBSERVER_REMOTE_USER = 'REMOTE_USER'
# Two-factor Authentication Configuration
```

```
# Set it to True, to enable the two-factor authentication
MFA ENABLED = True
# Set it to True, to ask the users to register forcefully for the
# two-authentication methods on logged-in.
MFA_FORCE_REGISTRATION = False
# pgAdmin supports Two-factor authentication by either sending an one-time code
# to an email, or using the TOTP based application like Google Authenticator.
MFA_SUPPORTED_METHODS = ["email", "authenticator"]
# NOTE: Please set the 'Mail server settings' to use 'email' as two-factor
     authentication method.
# Subject for the email verification code
# Default: <APP_NAME> - Verification Code
# e.g. pgAdmin 4 - Verification Code
MFA_EMAIL_SUBJECT = None
# PSOL tool settings
# This will enable PSQL tool in pgAdmin when running in server mode.
# PSQL is always enabled in Desktop mode, however in server mode it is
# disabled by default because users can run arbitrary commands on the
# server through it.
ENABLE\_PSQL = False
# ENABLE_BINARY_PATH_BROWSING setting is used to enable the browse button
# while selecting binary path for the database server in server mode.
# In Desktop mode it is always enabled and setting is of no use.
ENABLE_BINARY_PATH_BROWSING = False
# In server mode, the SHARED_STORAGE setting is used to enable shared storage.
# Specify the name, path, and restricted_access values that should be shared
# between users. When restricted_access is set to True, non-admin users cannot
# upload/add, delete, or rename files/folders in shared storage, only admins
# can do that. Users must provide the absolute path to the folder, and the name
# can be anything they see on the user interface.
# [{ 'name': 'Shared 1', 'path': '/shared_folder',
 'restricted_access': True/False}]
SHARED\_STORAGE = []
# AUTO_DISCOVER_SERVERS setting is used to enable the pgAdmin to discover the
# database server automatically on the local machine.
                                                         (continues on next page)
```

\_ \_

```
# When it is set to False, pgAdmin will not discover servers installed on
# the local machine.
AUTO DISCOVER SERVERS = True
# SERVER_HEARTBEAT_TIMEOUT is used to send the server heartbeat to server
# from the client. This will resolve the orphan database issue once
# browser tab is closed.
SERVER_HEARTBEAT_TIMEOUT = 30 # In seconds
# ENABLE_SERVER_PASS_EXEC_CMD is used to enable/disable Password exec command
# field in server properties. This is used to specify a shell command to be
# executed to retrieve a password to be used for server authentication.
# This setting is applicable only for server mode.
ENABLE_SERVER_PASS_EXEC_CMD = False
# Patch the default config with custom config and other manipulations
from pgadmin.evaluate_config import evaluate_and_patch_config
locals().update(evaluate_and_patch_config(locals()))
```

## 1.1.2 Desktop Deployment

pgAdmin may be deployed as a desktop application by configuring the application to run in desktop mode and then utilising the desktop runtime to host the program on a supported Windows, Mac OS X or Linux installation.

The desktop runtime is a standalone application that when launched, runs the pgAdmin server and opens a window to render the user interface.

**Note:** Pre-compiled and configured installation packages are available for a number of platforms. These packages should be used by end-users whereever possible - the following information is useful for the maintainers of those packages and users interested in understanding how pgAdmin works.

#### See also:

For detailed instructions on building and configuring pgAdmin from scratch, please see the README file in the top level directory of the source code. For convenience, you can find the latest version of the file here, but be aware that this may differ from the version included with the source code for a specific version of pgAdmin.

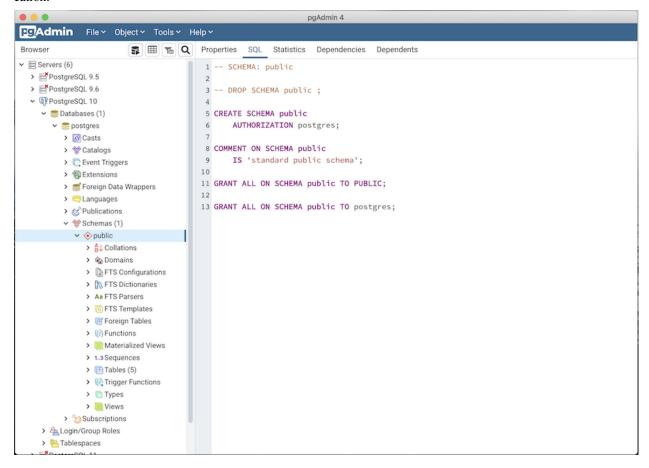
#### Configuration

From pgAdmin 4 v2 onwards, the default configuration mode is server, however, this is overridden by the desktop runtime at startup. In most environments, no Python configuration is required unless you wish to override other default settings.

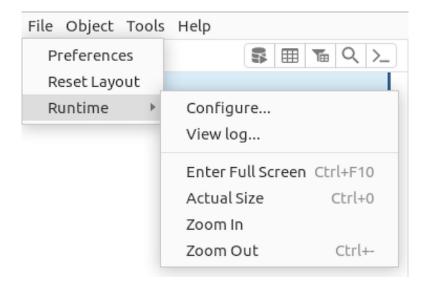
See *The config.py File* for more information on configuration settings.

#### **Desktop Runtime Standalone Application**

The Desktop Runtime is based on NWjs which integrates a browser and the Python server creating a standalone application.



#### **Runtime Menu**

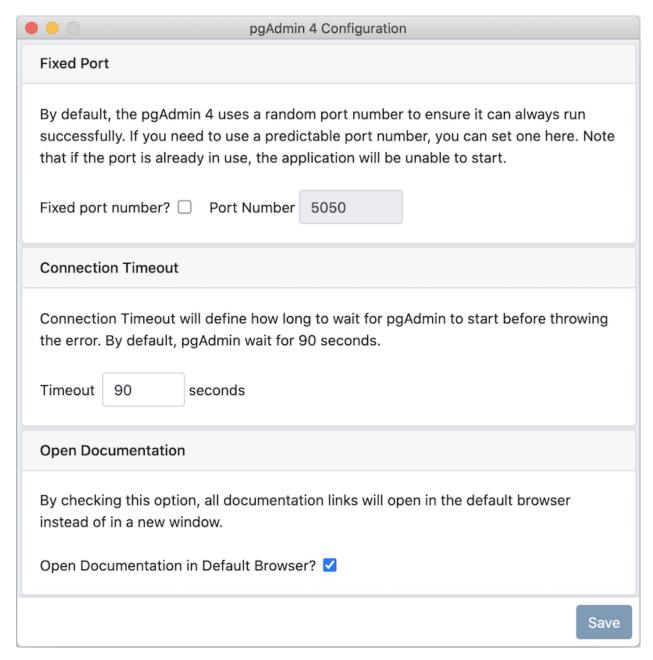


Use the *File Menu* to access the *Runtime Menu*:

Option	Action						
Configure	Click to open configuration dialog to configure fixed port, port number and connection timeout.						
View log	Click to open the view log dialog to view the pgAdmin 4 logs.						
Enter Full	Click to enter/exit the full screen mode. Keyboard Shortcuts: OSX (Cmd + Ctrl + F), Other OS						
Screen	(F10).						
Actual Size	Click to change the window size to it original size. Keyboard Shortcuts: $OSX (Cmd + 0)$ , Other $OS (Ctrl + 0)$ .						
Zoom In	Click to increase the zoom level. Keyboard Shortcuts: OSX (Cmd + +), Other OS (Ctrl + +).						
Zoom Out	Click to decrease the zoom level. Keyboard Shortcuts: OSX (Cmd + -), Other OS (Ctrl + -).						

### **Configuration Dialog**

Use the Runtime Menu to access the Configuration dialog:

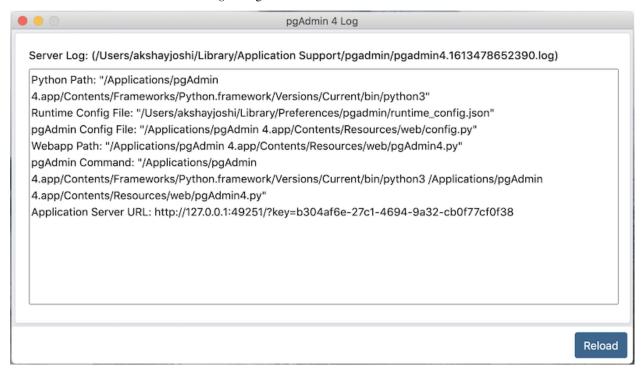


Following are the details of the *Fixed port number?*, *Port Number*, *Connection Timeout*, and 'Open Documentation in Default Browser?' configuration parameters:

Key	Туре	Purpose
FixedPort	Boolean	Use a fixed network port number rather than a random one.
PortNumber	Integer	The port number to use, if using a fixed port.
ConnectionTimeout	Integer	The number of seconds to wait for application server startup.
Open Documentation in De-	Boolean	By checking this option, all documentation links will open in the
fault Browser		default browser instead of in a new window.

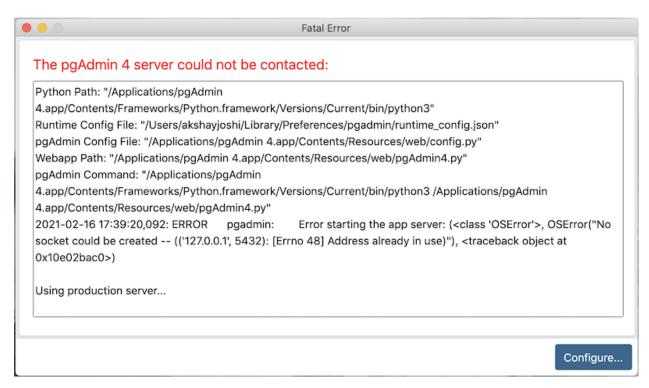
#### Log dialog

Use the Runtime Menu to access the Log dialog:



Click on the *Reload* button at the bottom to view the latest logs of pgAdmin 4 Server.

When executed, the runtime will automatically try to execute the pgAdmin Python application. If execution fails, it will prompt you with error message displaying a *Configure* button at the bottom. You can configure a fixed port number to avoid clashes of the default random port number with other applications and a connection timeout if desired.



If the error is related to Python Path or pgAdmin Python file then you need to create a file named 'dev\_config.json' and specify the following entries:

```
{
    "pythonPath": "/path/to/python.exe",
    "pgadminFile": "/path/to/pgAdmin4.py"
}
```

Note that the dev\_config.py file should only be required by developers who are working outside of a standard installation.

The configuration settings are stored in *runtime\_config.json* file, which will be available on Unix systems (~/.lo-cal/share/pgadmin/), on Mac OS X (~/Library/Preferences/pgadmin), and on Windows (%APPDATA%/pgadmin).

### 1.1.3 Server Deployment

pgAdmin may be deployed as a web application by configuring the app to run in server mode and then deploying it either behind a webserver running as a reverse proxy, or using the WSGI interface.

When deployed in server mode, there are two notable differences for users:

- Users must login before they can use pgAdmin. An initial superuser account is created when server mode is initialised, and this user can add additional superusers and non-superusers as required.
- File storage is restricted to a virtual root directory for each individual user under the directory configured using the STORAGE\_DIR configuration parameter. Users do not have access to the complete filesystem of the server.

The following instructions demonstrate how pgAdmin may be run as a WSGI application under Apache HTTPD, using mod\_wsgi, standalone using uWSGI or Gunicorn, or under NGINX using uWSGI or Gunicorn.

#### See also:

For detailed instructions on building and configuring pgAdmin from scratch, please see the README file in the top level directory of the source code. For convenience, you can find the latest version of the file here, but be aware that this may differ from the version included with the source code for a specific version of pgAdmin.

#### Requirements

**Important**: Some components of pgAdmin require the ability to maintain affinity between client sessions and a specific database connection (for example, the Query Tool in which the user might run a BEGIN command followed by a number of DML SQL statements, and then a COMMIT). pgAdmin has been designed with built-in connection management to handle this, however it requires that only a single Python process is used because it is not easily possible to maintain affinity between a client session and one of multiple WSGI worker processes.

On Windows systems, the Apache HTTP server uses a single process, multi-threaded architecture. WSGI applications run in embedded mode, which means that only a single process will be present on this platform in all cases.

On Unix systems, the Apache HTTP server typically uses a multi-process, single threaded architecture (this is dependent on the MPM that is chosen at compile time). If embedded mode is chosen for the WSGI application, then there will be one Python environment for each Apache process, each with it's own connection manager which will lead to loss of connection affinity. Therefore one should use mod\_wsgi's daemon mode, configured to use a single process. This will launch a single instance of the WSGI application which is utilised by all the Apache worker processes.

Whilst it is true that this is a potential performance bottleneck, in reality pgAdmin is not a web application that's ever likely to see heavy traffic unlike a busy website, so in practice should not be an issue.

Future versions of pgAdmin may introduce a shared connection manager process to overcome this limitation, however that is a significant amount of work for little practical gain.

#### Configuration

In order to configure pgAdmin to run in server mode, it may be necessary to configure the Python code to run in multi-user mode, and then to configure the web server to find and execute the code.

See *The config.py File* for more information on configuration settings.

#### **Python**

From pgAdmin 4 v2 onwards, server mode is the default configuration. If running under the desktop runtime, this is overridden automatically. There should typically be no need to modify the configuration simply to enable server mode to work, however it may be desirable to adjust some of the paths used.

In order to configure the Python code, follow these steps:

- 1. Create a config\_local.py file alongside the existing config.py file.
- 2. Edit config\_local.py and add the following settings. In most cases, the default file locations should be appropriate:

NOTE: You must ensure the directories specified are writeable by the user that the web server processes will be running as, e.g. apache or www-data. You may specify DATA\_DIR in order to create all required directories and files under DATA\_DIR folder.

```
LOG_FILE = '/var/log/pgadmin4/pgadmin4.log'
SQLITE_PATH = '/var/lib/pgadmin4/pgadmin4.db'
SESSION_DB_PATH = '/var/lib/pgadmin4/sessions'
STORAGE_DIR = '/var/lib/pgadmin4/storage'
AZURE_CREDENTIAL_CACHE_DIR = '/var/lib/pgadmin4/azurecredentialcache'
KERBEROS_CCACHE_DIR = '/var/lib/pgadmin4/kerberoscache'
```

4. Run the following command to create the configuration database:

```
# python setup.py setup-db
```

5. Change the ownership of the configuration database to the user that the web server processes will run as, for example, assuming that the web server runs as user www-data in group www-data, and that the SQLite path is /var/lib/pgadmin4/pgadmin4.db:

```
# chown www-data:www-data /var/lib/pgadmin4/pgadmin4.db
```

## Hosting

There are many possible ways to host pgAdmin in server mode. Some examples are given below:

## **Apache HTTPD Configuration (Windows)**

Once Apache HTTP has been configured to support mod\_wsgi, the pgAdmin application may be configured similarly to the example below:

Now open the file C:\Program Files\pgAdmin4\web\pgAdmin4.wsgi with your favorite editor and add the code below which will activate Python virtual environment when Apache server runs.

```
activate_this = 'C:\Program Files\pgAdmin4\venv\Scripts\activate_this.py'
exec(open(activate_this).read())
```

Note: The changes made in pgAdmin4.wsgi file will revert when pgAdmin4 is either upgraded or downgraded.

## **Apache HTTPD Configuration (Linux/Unix)**

Once Apache HTTP has been configured to support mod\_wsgi, the pgAdmin application may be configured similarly to the example below:

(continues on next page)

1.1. Deployment 29

(continued from previous page)

```
Allow from all
</Directory>
</VirtualHost>
```

**Note:** If you're using Apache HTTPD 2.4 or later, replace the lines:

```
Order deny,allow
Allow from all
```

with:

```
Require all granted
```

Adjust as needed to suit your access control requirements.

## **Standalone Gunicorn Configuration**

pgAdmin may be hosted by Gunicorn directly simply by running a command such as the one shown below. Note that this example assumes pgAdmin was installed using the Python Wheel (you may need to adjust the path to suit your installation):

```
gunicorn --bind 0.0.0.0:80 \
    --workers=1 \
    --threads=25 \
    --chdir /usr/lib/python3.7/dist-packages/pgadmin4 \
    pgAdmin4:app
```

## Standalone uWSGI Configuration

pgAdmin may be hosted by uWSGI directly simply by running a command such as the one shown below. Note that this example assumes pgAdmin was installed using the Python Wheel (you may need to adjust the path to suit your installation):

```
uwsgi --http-socket 0.0.0.0:80 \
    --processes 1 \
    --threads 25 \
    --chdir /usr/lib/python3.7/dist-packages/pgadmin4/ \
    --mount /=pgAdmin4:app
```

## **NGINX Configuration with Gunicorn**

pgAdmin can be hosted by Gunicorn, with NGINX in front of it. Note that these examples assume pgAdmin was installed using the Python Wheel (you may need to adjust the path to suit your installation).

To run with pgAdmin in the root directory of the server, start Gunicorn using a command similar to:

(continues on next page)

(continued from previous page)

```
--chdir /usr/lib/python3.7/dist-packages/pgadmin4 \
pgAdmin4:app
```

And configure NGINX:

```
location / {
   include proxy_params;
   proxy_pass http://unix:/tmp/pgadmin4.sock;
}
```

Alternatively, pgAdmin can be hosted in a sub-directory (/pgadmin4 in this case) on the server. Start Gunicorn as when using the root directory, but configure NGINX as follows:

```
location /pgadmin4/ {
    include proxy_params;
    proxy_pass http://unix:/tmp/pgadmin4.sock;
    proxy_set_header X-Script-Name /pgadmin4;
}
```

## NGINX Configuration with uWSGI

pgAdmin can be hosted by uWSGI, with NGINX in front of it. Note that these examples assume pgAdmin was installed using the Python Wheel (you may need to adjust the path to suit your installation).

To run with pgAdmin in the root directory of the server, start uWSGI using a command similar to:

```
uwsgi --socket /tmp/pgadmin4.sock \
    --processes 1 \
    --threads 25 \
    --chdir /usr/lib/python3.7/dist-packages/pgadmin4/ \
    --manage-script-name \
    --mount /=pgAdmin4:app
```

And configure NGINX:

```
location / { try_files $uri @pgadmin4; }
location @pgadmin4 {
   include uwsgi_params;
   uwsgi_pass unix:/tmp/pgadmin4.sock;
}
```

Alternatively, pgAdmin can be hosted in a sub-directory (/pgadmin4 in this case) on the server. Start uWSGI, noting that the directory name is specified in the mount parameter:

```
uwsgi --socket /tmp/pgadmin4.sock \
    --processes 1 \
    --threads 25 \
    --chdir /usr/lib/python3.7/dist-packages/pgadmin4/ \
    --manage-script-name \
    --mount /pgadmin4=pgAdmin4:app
```

Then, configure NGINX:

1.1. Deployment 31

```
location = /pgadmin4 { rewrite ^ /pgadmin4/; }
location /pgadmin4 { try_files $uri @pgadmin4; }
location @pgadmin4 {
  include uwsgi_params;
  uwsgi_pass unix:/tmp/pgadmin4.sock;
}
```

#### **Additional Information**

**Note:** pgAdmin will spawn additional Python processes from time to time, and relies on the *sys.executable* variable in Python to do this. In some cases, you may need to override that value to ensure the correct interpreter is used, instead of the WSGI host process. For example, uWSGI offers the *-py-sys-executable* command line option to achieve this.

## 1.1.4 Container Deployment

pgAdmin can be deployed in a container using the image at:

https://hub.docker.com/r/dpage/pgadmin4/

There are various tags that you can select from to get the version of pgAdmin that you want, using a command such as this if you're using Docker:

```
docker pull dpage/pgadmin4:<tag name>
```

where <tag name> is one of the following:

Tag name	Description
latest	The most recent release.
8.4	A specific version (8.4 in this case).
8	the latest release of a specific major version (major version 8 in this case).
snapshot	The latest nightly test build.

## **PostgreSQL Utilities**

The PostgreSQL utilities  $pg\_dump$ ,  $pg\_dumpall$ ,  $pg\_restore$  and psql are included in the container to allow backups to be created and restored and other maintenance functions to be executed. Multiple versions are included in the following directories to allow use with different versions of the database server:

- PostgreSQL 12: /usr/local/pgsql-12
- PostgreSQL 13: /usr/local/pgsql-13
- PostgreSQL 14: /usr/local/pgsql-14
- PostgreSQL 15: /usr/local/pgsql-15
- PostgreSQL 16: /usr/local/pgsql-16

The default binary paths set in the container are as follows:

```
DEFAULT_BINARY_PATHS = {
    'pg-16': '/usr/local/pgsql-16',
    'pg-15': '/usr/local/pgsql-15',
    'pg-14': '/usr/local/pgsql-14',
    'pg-13': '/usr/local/pgsql-13',
    'pg-12': '/usr/local/pgsql-12'
}
```

this may be changed in the *Preferences Dialog*.

#### **Environment Variables**

The container will accept the following variables at startup:

### PGADMIN\_DEFAULT\_EMAIL

This is the email address used when setting up the initial administrator account to login to pgAdmin. This variable is required and must be set at launch time.

#### PGADMIN DEFAULT PASSWORD

This is the password used when setting up the initial administrator account to login to pgAdmin. This variable is required and must be set at launch time.

### PGADMIN\_DEFAULT\_PASSWORD\_FILE

This is the password used when setting up the initial administrator account to login to pgAdmin. This value should be set to *docker secret* in order to set the password. This variable is supported in docker swarm environment or while creating container with docker compose. PGADMIN\_DEFAULT\_PASSWORD or PGADMIN\_DEFAULT\_PASSWORD\_FILE variable is required and must be set at launch time.

## PGADMIN\_DISABLE\_POSTFIX

Default: <null>

If left unset, a Postfix server will be started to deliver password reset emails.

If set to any value, the Postfix server will not be started, and pgAdmin will need to be configured to use an external mail server using the *PGADMIN\_CONFIG\_* options below.

This option is useful if you're running in an environment that prevents the use of sudo to start Postfix, or if you wish to use an external mail server.

## PGADMIN\_ENABLE\_TLS

Default: <null>

If left un-set, the container will listen on port 80 for connections in plain text. If set to any value, the container will listen on port 443 for TLS connections.

When TLS is enabled, a certificate and key must be provided. Typically these should be stored on the host file system and mounted from the container. The expected paths are /certs/server.cert and /certs/server.key

## PGADMIN\_LISTEN\_ADDRESS

Default: [::]

Specify the local address that the servers listens on. The default should work for most users - in IPv4-only environments, this may need to be set to 0.0.0.0.

### PGADMIN\_LISTEN\_PORT

Default: 80 or 443 (if TLS is enabled)

1.1. Deployment 33

Allows the port that the server listens on to be set to a specific value rather than using the default.

### PGADMIN\_SERVER\_JSON\_FILE

Default: /pgadmin4/servers.json

Override the default file path for the server definition list. See the /pgadmin4/servers.json mapped file below for more information. See the format of the Servers JSON file.

#### PGADMIN PREFERENCES JSON FILE

Default: /pgadmin4/preferences.json

Override the default file path for the preferences customization at the container creation. See the /pgad-min4/preferences.json mapped file below for more information. See the format of the Preferences JSON file.

### GUNICORN\_ACCESS\_LOGFILE

Default: - (stdout)

Specify an output file in which to store the Gunicorn access logs, instead of sending them to stdout.

## GUNICORN\_LIMIT\_REQUEST\_LINE

Default: 8190

Set the maximum size of HTTP request line in bytes. By default the pgAdmin container uses the maximum limited size offered by Gunicorn as some requests can be quite large. In exceptional cases this value can be set to 0 (zero) to specify "unlimited", however this poses a potential denial of service hazard.

### **GUNICORN\_THREADS**

Default: 25

Adjust the number of threads the Gunicorn server uses to handle incoming requests. This should typically be left as-is, except in highly loaded systems where it may be increased.

#### **PGADMIN CONFIG \***

This is a variable prefix that can be used to override any of the configuration options in pgAdmin's *config.py* file. Add the *PGADMIN\_CONFIG\_* prefix to any variable name from *config.py* and give the value in the format 'string value' for strings, True/False for booleans or 123 for numbers. See below for an example.

Settings are written to /pgadmin4/config\_distro.py within the container, which is read after /pgadmin4/config\_py and before /pgadmin4/config\_local.py. Any settings given will therefore override anything in config\_py, but can be overridden by settings in config\_local.py.

Settings are only written to /pgadmin4/config\_distro.py once, typically on first launch of the container. If /pgad-min4/config\_distro.py contains one or more lines, then no changes are made; for example, if the container instance is restarted, or /pgadmin4/config\_distro.py is mapped to a file on persistent storage (not recommended - use /pgad-min4/config\_local.py instead)!

See *The config.py File* for more information on the available configuration settings.

## **Mapped Files and Directories**

The following files or directories can be mapped from the container onto the host machine to allow configuration to be customised and shared between instances.

**Warning:** Warning: pgAdmin runs as the *pgadmin* user (UID: 5050) in the *pgadmin* group (GID: 5050) in the container. You must ensure that all files are readable, and where necessary (e.g. the working/session directory) writeable for this user on the host machine. For example:

```
sudo chown -R 5050:5050 <host_directory>
```

On some filesystems that do not support extended attributes, it may not be possible to run pgAdmin without specifying a value for *PGADMIN\_LISTEN\_PORT* that is greater than 1024. In such cases, specify an alternate port when launching the container by adding the environment variable, for example:

```
-e 'PGADMIN_LISTEN_PORT=5050'
```

Don't forget to adjust any host-container port mapping accordingly.

#### /var/lib/pgadmin

This is the working directory in which pgAdmin stores session data, user files, configuration files, and it's configuration database. Mapping this directory onto the host machine gives you an easy way to maintain configuration between invocations of the container.

### /pgadmin4/config\_local.py

This file can be used to override configuration settings in pgAdmin. Settings found in config.py can be overridden with deployment specific values if required. Settings in config\_local.py will also override anything specified in the container environment through *PGADMIN\_CONFIG\_* prefixed variables.

## /pgadmin4/servers.json

If this file is mapped, server definitions found in it will be loaded at launch time. This allows connection information to be pre-loaded into the instance of pgAdmin in the container. Note that server definitions are only loaded on first launch, i.e. when the configuration database is created, and not on subsequent launches using the same configuration database.

#### /pgadmin4/preferences.json

If this file is mapped, preferences defined in it will be updated at launch time. This allows customization of preferences settings into the instance of pgAdmin in the container. Note that preferences are only set on first launch, i.e. when the configuration database is created, and not on subsequent launches using the same configuration database.

#### /certs/server.cert

If TLS is enabled, this file will be used as the servers TLS certificate.

## /certs/server.key

If TLS is enabled, this file will be used as the key file for the servers TLS certificate.

1.1. Deployment 35

### **Examples**

Run a simple container over port 80:

```
docker pull dpage/pgadmin4
docker run -p 80:80 \
    -e 'PGADMIN_DEFAULT_EMAIL=user@domain.com' \
    -e 'PGADMIN_DEFAULT_PASSWORD=SuperSecret' \
    -d dpage/pgadmin4
```

Run a simple container over port 80, setting some configuration options:

```
docker pull dpage/pgadmin4
docker run -p 80:80 \
    -e 'PGADMIN_DEFAULT_EMAIL=user@domain.com' \
    -e 'PGADMIN_DEFAULT_PASSWORD=SuperSecret' \
    -e 'PGADMIN_CONFIG_ENHANCED_COOKIE_PROTECTION=True' \
    -e 'PGADMIN_CONFIG_LOGIN_BANNER="Authorised users only!"' \
    -e 'PGADMIN_CONFIG_CONSOLE_LOG_LEVEL=10' \
    -d dpage/pgadmin4
```

Run a TLS secured container using a shared config/storage directory in /private/var/lib/pgadmin on the host, and servers pre-loaded from /tmp/servers.json on the host:

```
docker pull dpage/pgadmin4
docker run -p 443:443 \
    -v /private/var/lib/pgadmin:/var/lib/pgadmin \
    -v /path/to/certificate.cert:/certs/server.cert \
    -v /path/to/certificate.key:/certs/server.key \
    -v /tmp/servers.json:/pgadmin4/servers.json \
    -e 'PGADMIN_DEFAULT_EMAIL=user@domain.com' \
    -e 'PGADMIN_DEFAULT_PASSWORD=SuperSecret' \
    -e 'PGADMIN_ENABLE_TLS=True' \
    -d dpage/pgadmin4
```

### **Reverse Proxying**

Sometimes it's desirable to have users connect to pgAdmin through a reverse proxy rather than directly to the container it's running in. The following examples show how this can be achieved. With traditional reverse proxy servers such as Nginx, pgAdmin is running in a container on the same host, with port 5050 on the host mapped to port 80 on the container, for example:

```
docker pull dpage/pgadmin4
docker run -p 5050:80 \
    -e "PGADMIN_DEFAULT_EMAIL=user@domain.com" \
    -e "PGADMIN_DEFAULT_PASSWORD=SuperSecret" \
    -d dpage/pgadmin4
```

## pgAdmin X-Forwarded-\* Configuration

pgAdmin needs to understand how many proxies set each header so it knows what values to trust. The configuration parameters for the X-Forwarded-\* options which are used for this purpose are shown below, along with their default values.

pgAdmin is configured by default to be able to run behind a reverse proxy even on a non-standard port and these config options don't normally need to be changed. If you're running an unusual configuration (such as multiple reverse proxies) you can adjust the configuration to suit.

```
# Number of values to trust for X-Forwarded-For
PROXY_X_FOR_COUNT = 1

# Number of values to trust for X-Forwarded-Proto.
PROXY_X_PROTO_COUNT = 0

# Number of values to trust for X-Forwarded-Host.
PROXY_X_HOST_COUNT = 0

# Number of values to trust for X-Forwarded-Port.
PROXY_X_PORT_COUNT = 1

# Number of values to trust for X-Forwarded-Prefix.
PROXY_X_PREFIX_COUNT = 0
```

### **HTTP via Nginx**

A configuration similar to the following can be used to create a simple HTTP reverse proxy listening for all hostnames with Nginx:

```
server {
    listen 80;
    server_name _;

    location / {
        proxy_set_header Host $host;
        proxy_pass http://localhost:5050/;
        proxy_redirect off;
    }
}
```

If you wish to host pgAdmin under a subdirectory rather than on the root of the server, you must specify the location and set the *X-Script-Name* header which tells the pgAdmin container how to rewrite paths:

```
server {
    listen 80;
    server_name _;

    location /pgadmin4/ {
        proxy_set_header X-Script-Name /pgadmin4;
        proxy_set_header Host $host;
        proxy_pass http://localhost:5050/;
```

(continues on next page)

1.1. Deployment 37

(continued from previous page)

```
proxy_redirect off;
}
}
```

If Nginx is also running in a container, there is no need to map the pgAdmin port to the host, provided the two containers are running in the same Docker network. In such a configuration, the *proxy\_pass* option would be changed to point to the pgAdmin container within the Docker network.

## **HTTPS via Nginx**

The following configuration can be used to serve pgAdmin over HTTPS to the user whilst the backend container is serving plain HTTP to the proxy server. In this configuration we not only set *X-Script-Name*, but also *X-Scheme* to tell the pgAdmin server to generate any URLs using the correct scheme. A redirect from HTTP to HTTPS is also included. The certificate and key paths may need to be adjusted as appropriate to the specific deployment:

```
server {
   listen 80:
   return 301 https://$host$request_uri;
}
server {
   listen 443;
   server_name _;
   ssl_certificate /etc/nginx/server.cert;
   ssl_certificate_key /etc/nginx/server.key;
   ssl on;
   ssl_session_cache builtin:1000 shared:SSL:10m;
    ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
   ssl_ciphers HIGH:!aNULL:!eNULL:!EXPORT:!CAMELLIA:!DES:!MD5:!PSK:!RC4;
    ssl_prefer_server_ciphers on:
   location /pgadmin4/ {
       proxy_set_header X-Script-Name /pgadmin4;
        proxy_set_header X-Scheme $scheme;
        proxy_set_header Host $host;
       proxy_pass http://localhost:5050/;
       proxy_redirect off;
   }
}
```

### **Traefik**

Configuring Traefik is straightforward for either HTTP or HTTPS when running pgAdmin in a container as it will automatically configure itself to serve content from containers that are running on the local machine, virtual hosting them at *<container\_name>*. *<domain\_name>*, where the domain name is that specified in the Traefik configuration. The container is typically launched per the example below:

```
docker pull dpage/pgadmin4
docker run --name "pgadmin4" \
   -e "PGADMIN_DEFAULT_EMAIL=user@domain.com" \
   -e "PGADMIN_DEFAULT_PASSWORD=SuperSecret" \
   -d dpage/pgadmin4
```

Note that the TCP/IP port has not been mapped to the host as it was in the Nginx example, and the container name has been set to a known value as it will be used as the hostname and may need to be added to the DNS zone file.

The following configuration will listen on ports 80 and 443, redirecting 80 to 443, using the default certificate shipped with Traefik. See the Traefik documentation for options to use certificates from LetsEncrypt or other issuers.

```
defaultEntryPoints = ["http", "https"]

[entryPoints.http]
  address = ":80"
     [entryPoints.http.redirect]
       entryPoint = "https"
  [entryPoints.https]
  address = ":443"
      [entryPoints.https.tls]

[docker]
  domain = "domain_name"
watch = true
```

If you wish to host pgAdmin under a subdirectory using Traefik, the configuration changes are typically made to the way the container is launched and not to Traefik itself. For example, to host pgAdmin under /pgadmin4/ instead of at the root directory, the Traefik configuration above may be used if the container is launched like this while using the version v1 of Traefik:

```
docker pull dpage/pgadmin4
docker run --name "pgadmin4" \
    -e "PGADMIN_DEFAULT_EMAIL=user@domain.com" \
    -e "PGADMIN_DEFAULT_PASSWORD=SuperSecret" \
    -e "SCRIPT_NAME=/pgadmin4" \
    -1 "traefik.frontend.rule=PathPrefix:/pgadmin4" \
    -d dpage/pgadmin4
```

The *SCRIPT\_NAME* environment variable has been set to tell the container it is being hosted under a subdirectory (in the same way as the *X-Script-Name* header is used with Nginx), and a label has been added to tell Traefik to route requests under the subdirectory to this container.

While using the Traefik configuration for version v2 for hosting pgAdmin under subdirectory the container is typically launched per the example below:

1.1. Deployment 39

```
docker pull dpage/pgadmin4
docker run --name "pgadmin4" \
    -e "PGADMIN_DEFAULT_EMAIL=user@domain.com" \
    -e "PGADMIN_DEFAULT_PASSWORD=SuperSecret" \
    -e "SCRIPT_NAME=/pgadmin4" \
    -1 "traefik.frontend.pgadmin4.rule=Host(`host.example.com`) && PathPrefix(`/
    →pgadmin4`)" \
    -d dpage/pgadmin4
```

# 1.2 Login Page

Use the *Login* page to log in to pgAdmin:



Use the fields in the *Login* page to authenticate your connection. There are two ways to authenticate your connection:

- From pgAdmin version 4.21 onwards, support for LDAP authentication has been added. If LDAP authentication has been enabled for your pgAdmin application, you can use your LDAP credentials to log in to pgAdmin:
  - Provide the LDAP username in the *Email Address/Username* field.
  - Provide your LDAP password in the Password field.
- Alternatively, you can use the following information to log in to pgAdmin:
  - Provide the email address associated with your account in the Email Address/Username field.
  - Provide your password in the Password field.

Click the *Login* button to securely log into pgAdmin.

Please note that if the pgAdmin server is restarted, then you will be logged out. You need to re-login to continue.

## 1.2.1 Recovering a Lost Password

If you cannot supply your password, click the Forgotten your password? button to launch a password recovery utility.



- Provide the email address associated with your account in the *Email Address* field.
- Click the *Recover Password* button to initiate recovery. An email, with directions on how to reset a password, will be sent to the address entered in the *Email Address* field.

If you have forgotten the email associated with your account, please contact your administrator.

Please note that your LDAP password cannot be recovered using this page. If you enter your LDAP username in the *Email Address/Username* field, and then enter your email to recover your password, an error message will be displayed asking you to contact the LDAP administrator to recover your LDAP password.

## 1.2.2 Avoiding a bruteforce attack

You have the possibility to lock an account by setting MAX\_LOGIN\_ATTEMPTS once it has reached the maximum number of login attempts. You can disable this feature by setting the value to zero.

# 1.3 Enabling two-factor authentication (2FA)

### 1.3.1 About two-factor authentication

Two-factor authentication (2FA) is an extra layer of security used when logging into websites or apps. With 2FA, you have to log in with your username and password and provide another form of authentication that only you know or have access to.

## 1.3.2 Setup two-factor authentication

To set up 2FA for pgAdmin 4, you must configure the Two-factor Authentication settings in *config\_local.py* or *config\_system.py* (see the *config.py* documentation) on the system where pgAdmin is installed in Server mode. You can copy these settings from *config.py* file and modify the values for the following parameters.

Parameter	Description
MFA_ENABLED	The default value for this parameter is True. To disable 2FA, set the value to <i>False</i>
SUPPORTED_MFA_LIST	Set the authentication methods to be supported
MFA_EMAIL_SUBJECT	<app_name> - Verification Code e.g. pgAdmin 4 - Verification Code</app_name>
MFA_FORCE_REGISTRATION	Force the user to configure the authentication method on login (if no authentication is already configured).

NOTE: You must set the 'Mail server settings' in config\_local.py or config\_system.py in order to use 'email' as two-factor authentication method (see the config.py documentation).

## 1.3.3 Configure two-factor authentication

To configure 2FA for a user, you must click on 'Two-factor Authentication' in the *User* menu in right-top corner. It will list down all the supported multi factor authentication methods. Click on 'Setup' of one of those methods and follow the steps for each authentication method. You will see the *Delete* button for the authentication method, which is already been configured. Clicking on *Delete* button will deregister the authentication method for the current user.



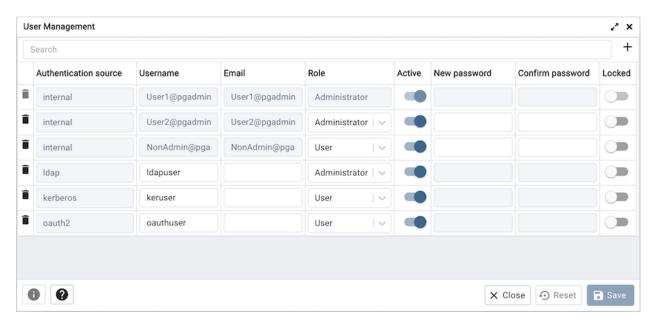
You can also force users to configure the two-factor authentication methods on login by setting MFA\_FORCE\_REGISTRATION parameter to *True*.

# 1.4 User Management Dialog

When invoking pgAdmin in desktop mode, a password is randomly generated, and then ignored. If you install pgAdmin in server mode, you will be prompted for an administrator email and password for the pgAdmin client.

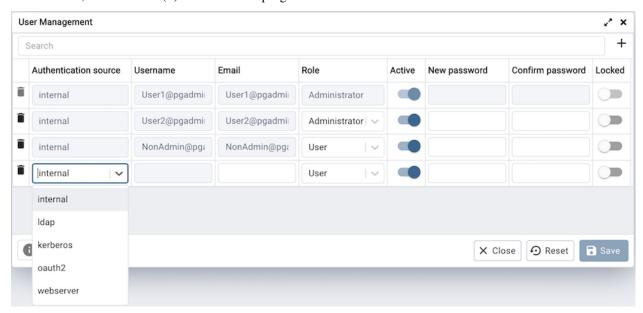
When you authenticate with pgAdmin, the server definitions associated with that login role are made available in the tree control. An administrative user can use the *User Management* dialog to:

- add or delete pgAdmin roles
- · assign privileges
- manage the password associated with a role



Use the *Search* field to specify a criteria and review a list of users that matches with the specified criteria. You can enter a value that matches the following criteria types: *Authentication source*, *Username*, or *Email*. For example, you can enter *ldap* in the search box and only the records having *ldap* as authentication source will be displayed in the *User Management* table.

To add a user, click the Add (+) button at the top right corner.



Provide information about the new pgAdmin role in the row:

- Use the drop-down list box next to *Authentication source* field to select the type of authentication that should be used for the user. If authentication source is only 'internal' then *Authentication source* field is disabled. Supported *Authentication source* are internal, ldap, kerberos, oauth2 and webserver.
- Click in the *Username* field, and provide a username for the user. This field is enabled only when you select authentication source except *internal*. If you select *internal* as authentication source, your email address is displayed in the username field.
- Click in the *Email* field, and provide an email address for the user.

- Use the drop-down list box next to *Role* to select whether a user is an *Administrator* or a *User*.
  - Select Administrator if the user will have administrative privileges within the pgAdmin client.
  - Select *User* to create a non-administrative user account.
- Move the *Active* switch to the *No* position if the account is not currently active; the default is *Yes*. Use this switch to disable account activity without deleting an account.
- Use the *New password* field to provide the password associated with the user specified in the *Email* field. This field is disabled if you select any authentication source except *internal*.
- Re-enter the password in the *Confirm password* field. This field is disabled if you select *ldap* as authentication source.
- *Locked* switch is disabled by default when set to *False*. It is only enabled when the user is locked by trying unsuccessful login attempts. Move the switch to the *False* position if you want to unlock the account.

To discard a user, and revoke access to pgAdmin, click the trash icon to the left of the row and confirm deletion in the *Delete user?* dialog. If the user has created some shared servers, then the *Change Ownership* dialog will appear to change the ownership of a shared server.

Users with the *Administrator* role are able to add, edit and remove pgAdmin users, but otherwise have the same capabilities as those with the *User* role.

- Click the *Help* button (?) to access online help.
- Click the Close button to save work. You will be prompted to return to the dialog if your selections cannot be saved.

## 1.4.1 Using 'setup.py' command line script

**Note:** To manage users using setup.py script, you must use the Python interpreter that is normally used to run pgAdmin to ensure that the required Python packages are available. In most packages, this can be found in the Python Virtual Environment that can be found in the installation directory. When using platform-native packages, the system installation of Python may be the one used by pgAdmin.

When using PIP wheel package to install pgadmin, all the commands can be used without Python interpreter.

Some of the examples: pgadmin4-cli add-user user1@gmail.com password -role 1 pgadmin4-cli get-prefs

## **Manage Users**

### Add User

To add user, invoke setup.py with add-user command line option, followed by email and password. role and active will be optional fields.

```
/path/to/python /path/to/setup.py add-user user1@gmail.com password

# to specify a role, admin and non-admin users:

/path/to/python /path/to/setup.py add-user user1@gmail.com password --admin
/path/to/python /path/to/setup.py add-user user1@gmail.com password --nonadmin

# to specify user's status
```

(continues on next page)

(continued from previous page)

```
/path/to/python /path/to/setup.py add-user user1@gmail.com password --active /path/to/python /path/to/setup.py add-user user1@gmail.com password --inactive
```

#### Add External User

To add external authentication user, invoke setup.py with add-external-user command line option, followed by email, password and authentication source. email, role and status will be optional fields.

```
/path/to/python /path/to/setup.py add-external-user user1@gmail.com ldap

# to specify an email:

/path/to/python /path/to/setup.py add-external-user ldapuser ldap --email user1@gmail.com

# to specify a role, admin and non-admin user:

/path/to/python /path/to/setup.py add-external-user ldapuser ldap --admin /path/to/python /path/to/setup.py add-external-user ldapuser ldap --nonadmin

# to specify user's status

/path/to/python /path/to/setup.py add-external-user user1@gmail.com ldap --active /path/to/python /path/to/setup.py add-external-user user1@gmail.com ldap --inactive
```

## **Update User**

To update user, invoke setup.py with update-user command line option, followed by email address. password, role and active are updatable fields.

```
/path/to/python /path/to/setup.py update-user user1@gmail.com --password new-password

# to specify a role, admin and non-admin user:

/path/to/python /path/to/setup.py update-user user1@gmail.com password --role --admin
/path/to/python /path/to/setup.py update-user user1@gmail.com password --role --nonadmin

# to specify user's status

/path/to/python /path/to/setup.py update-user user1@gmail.com password --active
/path/to/python /path/to/setup.py update-user user1@gmail.com password --inactive
```

### **Update External User**

To update the external user, invoke setup.py with update-external-user command line option, followed by username and auth source. email, password, role and active are updatable fields.

```
# to change email address:

/path/to/python /path/to/setup.py update-external-user ldap ldapuser --email_
-newemail@gmail.com

# to specify a role, admin and non-admin user:

/path/to/python /path/to/setup.py update-user user1@gmail.com password --role --admin /path/to/python /path/to/setup.py update-user user1@gmail.com password --role --nonadmin # to change user's status

/path/to/python /path/to/setup.py update-user ldap ldapuser --active /path/to/python /path/to/setup.py update-user ldap ldapuser --inactive
```

#### **Delete User**

To delete the user, invoke setup.py with delete-user command line option, followed by username and auth\_source. For Internal users, email address will be used instead of username. the option --yes is available to skip the confirmation prompt before deleting the user.

```
/path/to/python /path/to/setup.py delete-user user1@gmail.com --auth-source internal /path/to/python /path/to/setup.py delete-user user2@gmail.com --auth-source internal --
--yes
/path/to/python /path/to/setup.py delete-user ldapuser --auth-source ldap
```

#### **Get User**

To get the user details, invoke setup.py with get-users command line option, followed by username/email address.

```
# to list all the users:
/path/to/python /path/to/setup.py get-users
# to get the user's details:
/path/to/python /path/to/setup.py get-users --username user1@gmail.com
```

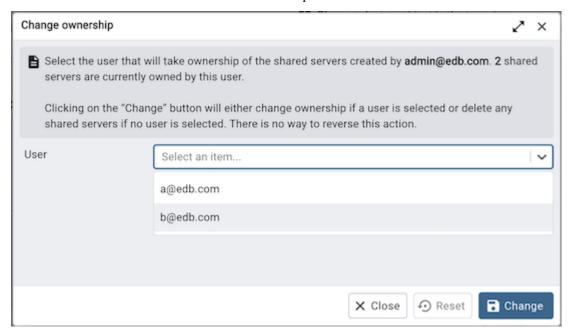
## Output

Each command output can be seen in the json format too by adding –json command line option.

# 1.5 Change Ownership Dialog

Use the *Change Ownership* dialog to change the ownership of the shared servers. This dialog will appear if a user has been deleted from *User Management* and owned some shared servers.

Choose the user who will own the shared servers from the drop-down.



Click the *Change* button to change the ownership.

The shared servers owned by the user will be deleted if the user is not selected from the drop-down.



Click the *Change* button to change the ownership; click *Close* to exit the dialog.

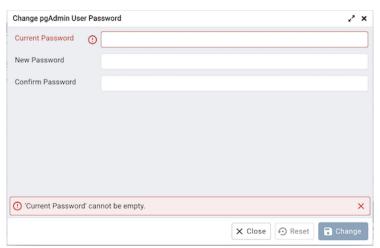
# 1.6 Change User Password Dialog

It is a good policy to routinely change your password to protect data, even in what you may consider a 'safe' environment. In the workplace, failure to apply an appropriate password policy could leave you in breach of Data Protection laws.

Please consider the following guidelines when selecting a password:

- Ensure that your password is an adequate length; 6 characters should be the absolute minimum number of characters in the password.
- The minimum password length is set by default to six characters. This value can be changed by setting the *PASSWORD\_LENGTH\_MIN* option to desired length in pgAdmin configuration; see *The config.py File* for more information.
- Ensure that your password is not open to dictionary attacks. Use a mixture of upper and lower case letters and numerics, and avoid words or names. Consider using the first letter from each word in a phrase that you will remember easily but is an unfamiliar acronym.
- Ensure that your password is changed regularly; at minimum, change it every ninety days.

The guidelines above should be considered a starting point: They are not a comprehensive list and they **will not guarantee security**.



Use the Change Password dialog to change your password:

- Enter your existing password in the Current Password field.
- Enter the desired password for in the *New Password* field.
- Re-enter the new password in the *Confirm Password* field.

Click the Change Password button to change your password; click Close to exit the dialog.

## 1.7 Lock/Restore Account

Account locking/unlocking via user management dialog:

An admin can lock and unlock user from the user management dialog. This allows the admin to lock or restore the user when there are several failed login attempts. This page guides you through configuring per-user locking/restoring. For more details visit *User management*.

Parameter	Description
MAX_LOGIN_ATTEMPTS	Which sets the number of failed login that are allowed. If this value is exceeded, the account is locked and can be reset by an administrator. By setting the variable to the value zero this feature is deactivated.

When Administrator itself gets locked, following steps may be considered to restore it:

• Increase MAX\_LOGIN\_ATTEMPTS, and try to successfully login to restore the account.

OR

By updating configuration database

SQLite DB (pgAdmin4.db):

- Locate the pgAdmin4.db file and open it using any DB Browser (or DB Browser for SQLite)
- After opening the DB file, head towards 'Execute SQL' section.
- Run below query -

 $\label{eq:update_user_set} \textit{update user set locked} = \textit{false}, \textit{login\_attempts} = \textit{0 where username} = \textit{<your\_email\_id} > \textit{one of the user set locked} = \textit{(a)} = \textit{(a)} = \textit{(b)} = \textit{(b$ 

External database:

- Connect to the database.
- Run below query -

UPDATE USER SET LOCKED = false, LOGIN\_ATTEMPTS = 0 WHERE USERNAME = <YOUR\_EMAIL\_ID>

• Make sure the query changes are committed.

Account locking by failed login attempts:



pgAdmin application is configured to lock a user account when a number of consecutive failed login attempts are exceeded.

MAX\_LOGIN\_ATTEMPTS is defaulted to 3 unsuccessful login attempts, after which the account would be locked.

The only way to restore the user account is by contacting the Administrator and ask to unlock it.

# 1.8 Enabling LDAP Authentication

To enable LDAP authentication for pgAdmin, you must configure the LDAP settings in the *config\_local.py* or *config\_system.py* file (see the *config.py* documentation) on the system where pgAdmin is installed in Server mode. You can copy these settings from *config.py* file and modify the values for the following parameters:

There are 3 ways to configure LDAP:

- · Bind as pgAdmin user
- · Anonymous bind
- · Dedicated user bind

Parameter	Description
AUTHENTICATION_SOURCES	The default value for this parameter is <i>internal</i> . To enable LDAP authentication, you must include <i>ldap</i> in the list of values for this parameter. you can modify the value as follows:  • ['ldap']: pgAdmin will use only LDAP authentication.  • ['ldap', 'internal']: pgAdmin will first try to authenticate the user through LDAP. If that authentication fails, then internal user entries of pgAdmin will be used for authentication.  • ['internal', 'ldap']: pgAdmin will first try to authenticate the user through internal user entries. If that authentication fails, then LDAP authentication will be used.
	continues on next nego

continues on next page

Table 5 – continued from previous page

Parameter	Description
LDAP_AUTO_CREATE_USER	Specifies if you want to automatically create a pgAdmin user corresponding to the LDAP user credentials. Please note that LDAP password is not stored in the pgAdmin database.
LDAP_CONNECTION_TIMEOUT	Specifies the connection timeout (in seconds) for LDAP authentication.
LDAP_SERVER_URI	An LDAP URI is a combination of connection protocol (ldap or ldaps), IP address/hostname and port of the directory server that you want to connect to. For example, 'ldap://172.16.209.35:389' is a valid LDAP_SERVER_URI where ldap is the connection protocol, 172.16.209.35 is the IP address and 389 is the port. Port 636 is used for the ldaps communication protocol.
LDAP_USERNAME_ATTRIBUTE	Specifies the LDAP attribute that contains the usernames. For LDAP authentication, you need to enter the value of that particular attribute as username. For example, if you set the value of LDAP_USERNAME_ATTRIBUTE as 'cn' and you have defined 'cn=admin' in your LDAP server entries, you should be able to authenticate by entering 'admin' in the <i>Email Address / Username</i> field and its corresponding password in the <i>Password</i> field.
LDAP_SEARCH_BASE_DN	Specifies the distinguished name (DN) for the top-most user directory that you want to search. You can use this parameter for limiting the search request to a specific group of users. For example, if you want to search only within the Organizational Unit named sales, you can define the value for LDAP_SEARCH_BASE_DN parameter as following: LDAP_SEARCH_BASE_DN = 'ou=sales,dc=example,dc=com' This is an optional parameter only while binding as pgAdmin user. If you do not specify any value for LDAP_SEARCH_BASE_DN, then the value for LDAP_BASE_DN will be considered for the same.
LDAP_SEARCH_FILTER	Defines the criteria to retrieve matching entries in an LDAP search request. For example, LDAP_SEARCH_FILTER = '(objectclass=HR)' setting searches only for users having HR as their objectClass attribute.
LDAP_SEARCH_SCOPE	Indicates the set of entries at or below the Base DN that maybe considered as potential matches for a search request. You can specify the scope of a search as either a <i>base</i> , <i>level</i> , or <i>subtree</i> search. A <i>base</i> search limits the search to the base object. A <i>level</i> search is restricted to the immediate children of a base object, but excludes the base object itself. A <i>subtree</i> search includes all child objects as well as the base object.
LDAP_DN_CASE_SENSITIVE	Indicates whether the DN (Distinguished Names) are case sensitive or not. Possible values are True or False. By default is set to False.
LDAP_USE_STARTTLS	Specifies if you want to use Transport Layer Security (TLS) for secure communication between LDAP clients and LDAP servers. If you specify the connection protocol in <i>LDAP_SERVER_URI</i> as <i>ldaps</i> , this parameter is ignored.
LDAP_CA_CERT_FILE	Specifies the path to the trusted CA certificate file. This parameter is applicable only if you are using <i>ldaps</i> as connection protocol or you have set <i>LDAP_USE_STARTTLS</i> parameter to <i>True</i> .
LDAP_CERT_FILE	Specifies the path to the server certificate file. This parameter is applicable only if you are using <i>ldaps</i> as connection protocol or you have set <i>LDAP_USE_STARTTLS</i> parameter to <i>True</i> .

continues on next page

Table 5 – continued from previous page

Parameter	Description
	•
LDAP_KEY_FILE	Specifies the path to the server private key file. This parameter is applicable only if you are using <i>ldaps</i> as connection protocol or you
	have set <i>LDAP_USE_STARTTLS</i> parameter to <i>True</i> .
LDAP_IGNORE_MALFORMED_SCHEM	Some flaky LDAP servers returns malformed schema. If this parame-
	ter set to <i>True</i> , no exception will be raised and schema is thrown away
	but authentication will be done. This parameter should remain False,
	as recommended.
Bind as pgAdmin user	
LDAP_BASE_DN	Specifies the base DN from where a server will start the search for users. For example, an LDAP search for any user will be performed
	by the server starting at the base DN (dc=example,dc=com). When
	the base DN matches, the full DN (cn=admin,dc=example,dc=com)
	is used to bind with the supplied password.
Anonymous bind	
LDAP_ANONYMOUS_BIND	Set this parameter to <i>True</i> for anonymous binding. After the connection is made, the pgadmin login user will be further authenticated by the username and password provided at the login screen.
Dedicated user bind	
LDAP_BIND_USER	The account of the user to log in for simple bind. Set this parameter to allow the connection to bind using a dedicated user. After the connection is made, the pgadmin login user will be further authenticated by the username and password provided at the login screen. at the login
	screen.
LDAP_BIND_PASSWORD	Password for simple bind. Specify the value if you have set the LDAP_BIND_USER parameter.

# 1.9 Enabling Kerberos Authentication

Prerequisite: Kerberos understanding and setup

Reference: https://web.mit.edu/kerberos/

To configure Kerberos authentication, you must setup your Kerberos Server and obtain a ticket on the client using kinit.

Note: Active Directory users with Kerberos support do not require kinit.

To enable Kerberos authentication for pgAdmin, you must configure the Kerberos settings in *config\_local.py* or *config\_system.py* (see the *config.py* documentation) on the system where pgAdmin is installed in Server mode. You can copy these settings from *config.py* file and modify the values for the following parameters.

Parameter	Description
AUTHENTICATION_SOURCES	The default value for this parameter is <i>internal</i> . To enable Kerberos authentication, you must include <i>kerberos</i> in the list of values for this parameter. you can modify the value as follows:  • ['kerberos']: pgAdmin will use only Kerberos authentication.  • ['kerberos', 'internal']: pgAdmin will first try to authenticate the user through kerberos. If that authentication fails, then it will return back to the login page where you need to provide internal pgAdmin user credentials for authentication.
KERBEROS_AUTO_CREATE_USER	Set the value to <i>True</i> if you want to automatically create a pgAdmin user corresponding to a successfully authenticated Kerberos user. Please note that password is not stored in the pgAdmin database.
KRB_APP_HOST_NAME	Specify the name of <i>pgAdmin webserver hostname</i> . Please note that if it is not set, it will take the value of <i>default_server</i> parameter.

## 1.9.1 Keytab file for HTTP Service

- Generate the *Keytab* file for the HTTP service principal HTTP/<host-name>@realm, and copy it to the *pgAdmin* webserver machine. Ensure that the operating system user owning the *pgAdmin* webserver is the owner of this file and should be accessible by that user.
- Please note that either you should set *default\_keytab\_name* parameter in *krb5.conf* file or the environment variable *KRB5\_KTNAME*. If not set then explicitly set *KRB\_KTNAME* to the location of your *Keytab* file in the *config\_local.py* or *config\_system.py* file.

## 1.9.2 Apache HTTPD Configuration

If the *pgAdmin* server is under the Apache Server, then you need to add the following parameters in *Directory* directive of *Apache HTTPD Configuration*:

- WSGIScriptReloading On
- WSGIPassAuthorization On

## 1.9.3 Browser settings to configure Kerberos Authentication

You need to configure the browser settings on the client machine to use Kerberos authentication via SPNEGO.

- · For Mozilla Firefox
  - Open the low level Firefox configuration page by entering *about:config* in the address bar.
  - In the Search text box, enter: network.negotiate-auth.trusted-uris
  - Double-click the network.negotiate-auth.trusted-uris preference and enter the hostname or the domain of
    the web server that is protected by Kerberos HTTP SPNEGO. Separate multiple domains and hostnames
    with a comma.
  - Click OK.
- · For Google Chrome
  - On Windows:
    - \* Open the Control Panel to access the Internet Options dialog.

- \* Select the Security tab.
- \* Select the Local Intranet zone and click the Sites button.
- \* Make sure that the first two options, *Include all local (intranet) sites not listed in other zones* and *Include all sites that bypass the proxy server* are checked.
- \* Click Advanced and add the names of the domains that are protected by Kerberos HTTP SPNEGO, one at a time, to the list of websites. For example, myhost.example.com. Click Close.
- \* Click OK to save your configuration changes.
- On Linux or macOS:
  - \* Add the *-auth-server-whitelist* parameter to the google-chrome command. For example, to run Chrome from a Linux prompt, run the google-chrome command as follows:

```
google-chrome --auth-server-whitelist = "hostname/domain"
```

## 1.9.4 PostgreSQL Server settings to configure Kerberos Authentication

- To connect the PostgreSQL server with Kerberos authentication, GSSAPI support has to be enabled when PostgreSQL is built and the necessary configuration has to be in place.
- In pgAdmin you need to enable Kerberos authentication for the PostgreSQL server by setting "Kerberos authentication" flag to True in the Server dialog. Once it is enabled, pgAdmin will not prompt for a password and will try to connect to the PostgreSQL server using Kerberos.
- Note that, you have to login into pgAdmin with Kerberos authentication to then connect to PostgreSQL using Kerberos.

#### 1.9.5 Master Password

In the multi user mode, pgAdmin uses user's login password to encrypt/decrypt the PostgreSQL server password. In the Kerberos authentication, the pgAdmin user does not have the password, so we need an encryption key to store the PostgreSQL server password for the servers which are not configured to use the Kerberos authentication. To accomplish this, set the configuration parameter MASTER\_PASSWORD to *True*, so upon setting the master password, it will be used as an encryption key while storing the password. If it is False, the server password can not be stored.

# 1.10 Enabling OAUTH2 Authentication

To enable OAUTH2 authentication for pgAdmin, you must configure the OAUTH2 settings in the *config\_local.py* or *config\_system.py* file (see the *config.py* documentation) on the system where pgAdmin is installed in Server mode. You can copy these settings from *config.py* file and modify the values for the following parameters:

Parameter	Description
AUTHENTICATION_SOURCES	The default value for this parameter is <i>internal</i> . To enable OAUTH2 authentication, you must include <i>oauth2</i> in the list of values for this parameter. you can modify the value as follows:  • ['oauth2', 'internal']: pgAdmin will display an additional button for authenticating with oauth2
	continues on next page

Table 7 – continued from previous page

Davamatav	Description
Parameter	Description
OAUTH2_NAME	The name of the Oauth2 provider, ex: Google, Github
OAUTH2_DISPLAY_NAME	Oauth2 display name in pgAdmin
OAUTH2_CLIENT_ID	Oauth2 Client ID
OAUTH2_CLIENT_SECRET	Oauth2 Client Secret
OAUTH2_TOKEN_URL	Oauth2 Access Token endpoint
OAUTH2_AUTHORIZATION_URL	Endpoint for user authorization
OAUTH2_SERVER_METADATA_URL	Server metadata url for your OAuth2 provider
OAUTH2_API_BASE_URL	Oauth2 base URL endpoint to make requests simple, ex: https://api.github.com/
OAUTH2_USERINFO_ENDPOINT	User Endpoint, ex: user (for github) and useinfo (for google)
OAUTH2_SCOPE	Oauth scope, ex: 'openid email profile'. Note that an 'email' claim is required in the resulting profile.
OAUTH2_ICON	The Font-awesome icon to be placed on the oauth2 button, ex: fagithub
OAUTH2_BUTTON_COLOR	Oauth2 button color
OAUTH2_USERNAME_CLAIM	
	The claim which is used for the username. If the value is empty the email is used as username, but if a value is provided, the claim has to exist. Ex: <i>oid</i> (for AzureAD)
OAUTH2_AUTO_CREATE_USER	Set the value to <i>True</i> if you want to automatically create a pgAdmin user corresponding to a successfully authenticated Oauth2 user. Please note that password is not stored in the pgAdmin database.
OAUTH2_ADDITIONAL_CLAIMS	If a dictionary is provided, pgAdmin will check for a matching key and value on the userinfo endpoint and in the Id Token. In case there is no match with the provided config, the user will receive an authorization error. Useful for checking AzureAD wids or groups, GitLab owner, maintainer and reporter claims.
OAUTH2_SSL_CERT_VERIFICATION	Set this variable to False to disable SSL certificate verification for OAuth2 provider.  This may need to set False, in case of self-signed certificates.

## 1.10.1 Redirect URL

The redirect url to configure Oauth2 server is <a href="https://epgAdmin Server URL">https://epgAdmin Server URL</a>/oauth2/authorize After successful application authorization, the authorization server will redirect the user back to the pgAdmin url specified here. Select https scheme if your pgAdmin server serves over https protocol otherwise select http.

## 1.10.2 Master Password

In the multi user mode, pgAdmin uses user's login password to encrypt/decrypt the PostgreSQL server password. In the Oauth2 authentication, the pgAdmin does not store the user's password, so we need an encryption key to store the PostgreSQL server password. To accomplish this, set the configuration parameter MASTER\_PASSWORD to *True*, so upon setting the master password, it will be used as an encryption key while storing the password. If it is False, the server password can not be stored.

## 1.10.3 Login Page

After configuration, on restart, you can see the login page with the Oauth2 login button(s).



# 1.11 Enabling Webserver Authentication

To configure Webserver authentication, you must setup your webserver with any authentication plug-in (such as Shibboleth, HTTP BASIC auth) as long as it sets the REMOTE\_USER environment variable. To enable Webserver authentication for pgAdmin, you must configure the Webserver settings in the *config\_local.py* or *config\_system.py* file (see the *config.py* documentation) on the system where pgAdmin is installed in Server mode. You can copy these settings from *config.py* file and modify the values for the following parameters:

Parameter	Description
AUTHENTICATION_SOURCES	The default value for this parameter is <i>internal</i> . To enable OAUTH2 authentication, you must include <i>webserver</i> in the list of values for this parameter. you can modify the value as follows:  • ['webserver']: pgAdmin will use only Webserver authentication.  • ['webserver', 'internal']: pgAdmin will first try to authenticate the user through webserver. If that authentication fails, then it will return back to the login page where you need to provide internal pgAdmin user credentials for authentication.
WEBSERVER_AUTO_CREATE_USER	Set the value to <i>True</i> if you want to automatically create a pgAdmin user corresponding to a successfully authenticated Webserver user. Please note that password is not stored in the pgAdmin database.
WEBSERVER_REMOTE_USER	To get the web server remote user details, set this variable to any header or environment variable name which comes from the web server after webserver authentication. The default value is RE-MOTE_USER and the possible values are REMOTE_USER, HTTP_X_FORWARDED_USER, X-Forwarded-User.

## 1.11.1 Master Password

In the multi user mode, pgAdmin uses user's login password to encrypt/decrypt the PostgreSQL server password. In the Webserver authentication, the pgAdmin does not store the user's password, so we need an encryption key to store the PostgreSQL server password. To accomplish this, set the configuration parameter MASTER\_PASSWORD to *True*, so upon setting the master password, it will be used as an encryption key while storing the password. If it is False, the server password can not be stored.

**Note:** Pre-compiled and configured installation packages are available for a number of platforms. These packages should be used by end-users whereever possible - the following information is useful for the maintainers of those packages and users interested in understanding how pgAdmin works.

The pgAdmin 4 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 tree control 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 provide quick access to statistical information about each object in the tree control, and pgAdmin tools and utilities (such as the Query tool and the debugger). pgAdmin opens additional feature tabs each time you access the extended functionality offered by pgAdmin tools; you can open, close, and re-arrange feature tabs as needed.

Use the *Preferences* dialog to customize the content and behaviour of the pgAdmin display. To open the *Preferences* dialog, select *Preferences* from the *File* menu.

*Help* buttons in the lower-left corner of each dialog will open the online help for the dialog. You can access additional Postgres help by navigating through the *Help* menu, and selecting the name of the resource that you wish to open.

You can search for objects in the database using the Search objects

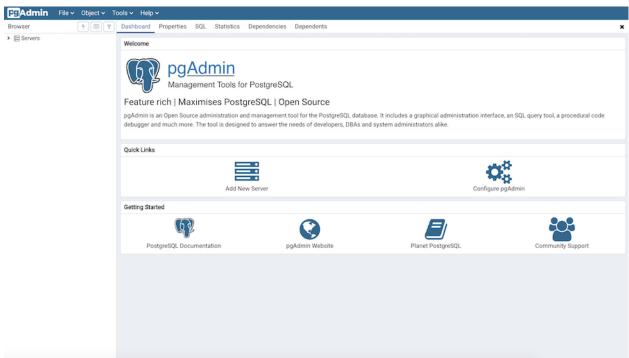
## 1.12 User Interface

pgAdmin 4 supports all PostgreSQL features, from writing simple SQL queries to developing complex databases. It is designed to query an active database (in real-time), allowing you to stay current with modifications and implementations.

Features of pgAdmin 4 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
- · redesigned graphical interfaces
- · powerful management dialogs and tools for common tasks
- · responsive, context-sensitive behavior
- · supportive error messages
- · helpful hints
- online help and information about using pgAdmin dialogs and tools.

When pgAdmin opens, the interface features a menu bar and a window divided into two panes: the *Object Explorer* tree control in the left pane, and a tabbed browser in the right pane.



Select an icon from the Quick Links panel on the Dashboard tab to:

- Click the Add New Server button to open the Create Server dialog to add a new server definition.
- Click the Configure pgAdmin button to open the Preferences dialog to customize your pgAdmin client.

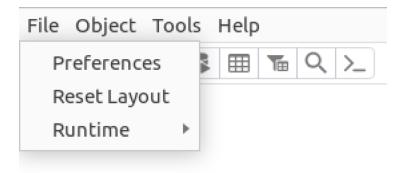
Links in the Getting Started panel open a new browser tab that provide useful information for Postgres users:

- Click the *PostgreSQL Documentation* link to navigate to the *Documentation* page for the PostgreSQL open-source project; once at the project site, you can review the manuals for the currently supported versions of the PostgreSQL server.
- Click the *pgAdmin Website* link to navigate to the pgAdmin project website. The pgAdmin site features news about recent pgAdmin releases and other project information.
- Click the Planet PostgreSQL link to navigate to the blog aggregator for Postgres related blogs.
- Click the *Community Support* link to navigate to the *Community* page at the PostgreSQL open-source project site; this page provides information about obtaining support for PostgreSQL features.

## 1.13 Menu Bar

The pgAdmin menu bar provides drop-down menus for access to options, commands, and utilities. The menu bar displays the following selections: *File*, *Object*, Tools\*, and *Help*. Selections may be grayed out which indicates they are disabled for the object currently selected in the *pgAdmin* tree control.

### 1.13.1 The File Menu

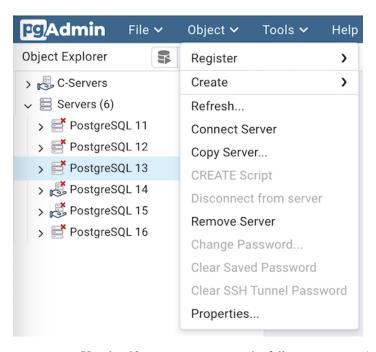


Use the File menu to access the following options:

Option	Action
Prefer- ences	Click to open the <i>Preferences</i> dialog to customize your pgAdmin settings.
Reset Layout	If you have modified the workspace, click to restore the default layout.
Run- time	Click to open a submenu to Configure, View Log and Zoom settings. Only visible when pgAdmin4 runs in desktop mode. To know more about runtime menu <i>click here</i>

1.13. Menu Bar 59

## 1.13.2 The Object Menu

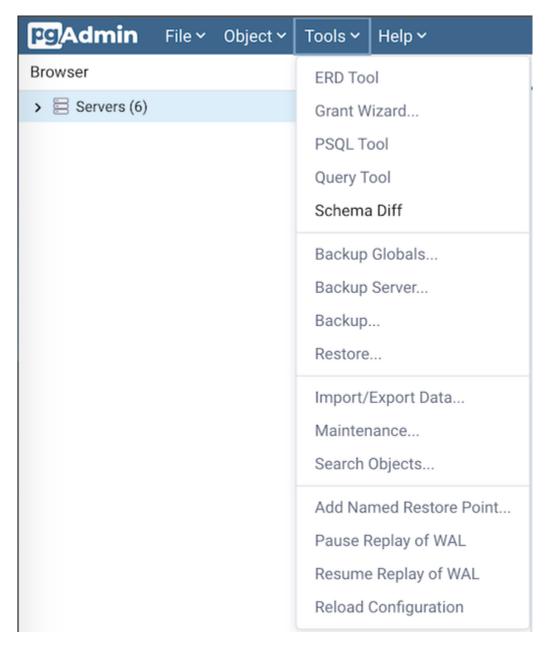


The *Object* menu is context-sensitive. Use the *Object* menu to access the following options (in alphabetical order):

Option	Action
Register 1) Server 2) Deploy Cloud Instance	Click to open the <i>Server</i> dialog to register a server. Click to open the <i>Cloud Deployment</i> dialog to deploy an cloud instance.
Change Password	Click to open the <i>Change Password</i> dialog to change your password.
Clear Saved Password	If you have saved the database server password, click to clear the saved password. Enable only when password is already saved.
Clear SSH Tunnel Password	If you have saved the ssh tunnel password, click to clear the saved password. Enable only when password is al- ready saved.
Connect Server	Click to open the <i>Connect to Server</i> dialog to establish a connection with a server.
Copy Server	Click to copy the currently selected server.
Create	Click <i>Create</i> to access a context menu that provides context-sensitive selections. Your selection opens a <i>Create</i> dialog for creating a new object.
Delete	Click to delete the currently selected object from the server.
Delete (Cascade)	Click to delete the currently selected object and all dependent objects from the server.
Delete (Force)	Click to delete the currently selected database with force option.
Disconnect from server	Click to disconnect from the currently selected server.
Properties	Click to review or modify the currently selected object's properties.
Refresh	Click to refresh the currently selected object.
Remove Server	Click to remove the currently selected server.
Scripts	Click to open the <i>Query tool</i> to edit or view the selected script from the flyout menu.
Trigger(s)	Click to <i>Disable</i> or <i>Enable</i> trigger(s) for the currently selected table. Options are displayed on the flyout menu.
Truncate	Click to remove all rows from a table ( <i>Truncate</i> ), to remove all rows from a table and its child tables ( <i>Truncate Cascade</i> ) or to remove all rows from a table and automatically restart sequences owned by columns ( <i>Truncate Restart Identity</i> ). Options are displayed on the flyout menu.
View Data	Click to access a context menu that provides several options for viewing data (see below).
ERD For Database	Click to open the ERD tool with automatically generated diagram for the database selected. This option is available only when a database is selected. Options are displayed on the flyout menu.
ERD For Table	Click to open the ERD tool with automatically generated diagram for the table selected. This option is available only when a table is selected. Options are displayed on the flyout menu.

1.13. Menu Bar 61

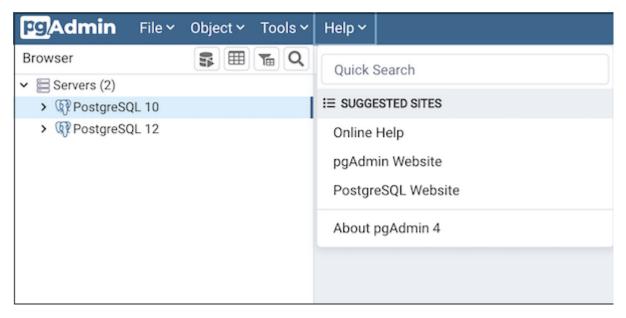
## 1.13.3 The Tools Menu



Use the *Tools* menu to access the following options (in alphabetical order):

Option	Action
ERD Tool	Click to open the <i>ERD Tool</i> and start designing your database.
Grant Wizard	Click to access the Grant Wizard tool.
PSQL Tool	Click to open the <i>PSQL Tool</i> and start PSQL in the current database context.
Query tool	Click to open the <i>Query tool</i> for the currently selected object.
Schema Diff	Click to open the Schema Diff and start comparing two database or two schema.
Backup Globals	Click to open the <i>Backup Globals</i> dialog to backup cluster objects.
Backup Server	Click to open the <i>Backup Server</i> dialog to backup a server.
Backup	Click to open the <i>Backup</i> dialog to backup database objects.
Restore	Click to access the <i>Restore</i> dialog to restore database files from a backup.
Import/Export	Click to open the <i>Import/Export data</i> dialog to import or export data from a table.
Data	
Maintenance	Click to open the <i>Maintenance</i> dialog to VACUUM, ANALYZE, REINDEX, or CLUSTER.
Search Objects	Click to open the Search Objects and start searching any kind of objects in a database.
Add named restore point	Click to open the <i>Add named restore point</i> dialog to take a point-in-time snapshot of the current server state.
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.
Reload Configuration	Click to update configuration files without restarting the server.
Storage Manager	Click to open the <i>Storage Manager</i> to upload, delete, or download the backup files.

## 1.13.4 The Help Menu



Use the options on the *Help* menu to access online help documents, or to review information about the pgAdmin installation (in alphabetical order):

1.13. Menu Bar 63

Op- tion	Action
. —	Type your keywords in the Quick Search field. Typing at least three characters will display all the matching possibilities under Menu items and the relevant documents under Help articles. Click on the options under Menu items to perform action of particular functionality or object. Click on any of the Help articles to open the help of that topic with highlighted text in a separate window.  Note:- If any of the option under Menu items is disabled, then it will provide information via info icon.
	Click to open a window where you will find information about pgAdmin; this includes the current version and the current user.
On- line Help	Click to open documentation support for using pgAdmin utilities, tools and dialogs. Navigate (in the newly opened tab?) help documents in the left browser pane or use the search bar to specify a topic.
pgAd- min Web- site	Click to open the pgAdmin.org website in a browser window.
Post- greSQ Web- site	Click to access the PostgreSQL core documentation hosted at the PostgreSQL site. The site also offers guides, tutorials, and resources.

## 1.14 Toolbar

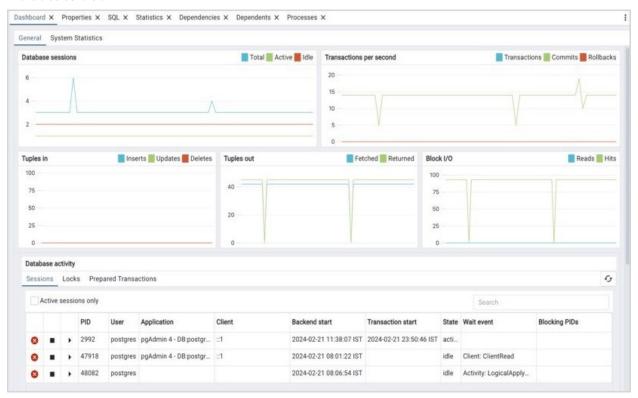
The pgAdmin toolbar provides shortcut buttons for frequently used features like the Query Tool, View/Edit Data, Search Object and the PSQL Tool. This toolbar is visible on the Object explorer panel. Buttons get enabled/disabled based on the selected object node.



- 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.
- Use the Search objects button to access the search objects dialog. It helps you search any database object.
- Use the *PSQL Tool* button to open the PSQL in the current database context.

## 1.15 Tabbed Browser

The right pane of the *pgAdmin* window features a collection of tabs that display information about the object currently selected in the *pgAdmin* tree control in the left window. Select a tab to access information about the highlighted object in the tree control.



The graphs and tables on the *Dashboard* tab provides an active analysis of system statistics and the usage statistics for the selected server or database.

Click the *General* tab to get the usage statistics for the selected server or database:

- 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* panel displays information about sessions, locks, prepared transactions, and server configuration (if applicable). The information is presented in context-sensitive tables. Use controls located above the table to:

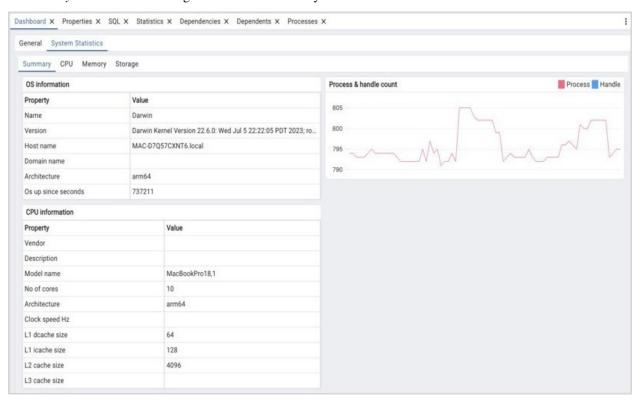
- Click the *Refresh* button to update the information displayed in each table.
- Select *Active sessions only* checkbox to get the information about active sessions only from the list of all the sessions.
- Enter a value in the *Search* box to restrict the table content to one or more sessions that satisfy the search criteria. For example, you can enter a process ID to locate a specific session, or a session state (such as *idle*) to locate all of the sessions that are in an idle state.

1.15. Tabbed Browser 65

You can use icons in the Sessions table to review or control the state of a session:

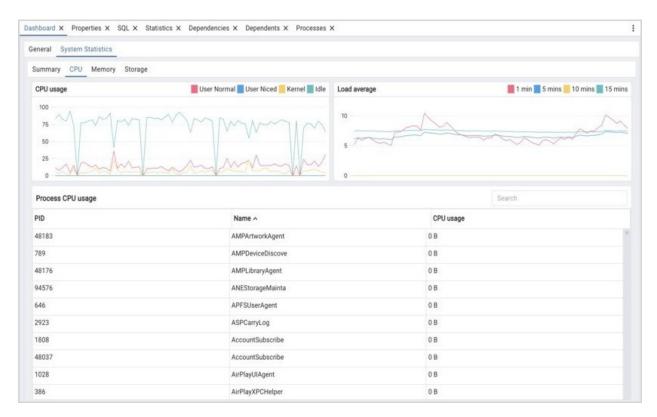
- Use the *Terminate* icon (located in the first column) to stop a session and remove the session from the table. Before the server terminates the session, you will be prompted to confirm your selection.
- Use the *Cancel* icon (located in the second column) to terminate an active query without closing the session. Before canceling the query, the server will prompt you to confirm your selection. When you cancel a query, the value displayed in the *State* column of the table will be updated from *Active* to *Idle*. The session will remain in the table until the session is terminated.
- Use the *Details* icon (located in the third column) to open the *Details* tab; the tab displays information about the selected session.

Click the System Statistics tab to get the statistics for the system:



Click the Summary tab to get the summary of the system:

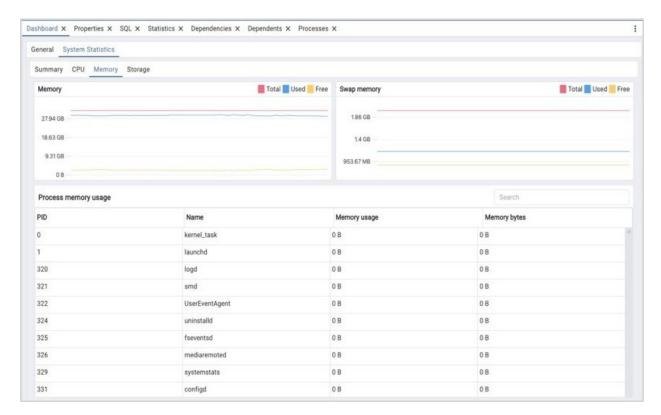
- The OS Information table displays the basic information about the operating system.
- The CPU Information table displays the information about system CPU.
- The *Process & Handle Count* graph displays the total count of processes running and handles opened for the system.



Click the *CPU* tab to get the detailed usage statistics of the system CPU:

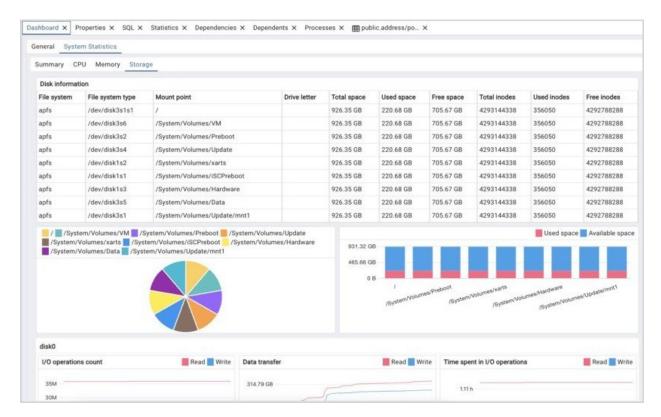
- The *CPU Usage* graph displays the percentage of time spent by CPU for user normal process, user niced process, kernel mode process, idle mode.
- The Load average graph displays the average load of the system over 1, 5, 10, and 15 minute intervals.
- The *Process CPU Usage* table displays information about CPU for each process ID. Enter a value like process ID or name in the Search box to restrict the table content to one or more processes that satisfy the search criteria.

1.15. Tabbed Browser 67



Click the *Memory* tab to get the detailed usage statistics of the system memory:

- The Memory Usage graph displays the size of total, free and used memory per seconds in Gigabytes.
- The Swap Memory graph displays the size of total, free and used swap memory per seconds in Gigabytes.
- The *Process Memory Usage* table displays information about memory for each process ID. Enter a value like process ID or name in the Search box to restrict the table content to one or more processes that satisfy the search criteria.



Click the *Storage* tab to get the detailed information of the system storage:

- The *Disk Information* table displays the information about the disk, such as file system type, mount point, total space, used space and free space and count of total, free, used inodes. The pie graph shows the total spaces in Gigabytes and stacked bar graph shows the used and available space in Gigabytes, for different mount points for file system.
- The Swap Memory graph displays the size of total, free and used swap memory per seconds in Gigabytes.

The disk0 panel displays statistics about the only I/O block device of system:

- The I/O Operations Count graph displays the count of read and write operations per second.
- The Data Transfer graph displays the size of read and writes happened in Gigabytes per second.
- The *Time spent in I/O operations* graph displays the time spent by the device for reading and writing in minutes per second.

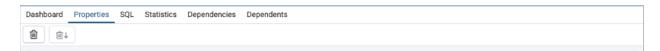


The *Properties* tab displays information about the object selected.

Click the *Delete* icon in the toolbar under the browser tab to delete the selected objects in the Properties panel.

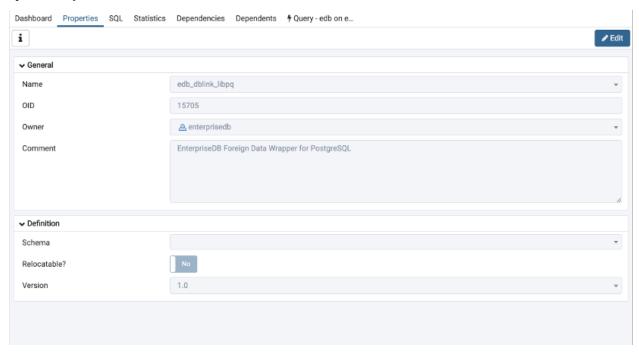
Click the *Drop Cascade* icon in the toolbar under the browser tab to delete the selected objects and all dependent objects in the Properties panel.

1.15. Tabbed Browser 69



Click the *Edit* icon in the toolbar under the browser tabs to launch the *Properties* dialog for the selected object.

To preserve any changes to the *Properties* dialog, click the *Save* icon; your modifications will be displayed in the updated *Properties* tab.

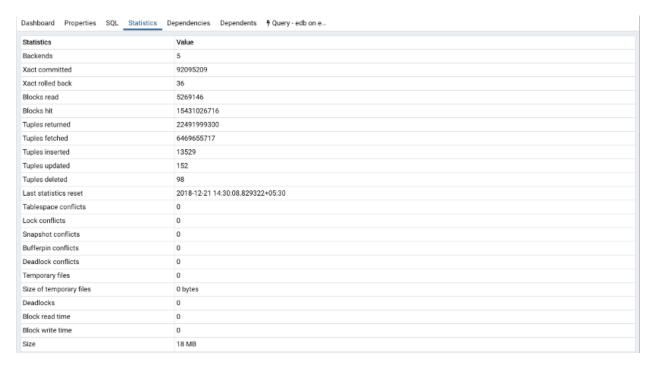


Details about the object highlighted in the tree control are displayed in one or more collapsible panels. You can use the arrow to the left of each panel label to open or close a panel.

```
Dashboard Properties SQL Statistics Dependencies Dependents

1 -- Database: edb
2
3 -- DROP DATABASE edb;
4
5 CREATE DATABASE edb
6 WITH
7 OWNER = enterprisedb
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;
```

The *SQL* tab displays the SQL script that created the highlighted object, and when applicable, a (commented out) SQL statement that will *DROP* the selected object. You can copy the SQL statements to the editor of your choice using cut and paste shortcuts.



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 selected. 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 are available:

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 ta- blespaces in database.	
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.	

continues on next page

1.15. Tabbed Browser 71

Table 9 – continued from previous page

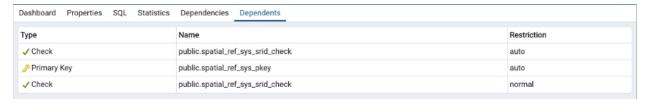
Panel	Description
Deadlocks	displays the number of queries canceled because of a recovery conflict with deadlocks in database.
Block read time	displays the number of milliseconds required to read the blocks read.
Block write time	displays the number of milliseconds required to write the blocks read.
Size	displays the size (in megabytes) of the selected database.



The *Dependencies* tab displays the objects on which the currently selected object depends. If a dependency is dropped, the object currently selected in the pgAdmin tree control will be affected. To ensure the integrity of the entire database structure, the database server makes sure that you do not accidentally drop objects that other objects depend on; you must use the DROP CASCADE command to remove an object with a dependency.

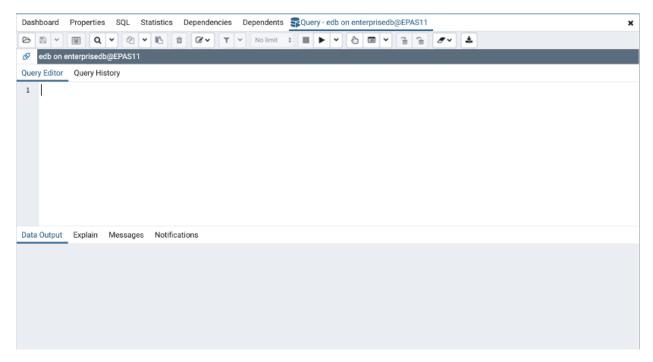
The *Dependencies* table displays the following information:

- 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:
  - If the field is *auto*, the selected object can be dropped separately from the parent object, and will be dropped
    if the parent object is dropped.
  - If the field is *internal*, the selected object was created during the creation of the parent object, and will be dropped if the parent object is dropped.
  - If the field is *normal*, the selected object can be dropped without dropping the parent object.
  - If the field is *blank*, the selected object is required by the system, and cannot be dropped.



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 *Database* field specifies the database in which the object resides.



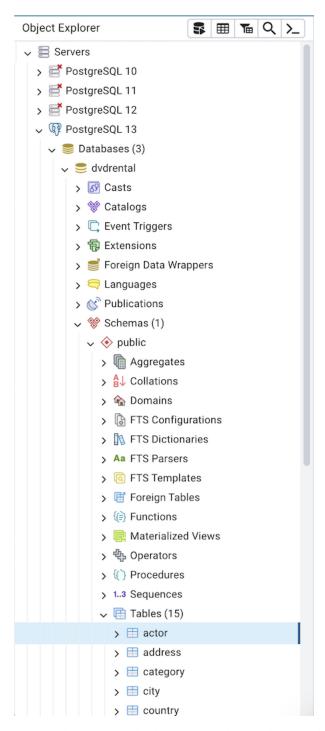
Additional tabs open when you access the extended functionality offered by pgAdmin tools (such as the Query tool, Debugger, or SQL editor). Use the close icon (X) located in the upper-right corner of each tab to close the tab when you are finished using the tool. Like permanent tabs, these tabs may be repositioned in the pgAdmin client window.

By default, each time you open a tool, pgAdmin will open a new browser tab. You can control this behavior by modifying the *Display* node of the *Preferences* dialog for each tool. To open the *Preferences* dialog, select *Preferences* from the *File* menu.

## 1.16 Tree Control

The left pane of the main window displays a tree control (Object explorer) that provides access to the objects that reside on a server.

1.16. Tree Control 73



You can expand nodes in the tree control to view the database objects that reside on a selected server. The tree control expands to display a hierarchical view:

- 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.

You can also **drag and drop** certain objects to the Query Tool 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.

Access context-sensitive menus by right-clicking on a node of the tree control to perform common tasks. Menus display options that include one or more of the following selections (options appear in alphabetical order):

Option	Action	
Add named re- store point	Click to create and enter the name of a restore point.	
Backup	Click to open the <i>Backup</i> dialog to backup database objects.	
Backup Glob- als	Click to open the <i>Backup Globals</i> dialog to backup cluster objects.	
Backup Server	Click to open the <i>Backup Server</i> dialog to backup a server.	
Connect Server	Click to open the <i>Connect to Server</i> dialog to establish a connection with a server.	
Create	Click to access a context menu that provides context-sensitive selections. Your selection oper a <i>Create</i> dialog for creating a new object.	
CREATE Script	Click to open the <i>Query tool</i> to edit or view the CREATE script.	
Debugging	Click through to open the <i>Debug</i> tool or to select <i>Set breakpoint</i> to stop or pause a script execution.	
Delete	Click to delete the currently selected object from the server.	
Delete (Cas-cade)	Click to delete the currently selected object and all dependent objects from the server.	
Delete (Force)	Click to delete the currently selected database with force option.	
Disconnect Database	Click to terminate a database connection.	
Disconnect from server	Click to disconnect from the currently selected server.	
Debugging	Click to access the <i>Debugger</i> tool.	
Grant Wizard	Click to access the Grant Wizard tool.	
Maintenance	Click to open the <i>Maintenance</i> dialog to VACUUM, ANALYZE, REINDEX, or CLUSTER.	
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 <i>Restore</i> dialog to restore database files from a backup.	
View Data	Use the <i>View Data</i> option to access the data stored in a selected table with the <i>Data Output</i> tab of the <i>Query Tool</i> .	

The context-sensitive menus associated with *Tables* and nested *Table* nodes provides additional display options (options appear in alphabetical order):

Option	Action	
Import/Export	Click open the Import/Export dialog to import data to or export data from the selected	
Data	table.	
Reset Statistics	Click to reset statistics for the selected table.	
Scripts	Click to open the <i>Query tool</i> 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 a data grid that displays the first 100 rows of the selected table.	
View Last 100 Rows	Click to access a data grid that displays the last 100 rows of the selected table.	
View All Rows	Click to access a a data grid that displays all rows of the selected table.	
View Filtered	Click to access the <i>Data Filter</i> popup to apply a filter to a set of data.	
Rows		

1.16. Tree Control 75

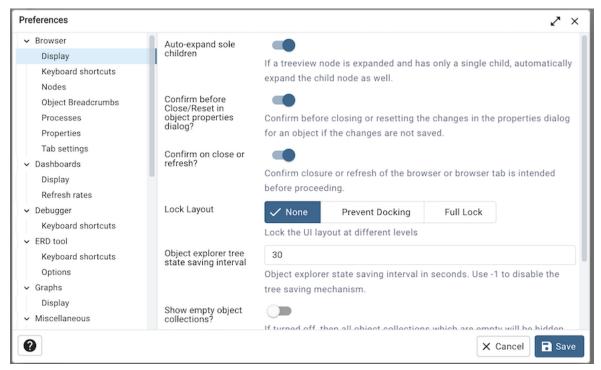
# 1.17 Preferences Dialog

Use options on the *Preferences* dialog to customize the behavior of the client. To open the *Preferences* 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.

#### 1.17.1 The Browser Node

Use preferences found in the *Browser* node of the tree control to personalize your workspace.

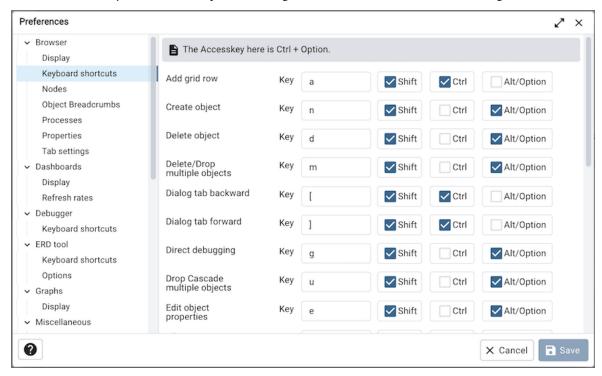


Use the fields on the *Display* panel to specify general display preferences:

- When the *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 *Object explorer 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 *Hide shared servers?* switch is set to *True*, the client will hide all the shared servers from the object explorer.
- When the *Show empty object collections?* switch is turned off, then all object collections which are empty will be hidden from browser tree.

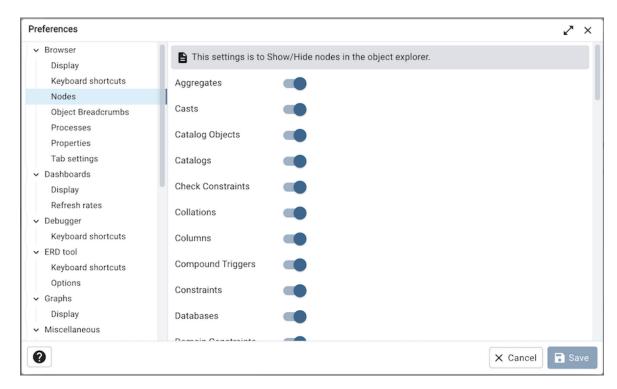
- When the *Show system objects?* switch is set to *True*, the client will display system objects such as system schemas (for example, *pg\_temp*) or system columns (for example, *xmin* or *ctid*) in the tree control.
- When the *Show template databases?* switch is set to *True*, the client will display template databases.

Use the fields on the Keyboard shortcuts panel to configure shortcuts for the main window navigation:



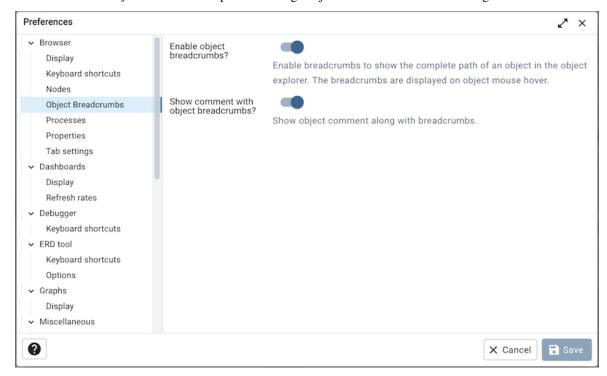
• The panel displays a list of keyboard shortcuts available for the main window; select the combination of the modifier keys along with the key to configure each shortcut.

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 to *Show* or *Hide* the database object. When querying system catalogs, you can reduce the number of object types displayed to increase speed.

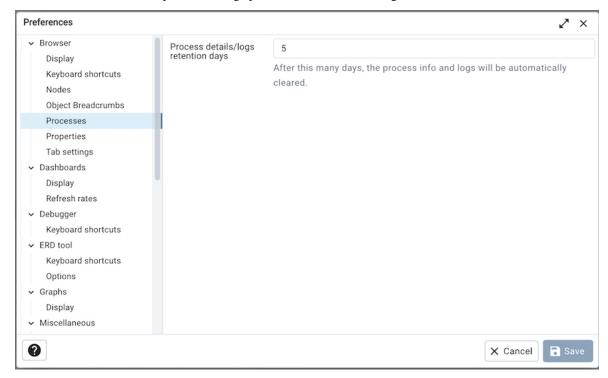
Use the fields on the Object Breadcrumbs panel to change object breadcrumbs related settings:



- Use Enable object breadcrumbs? to enable or disable object breadcrumbs displayed on on object mouse hover.
- Use Show comment with object breadcrumbs? to enable or disable the comment visibility which comes displayed

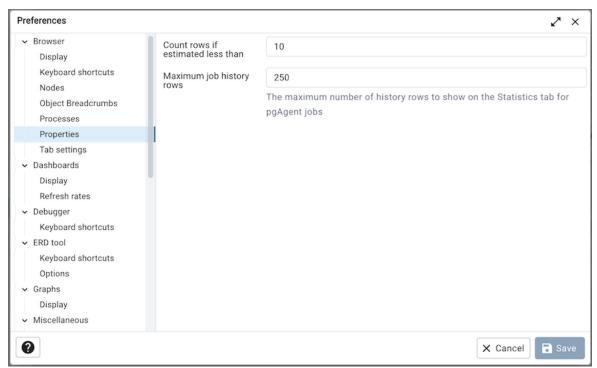
with object breadcrumbs.

Use the fields on the *Processes* panel to change processes tab related settings:



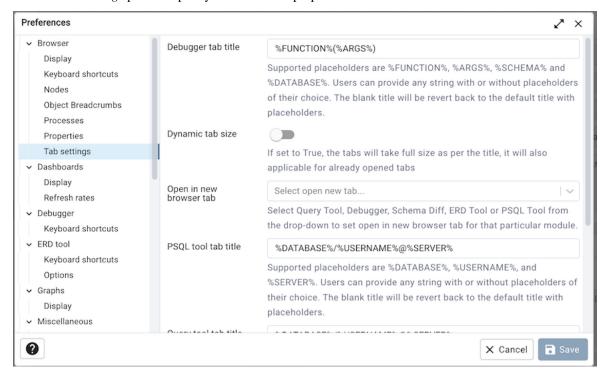
• Change *Process details/logs retention days* to the number of days, the process info and logs will be automatically cleared.

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.

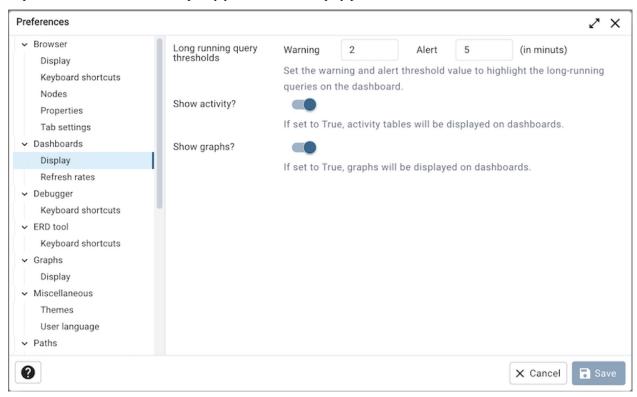
Use field on *Tab settings* panel to specify the tab related properties.



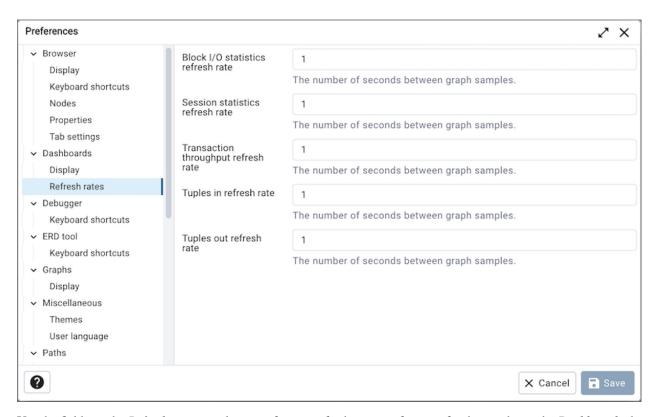
- Use Debugger tab title placeholder field to customize the Debugger tab title.
- When the *Dynamic tab size* If set to True, the tabs will take full size as per the title, it will also applicable for already opened tabs
- When the *Open in new browser tab* filed is selected for Query tool, Schema Diff 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.

#### 1.17.2 The Dashboards Node

Expand the Dashboards node to specify your dashboard display preferences.



- Set the warning and alert threshold value to highlight the long-running queries on the dashboard.
- When the Show activity? switch is set to True, activity tables will be displayed on dashboards.
- When the Show graphs? switch is set to True, graphs will be displayed on dashboards.



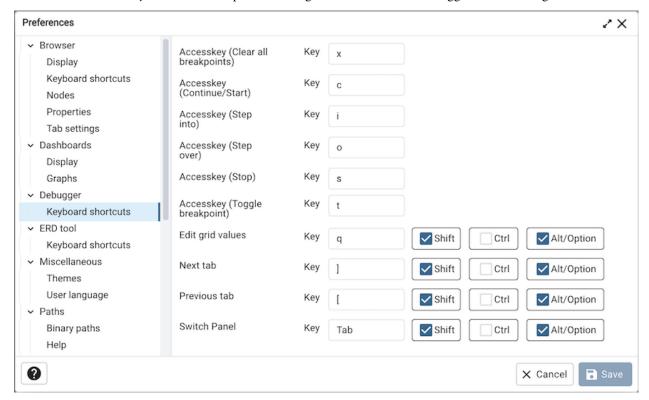
Use the fields on the *Refresh rates* panel to specify your refersh rates preferences for the graphs on the *Dashboard* tab:

- 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.

# 1.17.3 The Debugger Node

Expand the *Debugger* node to specify your debugger display preferences.

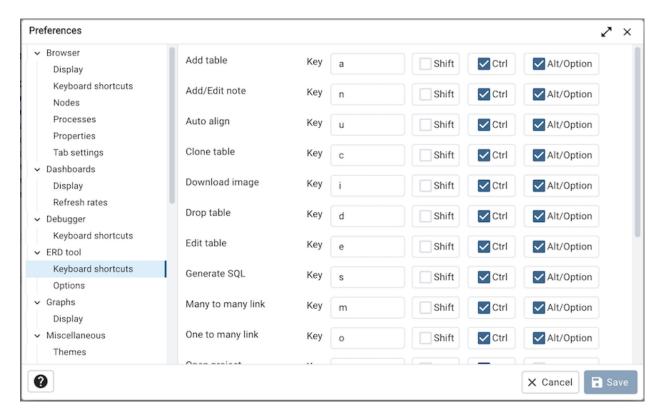
Use the fields on the Keyboard shortcuts panel to configure shortcuts for the debugger window navigation:



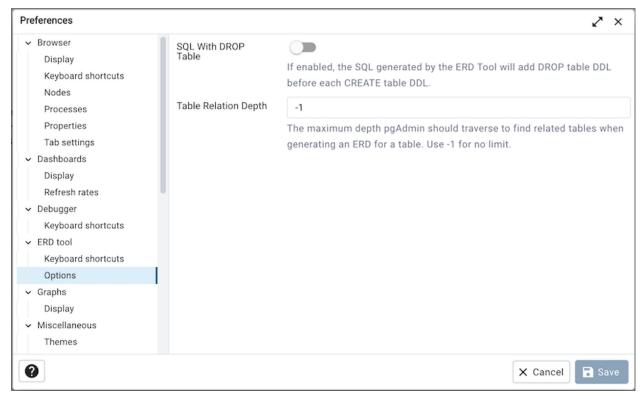
# 1.17.4 The ERD Tool Node

Expand the ERD Tool node to specify your ERD Tool display preferences.

Use the fields on the Keyboard shortcuts panel to configure shortcuts for the ERD Tool window navigation:



Use the fields on the Options panel to manage ERD preferences.

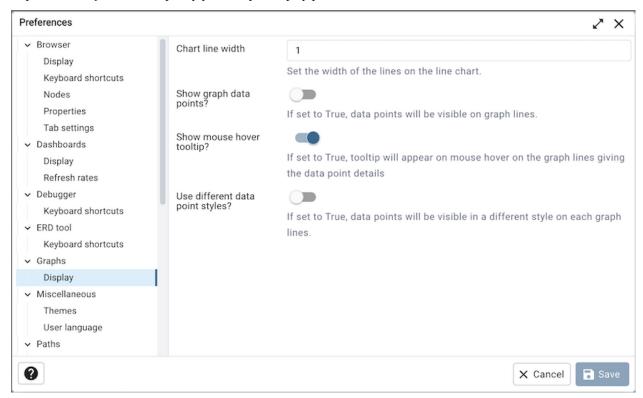


• When the *SQL With DROP Table* switch is set to *True*, the SQL generated by the ERD Tool will add DROP table DDL before each CREATE table DDL.

• *Table Relation Depth* is useful when generating an ERD for a table. It allows to set the limit on the depth level pgAdmin should traverse to find the relations. Use -1 to set no limit.

## 1.17.5 The Graphs Node

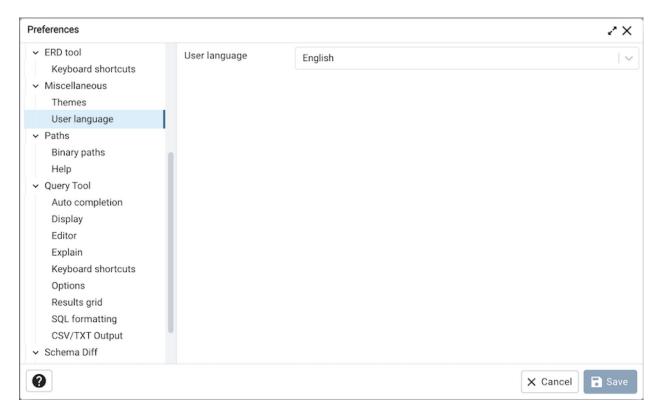
Expand the Graphs node to specify your Graphs display preferences.



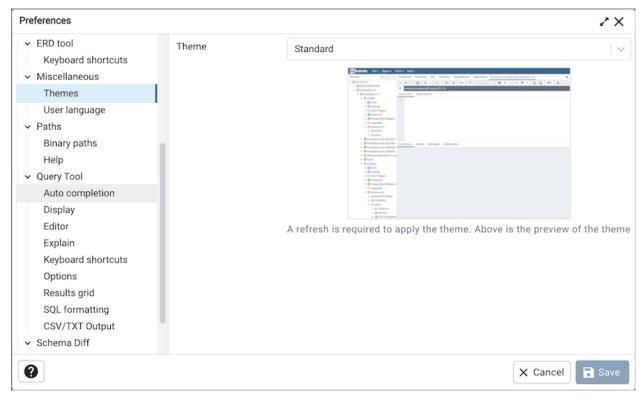
- Use the *Chart line width* field to specify the width of the lines on the line chart.
- When the Show graph data points? switch is set to True, data points will be visible on graph lines.
- When the *Show mouse hover tooltip?* switch is set to *True*, a tooltip will appear on mouse hover on the graph lines giving the data point details.
- When the *Use different data point styles?* switch is set to *True*, data points will be visible in a different style on each graph lines.

#### 1.17.6 The Miscellaneous Node

Expand the Miscellaneous node to specify miscellaneous display preferences.

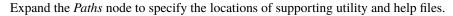


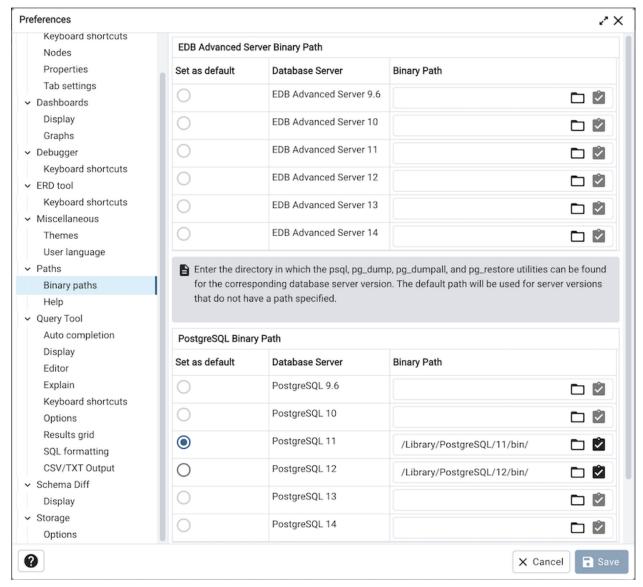
• Use the *User language* drop-down listbox to select the display language for the client.



• Use the *Themes* drop-down listbox to select the theme for pgAdmin. You'll also get a preview just below the drop down. Note that, to apply the theme you need to refresh the pgAdmin page. You can also submit your own themes, check here how. Currently we support Standard, Dark and High Contrast theme.

#### 1.17.7 The Paths Node

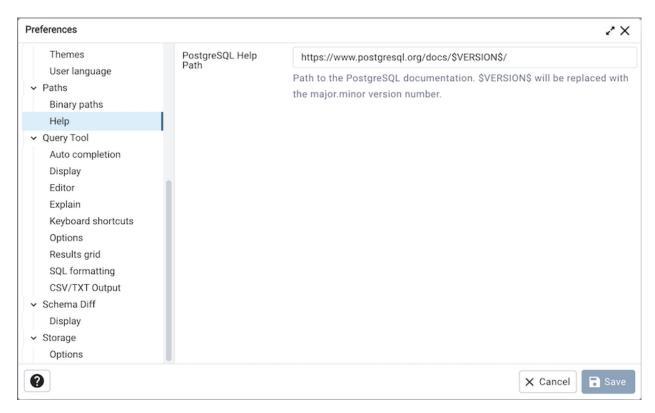




Use the fields on the *Binary paths* panel to specify the path to the directory that contains the utility programs (pg\_dump, pg\_dumpall, pg\_restore and psql) for monitored databases:

- Use the *EDB Advanced Server Binary Path* grid to specify the location of the EDB Postgres Advanced Server utility programs based on the server version. If the respective path is not set, then pgAdmin will pick up the path for which 'Set as default' is checked else pgAdmin will attempt to find the utilities in standard locations used by EnterpriseDB.
- Use the *PostgreSQL Binary Path* grid to specify the location of the PostgreSQL utility programs based on the server version. If the respective path is not set, then pgAdmin will pick up the path for which 'Set as default' is checked else pgAdmin will attempt to find the utilities in standard locations used by PostgreSQL.

**Note:** Use the 'Validate path' button to check the existence of the utility programs (pg\_dump, pg\_dumpall, pg\_restore and psql) and there respective versions.



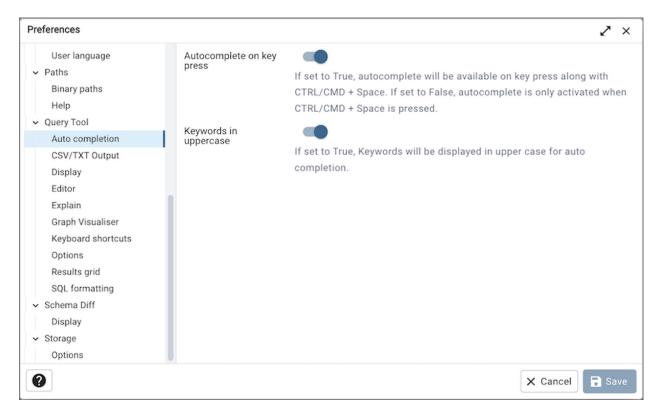
Use the fields on the *Help* panel to specify the location of help files.

- Use the EDB Advanced Server Help Path to specify the path to EDB Postgres Advanced Server documentation.
- Use the *PostgreSQL Help Path* 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.

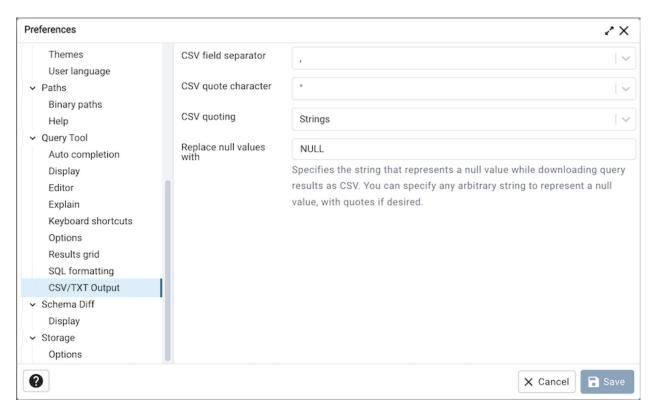
# 1.17.8 The Query Tool Node

Expand the Query Tool node to access panels that allow you to specify your preferences for the Query Editor tool.



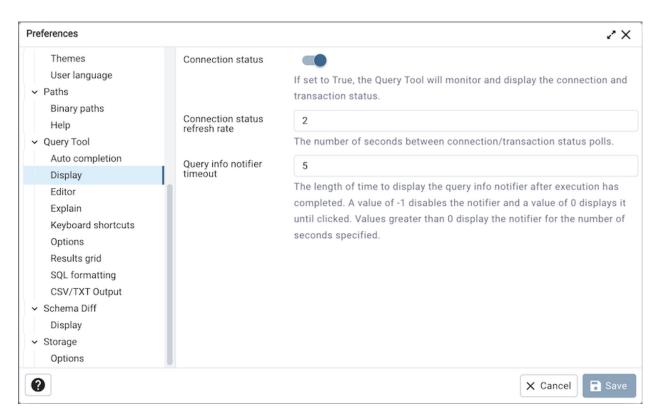
Use the fields on the Auto Completion panel to set the auto completion options.

- When the *Autocomplete on key press* switch is set to *True* then autocomplete will be available on key press along with CTRL/CMD + Space. If it is set to *False* then autocomplete is only activated when CTRL/CMD + Space is pressed.
- When the Keywords in uppercase switch is set to True then keywords are shown in upper case.



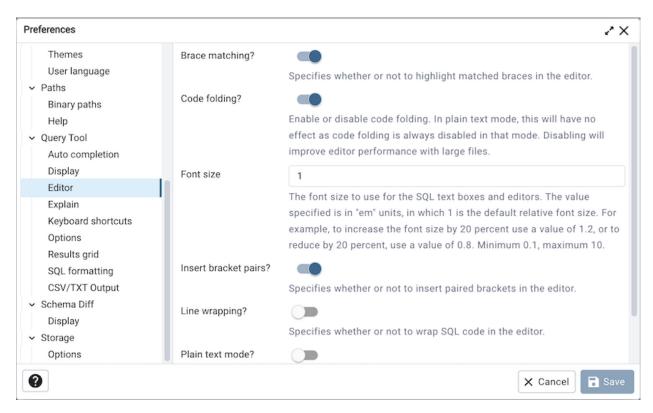
Use the fields on the CSV/TXT Output panel to control the CSV/TXT output.

- 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'.



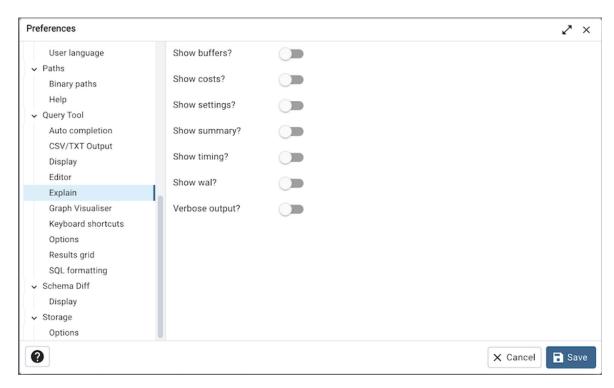
Use the fields on the *Display* panel to specify your preferences for the Query Tool display.

- 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.



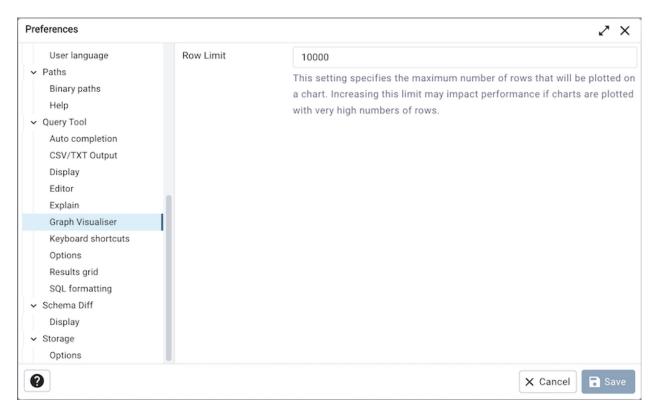
Use the fields on the *Editor* panel to change settings of the query editor.

- 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.



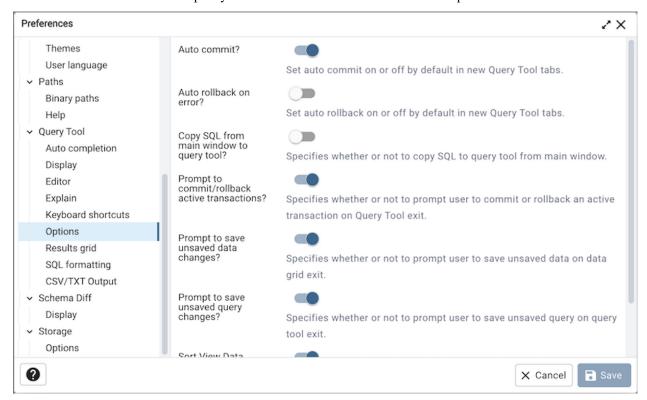
Use the fields on the Explain panel to specify the level of detail included in a graphical EXPLAIN.

- 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 settings?* switch is set to *True*, graphical explain details will include the information on the configuration parameters.
- When the *Show summary?* switch is set to *True*, graphical explain details will include the summary information about the query plan.
- 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 *Show wal?* switch is set to *True*, graphical explain details will include the information on WAL record generation.
- When the *Verbose output?* switch is set to *True*, graphical explain details will include extended information about the query execution plan.



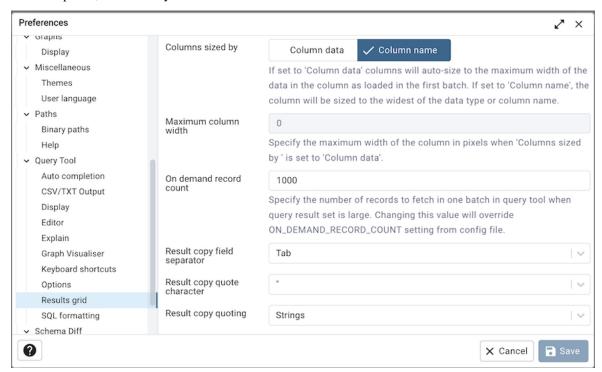
Use the fields on the *Graph Visualiser* panel to specify the settings related to graphs.

• Use the Row Limit field to specify the maximum number of rows that will be plotted on a chart.



Use the fields on the *Options* panel to manage editor preferences.

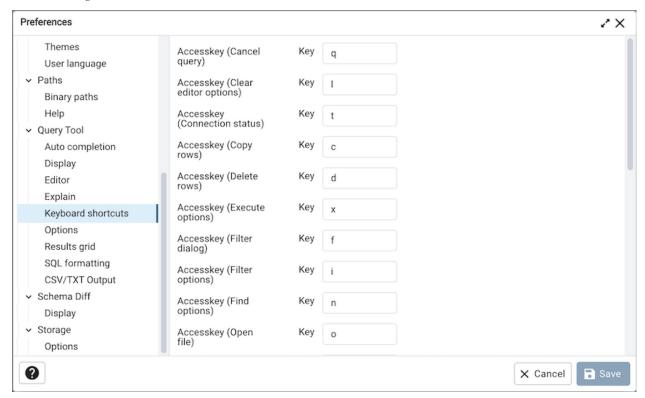
- When the Auto commit? switch is set to True, each successful query is committed after execution.
- When the Auto rollback on error? switch is set to True, failed queries are rolled back.
- When the *Copy SQL from main window to query tool?* switch is set to *True*, the CREATE sql of the selected object will be copied to query tool when query tool will open.
- 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?* If 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.



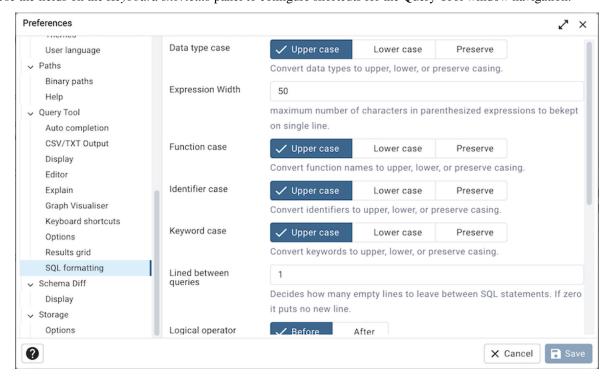
Use the fields on the Results grid panel to specify your formatting preferences for copied data.

- When the *Columns sized by* is set to *Column data*, then data columns will auto-size to the maximum width of the data in the column as loaded in the first batch. If set to *Column name*, the column will be sized to the widest of the data type or column name.
- Specify the maximum width of the column in pixels when 'Columns sized by' is set to *Column data*. If 'Columns sized by' is set to *Column name* then this setting won't have any effect.
- Specify the number of records to fetch in one batch in query tool when query result set is large. Changing this value will override ON\_DEMAND\_ROW\_COUNT setting from config file.
- 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*.



Use the fields on the Keyboard shortcuts panel to configure shortcuts for the Query Tool window navigation:



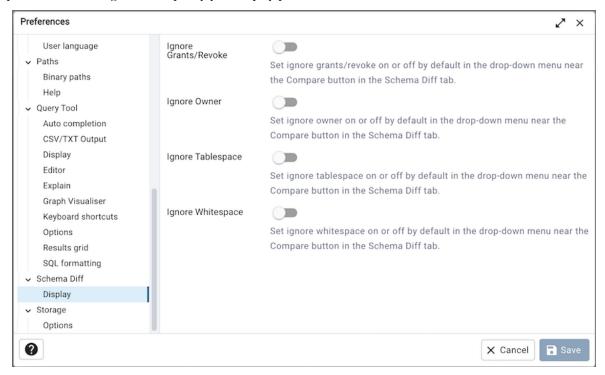
Use the fields on the SQL formatting panel to specify your preferences for reformatting of SQL.

• Use the Data type case option to specify whether to change data types into upper, lower, or preserve case.

- Use the *Expression width* option to specify maximum number of characters in parenthesized expressions to be kept on single line.
- Use the Function case option to specify whether to change function names into upper, lower, or preserve case.
- 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 preserve case.
- Use *Lines between queries* to specify how many empty lines to leave between SQL statements. If set to zero it puts no new line.
- Use Logical operator new line to specify newline placement before or after logical operators (AND, OR, XOR).
- Use New line before semicolon? to specify whether to place query separator (;) on a separate line.
- Use the Spaces around operators? option to specify whether or not to include spaces on either side of operators.
- 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.

#### 1.17.9 The Schema Diff Node

Expand the Schema Diff node to specify your display preferences.



Use the *Ignore Grant/Revoke* switch to ignores the grant and revoke command while comparing the objects.

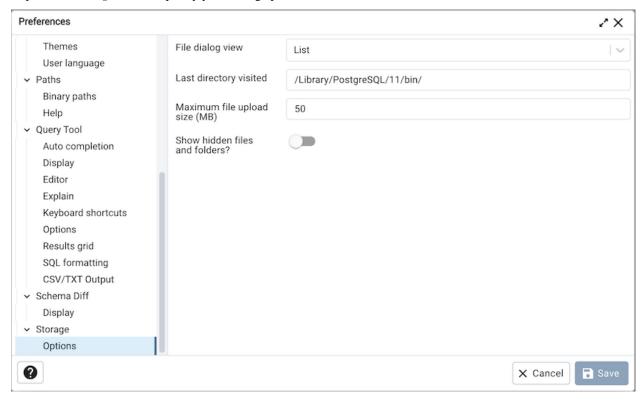
Use the *Ignore Owner* switch to ignores the owner while comparing the objects.

Use the *Ignore Tablespace* switch to ignores the tablespace while comparing the objects.

Use the *Ignore Whitespace* switch to ignores the whitespace while comparing the string objects. Whitespace includes space, tabs, and CRLF.

## 1.17.10 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.

### Using 'setup.py' command line script

**Note:** To manage preferences using setup.py script, you must use the Python interpreter that is normally used to run pgAdmin to ensure that the required Python packages are available. In most packages, this can be found in the Python Virtual Environment that can be found in the installation directory. When using platform-native packages, the system installation of Python may be the one used by pgAdmin.

## 1.17.11 Manage Preferences

#### 1.17.12 Get Preferences

To get all the preferences listed, invoke setup.py with get-prefs command line option. You can also get this mapping by hovering the individual preference in the Preference UI dialog.

```
/path/to/python /path/to/setup.py get-prefs
```

#### 1.17.13 Save Preferences

To save the preferences, invoke setup.py with set-prefs command line option, followed by username, preference\_key=value and auth\_source. Multiple preference can be given too by a space separated. If auth\_source is not given, Internal authentication will be consider by default.

```
/path/to/python /path/to/setup.py set-prefs user1@gmail.com sqleditor:editor:comma_
→first=true

# To specify multiple preferences options
/path/to/python /path/to/setup.py set-prefs user1@gmail.com sqleditor:editor:comma_
→first=true sqleditor:keyboard_shortcuts:toggle_case=true

# to specify an auth_source
/path/to/python /path/to/setup.py set-prefs user1@gmail.com sqleditor:editor:comma_
→first=true --auth-source=ldap
```

# 1.17.14 Updating Preferences preferences.json

To update preferences defined in json file, simply run setup.py `` with ``set-prefs command followed by username and JSON file containing the preferences.

```
/path/to/python /path/to/setup.py set-prefs user1@gmail.com --input-file /Users/
--yogeshmahajan/Desktop/pref.json
```

#### 1.17.15 **JSON** format

The JSON file simply contains preference\_key=value. Preference key value mapping can be obtained by hovering the individual preference in the Preference UI dialog.

The following is an typical example for preferences.json:

```
{
    "preferences":
    {
        "browser:display:show_system_objects": true,
        "browser:display:show_user_defined_templates": true,
        "browser:display:confirm_on_refresh_close": false,
        "misc:themes:theme": "dark",
```

(continues on next page)

(continued from previous page)

```
}
```

# 1.18 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.

#### 1.18.1 Main Browser Window

When using main browser window, the following keyboard shortcuts are available:

Shortcut for all platforms	Function
Shift+Ctrl+a	Add grid row
Shift+Alt+b	Object explorer
Shift+Alt+n	Create object
Shift+Alt+d	Delete object
Shift+Alt+g	Direct debugging
Shift+Alt+e	Edit object properties
Shift+Alt+f	File main menu
Shift+Alt+h	Help main menu
Shift+Alt+o	Object main menu
Shift+Alt+c	Open context menu
Shift+Alt+q	Open query tool
Shift+Ctrl+f	Quick Search
F5	Refresh object explorer
Shift+Alt+s	Search objects
Shift+Alt+[	Tabbed panel backward
Shift+Alt+]	Tabbed panel forward
Shift+Alt+l	Tools main menu
Shift+Alt+v	View data

# 1.18.2 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

### 1.18.3 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	
Ctrl + l	Cmd + 1	Go 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	

## 1.18.4 Query Tool

When using the Query Tool, the following shortcuts are available:

Shortcut (Windows/Linux)	Shortcut (Mac)	Function
<accesskey> + q</accesskey>	<accesskey> + q</accesskey>	Cancel query
<accesskey> + t</accesskey>	<accesskey> + t</accesskey>	Connection status
<accesskey> + d</accesskey>	<accesskey> + d</accesskey>	Delete rows
<accesskey> + x</accesskey>	<accesskey> + x</accesskey>	Execute options
<accesskey> + f</accesskey>	<accesskey> + f</accesskey>	Filter dialog
<accesskey> + i</accesskey>	<accesskey> + i</accesskey>	Filter options
<accesskey> + n</accesskey>	<accesskey> + n</accesskey>	Find options
<accesskey> + o</accesskey>	<accesskey> + o</accesskey>	Open file
<accesskey> + p</accesskey>	<accesskey> + p</accesskey>	Paste rows
<accesskey> + r</accesskey>	<accesskey> + r</accesskey>	Rows limit
<accesskey> + s</accesskey>	<accesskey> + s</accesskey>	Save file
Ctrl + Alt + L	Ctrl + option + L	Clear query
Shift + Ctrl + m	Shift + Ctrl + m	Commit
F8	F8	Download Results
Shift + F7	Shift + F7	EXPLAIN ANALYZE query
F7	F7	EXPLAIN query
F5	F5	Execute query
Shift + Alt + ]	Shift + option + ]	Next tab

continues on next page

Table 13 – continued from previous page

Shortcut (Windows/Linux)	Shortcut (Mac)	Function
Shift + Alt + [	Shift + option + [	Previous tab
Shift + Ctrl + r	Shift + Ctrl + r	Rollback
F6	F6	Save data changes
Shift + Alt + Tab	Shift + option +Tab	Switch Panel
Shift + Ctrl + u	Shift + Ctrl + u	Toggle case of selected text

## 1.18.5 Debugger

When using the Debugger, the following shortcuts are available:

Shortcut (Windows/Linux)	Shortcut (Mac)	Function
<accesskey> + x</accesskey>	<accesskey> + x</accesskey>	Clear all breakpoints
<accesskey> + c</accesskey>	<accesskey> + c</accesskey>	Continue/Start
<accesskey> + i</accesskey>	<accesskey> + i</accesskey>	Step into
<accesskey> + o</accesskey>	<accesskey> + o</accesskey>	Step over
<accesskey> + s</accesskey>	<accesskey> + s</accesskey>	Stop
<accesskey> + t</accesskey>	<accesskey> + t</accesskey>	Toggle breakpoint
Shift + Alt + q	Shift + option + q	Edit grid values
Shift + Alt + ]	Shift + option + ]	Next tab
Shift + Alt + [	Shift + option + ]	Previous tab
Shift + Alt + Tab	Shift + option +Tab	Switch Panel

### 1.18.6 ERD Tool

When using the ERD Tool, the following shortcuts are available:

Shortcut (Windows/Linux)	Shortcut (Mac)	Function
Ctrl + Alt + a	Ctrl + option + a	Add table
Ctrl + Alt + n	Ctrl + option + n	Add/Edit note
Ctrl + Alt + l	Ctrl + option + l	Auto align
Ctrl + Alt + c	Ctrl + option + c	Clone table
Ctrl + Alt + i	Ctrl + option + i	Download image
Ctrl + Alt + d	Ctrl + option + d	Drop table
Ctrl + Alt + e	Ctrl + option + e	Edit table
Ctrl + Alt + s	Ctrl + option + s	Generate SQL
Ctrl + Alt + m	Ctrl + option + m	Many to many link
Ctrl + Alt + o	Ctrl + option + o	One to many link
Ctrl + o	Ctrl + o	Open project
Ctrl + s	Ctrl + s	Save project
Shift + Ctrl + s	Shift + Ctrl + s	Save project as
Ctrl + Alt + t	Ctrl + option + t	Show more/fewer details
Shift + Alt + +	Shift + option + +	Zoom in
Shift + Alt + -	Shift + option + -	Zoom out
Shift + Alt + f	Shift + option + f	Zoom to fit

#### 1.18.7 Inner Tab and Panel Navigation

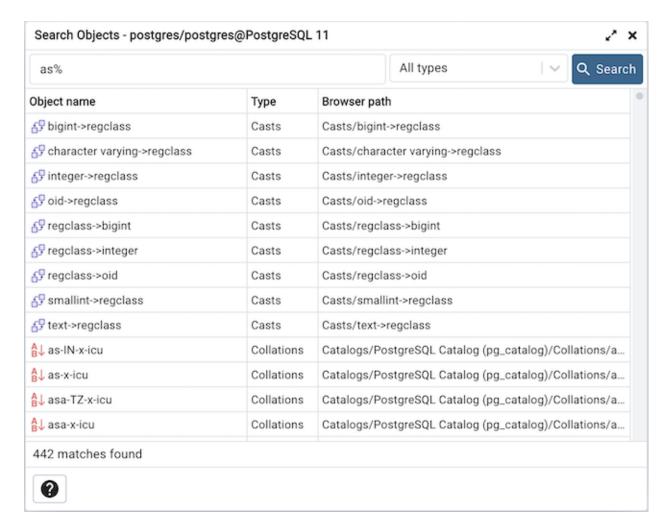
When using the Query Tool and Debugger, the following shortcuts are available for inner panel navigation:

### 1.18.8 Access Key

<accesskey> is browser and platform dependant. The following table lists the default access keys for supported browsers.

	Windows	Linux	Mac
Edge	Alt		
Chrome	Alt	Alt	Ctrl + Option
Firefox	Alt + Shift	Alt + Shift	Ctrl + Option
Safari	Alt		Ctrl + Option

# 1.19 Search objects



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%c, %%%, etc.

The result is presented in the grid with object name, object type and the object path in the *object explorer*. You can double click on a result row to select the object in the *object explorer*. 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 their 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*.

Before using pgAdmin to manage objects that reside on a server, you must define a connection to the server; for more information please see *Connecting to a Server* in the next section.

# External database for pgAdmin user settings

The user settings used by pgAdmin are stored in a SQLite database. In this database, many settings are stored, such as preferences, user accounts, auto-discovered servers, and many more.

As SQLite is a file-based database and it can be anywhere in the file system, so it is not designed to take care of failures (no HA support). Furthermore, it isn't designed to handle multiple connections concurrently reading/writing data to it. Example: In environments such as Kubernetes it may be useful to use an alternate backend to avoid using SQLite on non-ephemeral storage and to allow HA of the settings database.

In order to prevent this, pgAdmin now supports storing user settings in an external database using the new 'CON-FIG\_DATABASE\_URI' parameter in the *config.py* file.

#### 2.1 Use SQLite Database

In order to use SQLite Database, make sure CONFIG\_DATABASE\_URI parameter is set to an empty string like ''. By default it is set to an empty string in the config.py so if you would like to use SQLite database then no need to change anything.

#### 2.2 Use External Database

In order to use an external database, make sure CONFIG\_DATABASE\_URI parameter is set like "dialect+driver://username:password@host:port/database".

**Note** It is recommended to create the database in advance.

# 2.3 Use PostgreSQL Database

Following are the formats to use PostgreSQL as an external database.

Basic syntax:

postgresql://username:password@host:port/database

Using specific schema (It is recommended to create the schema in advance):

postgresql://username:password@host:port/database?options=-csearch\_path=<schema name>

Using default pgpass path for the service account:

postgresql://username@host:port?options=-csearch\_path=<schema name>

Specifying pgpass file path:

# Connecting To A Server

Before you can use the pgAdmin client to manage the objects that reside on your Postgres server, you must define a connection to the server. You can (optionally) use the *Server Group* dialog to create server groups to organize the server connections within the tree control for easier management. To open the *Server Group* dialog, right-click on the *Servers* node of the tree control, and select *Server Group* from the *Create* menu.

# 3.1 Server Group Dialog

Use the *Server Group* dialog to add a new server group. Assign servers to server groups to simplify management of multiple servers. Server groups are displayed as part of the *pgAdmin* tree control.



Use the *Name* field on the *Server Group* dialog to specify a name that will identify the server group in the *pgAdmin* tree control.

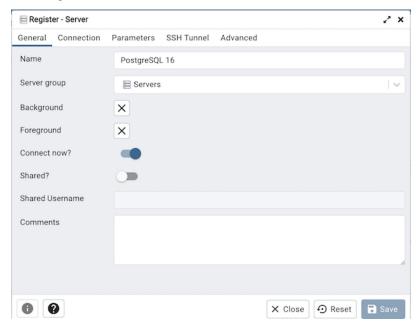
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

To create server connections in a server group, right click on the named server group and select the *Create* option to open the *Create* - *Server* dialog.

Use the fields on the *Server* dialog to define the connection properties for each new server that you wish to manage with pgAdmin. To open the *Server* dialog, right-click on the *Servers* node of the tree control, and select *Server* from the *Register* menu.

## 3.2 Server Dialog

Use the *Server* dialog to describe a connection to a server. Note: you must ensure that the pg\_hba.conf file of the server from which you are connecting allows connections from the host of the client.



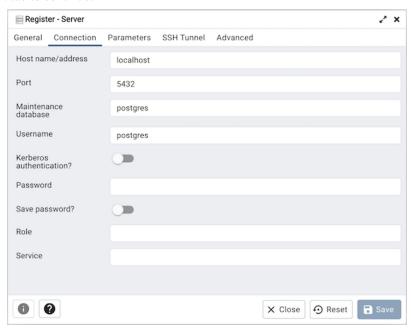
Use the fields in the *General* tab to identify the server:

- Use the *Name* field to add a descriptive name for the server; the name specified will be displayed in the *Object Explorer*.
- Use the drop-down list box in the *Server group* field to select the parent node for the server; the server will be displayed in the *Object Explorer* control within the specified group.
- Use the color-picker in the *Background* field to specify the background color for the server.
- Use the color-picker in the *Foreground* field to specify the foreground color for the server.
- If the *Connect now?* checkbox is checked, the client will attempt a connection to the server upon completion of the dialog; this is the default
- If the *Shared*? switch is moved to *Yes* then that server can be shared with all the other users. This option is available only to admin users. For more information on users see *User Management Dialog*. Users can access the shared servers with some restrictions the following operations on shared servers are not permitted:
  - Delete the server
  - Rename the server
  - Rename the group server
  - Change of host, port, and maintenance database

Please note that once the server is shared, it's icon is changed in the object explorer.

- Use the *Shared Username* field to fill the username of the shared server connection. By default, it will take the username of the server being shared.
- Provide a comment about the server in the *Comments* field.

Click the *Connection* tab to continue.

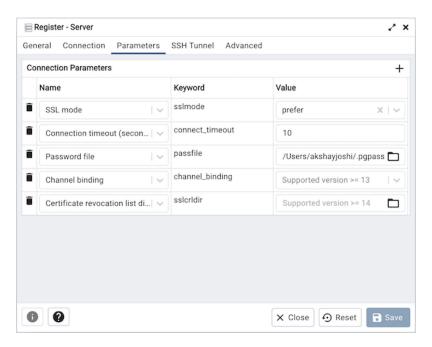


Use the fields in the Connection tab to configure a connection:

- Specify the IP address of the server host, or the fully qualified domain name in the *Host name/address* field. If you provide a unix domain socket, the directory name must begin with a "/".
- Enter the listener port number of the server host in the *Port* field. The default is 5432.
- Use the *Maintenance database* field to specify the name of the initial database to which the client will connect. If you will be using pgAgent or adminpack objects, the pgAgent schema and adminpack objects should be installed on that database.
- Use the *Username* field to specify the name of a role that will be used when authenticating with the server.
- When *Kerberos authentication?* is set to *True*, pgAdmin will try to connect the PostgreSQL server using Kerberos authentication.
- Use the Password field to provide a password that will be supplied when authenticating with the server.
- Check the box next to *Save password?* to instruct pgAdmin to save the password for future use. Use *Clear Saved Password* to remove the saved password.
- Use the *Role* field to specify the name of a role that has privileges that will be conveyed to the client after authentication with the server. This selection allows you to connect as one role, and then assume the permissions of this specified role after the connection is established. Note that the connecting role must be a member of the role specified.
- Use the Service field to specify the service name. For more information, see Section 33.16 of the Postgres
  documentation.

Click the Parameters tab to continue.

3.2. Server Dialog 109



Use the fields in the *Parameters* tab to configure a connection:

Click on the + button to add a new parameter. Some of the parameters are:

- Host address using this field to specify the host IP address may save time by avoiding a DNS lookup on connection, but it may be useful to specify both a host name and address when using Kerberos, GSSAPI, or SSPI authentication methods, as well as for verify-full SSL certificate verification.
- Password File field to specify the location of a password file (.pgpass). A .pgpass file allows a user to login
  without providing a password when they connect. For more information, see Section 33.15 of the Postgres
  documentation.
- Connection timeout field to specify the maximum wait for connection, in seconds. Zero or not specified means wait indefinitely. It is not recommended to use a timeout of less than 2 seconds. By default it is set to 10 seconds.
- *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.

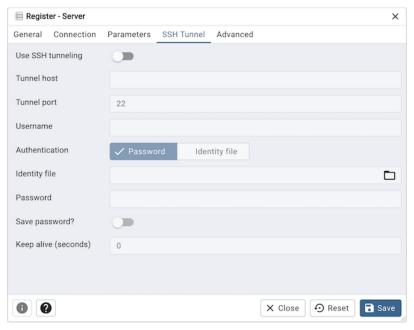
If pgAdmin is installed in Server mode (the default mode), you can use the platform-specific File manager dialog to upload files that support SSL encryption to the server. To access the File manager dialog, click the icon that is located to the right of each of the following fields.

- Client certificate field to specify the file containing the client SSL certificate. This file will replace the default ~/.postgresql/postgresql.crt if pgAdmin is installed in Desktop mode, and <STOR-AGE\_DIR>/<USERNAME>/.postgresql/postgresql.crt if pgAdmin is installed in Web mode. This parameter is ignored if an SSL connection is not made.
- 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 pgAdmin is installed in Desktop mode, and <STORAGE\_DIR>/<USERNAME>/.postgresql/postgresql.key if pgAdmin is installed in Web mode. This parameter
  is ignored if an SSL connection is not made.
- *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.
- 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.

• *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:** In Server mode, 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 pgAdmin 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.

Click the SSH Tunnel tab to continue.



Use the fields in the SSH Tunnel tab to configure SSH Tunneling:

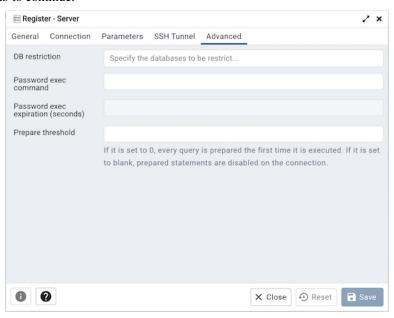
You can use the "SSH Tunnel" tab to connect pgAdmin (through an intermediary proxy host) to a server that resides on a network to which the client may not be able to connect directly.

- Set "Use SSH tunneling" to Yes to specify that pgAdmin 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 the *Password* option to specify that pgAdmin will use a password for authentication to the SSH host.
     This is the default.
  - Select the *Identity file* to specify that pgAdmin 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 of the user name or an identity file if being used, use the *Password* field to specify the password.

3.2. Server Dialog 111

- Check the box next to Save password? to instruct pgAdmin to save the password for future use. Use Clear SSH
  Tunnel Password to remove the saved password.
- Use the *Keep alive* field to specify interval in seconds defining the period in which, if no data was sent over the connection, a 'keepalive' packet will be sent (and ignored by the remote host). This can be useful to keep connections alive over a NAT. You can set to 0 for disable keepalive.

Click the Advanced tab to continue.



Use the fields in the Advanced tab to configure a connection:

- Use the *DB restriction* field to provide a SQL restriction that will be used against the pg\_database table to limit the databases that you see. For example, you might enter: *live\_db test\_db* so that only live\_db and test\_db are shown in the pgAdmin browser. Separate entries with a comma or tab as you type.
- Use the *Password exec command* field to specify a shell command to be executed to retrieve a password to be used for SQL authentication. The stdout of the command will be used as the SQL password. This may be useful when the password should be generated as a transient authorization token instead of providing a password when connecting in PAM authentication scenarios.
- Use the *Password exec expiration* field to specify a maximum age, in seconds, of the password generated with a *Password exec command*. If not specified, the password will not expire until your pgAdmin session does. Zero means the command will be executed for each new connection or reconnection that is made. If the generated password is not valid indefinitely, set this value to slightly before it will expire.
- Use the *Prepare threshold* field to specify the number of times a query is executed before it is prepared. If it is set to 0, every query is prepared the first time it is executed. If it is set to blank, prepared statements are disabled on the connection. This is particularly useful with external connection poolers, such as PgBouncer, which is not compatible with prepared statements. Set this to blank in such cases.

**Note:** The password file option is only supported when pgAdmin is using libpq v10.0 or later to connect to the server.

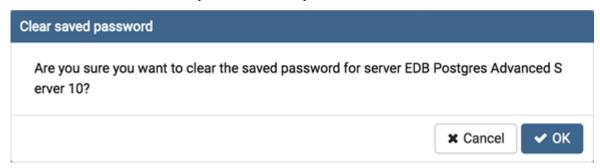
Note: The Password exec option is only supported when pgAdmin is run in desktop mode.

• Click the Save button to save your work.

- Click the *Close* button to exit without saving your work.
- Click the *Reset* button to return the values specified on the Server dialog to their original condition.

#### 3.2.1 Clear Saved Passwords

Use Clear Saved Password functionality to clear the saved password for the database server.



Clear Saved Password shows in the context menu for the selected server as well as under the Object menu on the top menu bar.

Use Clear SSH Tunnel Password functionality to clear the saved password of SSH Tunnel to connect to the database server.



*Clear SSH Tunnel Password* shows in the context menu for the selected server as well as under the *Object* menu on the top menu bar.

Note: It will be enabled/visible when the password for the selected database server is already saved.

Alternatively you can use the *Cloud Deployment* wizard to deploy a new PostgreSQL instance in the cloud, and connect to it.

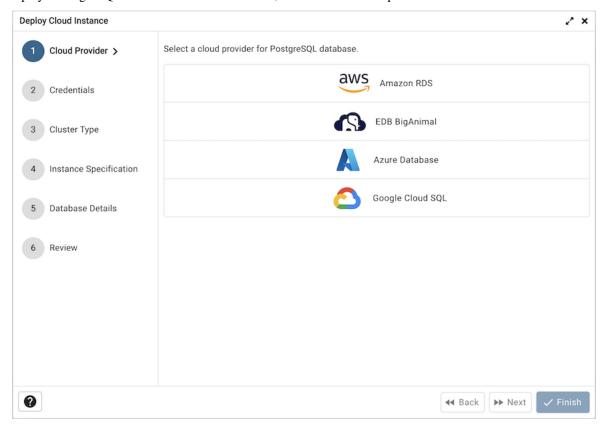
# 3.3 PostgreSQL Cloud Deployment

A PostgreSQL server can be deployed on the Amazon, EDB BigAnimal, Azure, Google cloud using this module. In future more cloud provider options will be available.

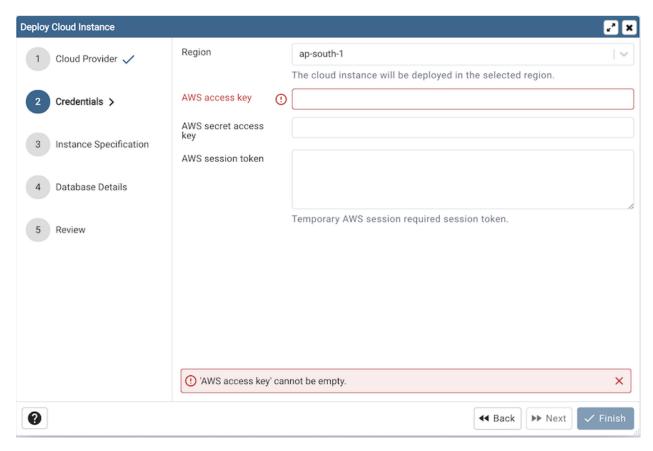
To launch the *Cloud Deployment...* tool, right click on the *Server Group* or *Server* of the tree control, and select *Deploy a Cloud Instance* from the *Register* menu.

## 3.3.1 Amazon RDS Cloud Deployment

To deploy a PostgreSQL server on the Amazon cloud, follow the below steps.

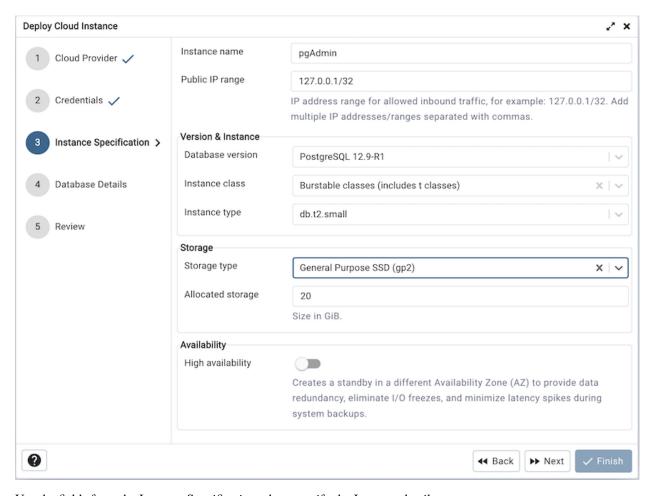


Once you launch the tool, select the Amazon RDS option. Click on the *Next* button to proceed further.



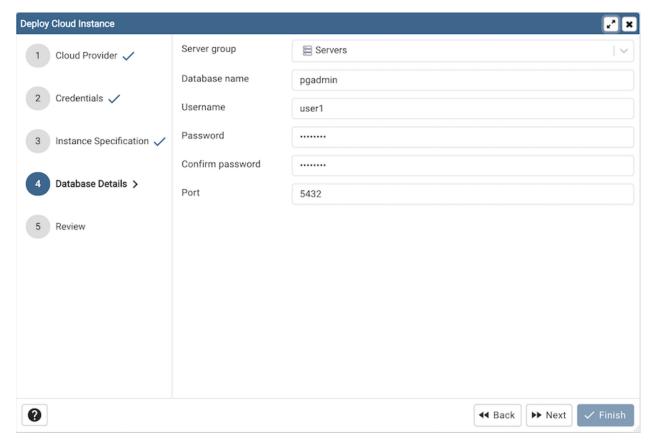
In the Credentials dialog, provide the region in which you want to deploy the instance along with the AWS access key and AWS secret access key. Provide AWS session token only if your AWS session is temporary.

 $To \ proceed \ further, \ click \ on \ the \ next \ button. \ Before \ going \ further, \ pgAdmin \ will \ validate \ your \ credentials.$ 



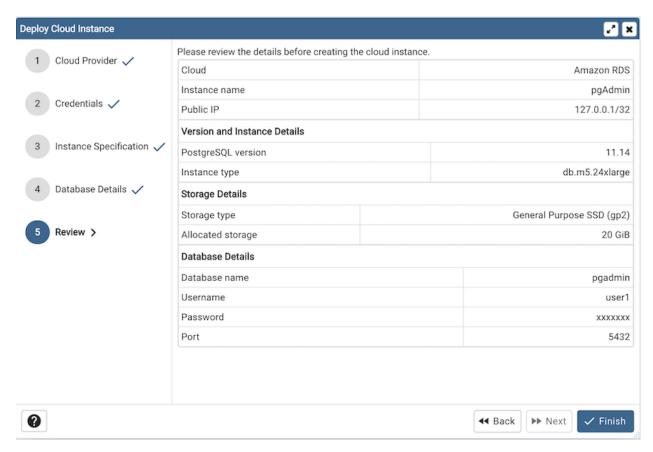
Use the fields from the Instance Specification tab to specify the Instance details.

- Use the *Instance name* field to add an instance name for the PostgreSQL server; the name specified will be displayed in the *Object Explorer* too.
- Use the *Public IP* field to specify the IP Address range for permitting the inbound traffic.
- Use the *Database version* field to specify the PostgreSQL version to deploy.
- Use the *Instance class* field to allocate the computational, network, and memory capacity required by planned workload of this DB instance.
- Use the *Instance type* field to select the instance type.
- Use the *Storage type* field to select the instance storage type. Three options are available. General Purpose (SSD) storage, Provisioned IOPS (SSD) and Magnetic storage.
- Use the *Allocated storage* field to specify the storage capacity in GiB.
- Use the *Provisioned IOPS* in case of Provisioned IOPS (SSD) storage type.
- Use the *High Availability* option to specify High Availability option. This option creates a standby in a different Availability Zone(AZ).

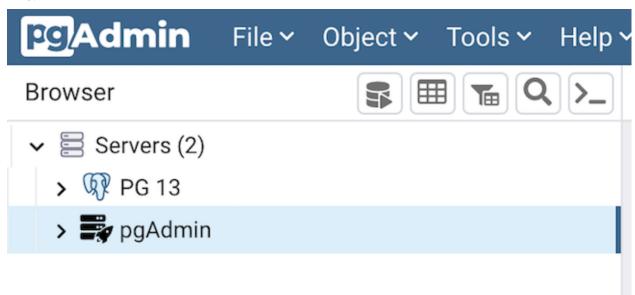


Use the fields from the Database Details tab to specify the Instance details.

- Use the drop-down list box in the *Server group* field to select the parent node for the server; the server will be displayed in the *Object Explorer* within the specified group.
- Use the *Database name* field to add the database name for the PostgreSQL server.
- Use the *Username* field to specify the name of a role that will be used when authenticating with the server.
- Use the *Password* field to provide a password that will be supplied when authenticating with the server.
- Use the *Confirm password* field to repeat the password.
- Enter the listener port number of the server host in the *Port* field.



At the end, review the Instance details that you provided. Click on Finish button to deploy the instance on Amazon RDS.

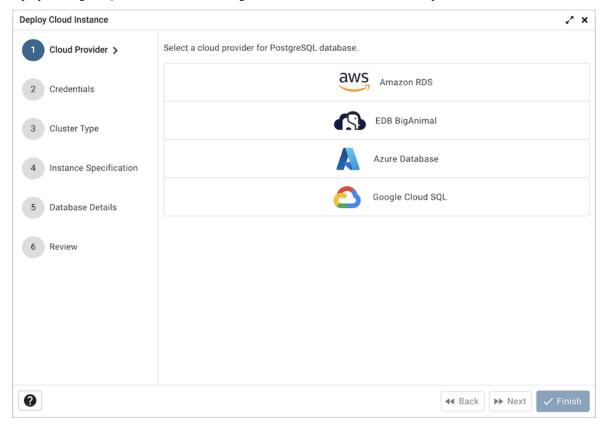


Once you click on the finish, one background process will start which will deploy the instance in the cloud and monitor the progress of the deployment. You can view all the background process with there running status and logs on the *Processes* tab

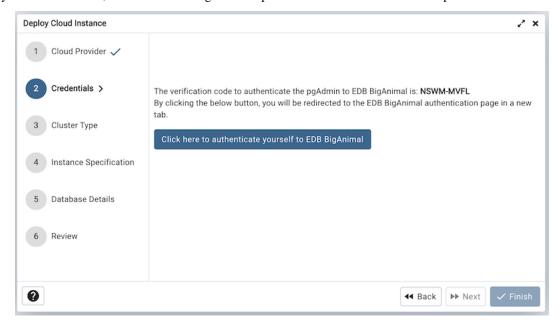
The Server will be added to the tree with the cloud deployment icon. Once the deployment is done, the server details will be updated.

### 3.3.2 EDB BigAnimal Cloud Deployment

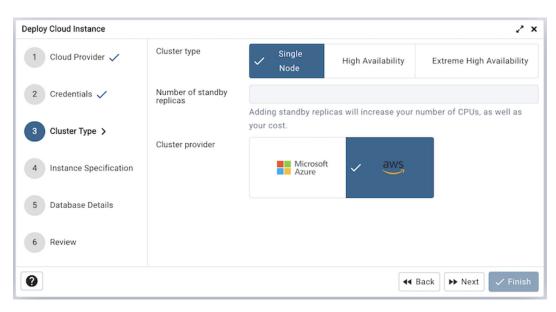
To deploy a PostgreSQL server on the EDB BigAnimal cloud, follow the below steps.



Once you launch the tool, select the EDB BigAnimal option. Click on the Next button to proceed further.



The next steps is to authenticate the user to EDB BigAninal. Click the given button to authenticate, by clicking the button, the user will be redirected to the new tab for the verification. Once you confirm the one time code, the pgAdmin will automatically detect it and the next button will be enabled. To proceed further, click on the next button.



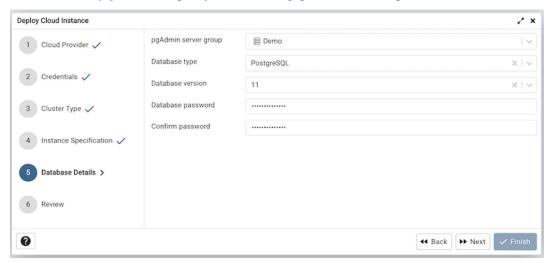
- Use the *Project* field to choose a project in your Biganimal account.
- Use the *Cluster type* field to choose a cluster type.
- Use the No. of Standby Replicas field to specify the replicas if you have selected the High Availability cluster.
- Use the *Cluster provider* field to choose the provider.



Use the fields from the Instance Specification tab to specify the Instance details.

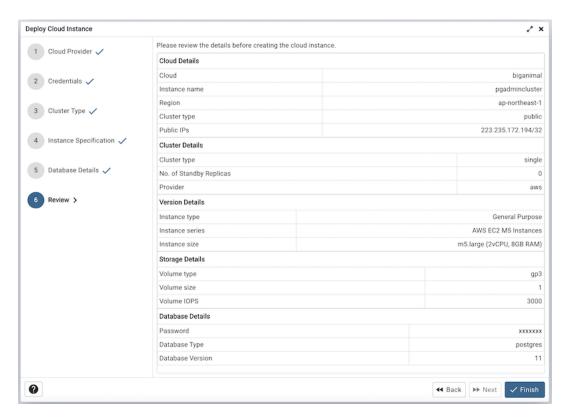
• Use the *Cluster name* field to add a cluster name for the PostgreSQL server; the name specified will be displayed in the *Object Explorer* too.

- Use the *Region* field to select the region.
- Use the Public IP range field to specify the IP Address range for permitting the inbound traffic. Leave it blank for 0.0.0.0/0
- Use the *Instance type* field to select the instance type.
- Use the *Instance series* field to select the instance series.
- Use the *Instance size* field to allocate the computational, network, and memory capacity required by planned workload of this DB instance.
- Use the *Volume type* field to select the instance storage type.
- Use the Volume properties field to specify the storage capacity. This field is specific to Azure.
- Use the *Volume size* field to specify the storage size. This field is specific to AWS.
- Use the *Volume IOPS* field to specify the storage IOPS. This field is specific to AWS.
- Use the *Disk throughput* field to specify the disk throughput. This field is specific to AWS.

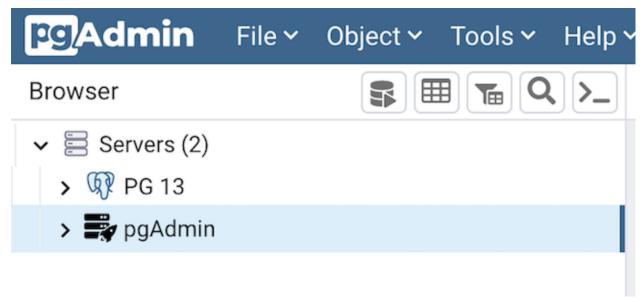


Use the fields from the Database Details tab to specify the Instance details.

- Use the drop-down list box in the *Server group* field to select the parent node for the server; the server will be displayed in the *Object Explorer* within the specified group.
- Use the *Database type* field to specify the PostgreSQL type, EnterpriseDB PostgreSQL Advanced Server or PostgreSQL.
- Use the *PostgreSQL version* field to select the database version.
- Use the *Database Password* field to provide a password that will be supplied when authenticating with the server.
- Use the *Confirm password* field to repeat the password.



At the end, review the Cluster details that you provided. Click on Finish button to deploy the instance on EDB BigAnimal.

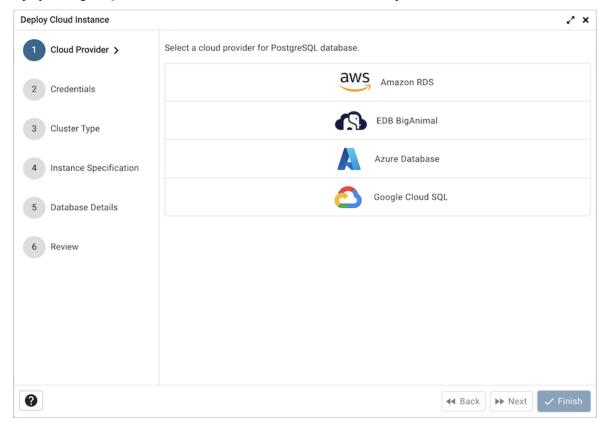


Once you click on the finish, one background process will start which will deploy the instance in the cloud and monitor the progress of the deployment. You can view all the background process with there running status and logs on the *Processes* tab

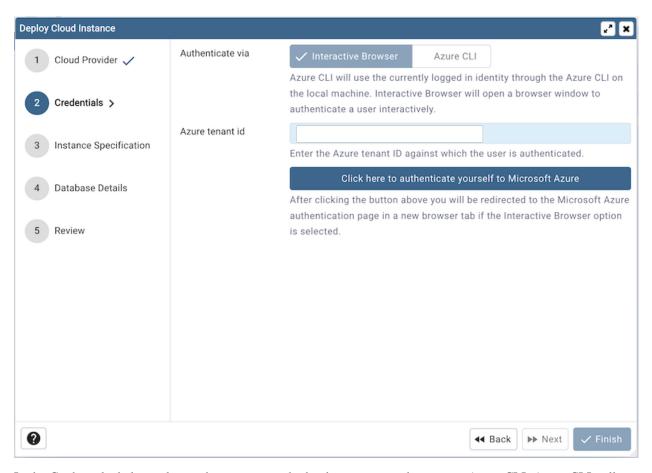
The Server will be added to the tree with the cloud deployment icon. Once the deployment is done, the server details will be updated.

## 3.3.3 Azure Database Cloud Deployment

To deploy a PostgreSQL server on the Azure Database, follow the below steps.



Once you launch the tool, select the Azure Database option. Click on the *Next* button to proceed further.



In the Credentials dialog, select authentication method either interactive browser or Azure CLI. Azure CLI will use the currently logged in identity through the Azure CLI on the local machine. Interactive Browser will open a browser window to authenticate a user interactively.

Use the Azure tenant id to specify Azure tenant ID against which user is authenticated.

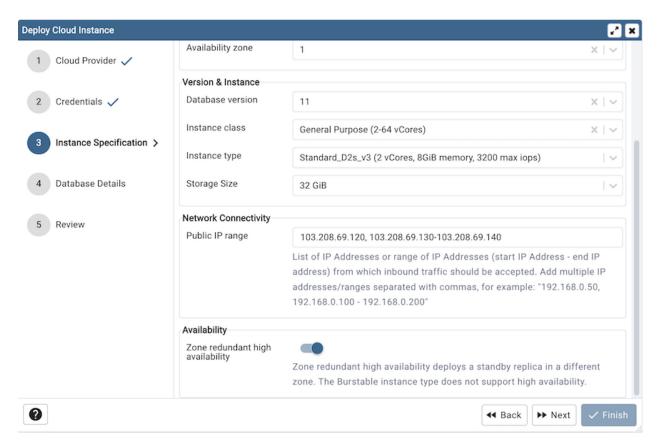
Clicking the *Click here to authenticate yourself to Microsoft Azure* button, user will be redirected to the Microsoft Azure authentication page in a new browser tab if the Interactive Browser option is selected. Azure CLI authentication can be used only in Desktop mode.

Once authentication is completed, click on the next button to proceed.

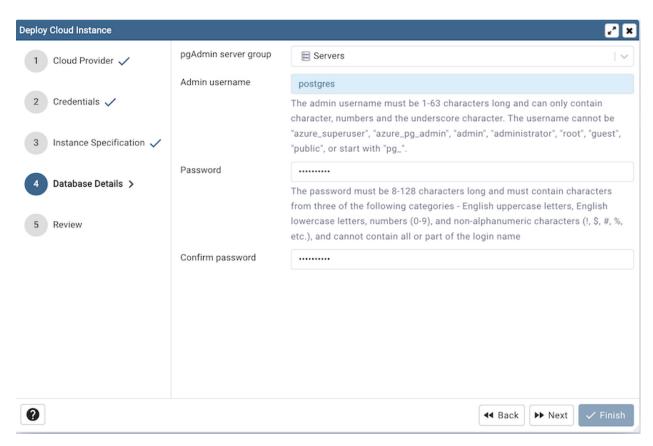


Use the fields from the Instance Specification tab to specify the Instance details.

- Use the *Cluster name* field to add a name for the PostgreSQL server; the name specified will be displayed in the *Object Explorer* too.
- Select a subscription from the *Subscription* options which are populated based on user access levels in Azure portal.
- Select the resource group from Resource Group dropdown under which the PostgreSQL instance will be created.
- Select the location to deploy PostgreSQL instance from *Location* options.
- Select the availability zone in specified region to deploy PostgreSQL instance from Availability zone options.
- Use Database version options to specify PostgreSQL database version.
- Use the *Instance class* field to allocate the computational, network, and memory capacity required by planned workload of this DB instance.
- Use the *Instance type* field to select the instance type.
- Use the *Storage size* option to specify the storage capacity.



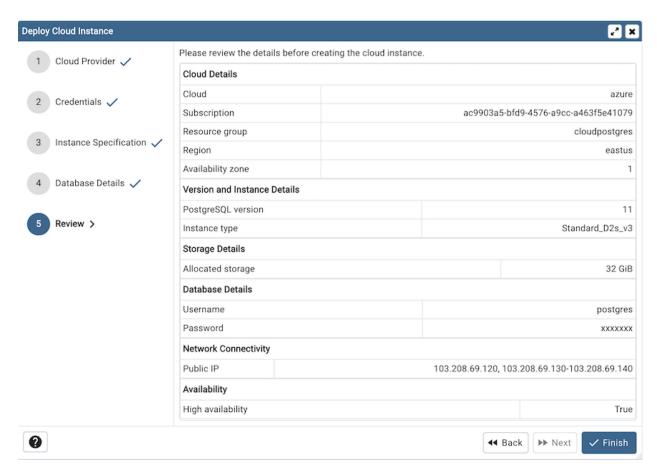
- Use the *Public IP* field to specify the List of IP Addresses or range of IP Addresses (start IP Address end IP address) from which inbound traffic should be accepted. Add multiple IP addresses/ranges separated with commas, for example: "192.168.0.50, 192.168.0.100 192.168.0.200"
- Use *Zone redundant high availability* option to specify High Availability option. Zone redundant high availability deploys a standby replica in a different zone. The Burstable instance type does not support high availability.



Use the fields from the Database Details tab to specify the PostgreSQL database details.

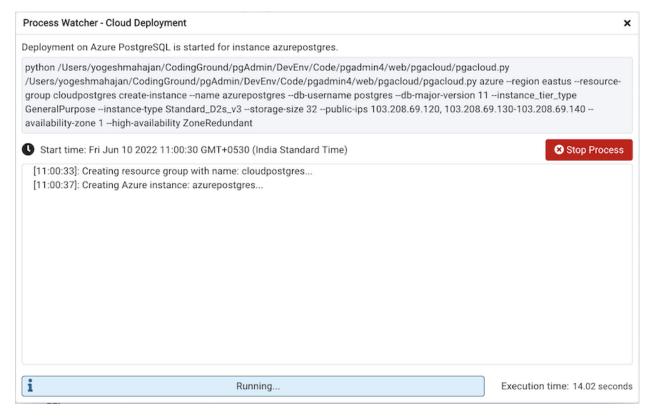
- Use the drop-down list in the *pgAdmin server group* field to select the parent node for the server; the server will be displayed in the *Object Explorer* within the specified group.
- Use the Admin username field to add the database name for the PostgreSQL server.
- Use the *Password* field to provide a password that will be supplied when authenticating with the server.
- Use the *Confirm password* field to repeat the password.

Click on the next button to proceed.

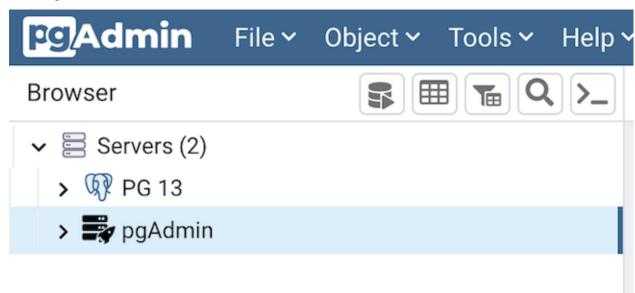


At the end, review the instance details that you provided. Click on Finish button to deploy the instance on Azure Database.

Once you click on the finish, one background process will start which will deploy the instance in the cloud and monitor the progress of the deployment. You can view all the background process with there running status and logs on the *Processes* tab

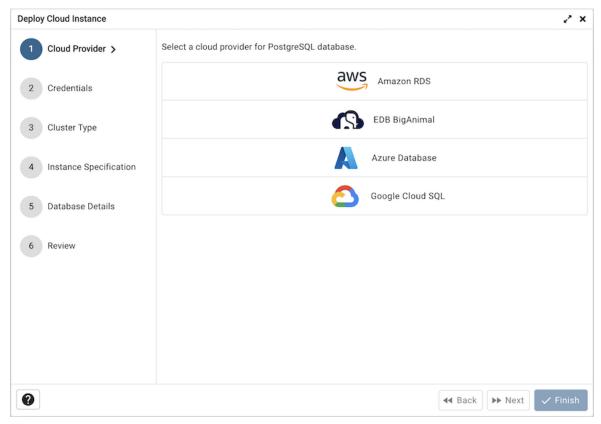


The Server will be added to the tree with the cloud deployment icon. Once the deployment is done, the server details will be updated.

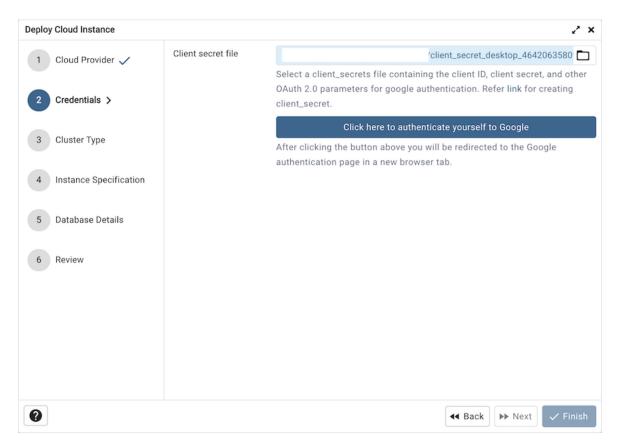


## 3.3.4 Google Cloud SQL Deployment

To deploy a PostgreSQL server on the Google Cloud SQL, follow the below steps.



Once you launch the tool, select the Google Cloud SQL option. Click on the *Next* button to proceed further.

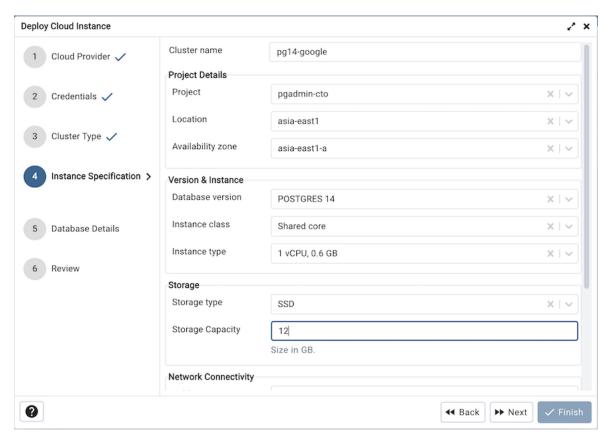


In the Credentials dialog, select client secret file to authenticate using google. You can download a client secret which is json formatted file from google cloud console once OAuth2 client ID is created.

**Note:** While creating client OAuth client ID, select Desktop App as application type. Refer this link for creating client secret.

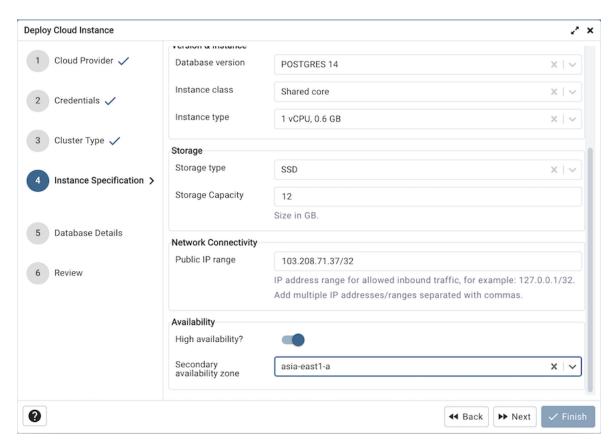
Clicking the *Click here to authenticate yourself to Google* button, user will be redirected to the Google authentication page in a new browser tab.

Once authentication is completed, click on the next button to proceed.

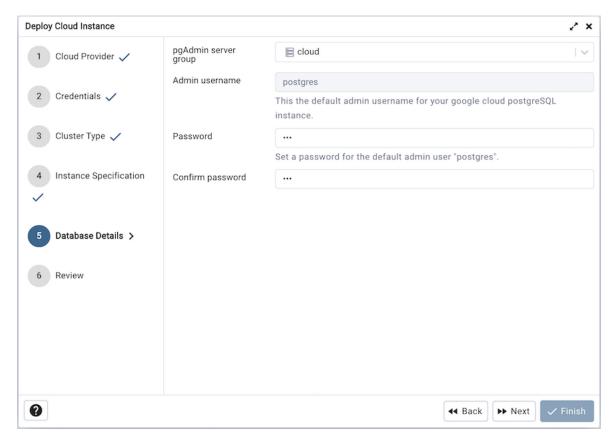


Use the fields from the Instance Specification tab to specify the Instance details.

- Use the *Cluster name* field to add a name for the PostgreSQL server; the name specified will be displayed in the *Object Explorer* too.
- Select the project from *project* dropdown under which the PostgreSQL instance will be created.
- Select the location to deploy PostgreSQL instance from *Location* options.
- Select the availability zone in specified region to deploy PostgreSQL instance from Availability zone options.
- Use *Database version* options to specify PostgreSQL database version.
- Use the *Instance class* field to allocate the computational and memory capacity required by planned workload of this DB instance.
- Use the *Instance type* field to select the instance type.
- Specify storage type by selecting option from *Storage type*.
- Use the Storage capacity option to specify the storage capacity.



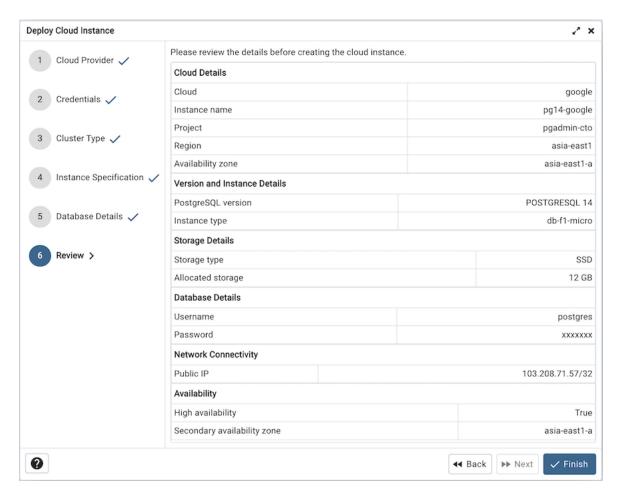
- Use the *Public IP* field to specify the list of IP address range for allowed inbound traffic, for example: 127.0.0.1/32. Add multiple IP addresses/ranges separated with commas.
- Use the *High Availability* option to specify High Availability option. This option creates a standby in a select Secondary Availability Zone.
- Select the secondary availability zone for high availability from Secondary Availability zone options.



Use the fields from the Database Details tab to specify the PostgreSQL database details.

- Use the drop-down list in the *pgAdmin server group* field to select the parent node for the server; the server will be displayed in the *Object Explorer* within the specified group.
- Admin username field will be default to postgres. server.
- Use the *Password* field to provide a password that will be supplied when authenticating with the server.
- Use the *Confirm password* field to repeat the password.

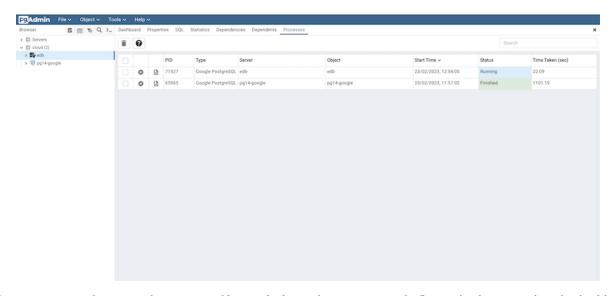
Click on the next button to proceed.



At the end, review the instance details that you provided. Click on Finish button to deploy the instance on Azure PostgreSQL.

Once you click on the finish, one background process will start which will deploy the instance in the cloud and monitor the progress of the deployment. You can view all the background process with there running status and logs on the *Processes* tab

The Server will be added to the tree with the cloud deployment icon. Once the deployment is done, the server details will be updated.



A master password is required to secure and later unlock saved server passwords. It is set by the user and can be disabled using config.

#### 3.4 Master Password

**Note:** pgAdmin 4 uses the operating system password store by default to store the saved server passwords in desktop mode from version 7.2 onwards and Master password will not be required. If the operating system password store is not available then pgAdmin 4 will continue to use a master password as per the configuration settings.

A master password is required to secure and later unlock the saved server passwords. This is applicable for desktop mode and in server mode if authentication source contains OAuth2 or Kerberos or Webserver.

- You are prompted to enter the master password when you open the window for the first time after starting the
  application.
- Once you set the master password, all the existing saved passwords will be re-encrypted using the master password.
- The server passwords which are saved in the SQLite DB file or External Database are encrypted and decrypted using the master password.



**Note:** pgAdmin aims to be **secure by default**, however, you can disable the master password by setting the configuration parameter *MASTER\_PASSWORD\_REQUIRED=False*. See *The config.py File* for more information on config-

uration parameters and how they can be changed or enforced across an organisation.

**Note:** If the master password is disabled, then all the saved passwords will be removed.

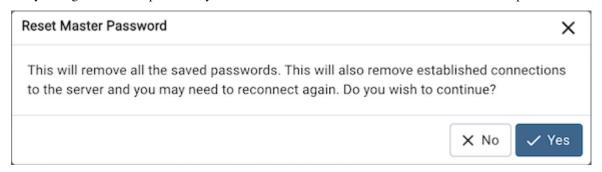
**Warning:** If the master password is disabled, then the saved passwords will be encrypted using a key which is derived from information within the configuration database. Use of a master password ensures that the encryption key does not need to be stored anywhere, and thus prevents possible access to server credentials if the configuration database becomes available to an attacker.

It is strongly recommended that you use the master password if you use the Save Password option.

- The master password is not stored anywhere on the physical storage. It is temporarily stored in the application memory and it does not get saved when the application is restarted.
- You are prompted to enter the master password when pgAdmin server is restarted.



• If you forget the master password, you can use the Reset Master Password button to reset the password.



**Warning:** Resetting the master password will also remove all saved passwords and close all existing established connections.

After defining a server connection, right-click on the server name, and select *Connect to server* to authenticate with the server, and start using pgAdmin to manage objects that reside on the server.

3.4. Master Password 137

### 3.5 Connect to Server

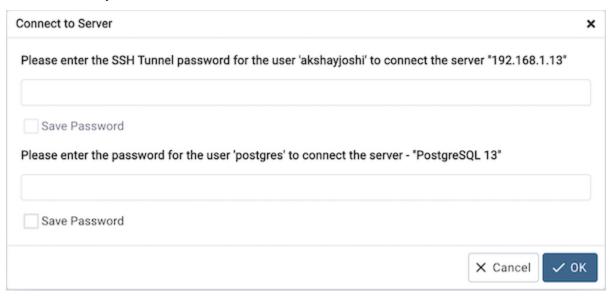
Use the *Connect to Server* dialog to authenticate with a defined server and access the objects stored on the server through the pgAdmin tree control. To access the dialog, right click on the server name in the *pgAdmin* tree control, and select *Connect Server*... from the context menu.



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.

When using SSH Tunneling, the *Connect to Server* dialog will prompt for the SSH Tunnel and Database server passwords if not already saved.



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 respective *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 pgAdmin client 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 the pgAdmin host 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.

### 3.6 Connection Error

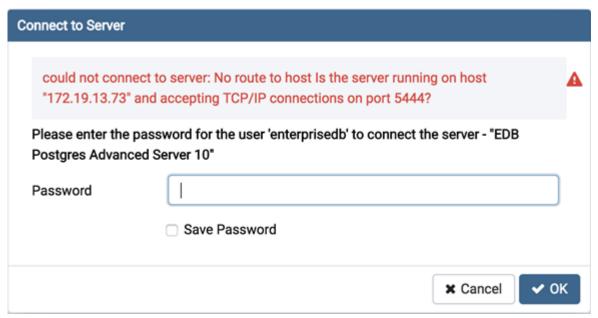
When connecting to a PostgreSQL server, you may get an error message. If you encounter an error message, please review the message carefully; each error message attempts to incorporate the information you'll need to resolve the problem. For more details about specific errors, please locate the error 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:

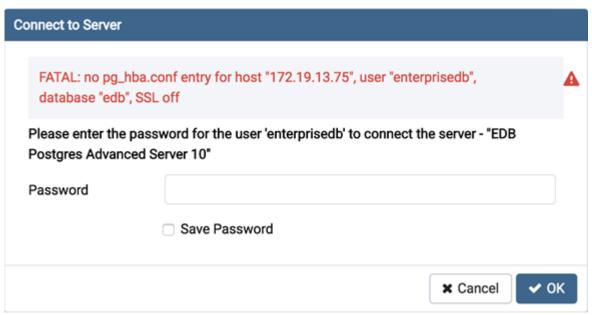
- the database server isn't running simply start it.
- the server isn't configured to accept TCP/IP requests on the address shown.

3.6. Connection Error 139

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 **listen\_addresses='\***; 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



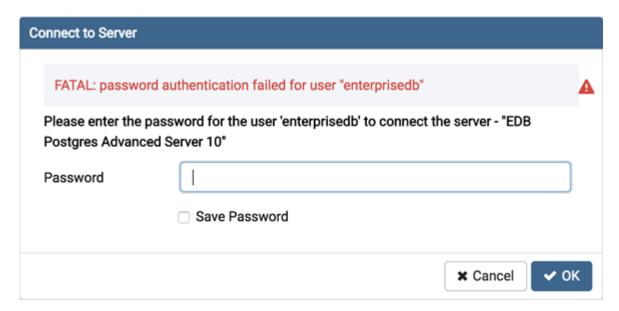
If pgAdmin displays this message when connecting, your server can be contacted correctly over the network, but is not configured to accept your connection. Your client has not been detected as a legal user for the database.

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.

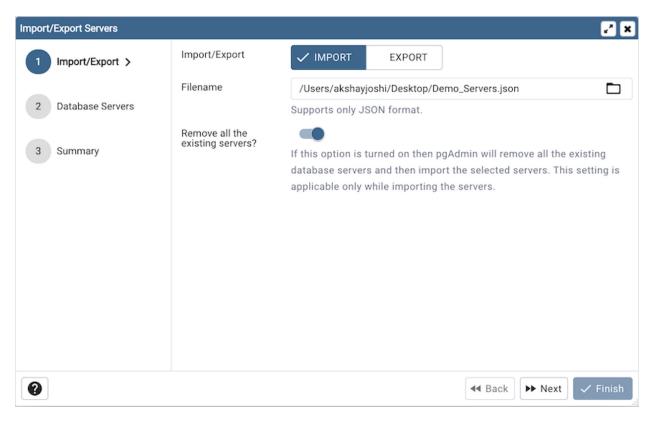
Server definitions (and their groups) can be exported to a JSON file and re-imported to the same or a different system to enable easy pre-configuration of pgAdmin.

### 3.7 Import/Export Servers

Server definitions (and their groups) can be exported to a JSON file and re-imported to the same or a different system to enable easy pre-configuration of pgAdmin.

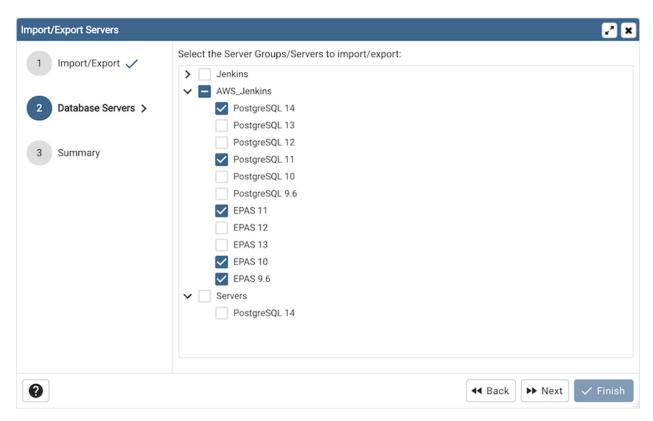
### 3.7.1 Using pgAdmin 4 GUI

To launch the *Import/Export Servers*... tool, navigate through *Tools* on the menu bar to click on the *Import/Export Servers* option.



- Use the *Import/Export* field to select the Server Groups/Servers to be imported or exported.
- Use the *Filename* field to select the JSON file to import servers or create the new file in case of Export where the servers to be exported in the JSON format.
- Use the *Remove all the existing servers?* field to specify whether to remove all the existing servers or not before importing the new selected servers. This field is applicable only in case of Import Servers.

Click the *Next* button to continue, or the *X* button to close the wizard.



• Select the Server Groups/ Servers to be imported/exported.

Click the *Next* button to continue, or the *X* button to close the wizard.



Check the summary of the servers that are going to be imported/exported on the Summary page.

Click the *Finish* button to close the wizard.

### 3.7.2 Using 'setup.py' command line script

**Note:** To export or import servers using setup.py script, you must use the Python interpreter that is normally used to run pgAdmin to ensure that the required Python packages are available. In most packages, this can be found in the Python Virtual Environment that can be found in the installation directory. When using platform-native packages, the system installation of Python may be the one used by pgAdmin.

### **Exporting Servers**

To export the servers defined in an installation, simply invoke setup.py with the dump-servers command line option, followed by the name (and if required, path) to the desired output file. By default, servers owned by the desktop mode user will be dumped (pgadmin4@pgadmin.org by default - see the DESKTOP\_USER setting in config.py). This can be overridden with the --user command line option. There can be multiple configurations of pgAdmin on the same system. To dump the servers from specific pgAdmin config DB file, --sqlite-path option can be used. It is also recommended to use this option when running pgAdmin in desktop mode. By default SQLITE\_PATH setting in config.py is taken. For example:

```
/path/to/python /path/to/setup.py dump-servers output_file.json
# or, to specify a non-default user name and auth source (the default is Internal):
/path/to/python /path/to/setup.py dump-servers output_file.json --user user@example.com -
--auth_source ldap
# to specify a pgAdmin config DB file:
/path/to/python /path/to/setup.py dump-servers output_file.json --sqlite-path /path/to/
--pgadmin4.db
```

To export only certain servers, use the --server option and list one or more server IDs. For example:

```
/path/to/python /path/to/setup.py dump-servers output_file.json --server 1 --server 2 --

⇒server 5
```

#### **Importing Servers**

To import the servers defined in a JSON file, simply invoke setup.py with the load-servers command line option, followed by the name (and if required, path) of the JSON file containing the server definitions. Servers will be owned by the desktop mode user (pgadmin4@pgadmin.org by default - see the DESKTOP\_USER setting in config.py). This can be overridden with the --user command line option. There can be multiple configurations of pgAdmin on the same system. The default behaviour is for the imported servers to be added to the existent list, which might lead to duplicates. This can be overridden with the --replace command line option, which will replace the list of servers with the newly imported one. To load the servers into a specific pgAdmin config DB file, --sqlite-path option can be used. It is also recommended to use this option when running pgAdmin in desktop mode. By default SQLITE\_PATH setting in config.py is taken. For example:

```
/path/to/python /path/to/setup.py load-servers input_file.json

# or, to replace the list of servers with the newly imported one:

/path/to/python /path/to/setup.py load-servers input_file.json --replace

# or, to specify a non-default user name and auth source (the default is Internal) to______own the new servers:

/path/to/python /path/to/setup.py load-servers input_file.json --user user@example.com

# to specify a pgAdmin config DB file:

/path/to/python /path/to/setup.py load-servers input_file.json --sqlite-path /path/to/_____pgadmin4.db
```

If any Servers are defined with a Server Group that is not already present in the configuration database, the required Group will be created.

#### **JSON format**

The JSON file format used when importing or exporting servers is quite straightforward and simply contains a list of servers, with a number of attributes. The following attributes are required to be present in every server definition: Name, Group, Port, Username, SSLMode, MaintenanceDB and one of Host, HostAddr or Service.

Password fields cannot be imported or exported.

The following example shows both a minimally defined and a fully defined server:

```
{
    "Servers": {
       "1": {
            "Name": "Minimally Defined Server",
            "Group": "Server Group 1",
            "Port": 5432.
            "Username": "postgres",
            "Host": "localhost",
            "SSLMode": "prefer",
            "MaintenanceDB": "postgres"
       },
        "2": {
            "Name": "Fully Defined Server",
            "Group": "Server Group 2",
            "Host": "host.domain.com".
            "HostAddr": "192.168.1.2".
            "Port": 5432,
            "MaintenanceDB": "postgres",
            "Username": "postgres",
            "Role": "my_role_name".
            "SSLMode": "require",
            "Comment": "This server has every option configured in the JSON",
            "DBRestriction": "live_db test_db",
            "PassFile": "/path/to/pgpassfile",
```

(continues on next page)

(continued from previous page)

```
"SSLCert": "/path/to/sslcert.crt",
            "SSLKey": "/path/to/sslcert.key",
            "SSLRootCert": "/path/to/sslroot.crt",
            "SSLCrl": "/path/to/sslcrl.crl",
            "SSLCompression": 1,
            "Shared": false,
            "SharedUsername": "postgres",
            "BGColor": "#ff9900",
            "FGColor": "#000000",
            "Service": "postgresql-10",
            "Timeout": 60,
            "UseSSHTunnel": 1,
            "TunnelHost": "192.168.1.253",
            "TunnelPort": 22,
            "TunnelUsername": "username",
            "TunnelAuthentication": 0,
            "PasswordExecCommand": "echo 'test'",
            "PasswordExecExpiration": 100
        }
    }
}
```

# CHAPTER 4

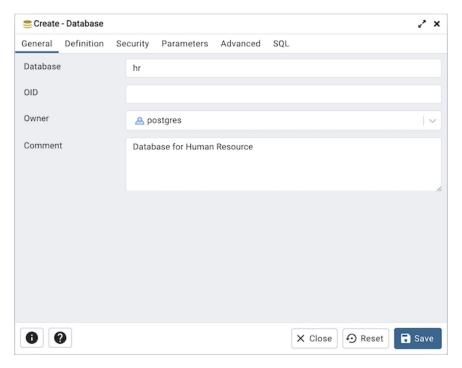
# Managing Cluster Objects

Some object definitions reside at the cluster level; pgAdmin 4 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 pgAdmin 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*...

### 4.1 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.

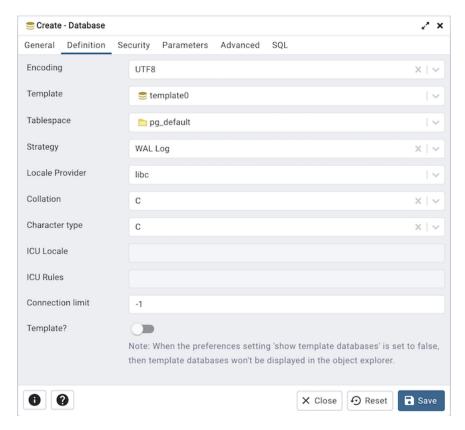
The *Database* dialog organizes the development of a database through the following dialog tabs: *General*, *Definition*, *Security*, and *Parameters*. The *SQL* tab displays the SQL code generated by dialog selections.



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 *pgAdmin* tree control.
- Use the *OID* field to specify the object identifier to be used for the new database. Users can specify the value greater than 16383. This option is available from v15 and above.
- Select the owner of the database from the drop-down listbox in the *Owner* field.
- Store notes about the database in the Comment field.

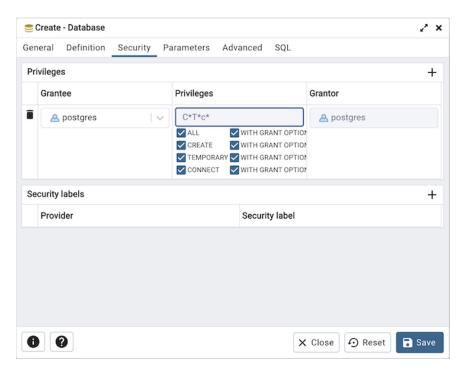
Click the Definition tab to continue.



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 strategy from the drop-down listbox in the *Strategy* field while creating a new database. This option is available from v15 and above.
- Select the locale provider from the drop-down listbox in the *Locale Provider* field to set the default collation for this database. Possible values are: icu, libc. This option is available from v15 and above.
- 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.
- Select the icu locale from the drop-down listbox in the *ICU Locale* to specifies the ICU locale ID if the ICU locale provider is used. This option is available from v15 and above.
- Specify the icu rules in the *ICU Rules* field as additional collation rules to customize the behavior of the default collation of this database. This option is available from v16 and above.
- 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.
- If the *templates?* is set to true, then database will be a template 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:

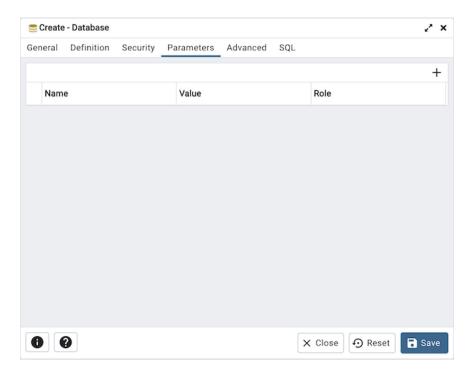
- Select the name of the role from the drop-down listbox in the *Grantee* field.
- Click inside the *Privileges* field. Check the boxes to the left of one or more privileges to grant the selected privilege to the specified user.
- The current user, who is the default grantor for granting the privilege, 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 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.

### 4.1.1 Example

The following is an example of the sql command generated by user selections in the *Database* dialog:

```
Create - Database
General Definition Security Parameters
                                      Advanced
    CREATE DATABASE hr
        WITH
        OWNER = postgres
3
4
        TEMPLATE = template0
5
        ENCODING = 'UTF8'
        STRATEGY = 'wal_log'
        LC_COLLATE = 'C'
        LC_CTYPE = 'C'
        LOCALE_PROVIDER = 'libc'
9
10
        TABLESPACE = pg_default
        CONNECTION LIMIT = -1
11
12
        IS_TEMPLATE = False;
13
14
    COMMENT ON DATABASE hr
        IS 'Database for Human Resource';
15
16
   GRANT ALL ON DATABASE hr TO postgres WITH GRANT OPTION;
17
 0
      0
                                                                        3 Save
                                                     X Close
                                                              Reset
```

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.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

## 4.2 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.

### 4.2.1 Example

The following is an example of the sql command generated by selections made in the Resource Group dialog:

```
General SQL

CREATE RESOURCE GROUP my_grp;

-- Following query will be executed in a separate transaction ALTER RESOURCE GROUP my_grp

SET cpu_rate_limit = 2, dirty_rate_limit = 6144;

X Close Paset Save
```

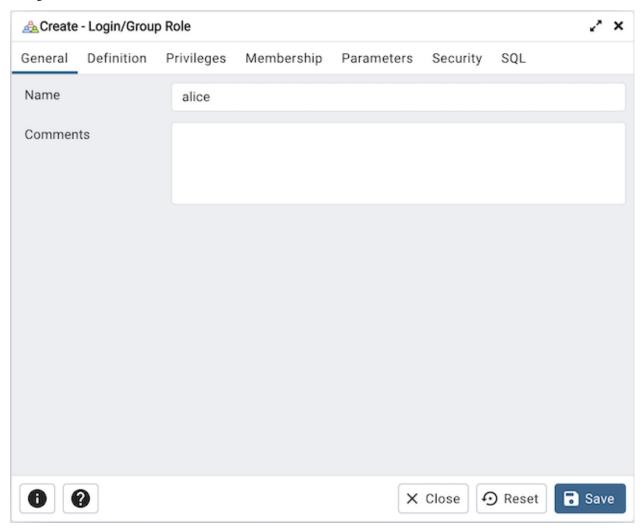
The example creates a resource group named acctg that sets cpu\_rate\_limit to 2, and dirty\_rate\_limit to 6144.

- 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 *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

# 4.3 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.

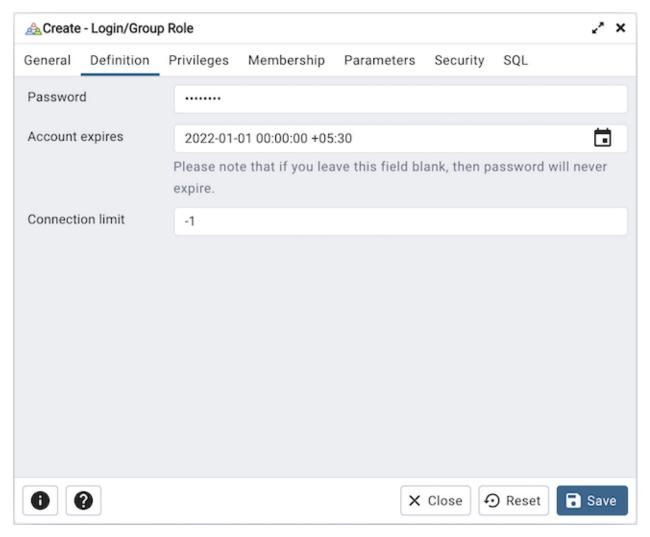
The Login/Group Role dialog organizes the creation and management of roles through the following dialog tabs: General, Definition, Privileges, Memberships Parameters, and Security. The SQL tab displays the SQL code generated by dialog selections.



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.



Use the *Definition* tab to set a password and configure connection rules:

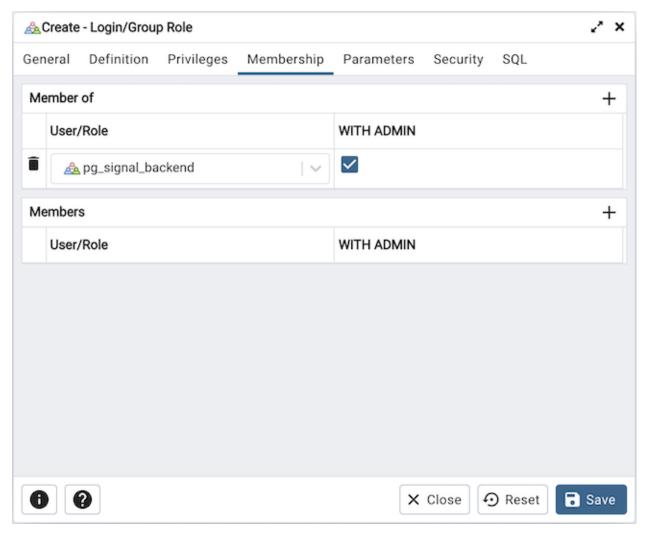
- Provide a password that will be associated with the role in the *Password* field.
- Provide an expiration date for the password in the *Account Expires* field (the role does not expire). The expiration date is not enforced when a user logs in with a non-password-based authentication method.
- If the 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.



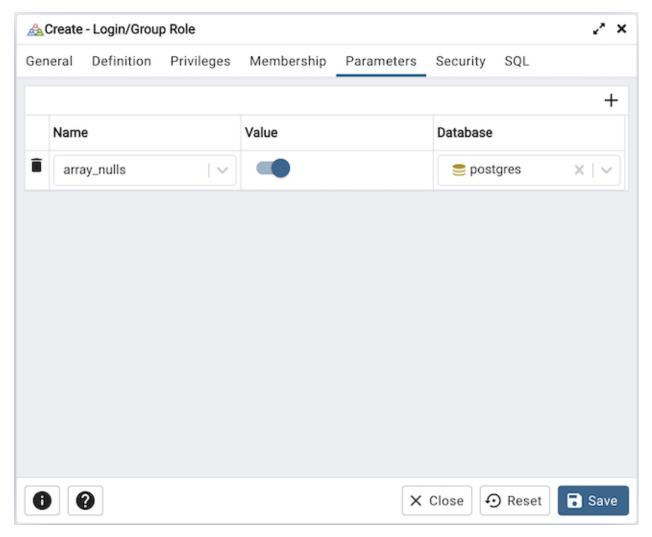
Use the *Privileges* tab to grant privileges to the role.

- 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*.
- 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*.
- Move the *Bypass RLS?* switch to the *Yes* position to control whether a role can bypasses every row-level security (RLS) policy. The default value is *No*.



• Specify member of the role in the *Member of* field and specify the members in the *Member* field. 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.

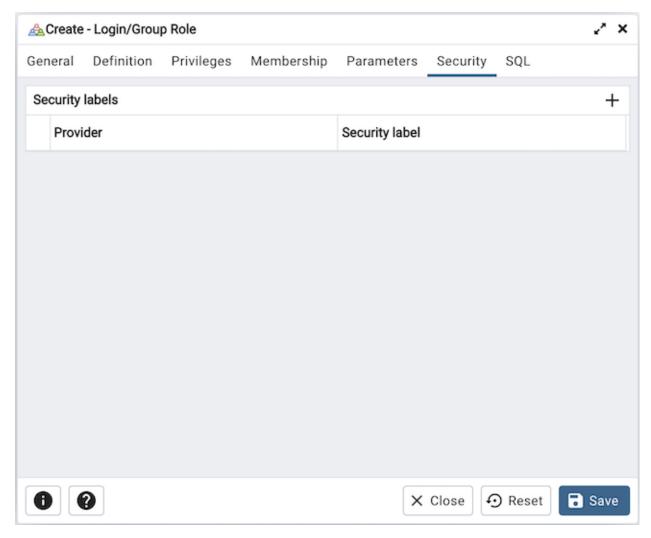


Use the fields on the *Parameters* tab to set session defaults for a selected configuration parameter when the role is connected to a specified database. This tab invokes the ALTER ROLE... SET configuration\_parameter syntax. Click the *Add* icon (+) to assign a value for a parameter.

- 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.

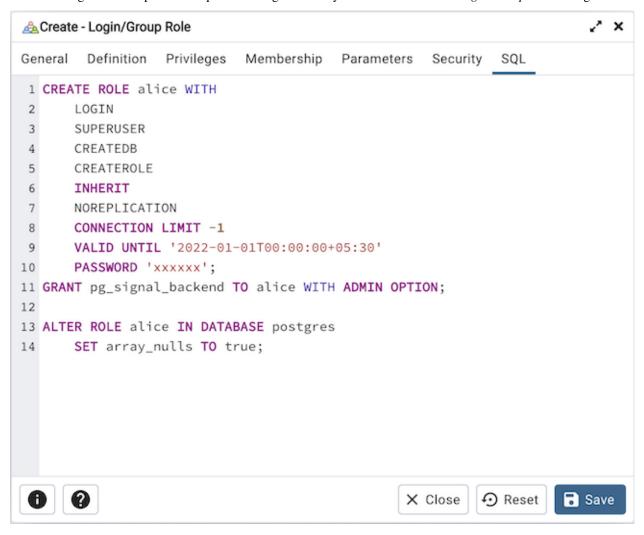
- Specify a security label provider in the *Provider* field. The named provider must be loaded and must consent to the proposed labeling operation.
- Specify a a security label in the *Security Label* field. The meaning of a given label is at the discretion of the label provider. PostgreSQL places no 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.

### 4.3.1 Example

The following is an example of the sql command generated by user selections in the Login/Group Role dialog:



The example creates a login role named *alice* with  $pg\_signal\_backend$  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 *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

## 4.4 Tablespace Dialog

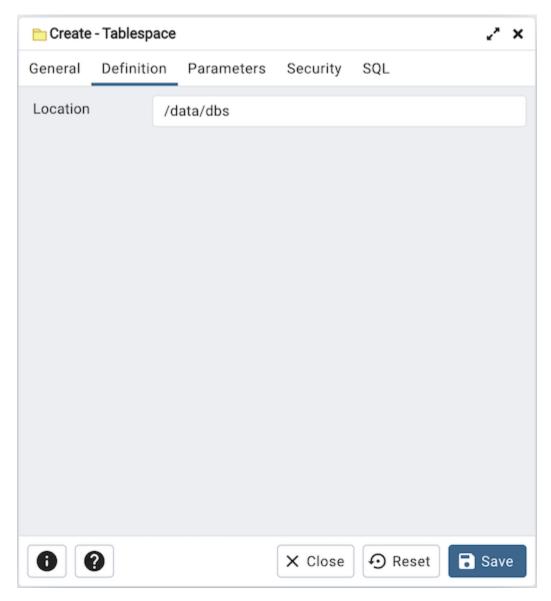
Use The *Tablespace* dialog to define a tablespace. A tablespace allows superusers to define an alternative location on the file system where the data files containing database objects (such as tables and indexes) reside. Tablespaces are only supported on systems that support symbolic links. Note that 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.

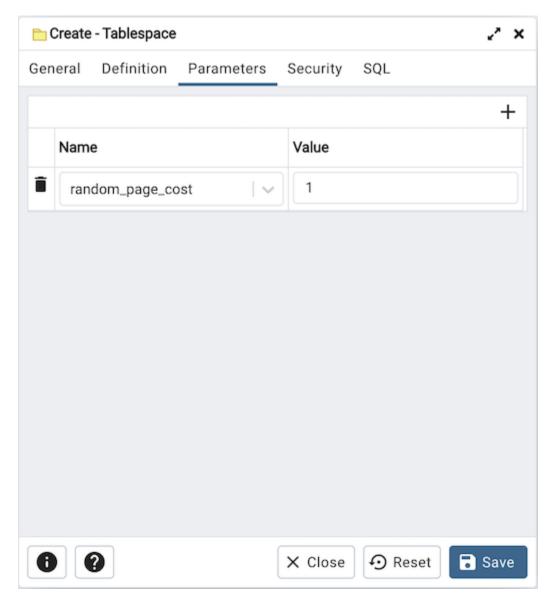


- Use the *Name* field to identify the tablespace with a descriptive name. The name cannot begin with pg\_; these names are reserved for system tablespaces.
- Select the owner of the tablespace from the drop-down listbox in the *Owner* field.
- Store notes about the tablespace in the *Comment* field.

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.

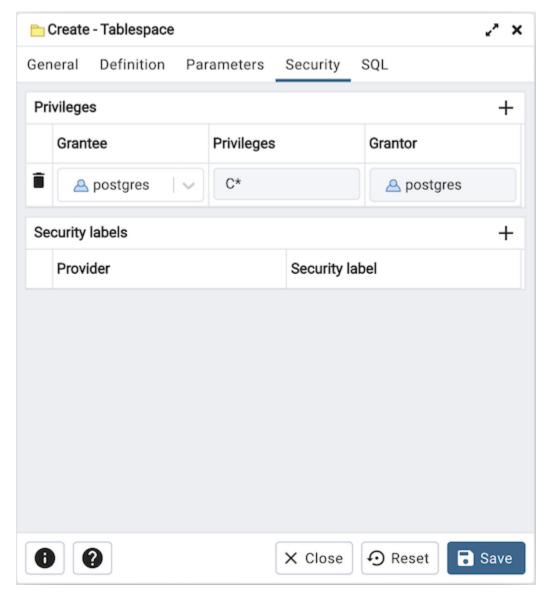


Use the *Parameters* tab to set parameters for the tablespace. Click the *Add* icon (+) to add a row to the table below.

- 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 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.

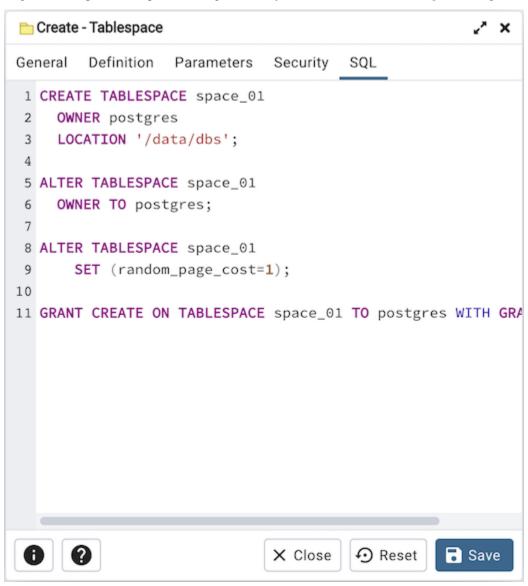
To discard a security label, click the trash icon to the left of the row and confirm deletion in the *Delete Row* popup.

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.

### 4.4.1 Example

The following is an example of the sql command generated by user selections in the *Tablespace* dialog:



The example shown demonstrates creating a tablespace named *space\_01*. It has a *random\_page\_cost* value equal to 1.

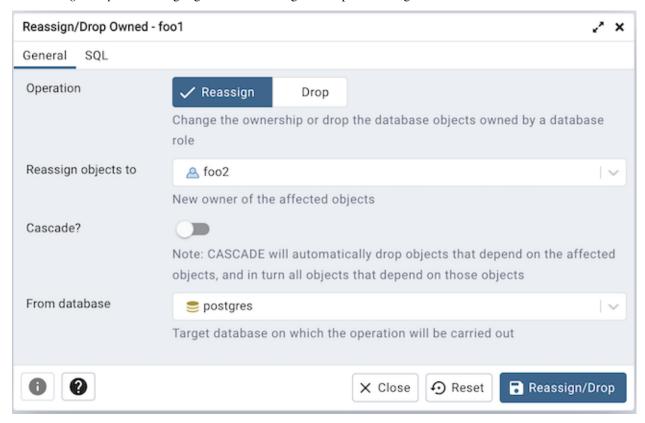
- Click the *Info* button (i) to access online help.
- Click the Save button to save work.

- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

### 4.5 Role Reassign/Drop Own Dialog

Use the *Reassign/Drop Own* dialog to change the ownership of database objects owned by a database role. This dialog instructs the system to change the ownership of database objects owned by any of the *old\_roles* to *new\_role*.

The Reassign/Drop Own dialog organizes the Reassign & Drop role through General tab.



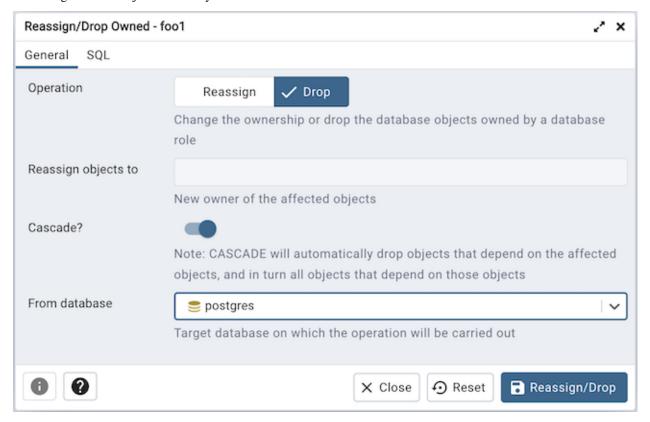
- Use the *Operation* field to provide Reassign option.
- Provide a new role in the *Reassign Objects to* field; The ownership of all the objects within the selected database, and of all shared objects (databases, tablespaces), owned by the *old\_role* will be reassigned to *new\_role*.
- Provide a database on which the reassignment is to be carried out.

The above example demonstrates reassigning *old\_role* to *new\_role*.

Click the SQL tab to continue.



Removing database objects owned by a database role.



• Use the *Operation* field to provide Drop option.

- Use the Cascade? field to provide Yes, No is default.
- Provide a database on which the drop of objects is to be carried out.

Click the SQL tab to continue.



The above examples demonstrates drop owned by role.

- Click the *Help* button (?) to access online help.
- Click the *OK* button to save work.
- Click the Cancel button to exit without saving work.

# CHAPTER 5

### Managing Database Objects

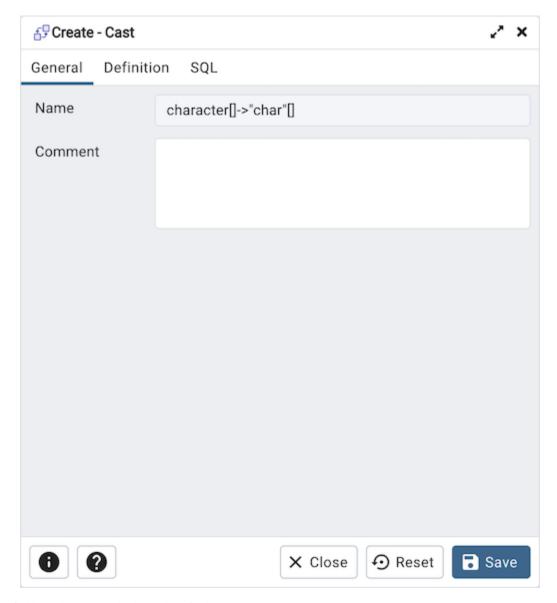
pgAdmin 4 provides simple but powerful dialogs that you can use to design and create database objects. Each dialog contains a series of tabs that you use to describe the object that will be created by the dialog; the SQL tab displays the SQL command that the server will execute when creating the object.

To access a dialog that allows you to create a database object, right-click on the object type in the pgAdmin 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.*..

# 5.1 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 *pgAdmin* 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.
- Select *Context* toggle to *Implicit* 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:

5.1. Cast Dialog 173

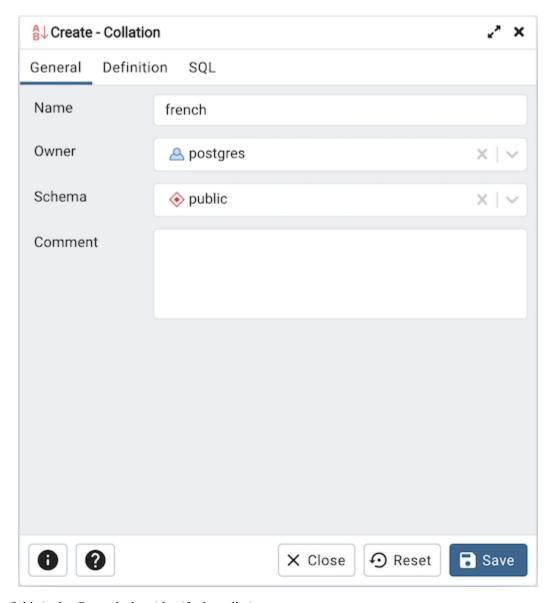


- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

# **5.2 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.

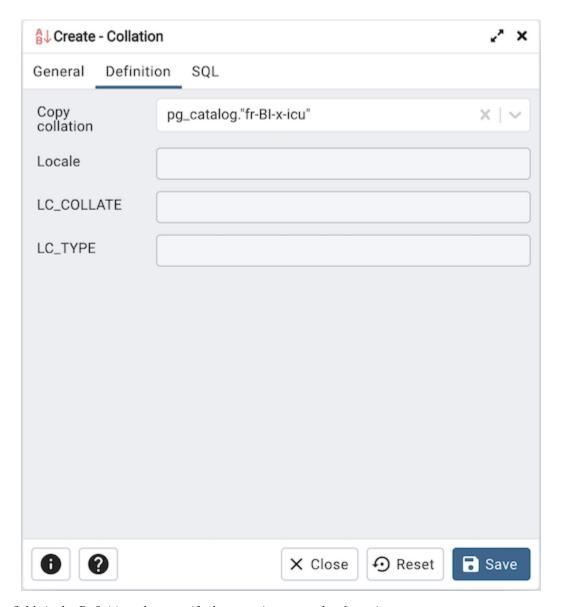


Use the fields in the *General* tab to identify the collation:

- 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 *pgAdmin* 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.

5.2. Collation Dialog 175



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 b elow). Use the *SQL* tab for review; revisit or switch tabs to make any changes to the SQL command.

#### 5.2.1 Example

The following is an example of the sql command generated by user selections in the *Collation* dialog:



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:

https://www.postgresql.org/docs/current/locale.html

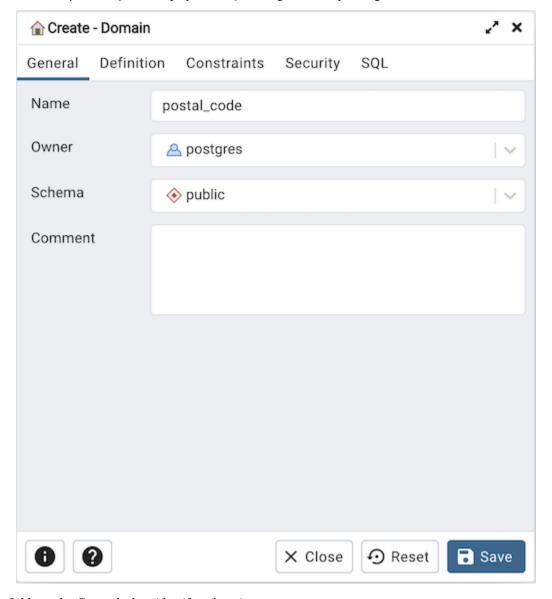
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

5.2. Collation Dialog 177

### 5.3 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.

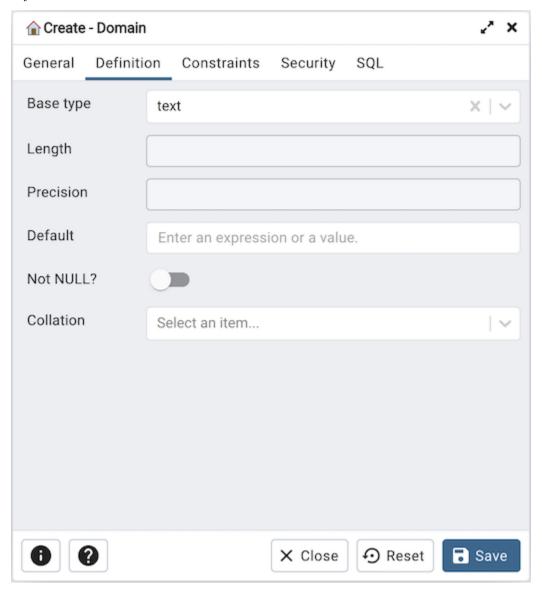
The *Domain* dialog organizes the development of a domain through the following tabs: *General*, *Definition*, *Constraints*, and *Security*. The *SQL* tab displays the SQL code generated by dialog selections.



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 *pgAdmin* 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.

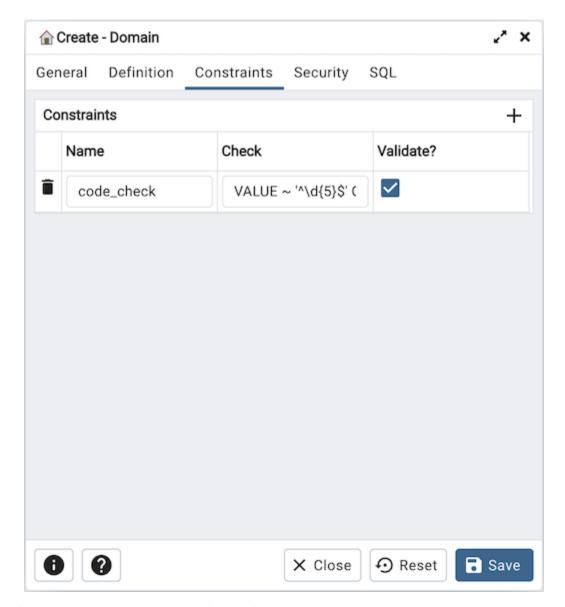


Use the fields in the *Definition* tab to describe the domain:

- 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.

5.3. Domain Dialog 179



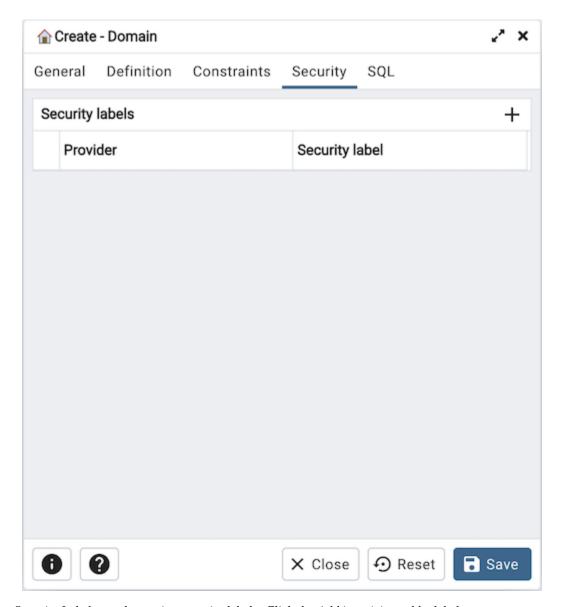
Use the fields in the *Constraints* tab to specify rules for the domain. Click the *Add* icon (+) to set constraints:

- 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 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.

5.3. Domain Dialog 181

#### 5.3.1 Example

The following is an example of the sql command generated by selections made in the *Domain* dialog:

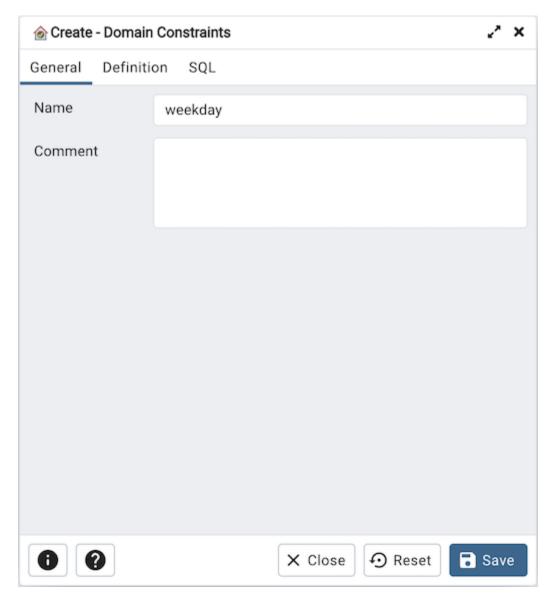
The example shown demonstrates creating a domain named *postal\_code* that confirms that the value entered is in proper format.

- Click the *Info* button (i) to access online help.
- Click the *Save* button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

## **5.4 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.

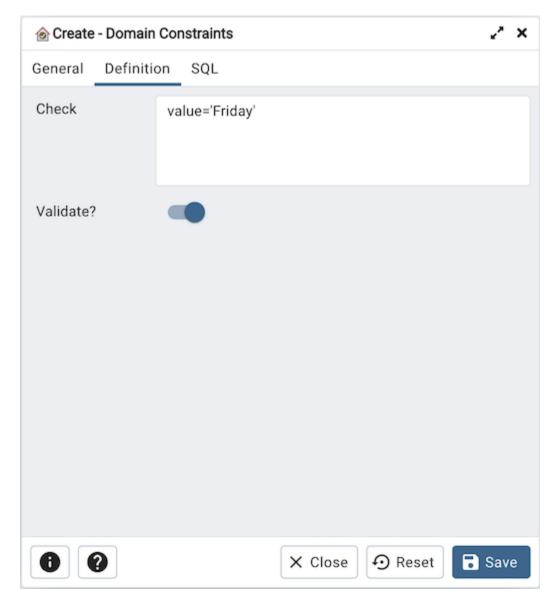
The *Domain Constraints* dialog organizes the development of a domain constraint through the following dialog tabs: *General* and *Definition*. The *SQL* tab displays the SQL code generated by dialog selections.



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 *pgAdmin* 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.

### 5.4.1 Example

The following is an example of the sql command generated by user selections in the *Domain Constraints* dialog:



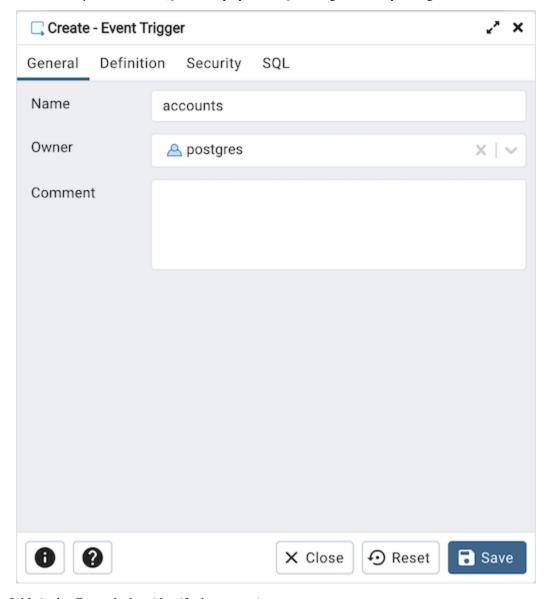
The example shown demonstrates creating a domain constraint on the domain *timesheets* named *weekday*. It constrains a value equal to *Friday*.

- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

## 5.5 Event Trigger Dialog

Use the *Event 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 *Event 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 *pgAdmin* 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.

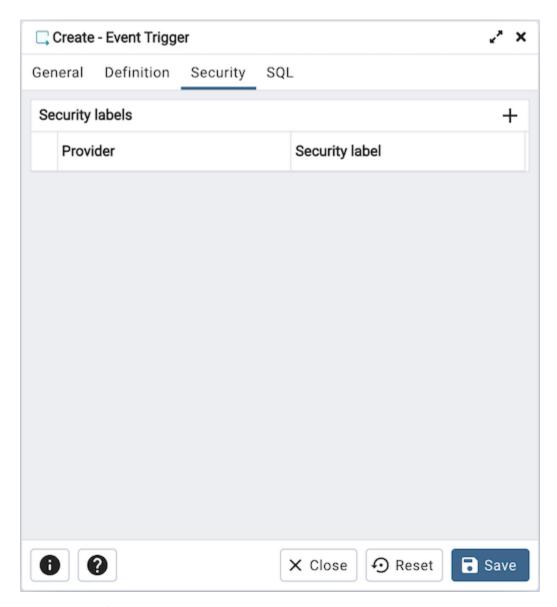
Click the *Definition* tab to continue.



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 for the trigger: *Enable Disable*, *Replica Always*.
- 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 *Event 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.

### 5.5.1 Example

The following is an example of the sql command generated by user selections in the Event Trigger dialog:



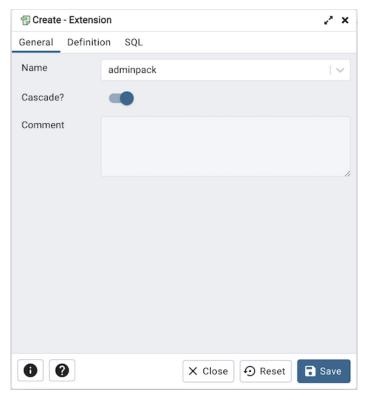
The command creates an event trigger named accounts that invokes the procedure named evt\_tri\_fun.

- Click the Info button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

## 5.6 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.

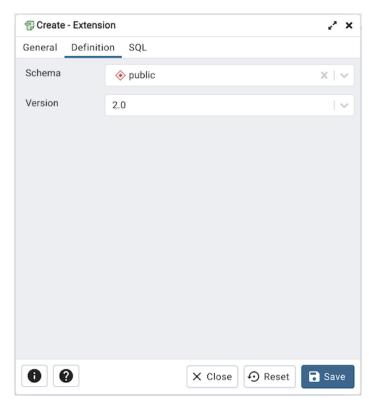
The *Extension* dialog allows you to implement options of the CREATE EXTENSION command through the following dialog tabs: *General* and *Definition*. The *SQL* tab displays the SQL code generated by dialog selections.



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.
- Move the switch next to *Cascade?* towards right position to automatically install any extensions that this extension depends on that are not already installed.
- 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*:

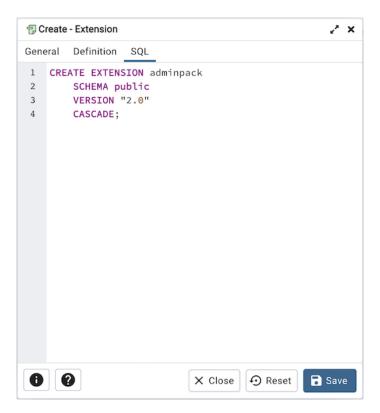
- Use the drop-down listbox next to *Schema* to select the name of the schema in which to install the extension's objects.
- Use the drop-down listbox next to *Version* to select the version of the extension to install.

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.

### 5.6.1 Example

The following is an example of the sql command generated by user selections in the Extension dialog:



The command creates the *adminpack* extension in the *public* schema. It is version 2.0 of *adminpack*.

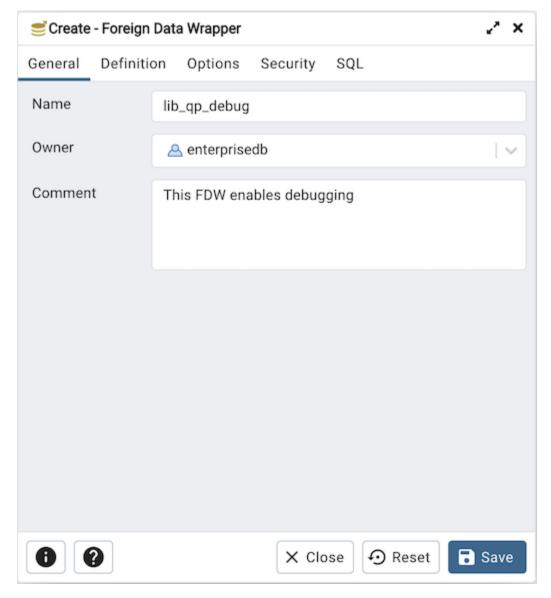
- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

## 5.7 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.

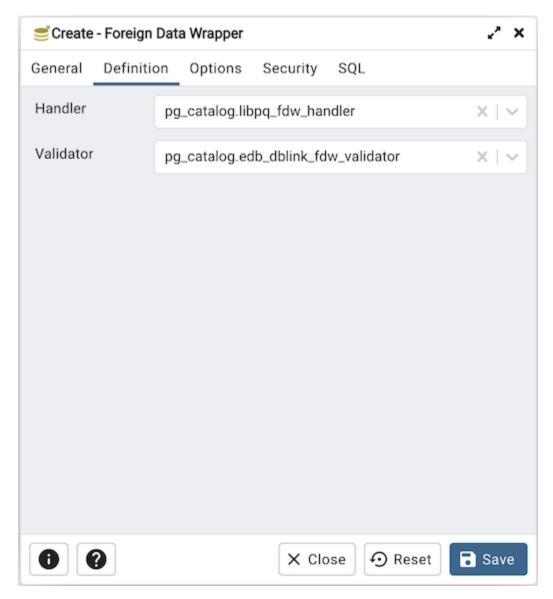
The *Foreign Data Wrapper* dialog organizes the development of a foreign data wrapper through the following dialog tabs: *General, Definition, Options*, and *Security*. The *SQL* tab displays the SQL code generated by dialog selections.



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 *pgAdmin* 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.

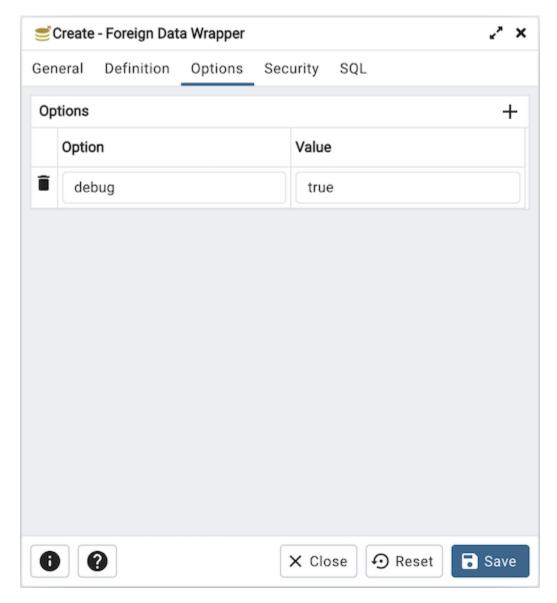
Click the *Definition* tab to continue.



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.

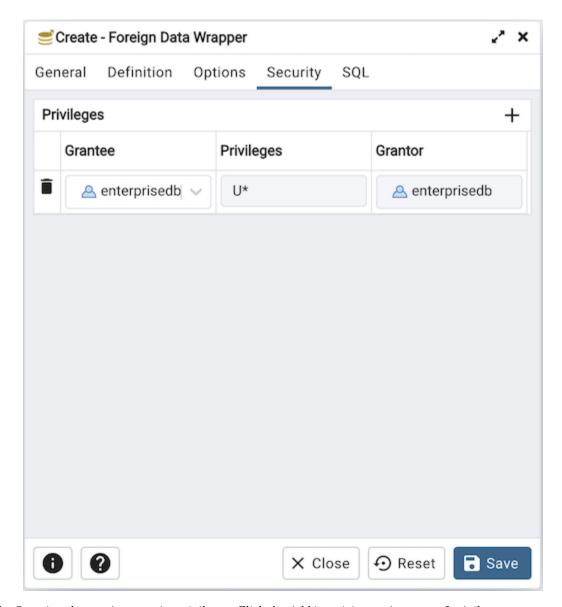


Use the fields in the *Options* tab to specify options:

- Click the *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.

### 5.7.1 Example

The following is an example of the sql command generated by user selections in the Foreign Data Wrapper dialog:

```
Create - Foreign Data Wrapper
General
        Definition
                                     SQL
                   Options
                            Security
1 CREATE FOREIGN DATA WRAPPER lib_qp_debug
       VALIDATOR pg_catalog.edb_dblink_fdw_validator
2
       HANDLER pg_catalog.libpq_fdw_handler
3
4
       OPTIONS (debug 'true');
5
  ALTER FOREIGN DATA WRAPPER lib_qp_debug
7
       OWNER TO enterprisedb;
8
9
  COMMENT ON FOREIGN DATA WRAPPER lib_qp_debug
10
       IS 'This FDW enables debugging';
11
12 GRANT USAGE ON FOREIGN DATA WRAPPER lib_qp_debug TO ente
                               X Close
                                         Reset
                                                    ■ Save
```

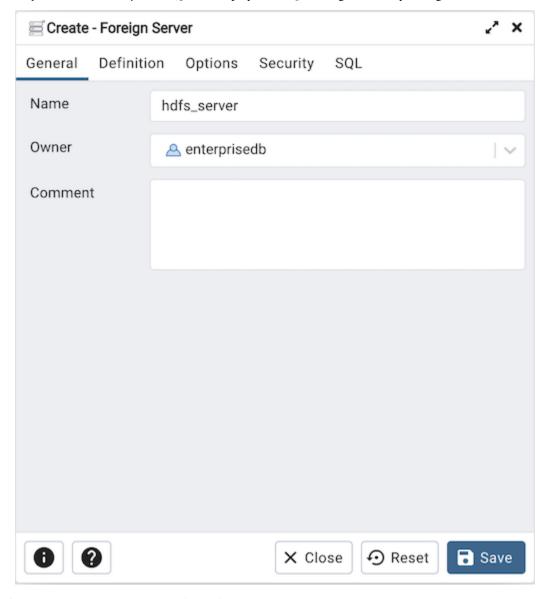
The example creates a foreign data wrapper named *lib\_qp\_debug* that uses pre-existing validator and handler functions, *dblink\_fdw\_validator* and *libpg\_fdw\_handler*. Selections on the *Options* tab set *debug* equal to *true*. The foreign data wrapper is owned by *postgres*.

- Click the *Help* button (?) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

## 5.8 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 *pgAdmin* tree control, expand the node of the applicable foreign data wrapper to launch the *Foreign Server* dialog.

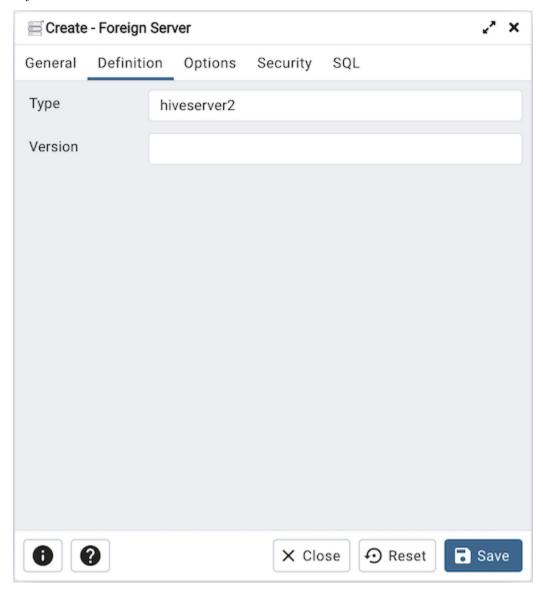
The *Foreign Server* dialog organizes the development of a foreign server through the following dialog tabs: *General*, *Definition*, *Options*, and *Security*. The *SQL* tab displays the SQL code generated by dialog selections.



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 *pgAdmin* 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.

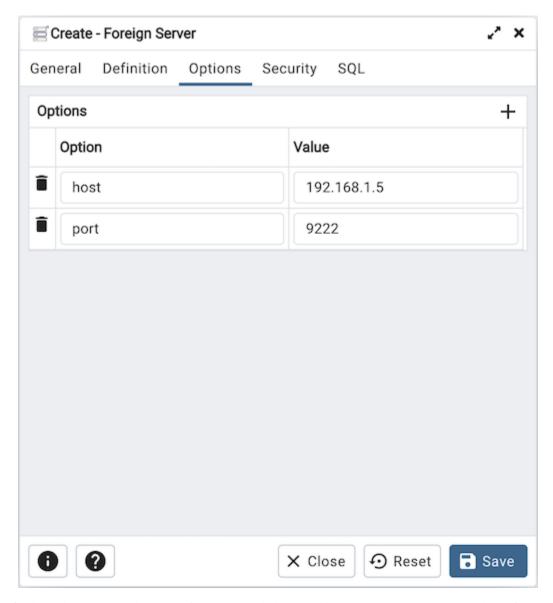
Click the *Definition* tab to continue.



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.

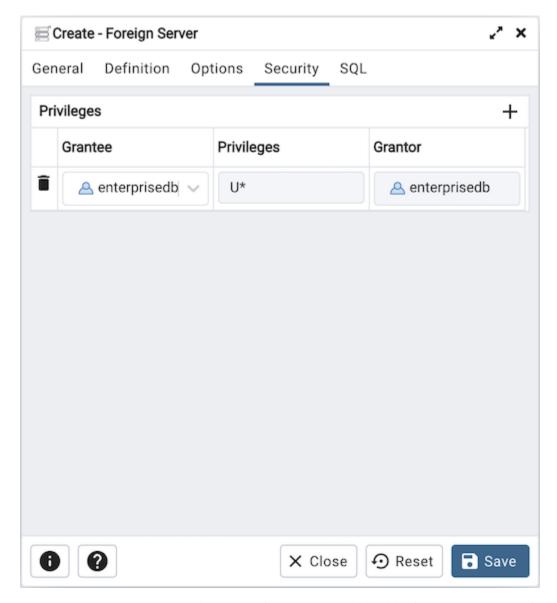


Use the fields in the *Options* tab to specify options. Click the *Add* button to create an option clause for the foreign server.

- 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.



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.

#### 5.8.1 Example

The following is an example of the sql command generated by user selections in the Foreign Server dialog:



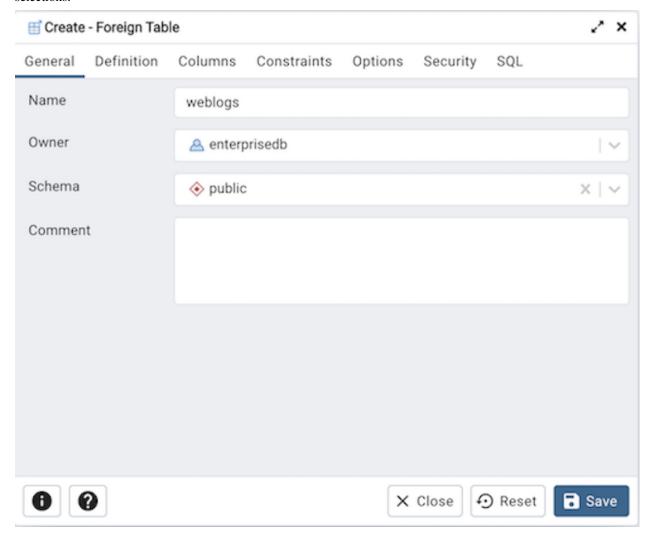
The example shown demonstrates creating a foreign server for the foreign data wrapper *hdfs\_fdw*. It has the name *hdfs\_server*; its type is *hiveserver*2. Options for the foreign server include a host and a port.

- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

## 5.9 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.

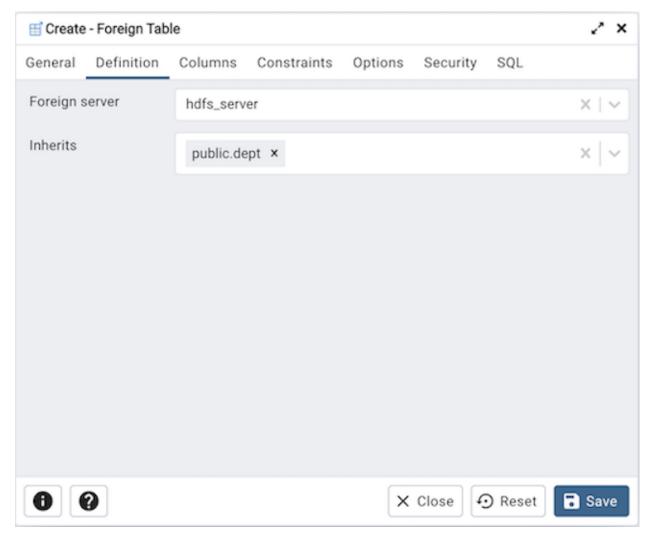
The Foreign Table dialog organizes the development of a foreign table through the following dialog tabs: General, Definition, Columns, Constraints, Options, and Security. The SQL tab displays the SQL code generated by dialog selections.



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 *pgAdmin* 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.

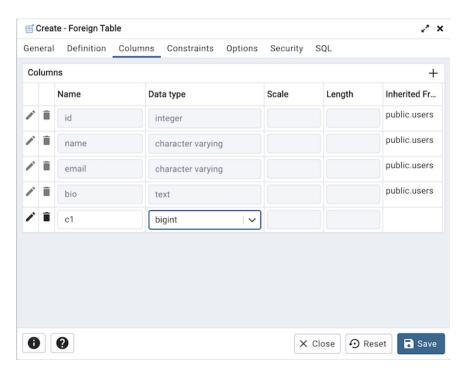
Click the *Definition* tab to continue.



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.

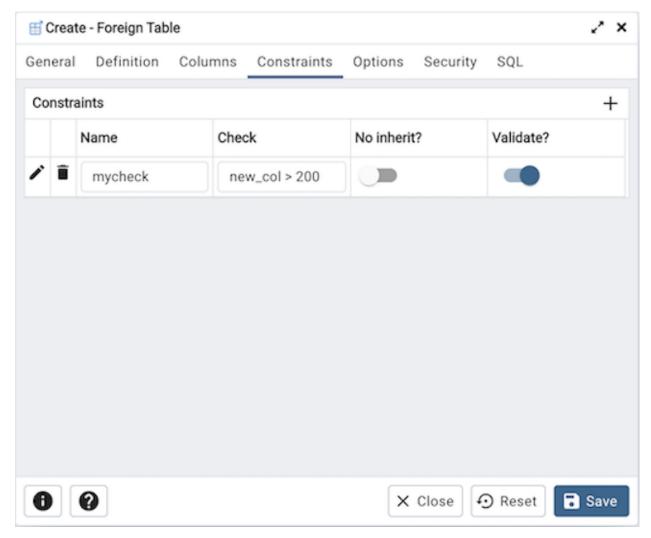


Use the fields in the Columns tab to add columns and their attributes to the table. Click the Add icon (+) to define a column:

- 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.

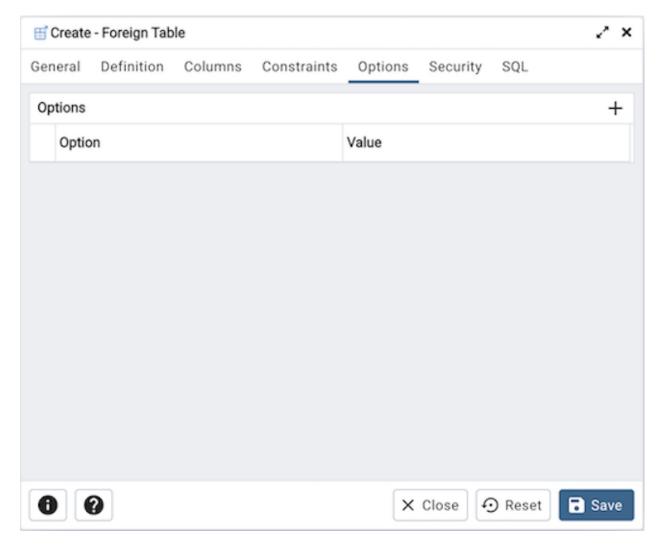


Use the fields in the *Constraints* tab to apply a table constraint to the foreign table. Click the *Add* icon (+) to define a constraint:

- 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 he *Delete Row* popup.

Click the *Options* tab to continue.

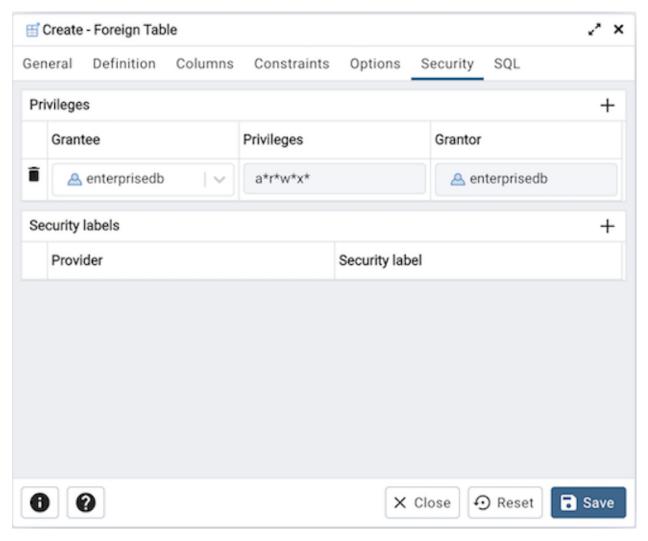


Use the fields in the *Options* tab to specify options to be associated with the new foreign table or one of its columns; the accepted option names and values are specific to the foreign data wrapper associated with the foreign server. Click the *Add* icon (+) to add an option/value pair.

- 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 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.

#### 5.9.1 Example

The following is an example of the sql command generated by user selections in the Foreign Table dialog:

```
Create - Foreign Table
General
        Definition
                   Columns
                             Constraints
                                          Options
                                                   Security
                                                            SQL
 1 CREATE FOREIGN TABLE public.weblogs(
       new_col bigint NULL
 2
 3)
 4
       INHERITS (public.dept)
 5
       SERVER hdfs_server;
 6
 7 ALTER FOREIGN TABLE public.weblogs
 8
       OWNER TO enterprisedb;
 9
10 ALTER FOREIGN TABLE public.weblogs
11
       ADD CONSTRAINT mycheck CHECK (new_col > 200);
12
13 GRANT ALL ON TABLE public.weblogs TO enterprised  WITH GRANT OPTION;
14
15
                                               X Close
                                                         Reset
                                                                     3 Save
```

The example shown demonstrates creating a foreign table weblogs with multiple columns.

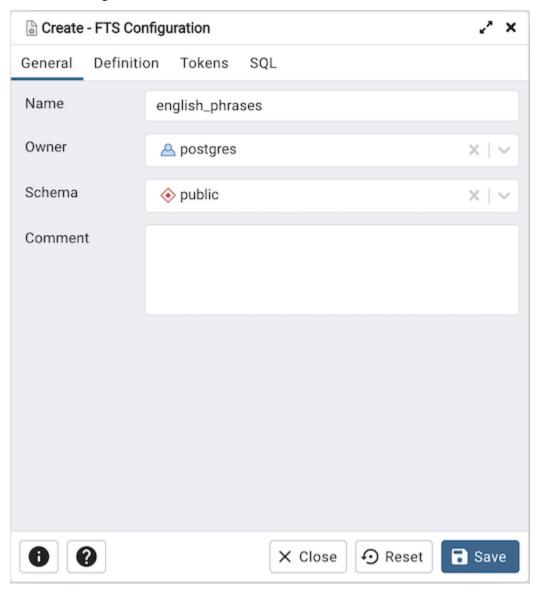
- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

### 5.10 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.

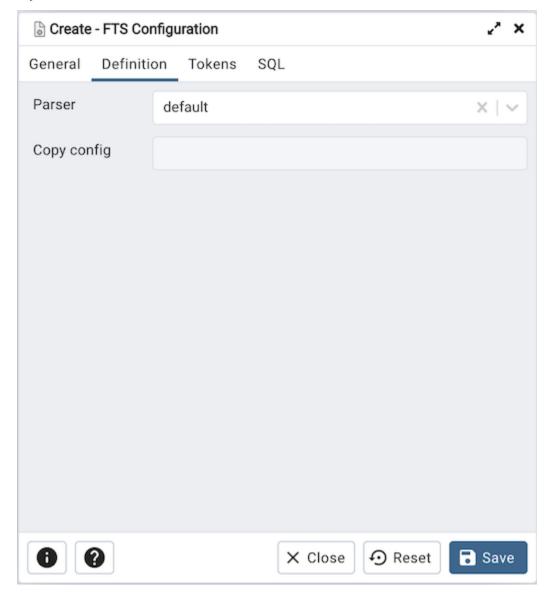


Use the fields in the *General* tab to identify a FTS configuration:

- Use the *Name* field to add a descriptive name for the FTS configuration. The name will be displayed in the *pgAdmin* 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.

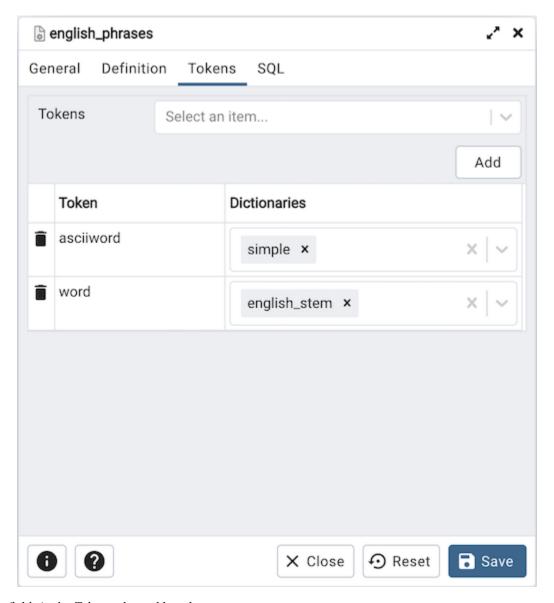
Click the *Definition* tab to continue.



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 button to create a token.
- Use the *Dictionaries* field to specify a dictionary.

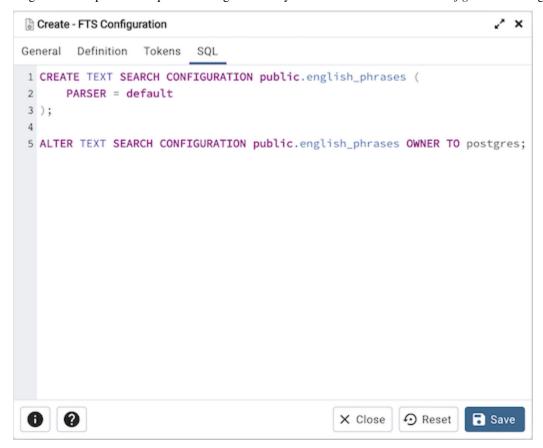
Repeat these steps to add additional tokens; to discard a token, click the trash icon to the left of the row and confirm deletion in the *Delete Row* popup.

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.

#### **5.10.1 Example**

The following is an example of the sql command generated by user selections in the FTS Configuration dialog:



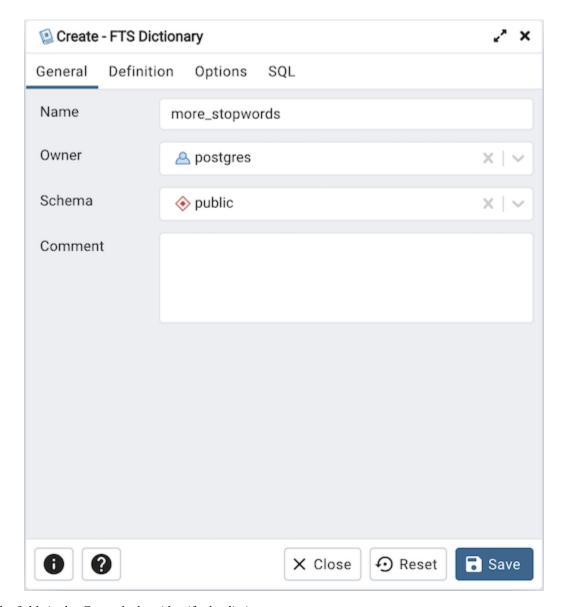
The example shown demonstrates creating a FTS configuration named english\_phrases. It uses the default parser.

- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

# **5.11 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.

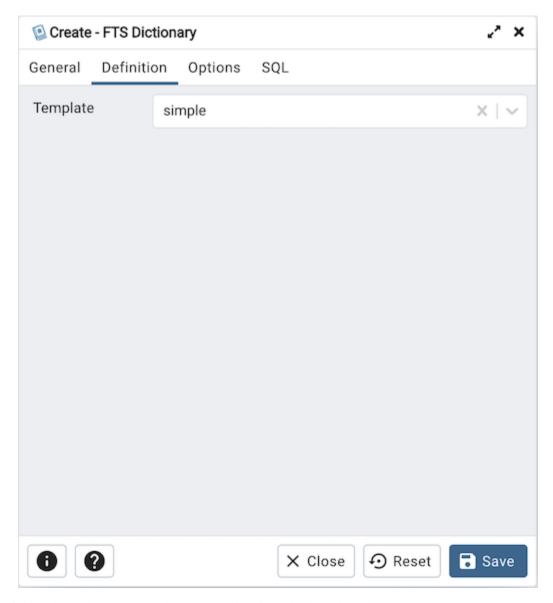
The FTS Dictionary dialog organizes the development of a FTS dictionary through the following dialog tabs: General, Definition, and Options. The SQL tab displays the SQL code generated by dialog selections.



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 *pgAdmin* 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.



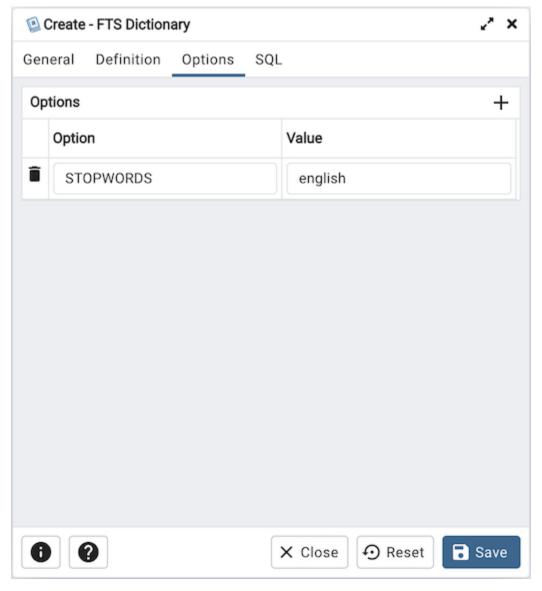
Use the field in the *Definition* tab to choose a template from the drop-down listbox:

- 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.

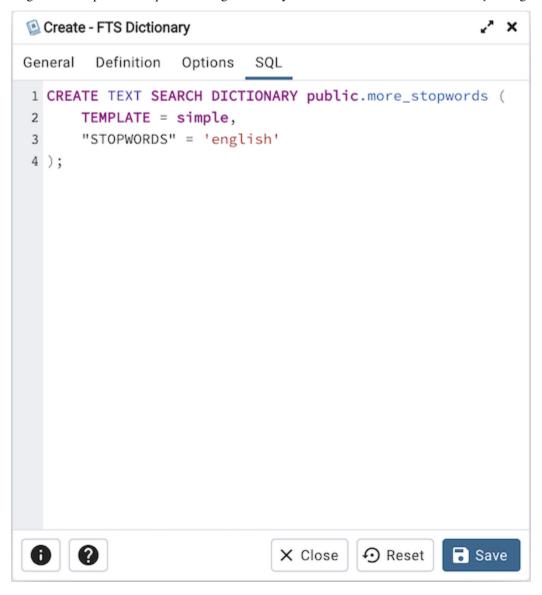
Click the *Add* icon (+) to specify each additional option/value pair; to discard an option, click the trash icon to the left of the row and confirm deletion in the *Delete Row* popup.

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.

#### **5.11.1 Example**

The following is an example of the sql command generated by user selections in the FTS Dictionary dialog:



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.

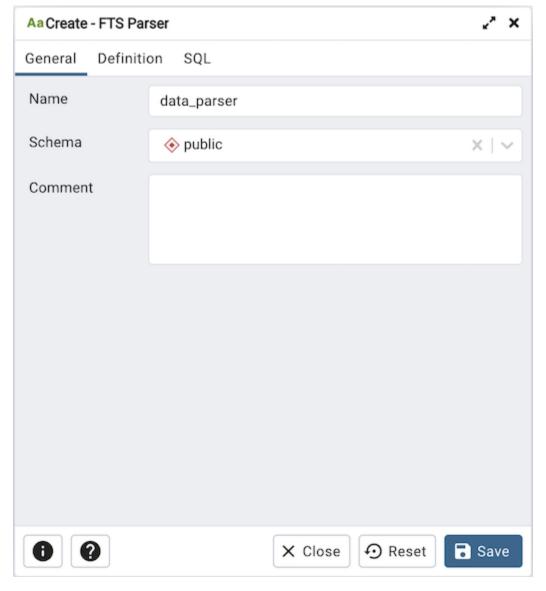
- Click the Info button (i) to access online help.
- Click the Save button to save work.

- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

### 5.12 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.

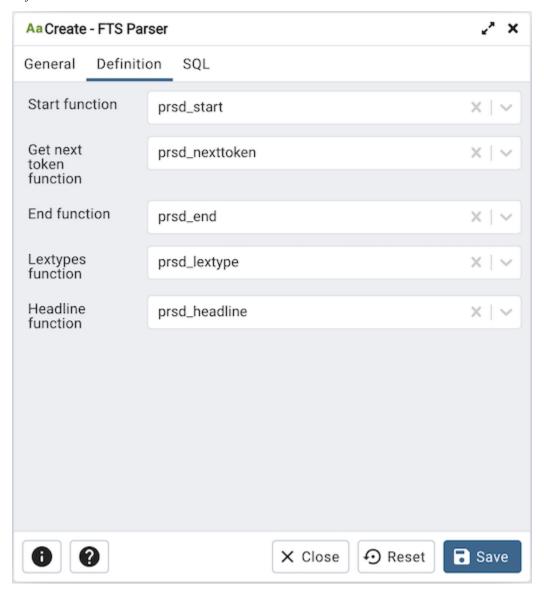


Use the fields in the *General* tab to identify a text search parser:

- Use the *Name* field to add a descriptive name for the parser. The name will be displayed in the *pgAdmin* 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.

```
Aa Create - FTS Parser
General
        Definition
                   SQL
1 CREATE TEXT SEARCH PARSER public.data_parser (
2
       START = prsd_start,
3
       GETTOKEN = prsd_nexttoken,
4
       END = prsd_end,
5
       LEXTYPES = prsd_lextype,
       HEADLINE = prsd_headline);
6
                                                     Save
                               X Close
                                          Reset
```

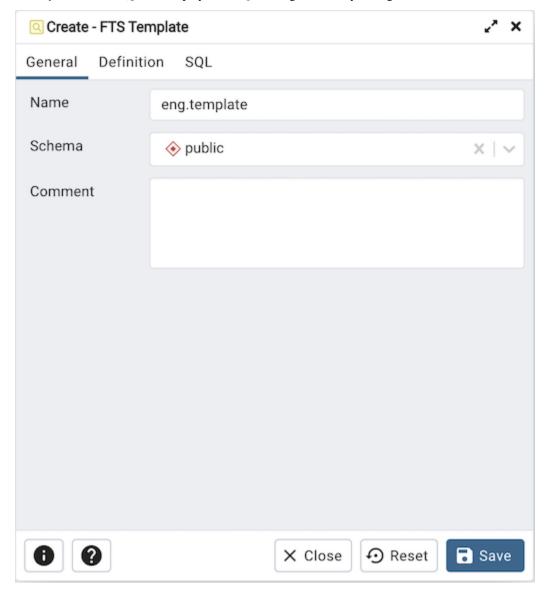
Your entries in the FTS Parser dialog generate a SQL command. Use the SQL tab for review; revisit or switch tabs to make any changes to the SQL command.

- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

# **5.13 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.

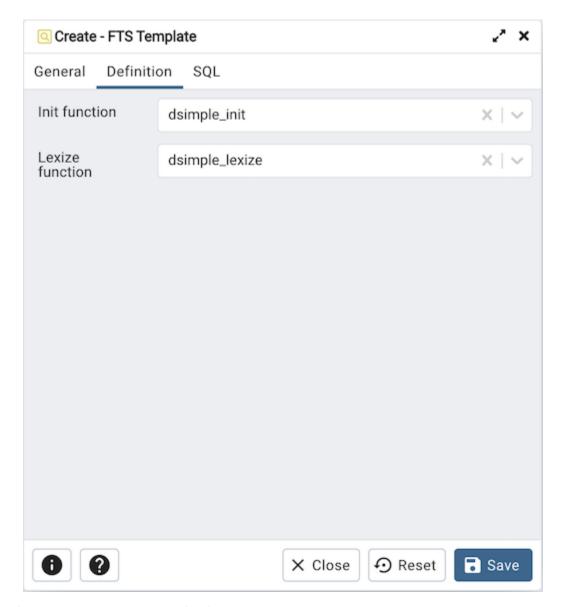
The FTS Template dialog organizes the development of a text search Template through the following dialog tabs: General, and Definition. The SQL tab displays the SQL code generated by dialog selections.



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 *pgAdmin* 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.

#### **5.13.1 Example**

The following is an example of the sql command generated by user selections in the FTS Template dialog:



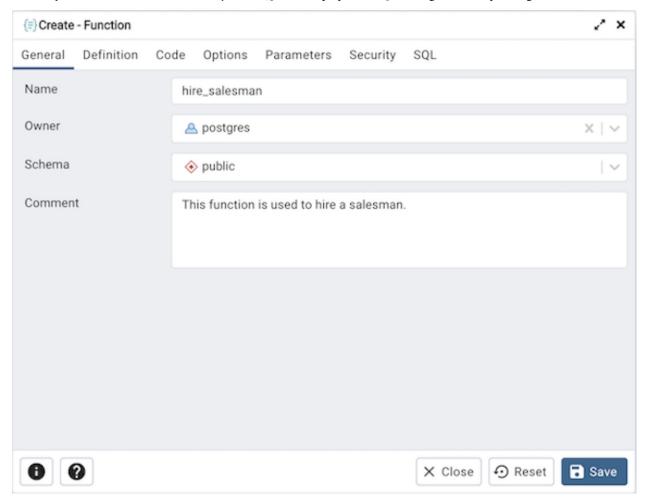
The example shown demonstrates creating a fts template named eng.template that uses the ispell dictionary.

- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

### 5.14 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.

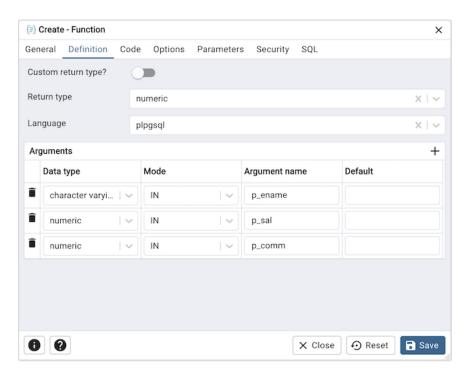
The *Function* dialog organizes the development of a function through the following dialog tabs: *General*, *Definition*, *Code*, *Options*, *Parameters*, and *Security*. The *SQL* tab displays the SQL code generated by dialog selections.



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 *pgAdmin* 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:

- Move the *Custom return type?* switch to provide a user defined return type.
- Use the drop-down listbox next to *Return type* to select the data type returned by the function, if any. If *Custom return type?* is enabled this field will change to an input text field.
- 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.

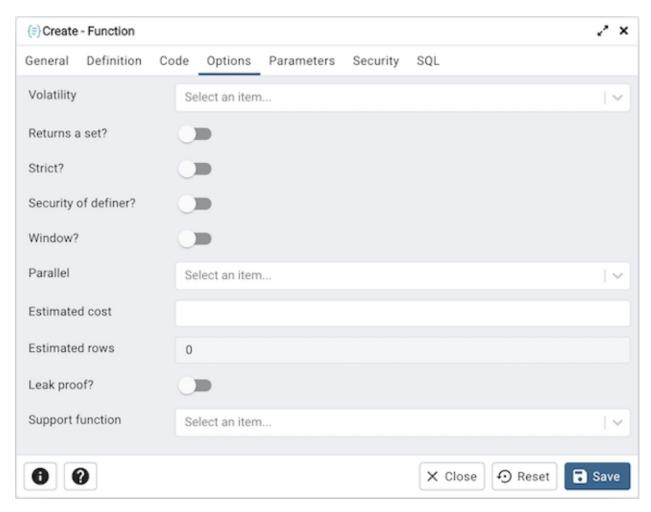
Click the *Add* icon (+) to define another argument; to discard an argument, click the trash icon to the left of the row and confirm deletion in the *Delete Row* popup.

Click the Code tab to continue.

```
∠ ×
(=) Create - Function
       Definition
General
                  Code
                         Options Parameters
                                              Security SQL
1 DECLARE
2
                       NUMERIC(4);
      v_empno
 3
      v_ename
                       VARCHAR(10);
 4
      v_job
                       VARCHAR(9);
 5
                       NUMERIC(4);
      v_mgr
 6
      v_hiredate
                       DATE:
 7
                       NUMERIC(7,2);
      v_sal
                       NUMERIC(7,2);
 8
      v_comm
9
       v_deptno
                       NUMERIC(2);
10 BEGIN
11
       v_empno := new_empno();
12
       INSERT INTO emp VALUES (v_empno, p_ename, 'SALESMAN', 7698,
13
           CURRENT_DATE, p_sal, p_comm, 30);
       SELECT INTO
14
15
           v_empno, v_ename, v_job, v_mgr, v_hiredate, v_sal, v_comm, v_deptno
16
           empno, ename, job, mgr, hiredate, sal, comm, deptno
17
           FROM emp WHERE empno = v_empno;
       RAISE INFO 'Department : %', v_deptno;
18
19
       RAISE INFO 'Employee No: %', v_empno;
20
       RAISE INFO 'Name
                              : %', v_ename;
                                                                             Save
                                                        X Close
                                                                  Reset
```

• Use the *Code* field to write the code that will execute when the function is called.

Click the *Options* tab to continue.



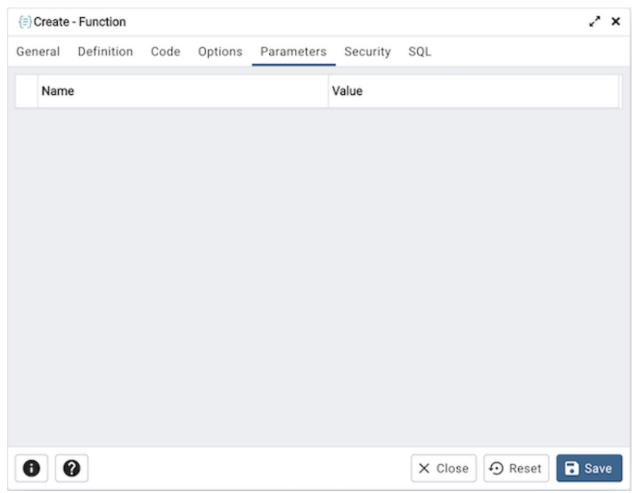
Use the fields in the *Options* tab to describe or modify the action of the function:

- 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:

https://www.postgresql.org/docs/current/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.

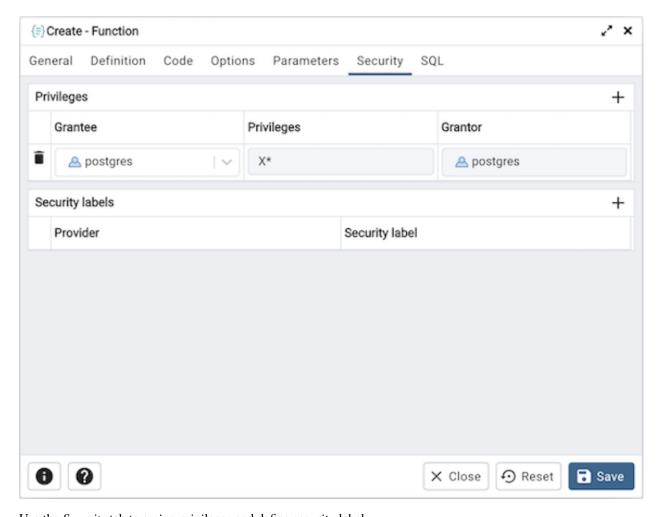
Click the *Parameters* tab to continue.



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 *NamelValue* 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 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 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.

#### **5.14.1 Example**

The following is an example of the sql command generated by selections made in the *Function* dialog:

```
(=) Create - Function
                                                                                      ×
                          Options
                                                         SQL
General
        Definition
                   Code
                                   Parameters
                                               Security
 1 CREATE FUNCTION public.hire_salesman(IN p_ename character varying, IN p_sal numer
2
       RETURNS numeric
 3
       LANGUAGE 'plpgsql'
 4
5 AS $BODY$
 6 DECLARE
                        NUMERIC(4);
7
       v_empno
8
       v_ename
                        VARCHAR(10);
9
       v_job
                        VARCHAR(9);
                        NUMERIC(4);
10
       v_mgr
11
       v_hiredate
                        DATE:
12
       v_sal
                        NUMERIC(7,2);
       v_comm
13
                        NUMERIC(7,2);
14
       v_deptno
                        NUMERIC(2);
15 BEGIN
16
       v_empno := new_empno();
       INSERT INTO emp VALUES (v_empno, p_ename, 'SALESMAN', 7698,
17
18
           CURRENT_DATE, p_sal, p_comm, 30);
19
       SELECT INTO
20
           v_empno, v_ename, v_job, v_mgr, v_hiredate, v_sal, v_comm, v_deptno
      0
 0
                                                          X Close

◆ Reset

                                                                                ∃ Save
```

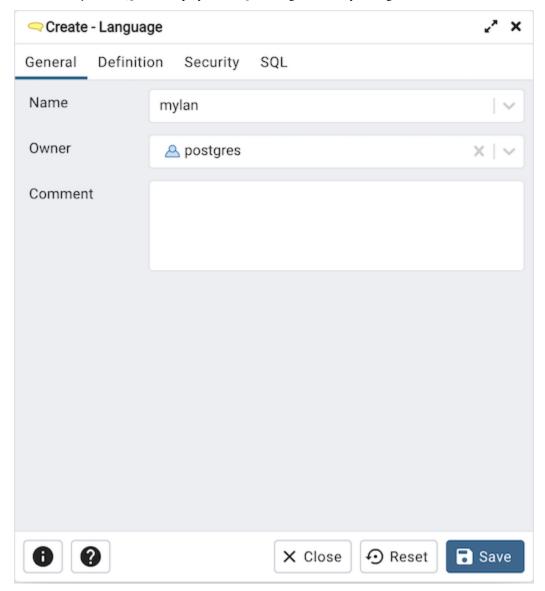
The example demonstrates creating an *plpgsql* function named *hire\_salesmen*. The function have three columns (p\_ename, p\_sal and p\_comm).

- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

# 5.15 Language Dialog

Use the CREATE LANGUAGE dialog to register a new procedural language.

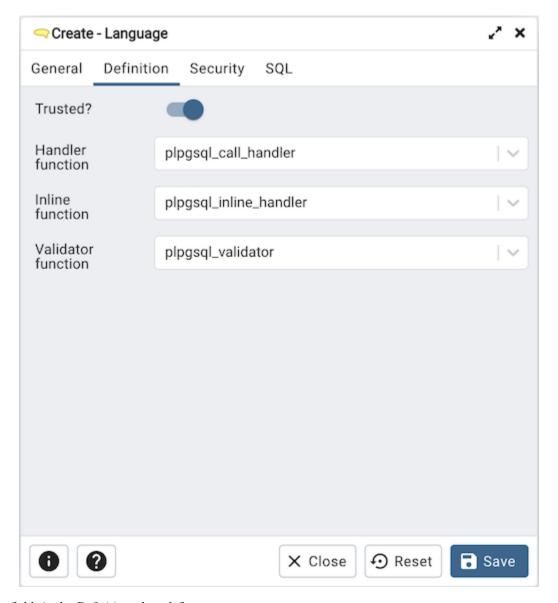
The *Language* dialog organizes the registration of a procedural language through the following dialog tabs: *General*, *Definition*, and *Security*. The *SQL* tab displays the SQL code generated by dialog selections.



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.

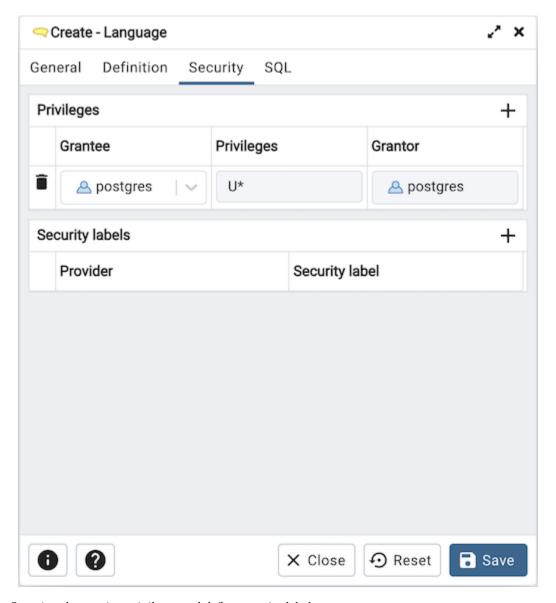
Click the *Definition* tab to continue.



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 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.

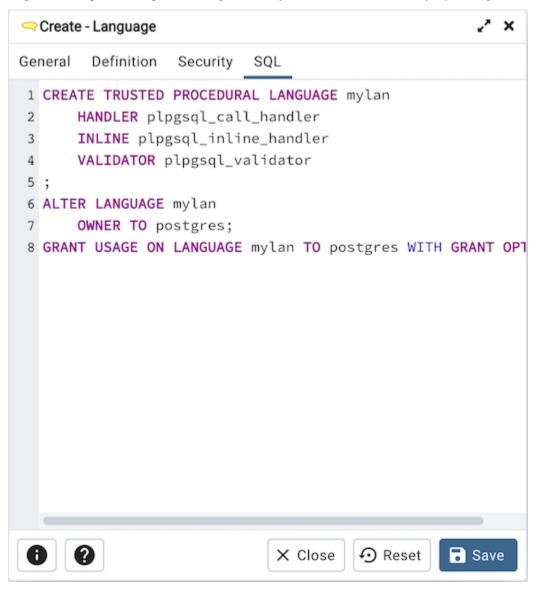
Click the *Add* icon (+) to assign additional security labels; to discard a security label, click the trash icon to the left of the row and confirm deletion in the *Delete Row* popup.

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.

#### **5.15.1 Example**

The following is an example of the sql command generated by user selections in the *Language* dialog:



The example shown demonstrates creating the trusted procedural language named mylan.

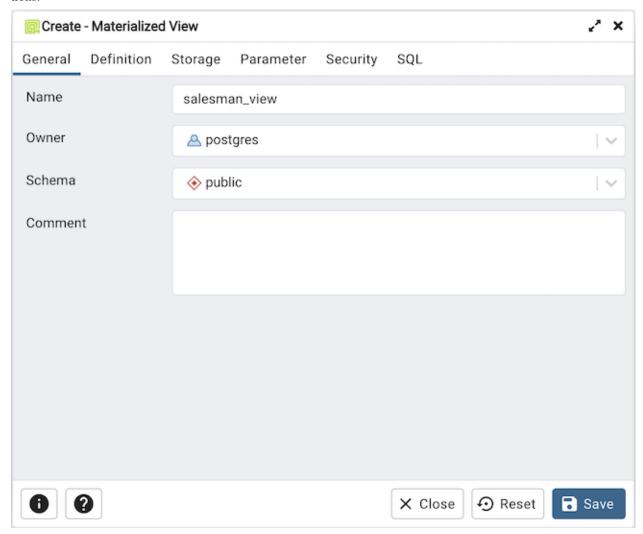
• Click the *Info* button (i) to access online help.

- Click the *Save* button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

### 5.16 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.

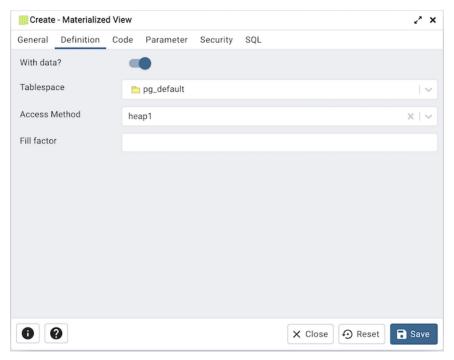


Use the fields in the *General* tab to identify the materialized view:

- Use the *Name* field to add a descriptive name for the materialized view. The name will be displayed in the *pgAdmin* 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.

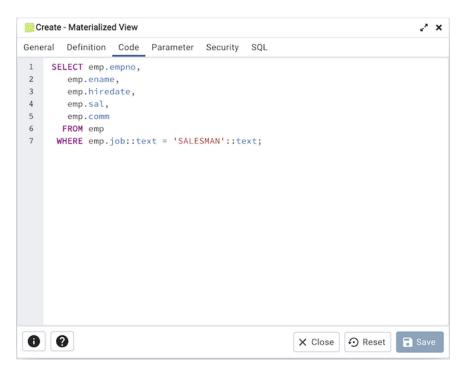
Click the *Definition* tab to continue.



Use the fields in the Storage tab to maintain the materialized view:

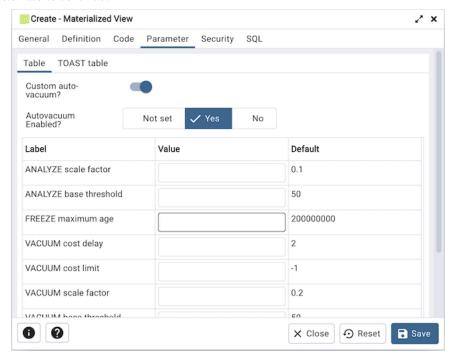
- Move the *With Data* switch to the *Yes* position to specify the materialized view should be populated at creation time. If not, the materialized view cannot be queried until you invoke REFRESH MATERIALIZED VIEW.
- Use the drop-down listbox next to *Tablespace* to select a location for the materialized view.
- Use the drop-down list box next to Access Method to specify the table access method to use to store the contents for the new materialized view; the method needs to be an access method of type TABLE. This field is optional. This option is available from v12 and above.
- 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 *Code* tab to continue.



Use the text editor field in the *Code* 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 Parameter tab to continue.



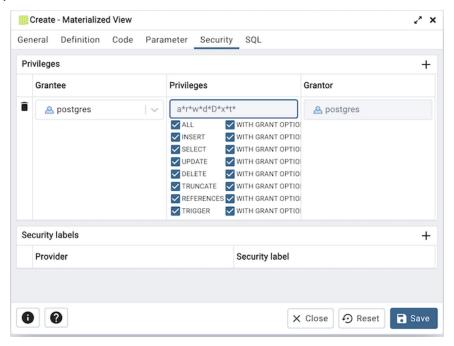
Use the tabs nested inside the *Parameter* tab to specify VACUUM and ANALYZE thresholds; use the *Table* tab and the *Toast Table* tab to customize values for the table and the associated toast table. To change the default values:

• 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 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.

#### **5.16.1 Example**

The following is an example of the sql command generated by user selections in the *Materialized View* dialog:

```
@ Create - Materialized View
General Definition Code Parameter Security SQL
1 CREATE MATERIALIZED VIEW public.salesman_view
3
   WITH (
        autovacuum_enabled = TRUE
5
   TABLESPACE pg_default
7
8
    SELECT emp.empno,
9
        emp.ename,
10
        emp.hiredate,
11
       emp.sal,
12
        emp.comm
1.3
      FROM emp
     WHERE emp.job::text = 'SALESMAN'::text
15 WITH DATA;
17 ALTER TABLE IF EXISTS public.salesman_view
18
        OWNER TO postgres;
19
20
   GRANT ALL ON TABLE public.salesman_view TO postgres WITH GRANT OPTION;
      a
                                                    X Close
                                                             • Reset
                                                                       3 Save
```

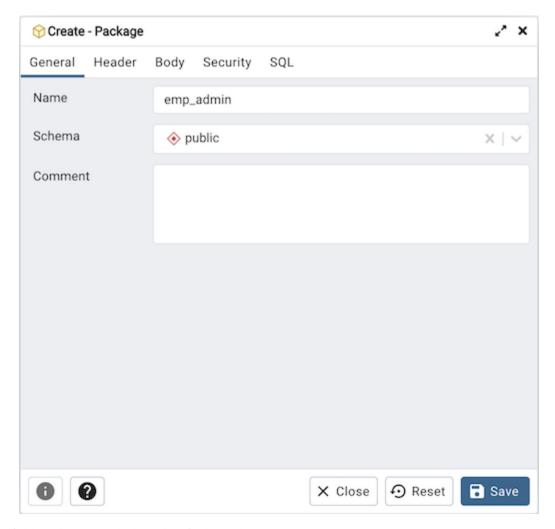
The example shown creates a query named *new\_hires* that stores the result of the displayed query in the *pg\_default* tablespace.

- Click the Info button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

### 5.17 Package Dialog

Use the *Package* dialog to create a (user-defined) package specification.

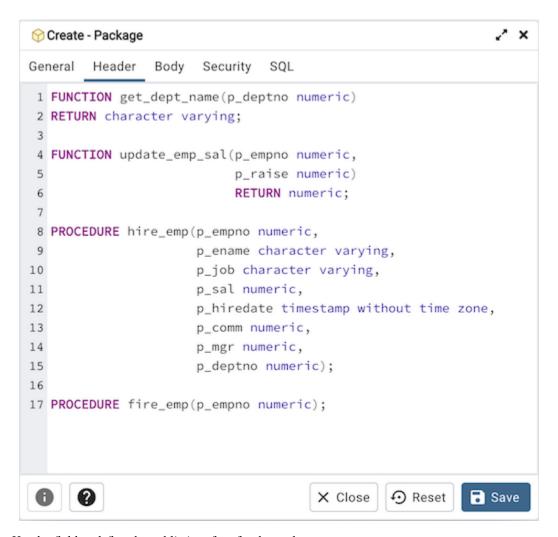
The *Package* dialog organizes the management of a package through the following dialog tabs: *General*, *Header*, *Body*, and *Security*. The *SQL* tab displays the SQL code generated by dialog selections.



Use the fields in the *General* tab to identify the package:

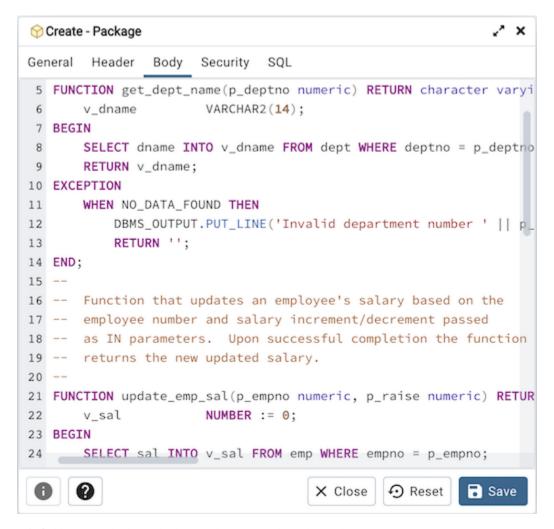
- Use the *Name* field to add a descriptive name for the package. The name of a new package must not match any existing package in the same schema.
- Select the schema in which the package will reside from the drop-down listbox in the Schema field.
- Store notes about the package in the *Comment* field.

Click the *Header* tab to continue.



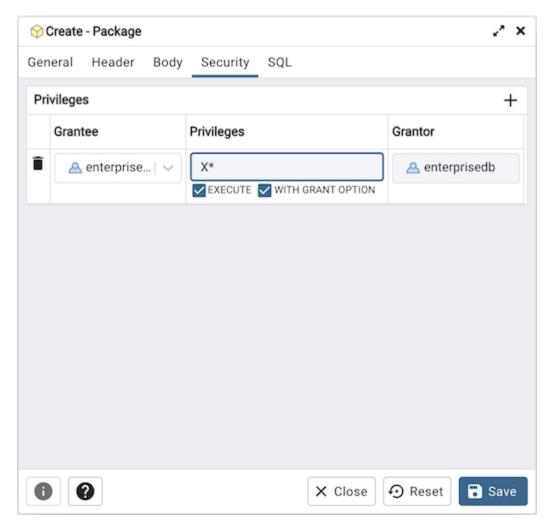
Use the *Header* field to define the public interface for the package.

Click the *Body* tab to continue.



Use the *Body* field to provide the code that implements each package object.

Click the Security tab to continue.



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:

```
Header
                                SQL
General
                Body
                       Security
1 CREATE OR REPLACE PACKAGE public.emp_admin
2 IS
3 FUNCTION get_dept_name(p_deptno numeric)
4 RETURN character varying;
5
 6 FUNCTION update_emp_sal(p_empno numeric,
7
                           p_raise numeric)
                           RETURN numeric;
8
9
10 PROCEDURE hire_emp(p_empno numeric,
11
                      p_ename character varying,
                      p_job character varying,
12
13
                      p_sal numeric,
14
                      p_hiredate timestamp without time zone,
15
                      p_comm numeric,
16
                      p_mgr numeric,
17
                      p_deptno numeric);
18
19 PROCEDURE fire_emp(p_empno numeric);
  END emp_admin;
                                      X Close
                                                Reset
                                                           Save
```

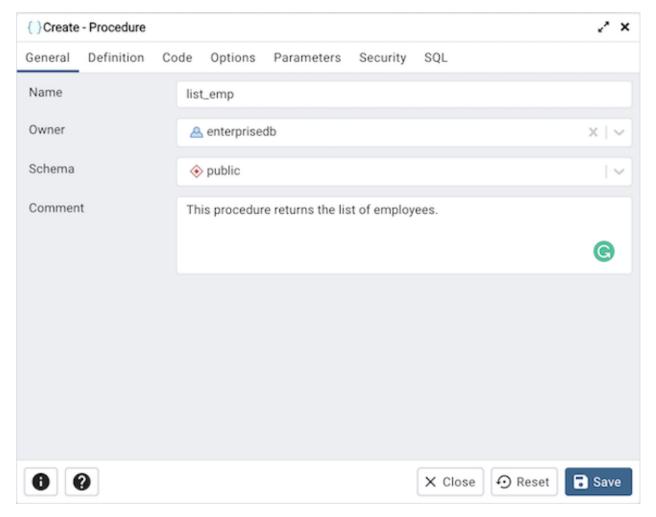
The example shown demonstrates creating a package named *empinfo* that includes two function and two procedure.

- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to delete any changes to the dialog.

### 5.18 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.

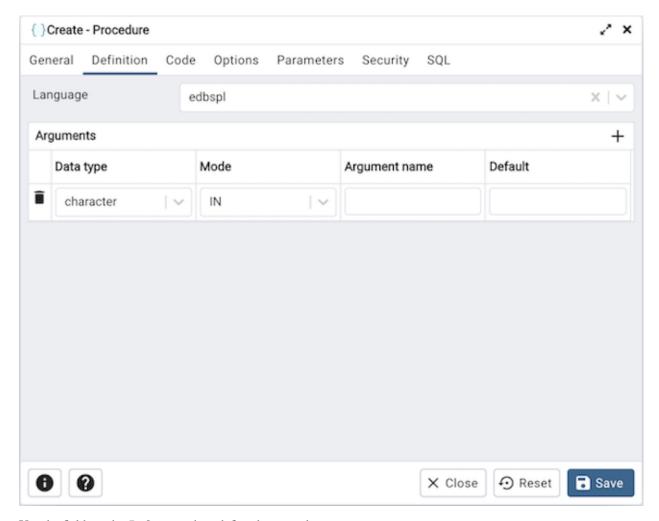
The *Procedure* dialog organizes the development of a procedure through the following dialog tabs: *General*, *Definition*, *Code*, *Options*, *Arguments*, *Parameters*, and *Security*. The *SQL* tab displays the SQL code generated by dialog selections.



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 *pgAdmin* 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.

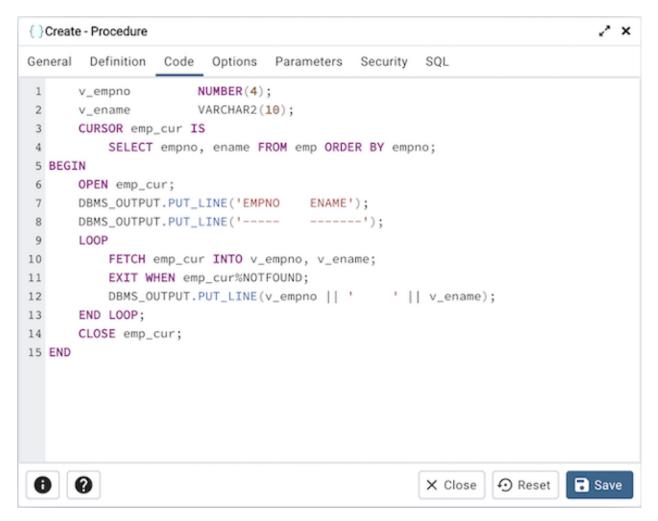


Use the fields in the *Definition* tab to define the procedure:

- 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.

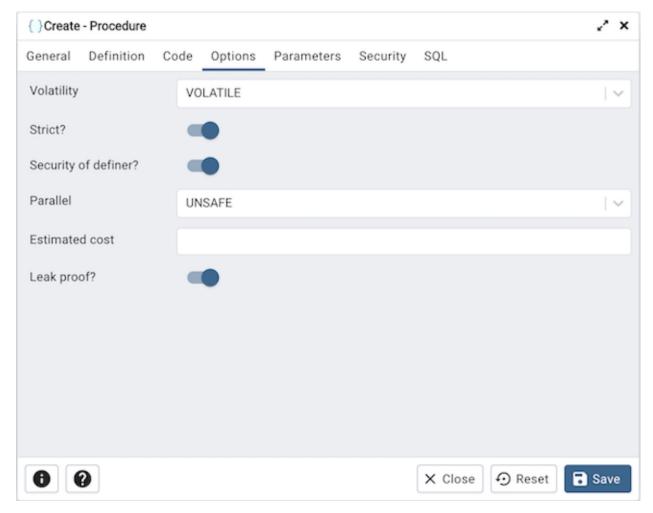
Click *Add* to define another argument; to discard an argument, click the trash icon to the left of the row and confirm deletion in the *Delete Row* popup.

Click the Code tab to continue.



• Use the *Code* field to specify the code that will execute when the procedure is called.

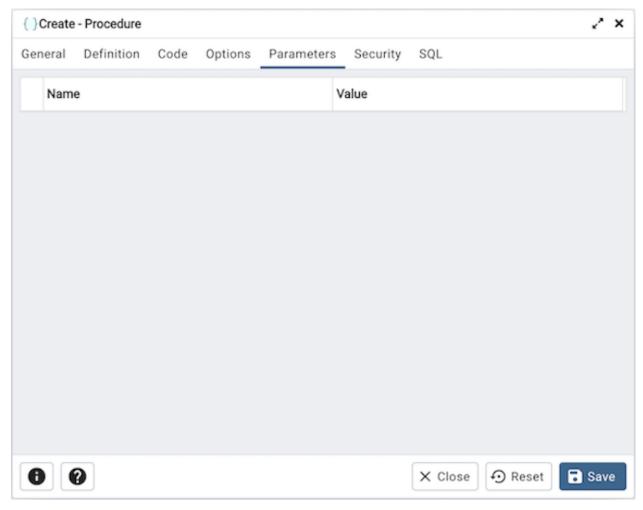
Click the Options tab to continue.



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*.

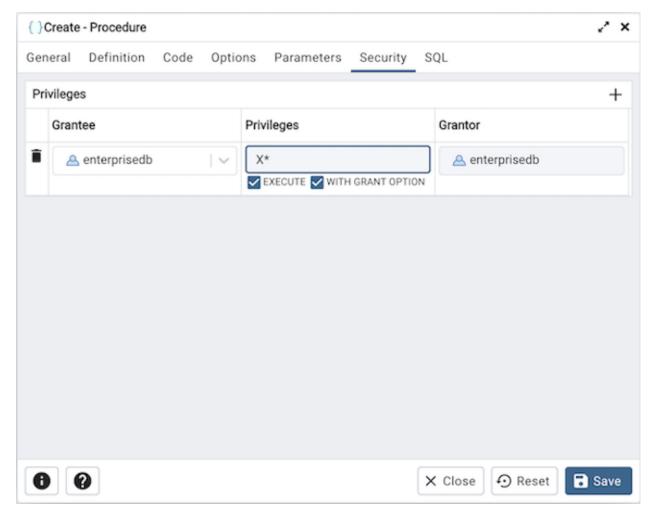
Click the *Parameters* tab to continue.



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 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.

#### **5.18.1 Example**

The following is an example of the sql command generated by selections made in the *Procedure* dialog:

```
. ×
Create - Procedure
        Definition
                   Code
                          Options
                                  Parameters
                                                        SQL
General
                                              Security
1 CREATE PROCEDURE public.list_emp(IN character)
2 LANGUAGE 'edbspl'
3
       SECURITY DEFINER VOLATILE LEAKPROOF STRICT PARALLEL UNSAFE
4 AS $BODY$
5
       v_empno
                       NUMBER(4);
       v_ename
                       VARCHAR2 (10);
6
       CURSOR emp_cur IS
7
8
           SELECT empno, ename FROM emp ORDER BY empno;
9 BEGIN
10
       OPEN emp_cur;
       DBMS_OUTPUT.PUT_LINE('EMPNO
11
                                       ENAME');
       DBMS_OUTPUT.PUT_LINE('----
12
13
       L00P
14
           FETCH emp_cur INTO v_empno, v_ename;
           EXIT WHEN emp_cur%NOTFOUND;
15
           DBMS_OUTPUT.PUT_LINE(v_empno || ' ' || v_ename);
16
17
       END LOOP;
18
       CLOSE emp_cur;
19 END
20 $BODY$;
                                                        X Close
                                                                  Reset
                                                                             Save
```

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.

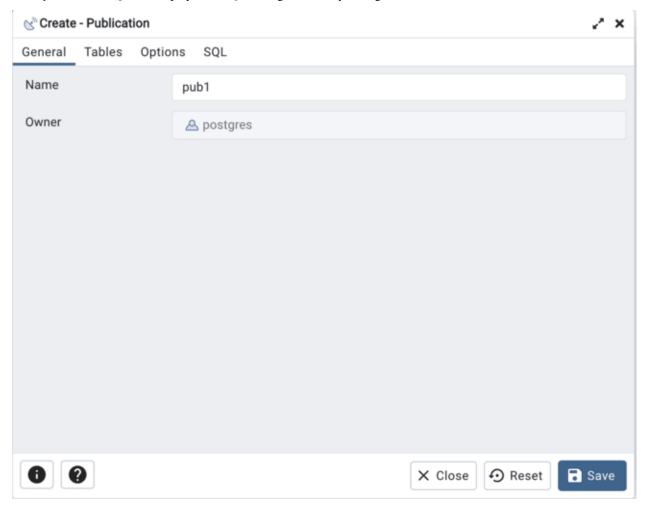
- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

### 5.19 Publication Dialog

Logical replication uses a *publish* and *subscribe* model with one or more *subscribers* subscribing to one or more *publications* on a publisher node.

Use the *publication* dialog to create a publication. A publication is a set of changes generated from a table or a group of tables or a schema or a group of schemas, and might also be described as a change set or replication set.

The *publication* dialog organizes the development of a publication through the following dialog tabs: *General, Tables* and *Options*. The *SQL* tab displays the SQL code generated by dialog selections.

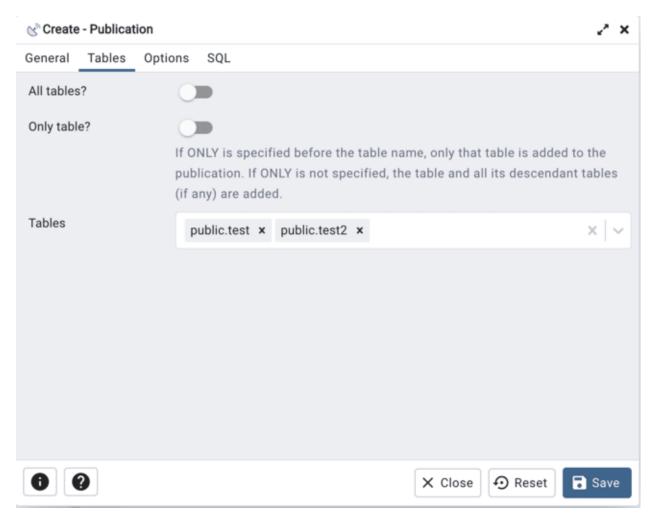


Use the fields in the *General* tab to identify the publication:

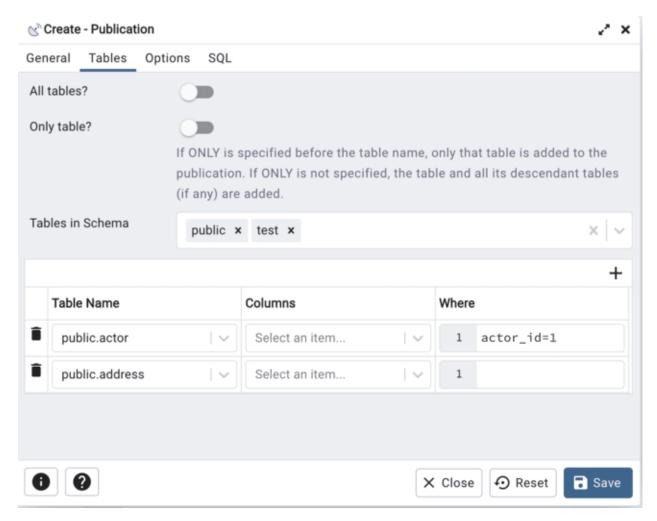
- Use the *Name* field to add a descriptive name for the publication. The name will be displayed in the *pgAdmin* tree control.
- The *Owner* field takes the name of the user automatically as current connected user. You can change the owner after creating publication using alter publication.

Click the Tables tab to continue.

Tables tab for PostgreSQL version <= 14



Tables tab for PostgreSQL version >= 15



Use the *Tables* tab to set table properties for the publication:

- Move the switch next to *All tables?* to *Yes* to replicate all the tables of the database, including tables created in the future.
- Move the switch next to Only table? to Yes to replicate only the listed tables excluding all its descendant tables.
- Specify a table or list of tables separated by a comma in *Tables* field to replicate all the listed table. With PostgreSQL 15 forward, specify a table or list of tables to replicate all the listed tables along with specific columns and/or WHERE clause to filter rows.
- With PostgreSQL 15 forward, specify a schema or list of schemas separated by a comma in *Tables In Schema* field to replicate all the listed schemas. This field will be disabled if any columns are selected for tables.

Click the *Options* tab to continue.



Use the *Options* tab to set option properties for the publication:

• Use the *With* section to determine which DML operations will be published by the new publication to the subscribers. Move the switch next to *INSERT*, *UPDATE*, *DELETE*, or *TRUNCATE* to *No* if you do not want to replicate any of these DML operations from Publication to Subscription. By default, all the switches are set to *Yes* allowing all the DML operations.

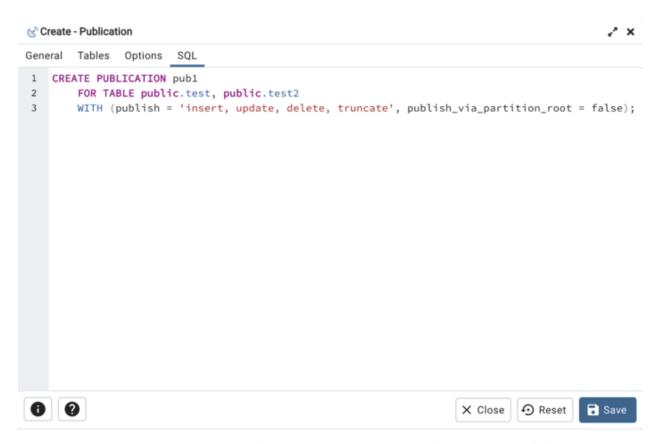
**Note:** A published table must have a "replica identity" configured in order to be able to replicate UPDATE and DELETE operations. You can change with ALTER TABLE statement. For more information on replica identity see Logical Replication Publication.

Click the SQL tab to continue.

Your entries in the *Publication* 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 *Publication* dialog:



The example creates a publication named *pub1* that is owned by *postgres*. It allows replication of all the DML operations.

- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore all the default settings.

## 5.20 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.

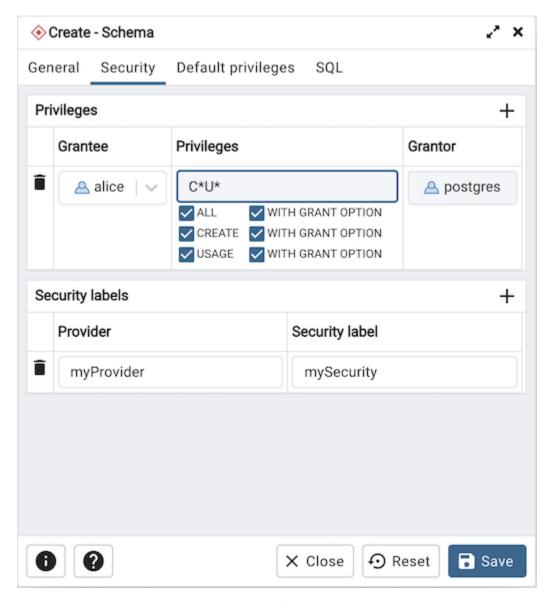
The *Schema* dialog organizes the development of schema through the following dialog tabs: *General* and *Security*. The *SQL* tab displays the SQL code generated by dialog selections.



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 *pgAdmin* 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.
- 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 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.

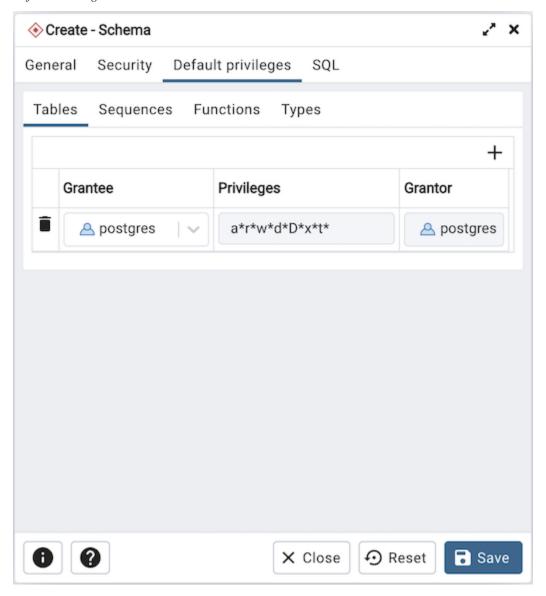
Click the *Add* icon (+) to assign a security label in the *Security Labels* panel:

- 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 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.

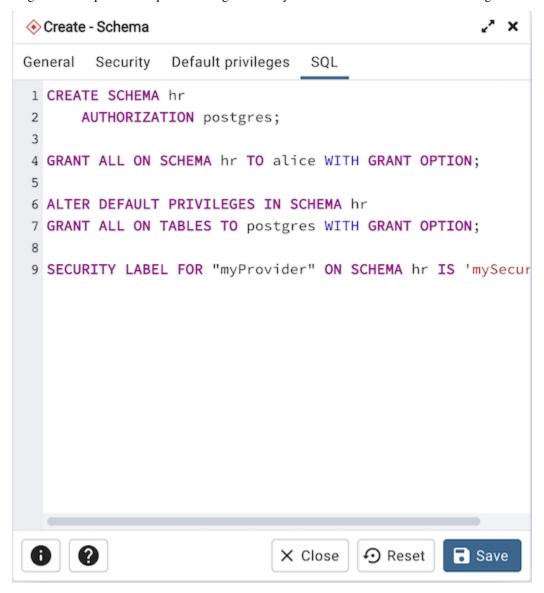
Your entries in the Schema dialog generate a SQL command (see an example below). Use the SQL tab for review;

5.20. Schema Dialog

revisit or switch tabs to make any changes to the SQL command.

### **5.20.1 Example**

The following is an example of the sql command generated by selections made in the *Schema* dialog:



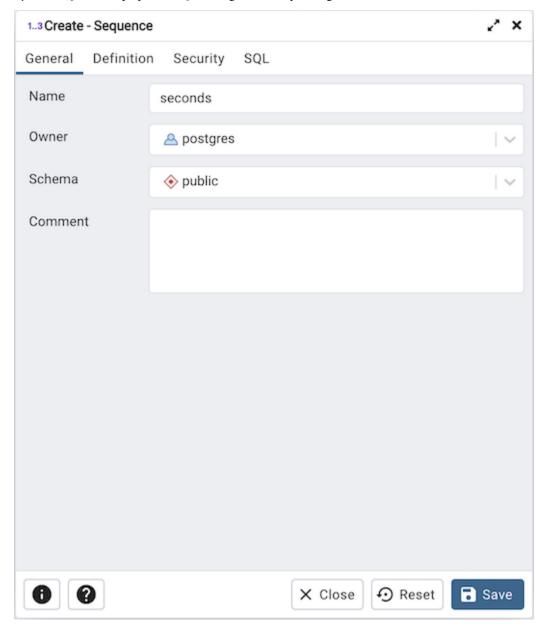
The example creates a schema named hr; the command grants *USAGE* privileges to *public* and assigns the ability to grant privileges to *alice*.

- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

### 5.21 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.

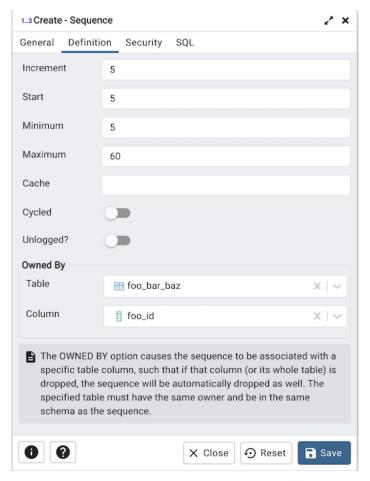


Use the fields in the *General* tab to identify a sequence:

- Use the *Name* field to add a descriptive name for the sequence. The name will be displayed in the *pgAdmin* 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.

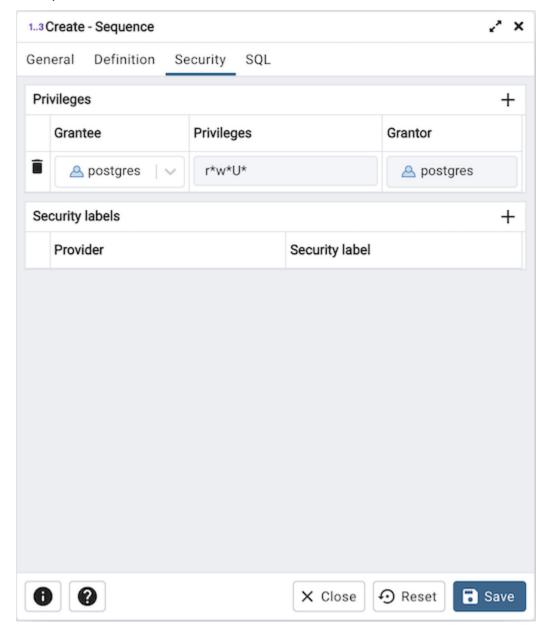


Use the fields in the *Definition* tab to define the sequence:

- 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 switch next to *Cycled* towards the *right position* to allow the sequence to wrap around when the MAX-VALUE 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*.
- Move the switch next to *Unlogged?* towards the *right position* to make the sequence Unlogged. The default is *No*. This option is available only on PostgreSQL 15 and above.

• The *OWNED BY* option causes the sequence to be associated with a specific table column, such that if that column (or its whole table) is dropped, the sequence will be automatically dropped as well. The specified table must have the same owner and be in the same schema as the sequence.

Click the Security tab to continue.



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 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.

### **5.21.1 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
1 CREATE SEQUENCE public.seconds
 2
       INCREMENT 5
 3
       START 5
 4
       MINVALUE 5
 5
       MAXVALUE 60
 6
       OWNED BY foo_bar_baz.foo_id;
 7
 8 ALTER SEQUENCE public.seconds
 9
       OWNER TO postgres;
10
11 GRANT ALL ON SEQUENCE public.seconds TO postgres WITH GRANT
                                    X Close
                                               Reset
                                                          Save
```

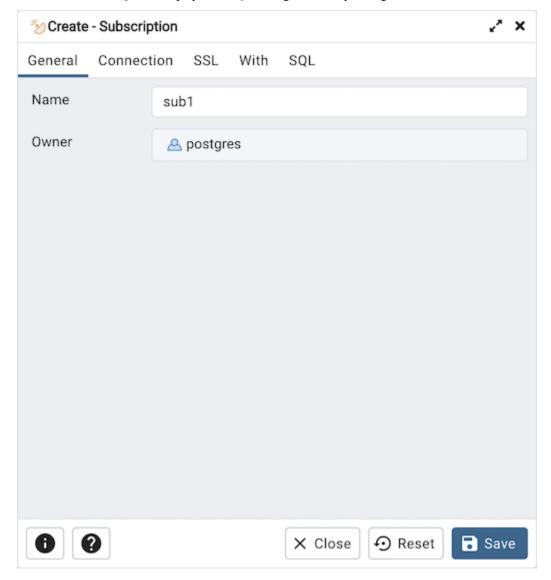
The example shown demonstrates a sequence named *seconds*. The sequence will increase in 5 second increments, and stop when it reaches a maximum value equal of 60.

- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

## 5.22 Subscription Dialog

Use the *Subscription* dialog to create a subscription. A subscription defines the connection to another database and set of publications (one or more) to which it wants to subscribe.

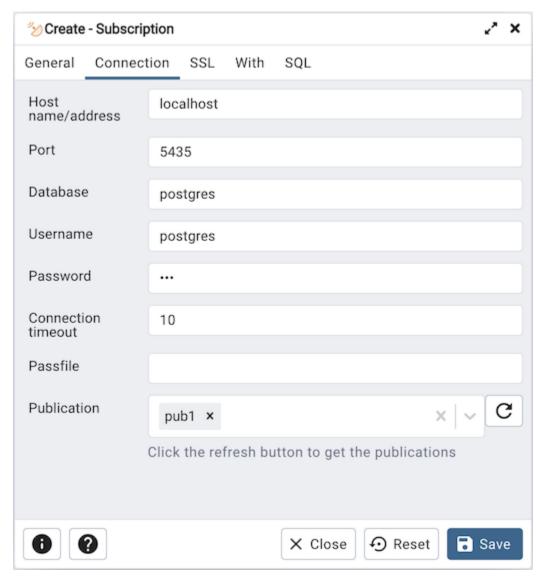
The *subscription* dialog organizes the development of a subscription through the following dialog tabs: *General*, *Connection*, *SSL* and *With*. The *SQL* tab displays the SQL code generated by dialog selections.



Use the fields in the *General* tab to identify the subscription:

- Use the *Name* field to add a descriptive name for the subscription. The name will be displayed in the *pgAdmin* tree control.
- The *Owner* field takes the name of the user automatically as current connected user. You can change the owner after creating subscription using alter subscription. Please note that owner of the subscription has superuser privileges.

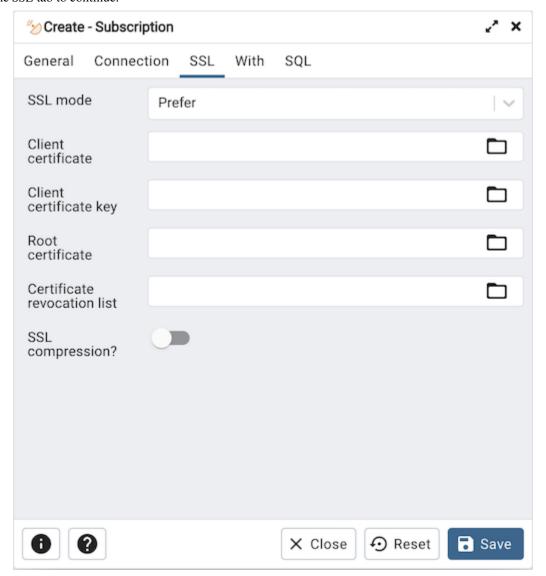
Click the Connection tab to continue.



Use the *Connection* tab to define the connection string to the publisher:

- Use the *Host name/address* field to provide the valid hostname or ip address of the publication.
- Use the Port field to provide port number to connect at Postgres Server in which publication is residing.
- Use the *Username* field to provide the name of the user to connect to the publication.
- Use the *Password* to provide the password of the user.
- Use the *Database* field to connect to the database in which publication is residing.
- Use the *Connection timeout* field to specify the maximum wait for connection, in seconds. Zero or not specified means wait indefinetly. It is not recommended to use a timeout of less than 2 seconds.
- Use the *Passfile* field to specify the location of a password file (.pgpass). A .pgpass file allows a user to login without providing a password when they connect. For more information, see Section 33.15 of the Postgres documentation.
- Use the *Publication* field to specify the publication name on the publishers to subscribe to. Click on the refresh button at the end to load the names of the existing publications and then select from the list. You can also write the name of the known publication in the field.

Click the SSL tab to continue.



Use the fields in the SSL tab to configure SSL:

• Use the drop-down list box in the *SSL* 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.

If pgAdmin is installed in Server mode (the default mode), 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.

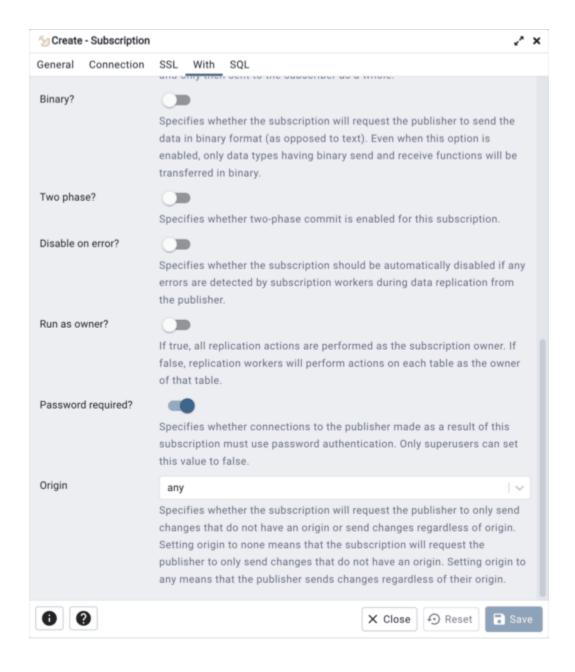
- Use the *Client certificate* field to specify the file containing the client SSL certificate. This file will replace the default ~/.postgresql/postgresql.crt if pgAdmin is installed in Desktop mode, and <STOR-AGE\_DIR>/<USERNAME>/.postgresql/postgresql.crt if pgAdmin is installed in Web mode. This 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 pgAdmin is installed in Desktop mode, and <*STOR-AGE\_DIR*>/<*USERNAME*>/.postgresql/postgresql.key if pgAdmin 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:** In Server mode, 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 pgAdmin 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.

Click the With tab to continue.





Use the *With* tab to define some parameters for a subscription:

- The *Copy data?* switch specifies whether the existing data in the publications that are being subscribed to should be copied once the replication starts. By default it is set to *Yes*.
- The *Create slot?* switch specifies whether the command should create the replication slot on the publisher. By default it is set to *Yes*. Please note: if your publisher and subscriber both are inside same PostgreSQL server then is is set to *No* by default.
- The *Enabled?* switch specifies whether the subscription should be actively replicating, or whether it should be just setup but not started yet. By default it is set to *Yes*.
- The *Connect?* specifies whether the CREATE SUBSCRIPTION should connect to the publisher at all. By default, it is set to *Yes*. Setting this to *No* will change default values of enabled, create\_slot and copy\_data to *No*.

- Use the *Slot Name* field to specify the name of the replication slot to use. By default, it uses the name of the subscription for the slot name.
- Use the *Synchronous commit* field to override the synchronous\_commit setting. By default, it is set to *off*. It is safe to use off for logical replication: If the subscriber loses transactions because of missing synchronization, the data will be sent again from the publisher.
- Use the *Streaming* field to specify whether to enable streaming of in-progress transactions for this subscription. The default value is *off*, meaning all transactions are fully decoded on the publisher and only then sent to the subscriber as a whole. This option is available only on PostgreSQL 14 and above.
- Use the *Binary?* switch to specify whether the subscription will request the publisher to send the data in binary format (as opposed to text). The default is *false*. This option is available only on PostgreSQL 14 and above.
- Use the *Two phase?* switch to specify whether two-phase commit is enabled for this subscription. The default is *false*. This option is available only on PostgreSQL 15 and above.
- Use the *Disable on error?* switch to specify whether the subscription should be automatically disabled if any errors are detected by subscription workers during data replication from the publisher. The default is *false*. This option is available only on PostgreSQL 15 and above.
- Move the *Run as owner?* switch to *true* position to specify all replication actions are performed as the subscription owner. If *false*, replication workers will perform actions on each table as the owner of that table. The default is *false*. This option is available only on PostgreSQL 16 and above.
- Use the Password required? to specify whether connections to the publisher made as a result of this subscription
  must use password authentication. This setting is ignored when the subscription is owned by a superuser. The
  default is true. Only superusers can set this value to false. This option is available only on PostgreSQL 16 and
  above.
- Use the *Origin* to specify whether the subscription will request the publisher to only send changes that don't have an origin or send changes regardless of origin. The default is *any*. This option is available only on PostgreSQL 16 and above.

Click the SQL tab to continue.

Your entries in the *Subscription* 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 Subscription dialog:



The example creates a subscription named *sub1* that is owned by *postgres*. It will replicate the data from the publication *pub1*.

- Click the Info button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore all the default settings.

## 5.23 Synonym Dialog

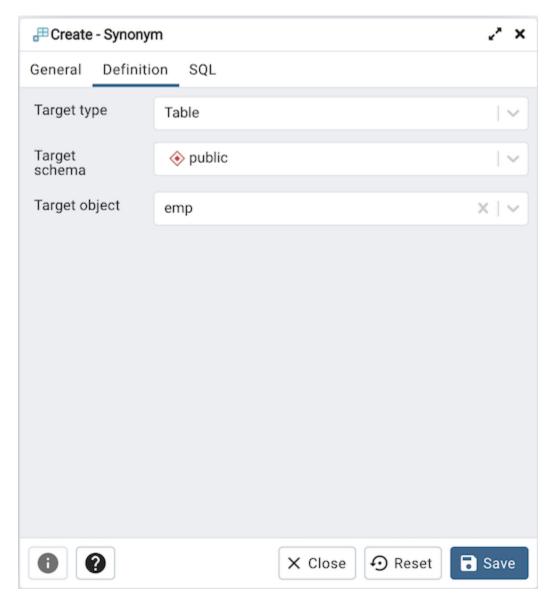
Use the *Synonym* dialog to substitute the name of a target object with a user-defined synonym.

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 *pgAdmin* 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.



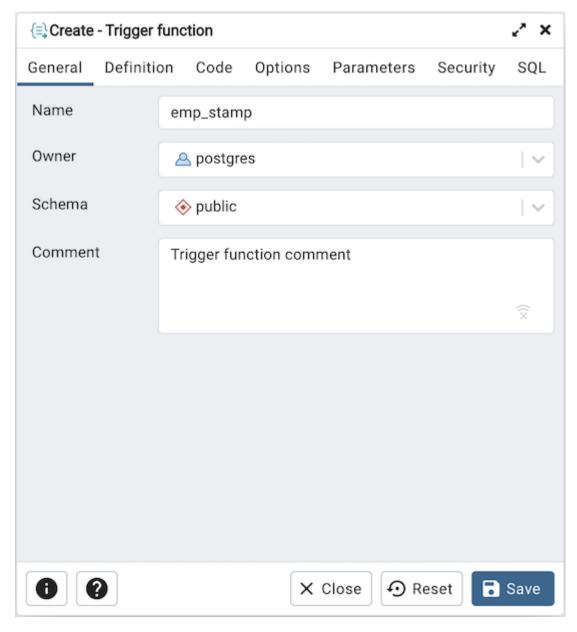
The example creates a synonym for the *emp* table named *emp\_hist*.

- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

# **5.24 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.

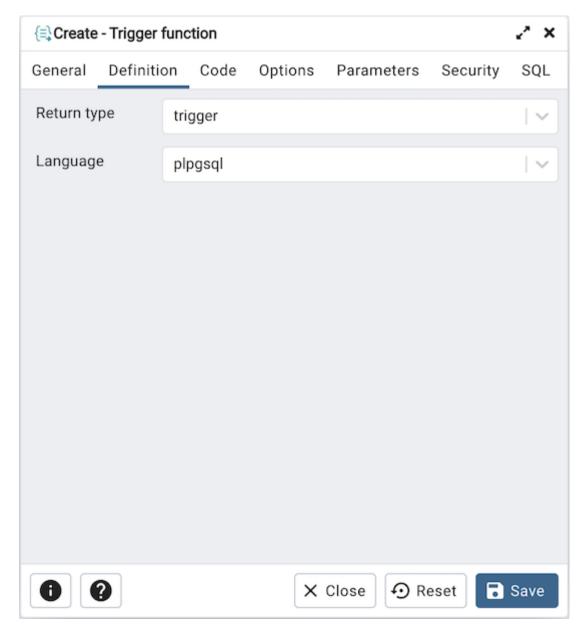
The *Trigger function* dialog organizes the development of a trigger function through the following dialog tabs: *General*, *Definition*, *Code*, *Options*, *Parameters* and *Security*. The *SQL* tab displays the SQL code generated by dialog selections.



Use the fields in the *General* tab to identify the trigger function:

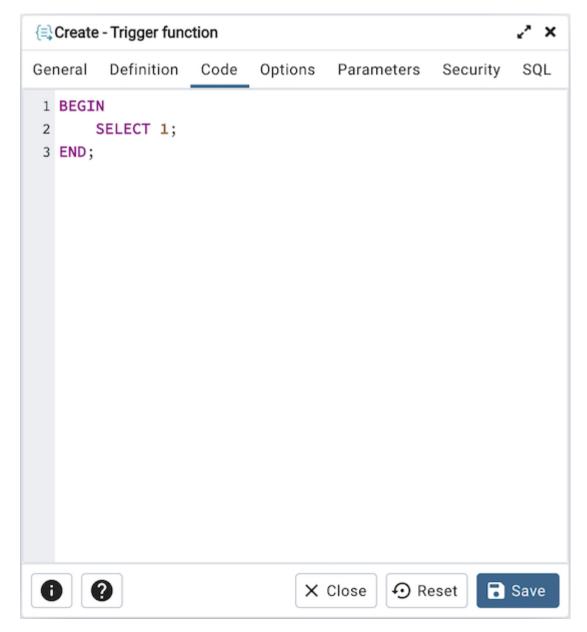
- Use the *Name* field to add a descriptive name for the trigger function. The name will be displayed in the *pgAdmin* tree control. Please note that trigger functions will be invoked in alphabetical order.
- Use the drop-down listbox next to Owner to select the role that will own the trigger function.
- 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.

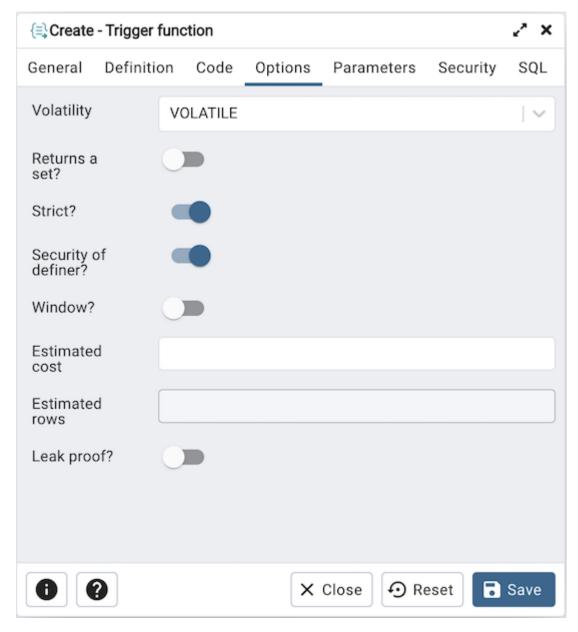


Use the fields in the *Definition* tab to define the trigger function:

- 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.

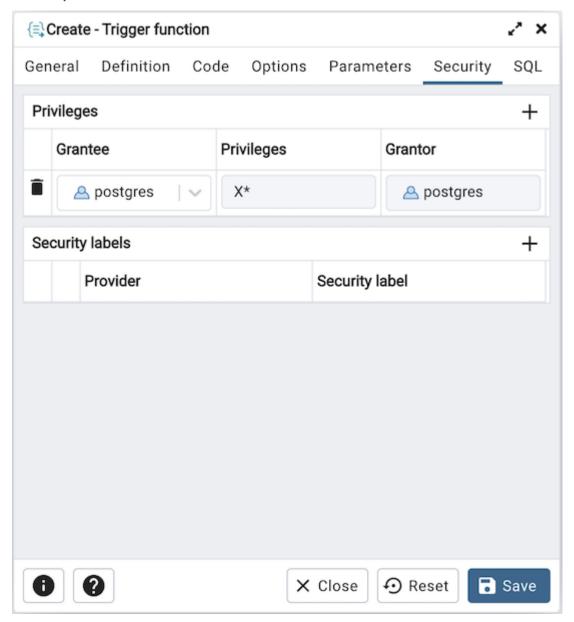


Use the fields in the *Parameters* tab to specify settings that will be applied when the trigger function is invoked. Click the *Add* icon (+) to add a *Name/Value* pair to the table below.

- 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 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.

### **5.24.1 Example**

The following is an example of the sql command generated by user selections in the Trigger function dialog:

```
Create - Trigger function
                                                                           2 X
General Definition Code
                        Options
                                 Parameters
                                                      SQL
                                             Security
1 CREATE FUNCTION public.emp_stamp()
2
      RETURNS trigger
      LANGUAGE 'plpgsql'
3
      VOLATILE NOT LEAKPROOF STRICT SECURITY DEFINER
      SET array_nulls=false
6 AS $BODY$
7 BEGIN
      SELECT 1;
9 END;
10 $BODY$;
11
12 ALTER FUNCTION public.emp_stamp()
      OWNER TO postgres;
13
14
15 GRANT EXECUTE ON FUNCTION public.emp_stamp() TO postgres WITH GRANT OPTION;
16
17 REVOKE ALL ON FUNCTION public.emp_stamp() FROM PUBLIC;
18
19 COMMENT ON FUNCTION public.emp_stamp()
20
      IS 'Trigger function comment';
0
                                                   X Close
                                                             Reset
                                                                        Save
```

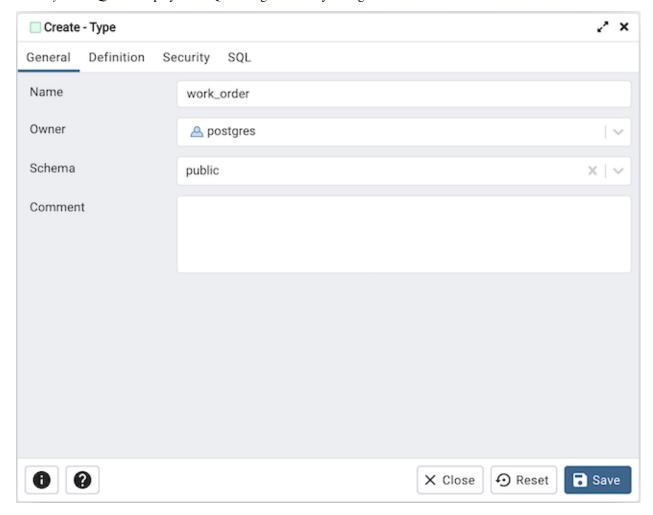
The example shown demonstrates creating a trigger function named *emp\_stamp*.

- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

# 5.25 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 *SOL* 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 *pgAdmin* 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.

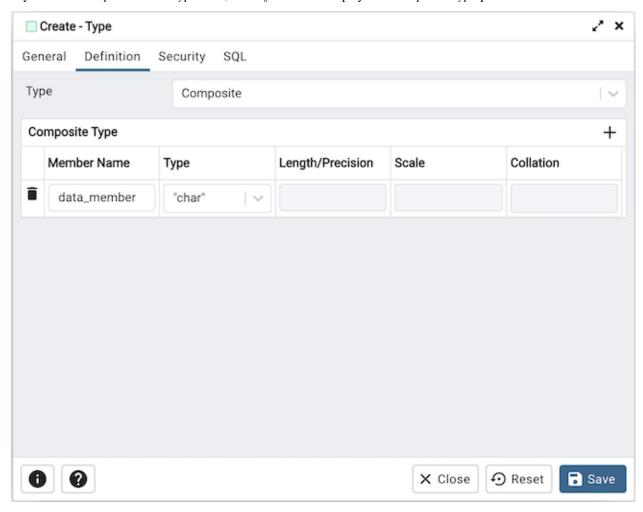
There are five data types:

· Composite Type

5.25. Type Dialog 285

- 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:

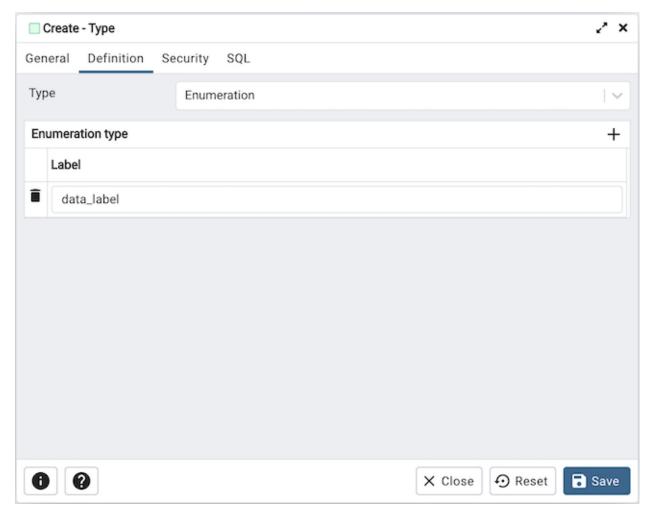


Click the Add icon (+) to provide attributes of the type. Fields on the General panel are context sensitive and may be disabled.

- 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:



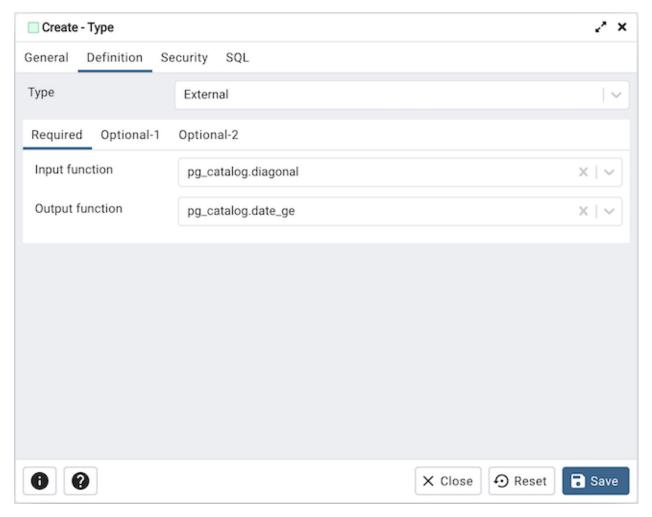
Click the *Add* icon (+) to provide a label for the type.

• 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:

5.25. Type Dialog 287



#### 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.

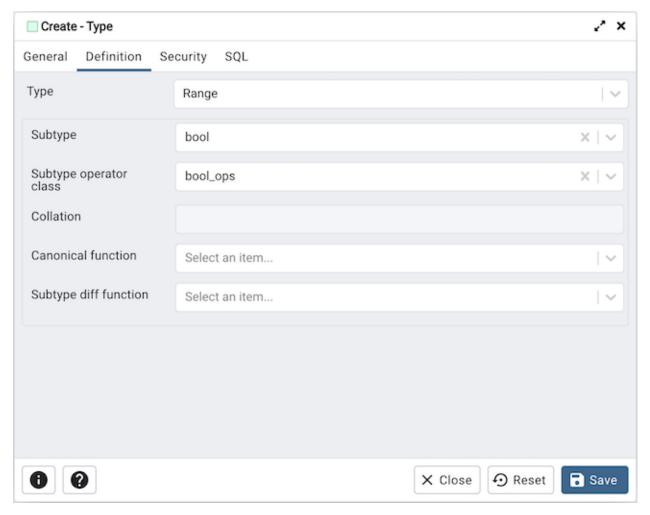
- Use the optional *Internal length* to specify a value for internal representation.
- Move the *Variable?* switch to specify the internal representation is of variable length (VARIABLE). The default is a fixed length positive integer.
- Specify a default value in the optional *Default* field in cases 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*.

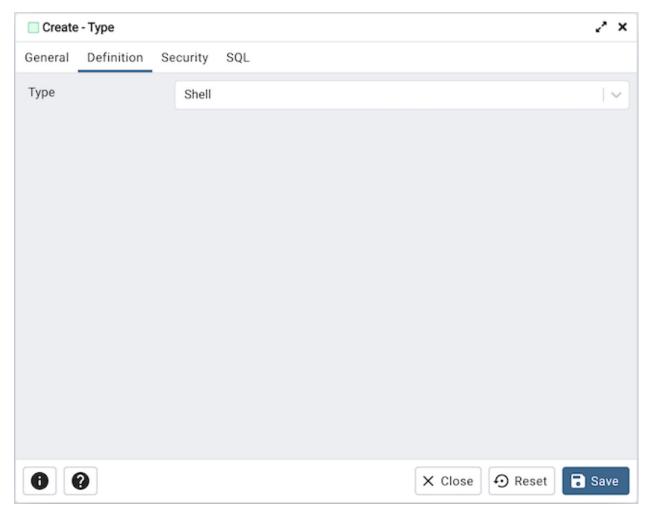
If you select *Range* in the *Type* field, the *Definition* tab displays the *Range* panel. Fields on the *Range* panel are context-sensitive and may be disabled.

5.25. Type Dialog 289



- 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.

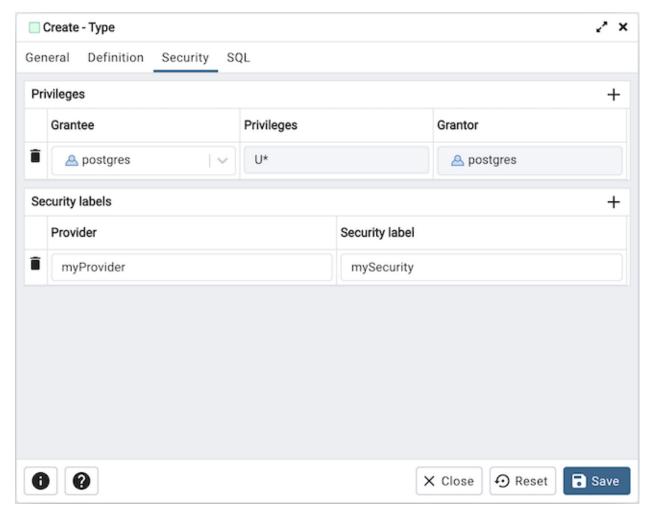
If you select *Shell* in the *Type* field, the *Definition* tab displays the *Shell* panel:



A shell type is a placeholder for a type and has no parameters.

Click the Security tab to continue.

5.25. Type Dialog 291



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.

### **5.25.1 Example**

The following is an example of a sql command generated by user selections made in the *Type* dialog:

```
. ×
Create - Type
General
        Definition
                            SQL
                   Security
1 CREATE TYPE public.work_order
2 (
3
       INPUT = pg_catalog.diagonal,
       OUTPUT = pg_catalog.date_ge
 4
5);
 6
7 ALTER TYPE public.work_order
8
       OWNER TO postgres;
 9
10 GRANT USAGE ON TYPE public.work_order TO postgres WITH GRANT OPTION;
11
12
13 SECURITY LABEL FOR "myProvider" ON TYPE public.work_order IS 'mySecurity';
                                                                 Reset
                                                                            Save
                                                       X Close
```

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.

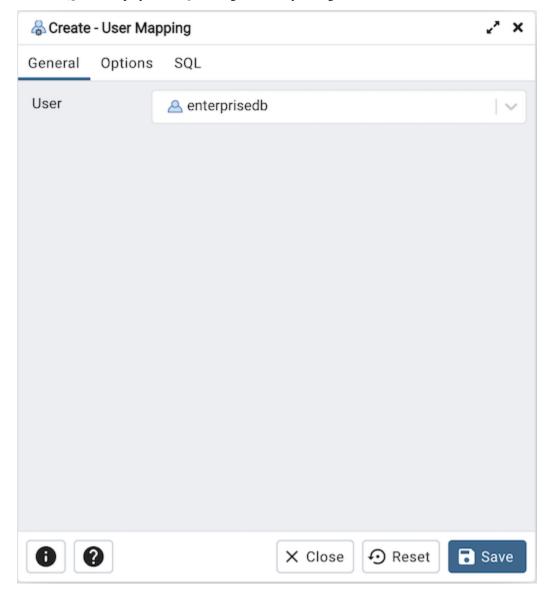
- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

5.25. Type Dialog 293

## 5.26 User Mapping Dialog

Use the *User Mapping* dialog to define a new mapping of a user to a foreign server.

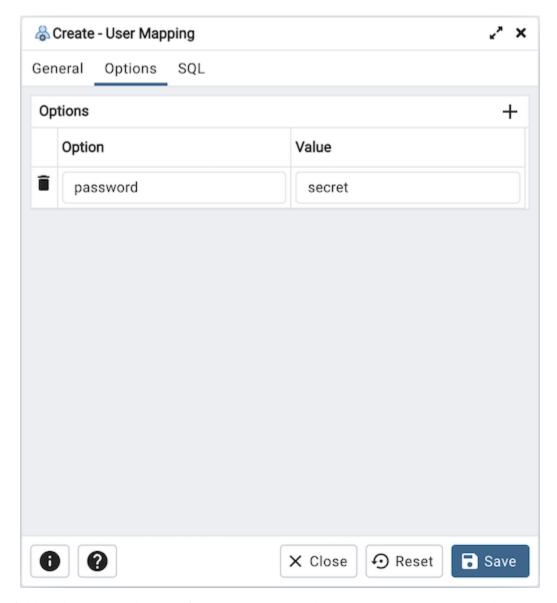
The *User Mapping* dialog organizes the development of a user mapping through the following dialog tabs: *General* and *Options*. The *SQL* tab displays the SQL code generated by dialog selections.



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.

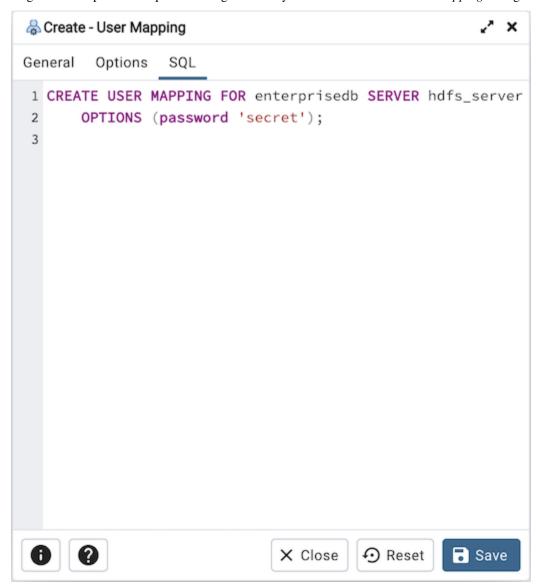
Click *Add* to specify each additional option/value pair; to discard an option, click the trash icon to the left of the row and confirm deletion in the *Delete Row* popup.

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.

### **5.26.1 Example**

The following is an example of the sql command generated by user selections in the *User Mapping* dialog:



The example shown demonstrates a user mapping for the *hdfs\_server*. The user is *CURRENT\_USER* with a password *secret*.

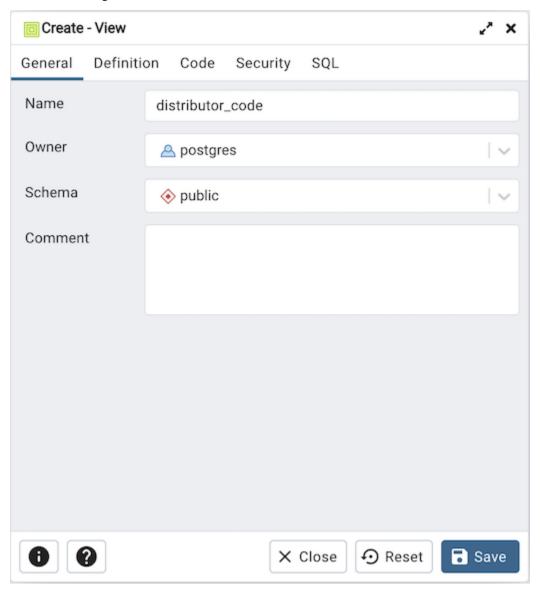
- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

### 5.27 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.



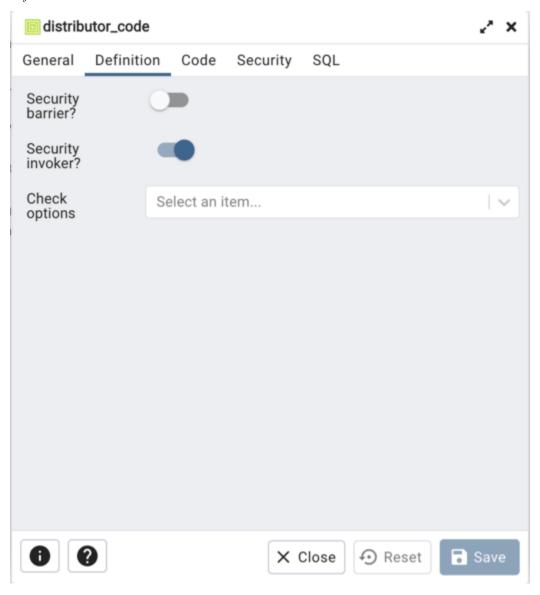
Use the fields in the *General* tab to identify a view:

- 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 *pgAdmin* 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.

5.27. View Dialog 297

• 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 Rules and Privileges of the PostgreSQL documentation.
- Set the *Security Invoker* switch to *Yes* to indicate that the underlying base relations are to be checked against the privileges of the user of the view rather than the view owner. This option is available from PostgreSQL 15 onwards.
- 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.

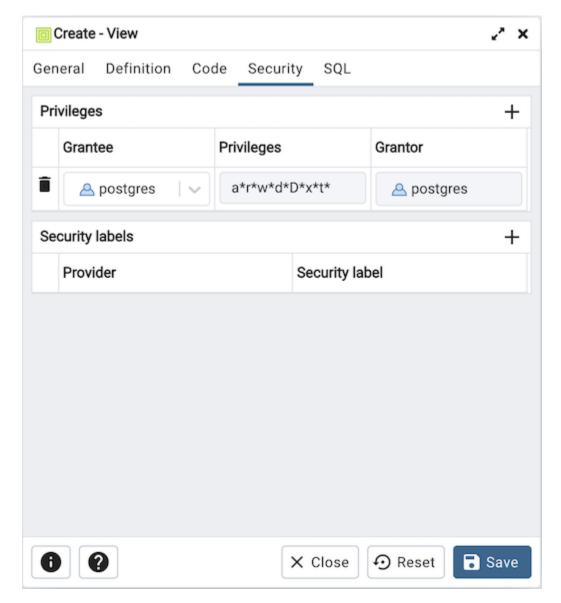
Click the Code tab to continue.



Use the workspace in the *Code* tab to write a query to create a view.

Click the Security tab to continue.

5.27. View Dialog 299



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 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.

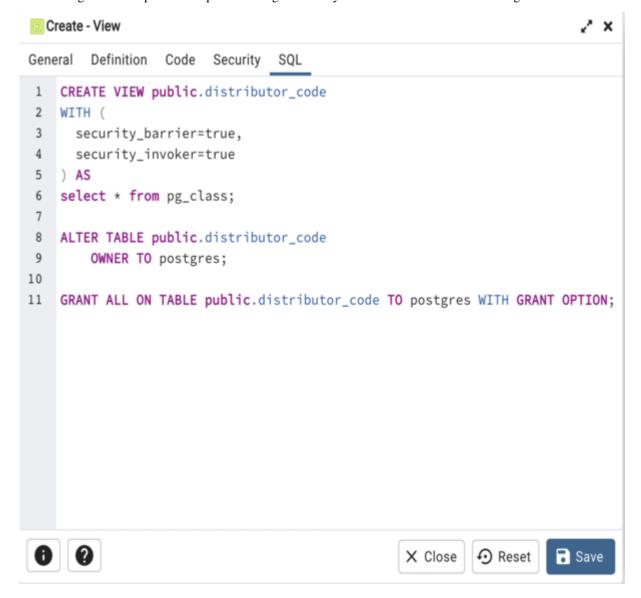
Click the *Add* icon (+) to assign additional security labels; to discard a security label, click the trash icon to the left of the row and confirm deletion in the *Delete Row* popup.

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.

### **5.27.1 Example**

The following is an example of the sql command generated by user selections in the *View* dialog:



The example shown demonstrates creating a view named distributor\_code.

• Click the *Info* button (i) to access online help.

5.27. View Dialog 301

- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

## Creating or Modifying a Table

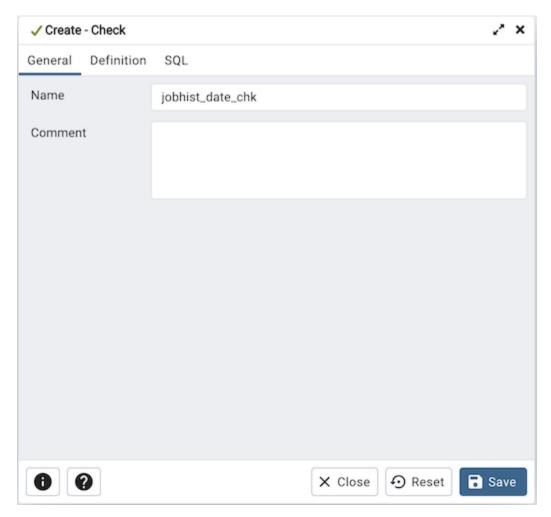
pgAdmin 4 provides dialogs that allow you to modify all table properties and attributes.

To access a dialog that allows you to create a database object, right-click on the object type in the pgAdmin 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*...

## 6.1 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.

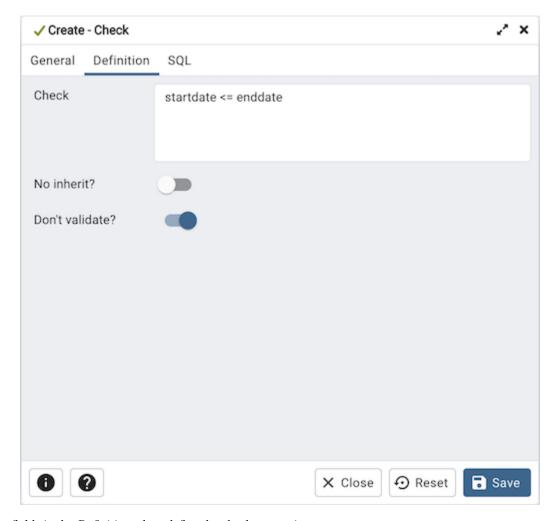
The *Check* dialog organizes the development of a check constraint through the *General* and *Definition* tabs. The *SQL* tab displays the SQL code generated by dialog selections.



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 *pgAdmin* 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 that this constraint is not automatically inherited by a table's children. The default is *No*, meaning that the constraint will be inherited by any children.
- 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*.

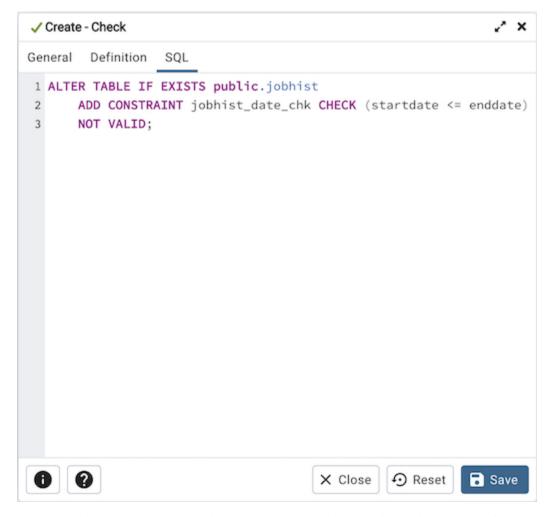
Click the SQL tab to continue.

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:

6.1. Check Dialog 305



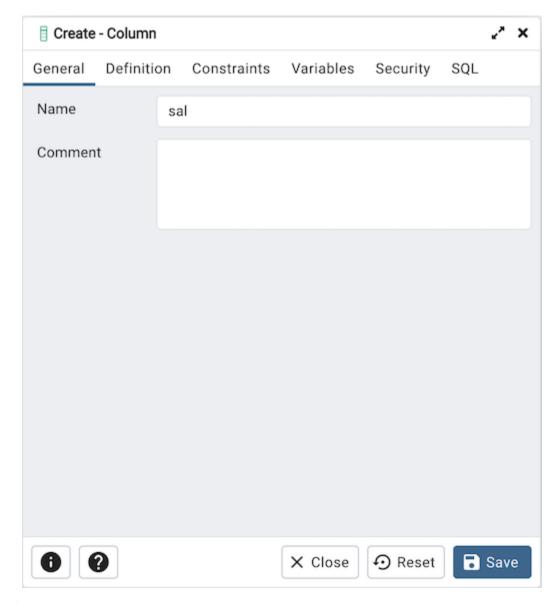
The example shown demonstrates creating a check constraint named *jobhist\_date\_chk* on the *startdate* column of the *jobhist* table.

- Click the Info button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

## 6.2 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.

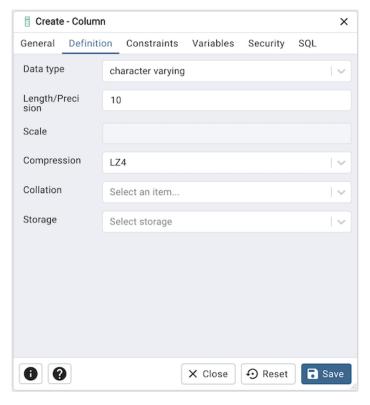


Use the fields in the *General* tab to identify the column:

- Use the *Name* field to add a descriptive name for the column. The name will be displayed in the *pgAdmin* tree control. This field is required.
- Store notes about the column in the *Comment* field.

Click the Definition tab to continue.

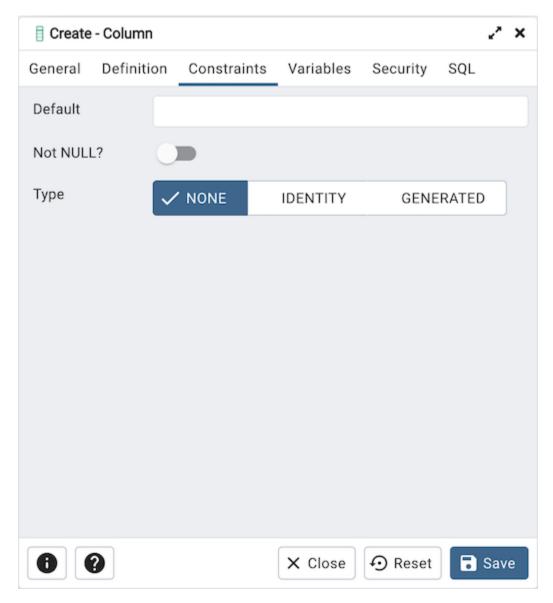
6.2. Column Dialog 307



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.
- Use the drop-down listbox next to *Compression* to set the compression method for the column. Compression is supported only for variable-width data types, and is used only when the column's storage mode is main or extended. This option is available from v14 and above.
- Use the drop-down listbox next to *Storage* to set the storage mode for the column. This option is available from v16 and above.

Click the Constraints tab to continue.

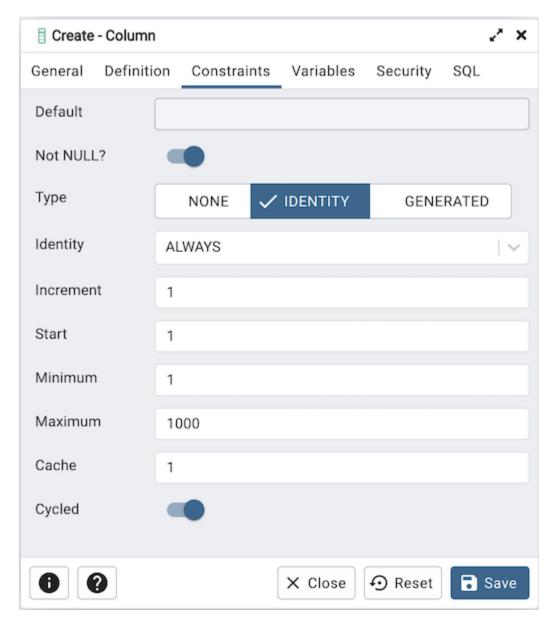


Use the fields in the *Constraints* tab to specify constraints for the column. (Fields are disabled if inapplicable.)

- 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.*
- ullet Use the  $\mathit{Type}$  field to specify the column type (NONE/IDENTITY/GENERATED). The default is NONE.

Click the *IDENTITY* type to create Identity column.

6.2. Column Dialog 309

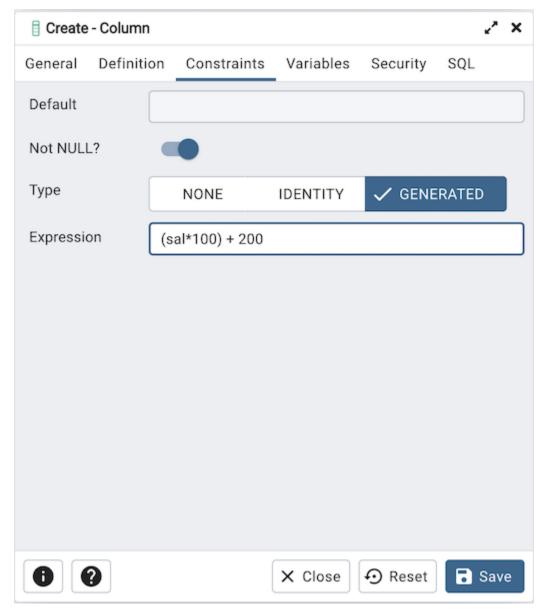


Use the following fields to create *IDENTITY* column. Identity columns are applicable for PG/EPAS version 10 and above.

- 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.

6.2. Column Dialog 311

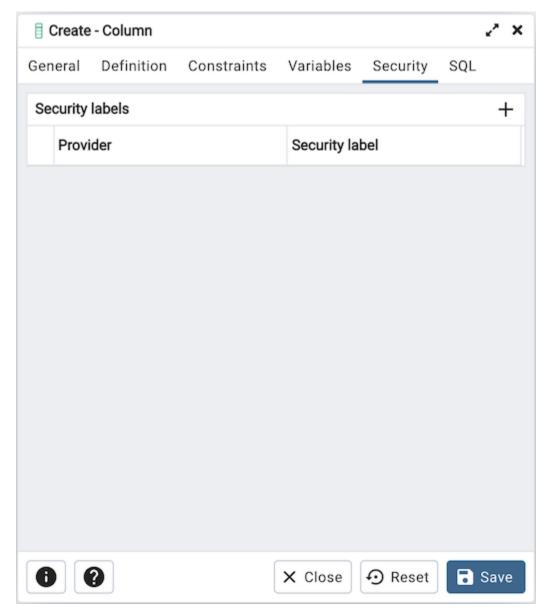


Use the *Variables* tab to specify the number of distinct values that may be present in the column; this value overrides estimates made by the ANALYZE command. Click the *Add* icon (+) to add a *Name/Value* pair:

- 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 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.

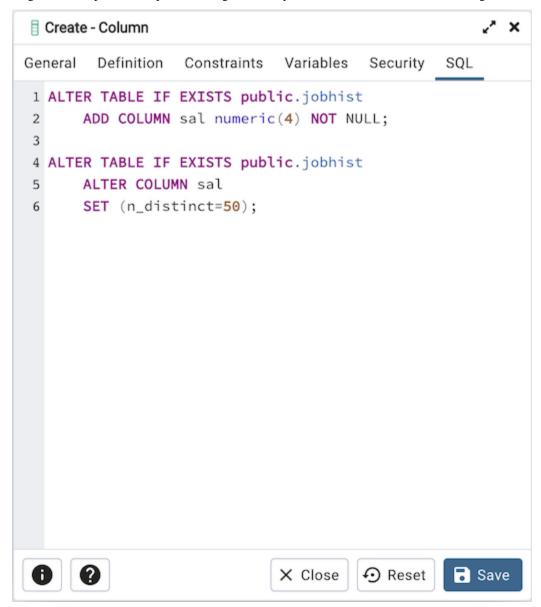
Click the SQL tab to continue.

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.

6.2. Column Dialog 313

### 6.2.1 Example

The following is an example of the sql command generated by user selections in the Column dialog:



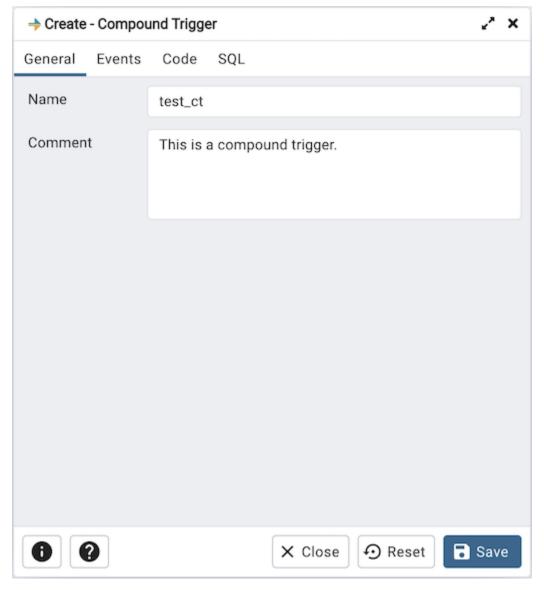
The example shown demonstrates creating a column named sal in the table named jobhist.

- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

# 6.3 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.



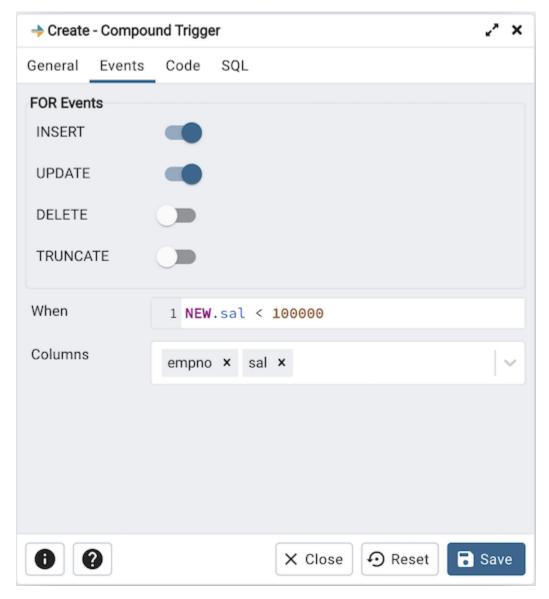
Use the fields in the *General* tab to identify the compound trigger:

- 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 *pgAdmin* 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. You can select one of the four options available.

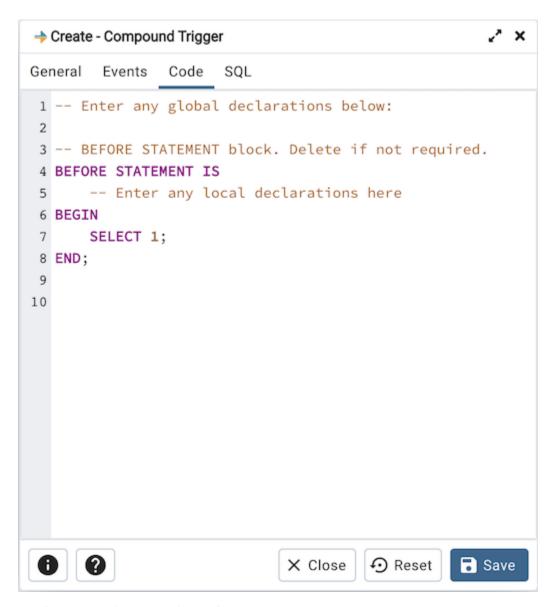
Click the *Events* tab to continue.



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.

#### 6.3.1 Example

The following is an example of the sql command generated by user selections in the Compound Trigger dialog:

```
Create - Compound Trigger
General
                Code
        Events
                       SOL
 1 CREATE OR REPLACE TRIGGER test_ct
 2
       FOR INSERT OR UPDATE OF empno, sal
 3
       ON public.emp
 4
       WHEN (NEW.sal < 100000)
       COMPOUND TRIGGER
 6
  -- Enter any global declarations below:
 7
 8 -- BEFORE STATEMENT block. Delete if not required.
 9 BEFORE STATEMENT IS
       -- Enter any local declarations here
10
11 BEGIN
12
       SELECT 1;
13 END;
14
15
16 END test_ct;
17
18 COMMENT ON TRIGGER test_ct ON public.emp
19
       IS 'This is a compound trigger.';
                               X Close
                                         Reset
                                                    Save
```

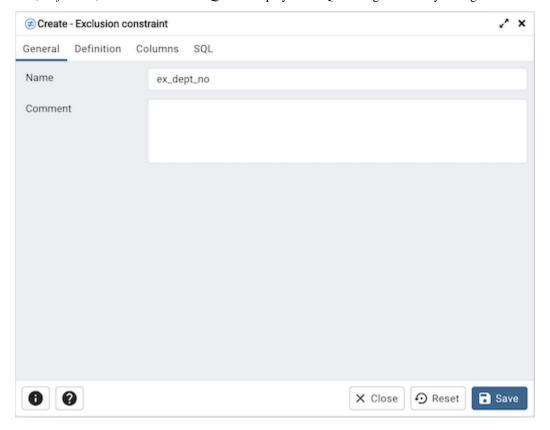
The example demonstrates creating a compound trigger named *test\_ct*.

- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

# **6.4 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.

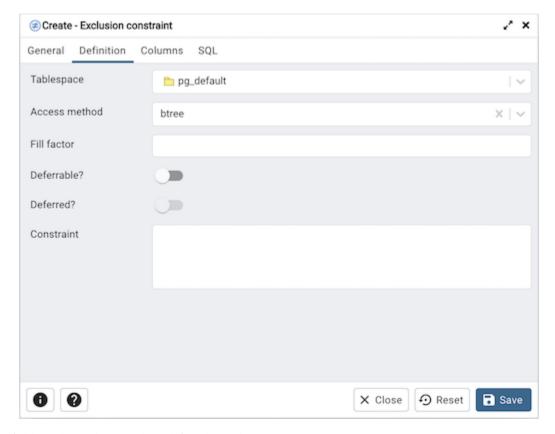
The *Exclusion constraint* dialog organizes the development of an exclusion constraint through the following dialog tabs: *General*, *Definition*, and *Columns*. The *SQL* tab displays the SQL code generated by dialog selections.



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 *pgAdmin* tree control.

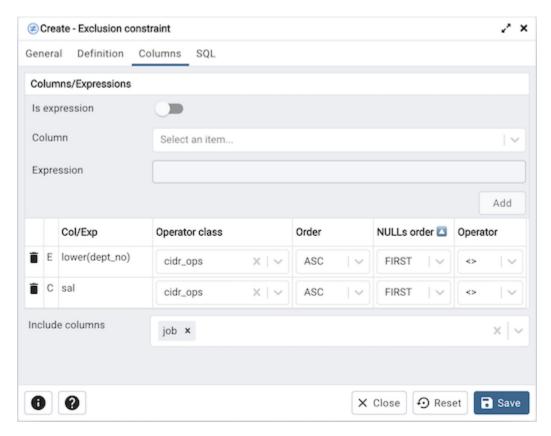
Click the *Definition* tab to continue.



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 *Is 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.

#### 6.4.1 Example

The following is an example of the sql command generated by user selections in the Exclusion Constraint dialog:

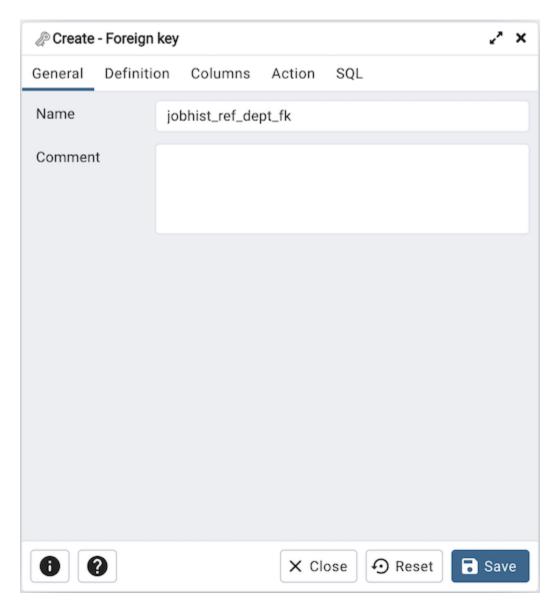
The example shown demonstrates creating an exclusion constraint named *ex\_dept\_no*. The constraint uses a btree index.

- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

## 6.5 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 *pgAdmin* 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 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:

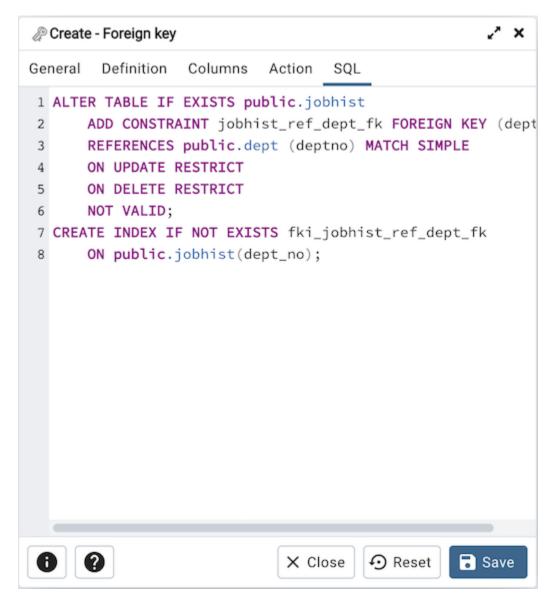
NO AC-	Produce an error indicating that the deletion or update will create a foreign key constraint violation. If the constraint is deferred, this error will be produced at constraint check time if any referencing rows still
TION	exist. This is the default.
RE- STRICT	Throw an error indicating that the deletion or update would create a foreign key constraint violation. This is the same as NO ACTION except that the check is not deferrable.
CAS- CADE	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 DE- FAULT	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.

### 6.5.1 Example

The following is an example of the sql command generated by user selections in the Foreign key dialog:



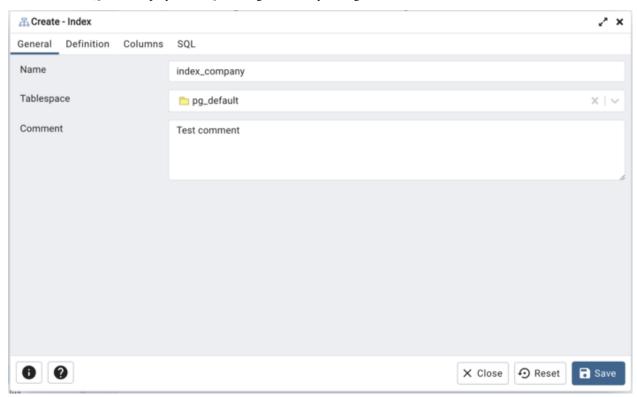
The example shown demonstrates creating a foreign key constraint named *jobhist\_ref\_dept\_fk* that matches values in the *jobhist* table *dept\_no* column with those of the *dept* table *deptno* column.

- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

# 6.6 Index Dialog

Use the *Index* dialog to create an index on a specified table or materialized view.

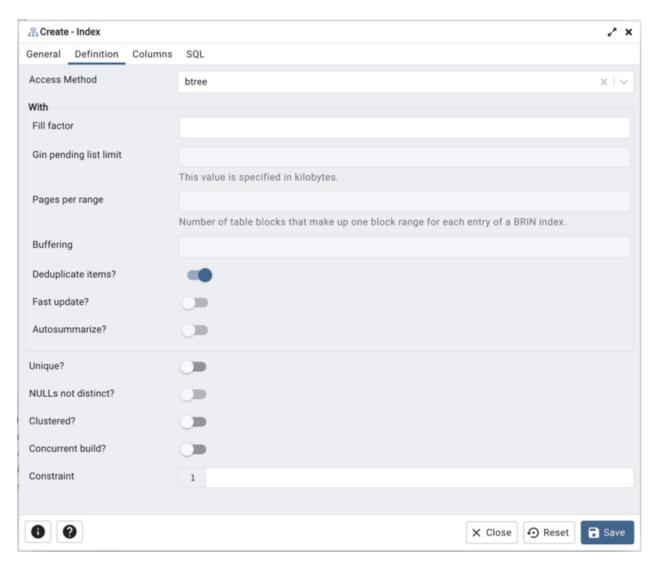
The *Index* dialog organizes the development of a index through the following dialog tabs: *General*, *Definition*, and *Columns*. The *SQL* tab displays the SQL code generated by dialog selections.



Use the fields in the *General* tab to identify the index:

- Use the *Name* field to add a descriptive name for the index. The name will be displayed in the *pgAdmin* 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.



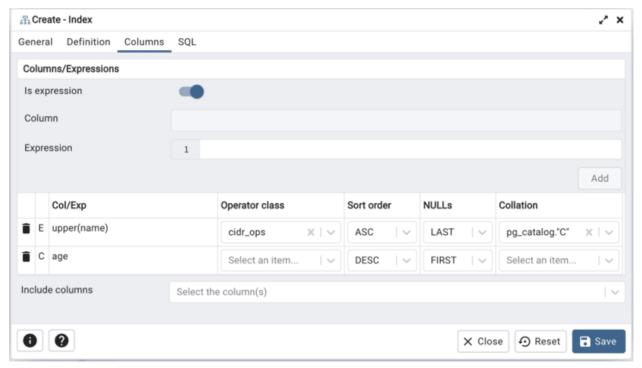
Use the fields in the *Definition* tab to define the index:

- 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 twodimensional geometric data types and nearest-neighbor searches.
  - Select gin to create a GIN index. A GIN index may improve performance 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.

6.6. Index Dialog 331

- Use the *Gin pending list limit* field to specify the maximum size of a GIN index's pending list, which is used when fastupdate is enabled. This value is specified in kilobytes.
- Use the *Pages per range* field to specify the number of table blocks that make up one block range for each entry of a BRIN index.
- Select Buffering to specify whether the buffering build technique is used to build the index. The default is Auto
- Move the switch next to *Deduplicate items?* towards the *right position* to control usage of the B-tree deduplication technique. The default is *Yes*. This option is available only on PostgreSQL 13 and above.
- Move the switch next to *Fast update*? towards the *right position* to control usage of the fast update technique. The default is *Yes*.
- Move the switch next to *Autosummarize* towards the *right position* to define whether a summarization run is queued for the previous page range whenever an insertion is detected on the next one. The default is *No*
- Move the switch next to *Unique?* towards the *right 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 switch next to *NULLs not distinct?* towards the *right position* to treat null values as not distinct. The default is\*No\*. This option is available only on PostgreSQL 15 and above.
- 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.

Click the Columns tab to continue.



Use the fields in the *Columns* tab to specify which column(s) or expression(s) the index queries. Use the *Is expression*? switch to enable expression sql 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 enabled, use the drop-down listbox to select an available *Operator class* to specify the type of action performed on the column.
- If enabled, use the drop-down listbox to select *Sort order*:
  - Select ASC to specify an ascending sort order (the default);
  - Select *DESC* to specify a descending sort order.
- If enabled, use the drop-down listbox to select 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 constraint. 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.

#### 6.6.1 Example

The following is an example of the sql command generated by user selections in the *Index* dialog:

```
R Create - Index
General Definition
                  Columns
                           SOL
  CREATE INDEX index_company
        ON public.company USING btree
2
        ((upper(name)) COLLATE pg_catalog."C" cidr_ops ASC NULLS LAST, age DESC NULLS FIRST)
3
4
        WITH (deduplicate_items=True)
5
        TABLESPACE pg_default
6
        WHERE id<10;
    COMMENT ON INDEX public.index_company
        IS 'Test comment';
                                                                                                  Save
                                                                              X Close
                                                                                       • Reset
```

The example shown demonstrates creating an index named *index\_sal* that indexes the values in the *job* column of the *jobhist* table.

• Click the *Info* button (i) to access online help.

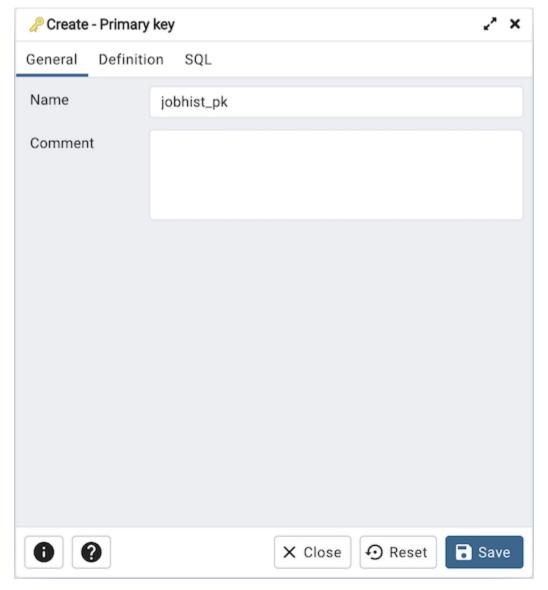
6.6. Index Dialog 333

- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

### 6.7 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.

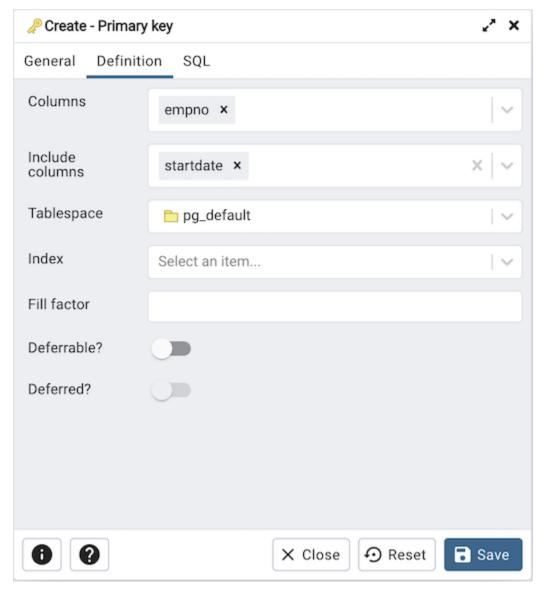
The *Primary key* dialog organizes the development of a primary key constraint through the *General* and *Definition* tabs. The *SQL* tab displays the SQL code generated by dialog selections.



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 *pgAdmin* 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.

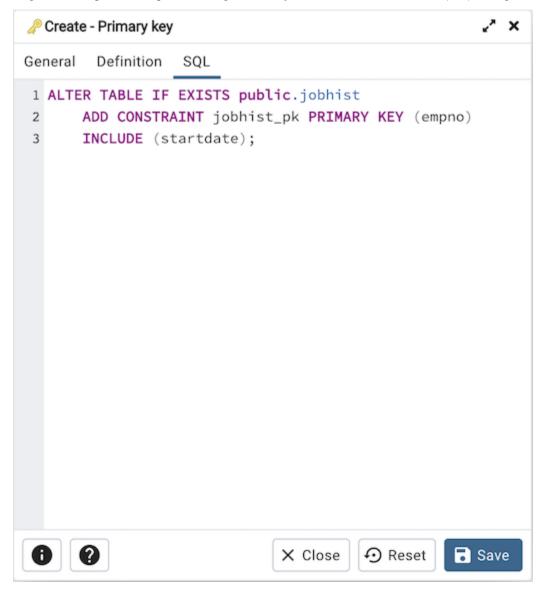
- Use the *Fill Factor* field to specify a fill factor for the table and index. The fill factor for a table is a percentage between 10 and 100. 100 (complete packing) is the default.
- Move the *Deferrable?* switch to the *Yes* position to specify the timing of the constraint is deferrable and can be 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.

### 6.7.1 Example

The following is an example of the sql command generated by user selections in the *Primary key* dialog:



The example shown demonstrates creating a primary key constraint named *jobhist\_pk* on the *empno* column of the *jobhist* table.

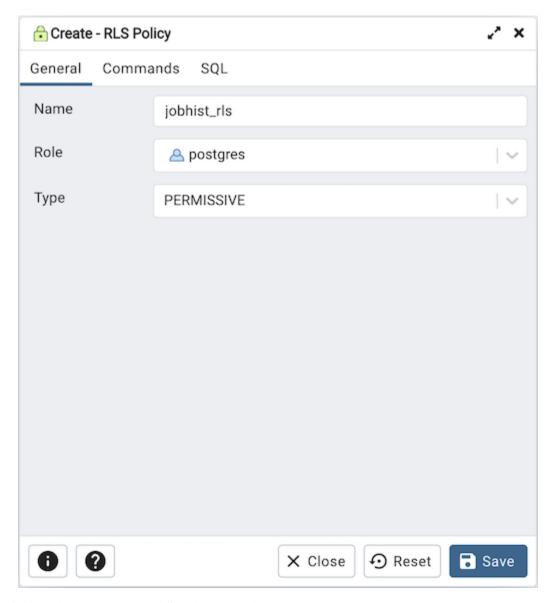
- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

## 6.8 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.

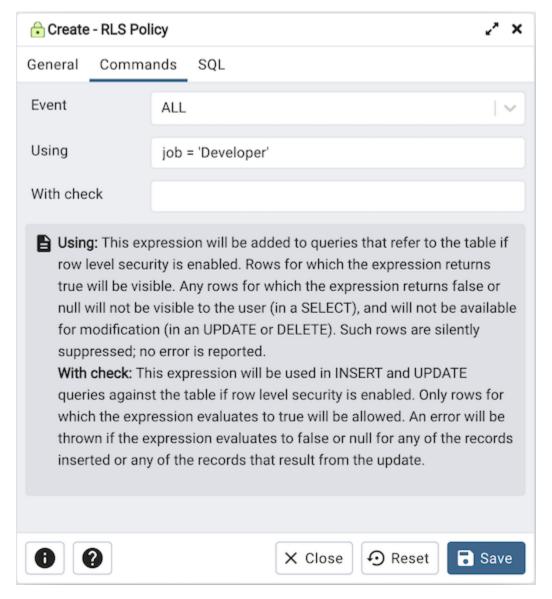
The *RLS Policy* dialog creates a Row Level Security Policy through the following dialog tabs: *General*, and *Commands*. The *SQL* tab displays the SQL code generated by dialog selections.



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:

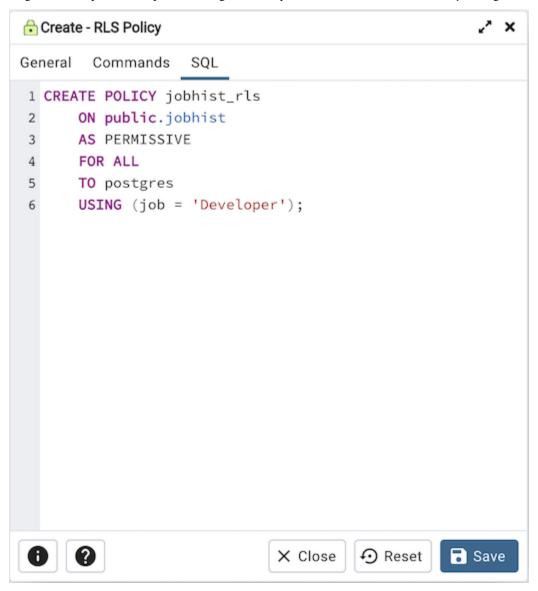
- Use the drop-down listbox next to *Event* to select the command to which policy applies. Valid options are ALL, SELECT, INSERT, UPDATE, and DELETE. Default is ALL.
- Use the *Using* field to add a SQL conditional expression returning boolean. This expression will be added to queries that refer to the 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.

#### 6.8.1 Example

The following is an example of the sql command generated by user selections in the RLS Policy dialog:



The example shown demonstrates creating a RLS Policy named *jobhist\_rls* that applies the Row Level Security on the *jobhist* table.

- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

# 6.9 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.

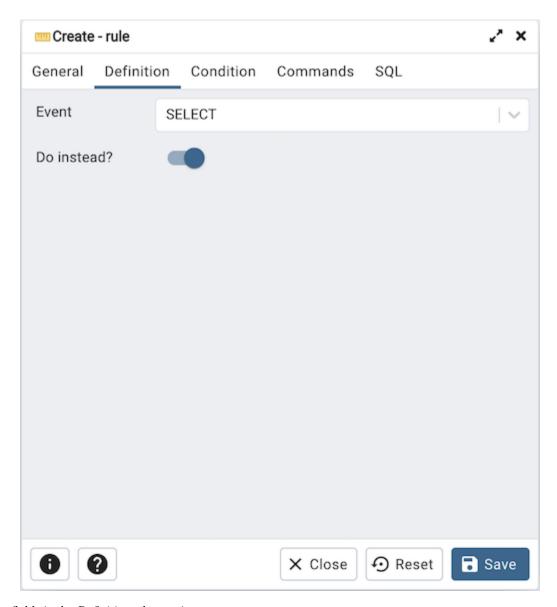


Use the fields in the *General* tab to identify the rule:

- Use the *Name* field to add a descriptive name for the rule. The name will be displayed in the *pgAdmin* tree control. Multiple rules on the same table are applied in alphabetical name order.
- Store notes about the rule in the *Comment* field.

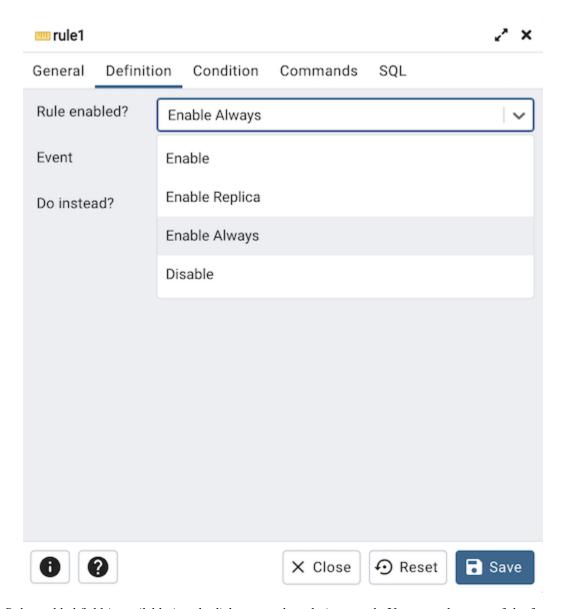
Click the Definition tab to continue.

6.9. Rule Dialog 341



Use the fields in the *Definition* tab to write parameters:

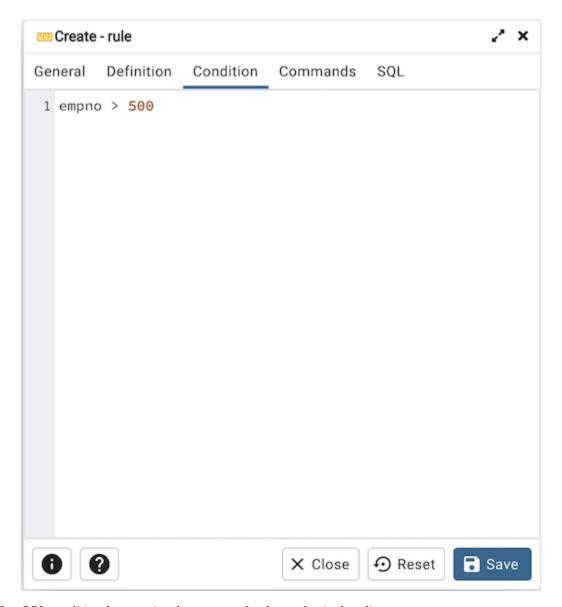
- Click inside the *Event* field to select the type of event that will invoke the rule; event may be *Select*, *Insert*, *Update*, or *Delete*.
- Move the *Do Instead* switch to *Yes* indicate that the commands should be executed instead of the original command; if Do Instead specifies *No*, the rule will be invoked in addition to the original command.



• *Rule enabled* field is available in rule dialog once the rule is created. You can select one of the four options available.

Click the *Condition* tab to continue.

6.9. Rule Dialog 343



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.

6.9. Rule Dialog 345

### 6.9.1 Example

The following is an example of the sql command generated by user selections in the *Rule* dialog:

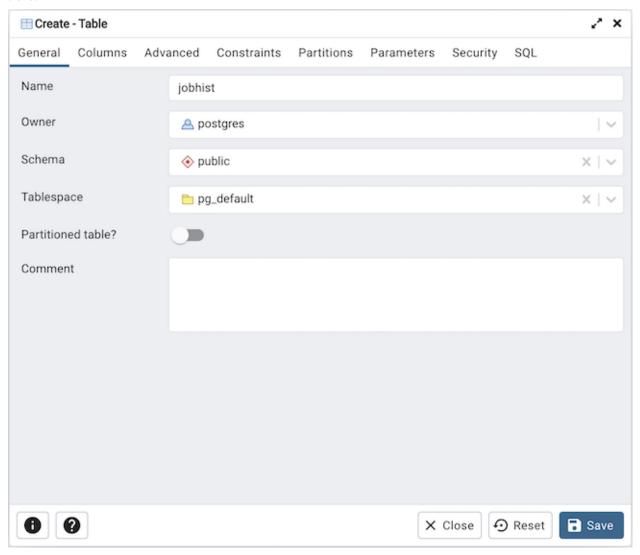


- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

## 6.10 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*, *Parition*, *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:

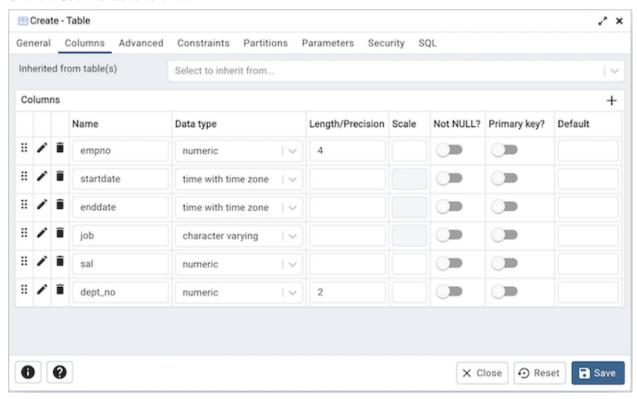
- Use the *Name* field to add a descriptive name for the table. A table cannot have the same name as any existing table, sequence, index, view, foreign table, or data type in the same schema. The name specified will be displayed in the *pgAdmin* tree control. This field is required.
- Select the owner 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

6.10. Table Dialog 347

for PostgreSQL 10 and above.

• Store notes about the table in the *Comment* field.

Click the Columns tab to continue.



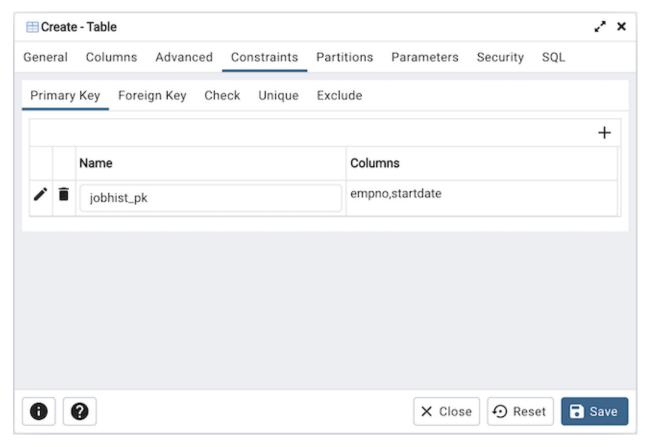
Use the drop-down listbox next to *Inherited from table(s)* to specify any parent table(s); the table will inherit columns from the selected parent table(s). Click inside the *Inherited from table(s)* field to select a table name from a drop-down list. Repeat to add any other parent tables. Delete a 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/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.
- 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.

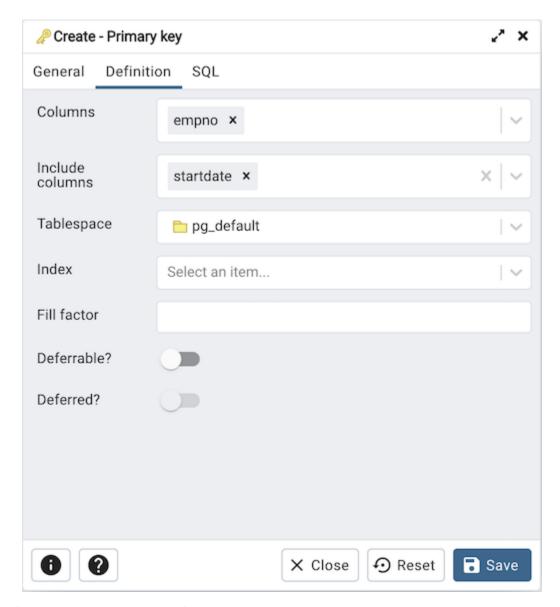
To add a primary key for the table, select the *Primary Key* tab, and click the *Add* icon (+). To define the primary key, click the *Edit* icon to the left of the *Trash* icon. A dialog similar to the *Primary key* dialog (accessed by right clicking on *Constraints* in the *pgAdmin* tree control) opens.

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 *pgAdmin* tree control.
- Provide notes about the primary key in the *Comment* field.

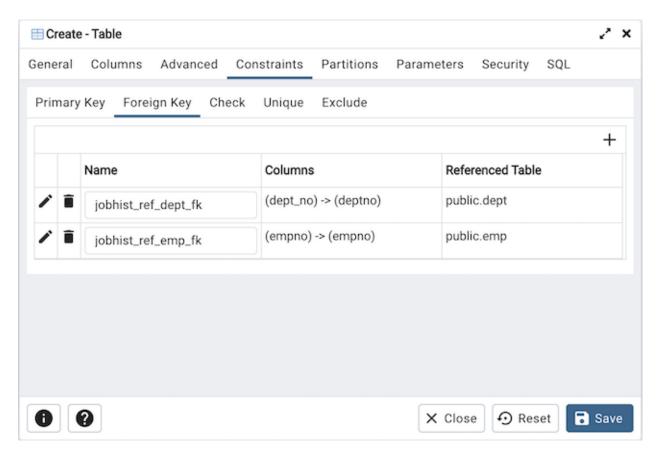
Click the *Definition* tab to continue.

6.10. Table Dialog 349



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 *pgAdmin* 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 *pgAdmin* tree control.
- Provide notes about the foreign key in the Comment field.

Click the *Definition* tab to continue.

6.10. Table Dialog 351



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.



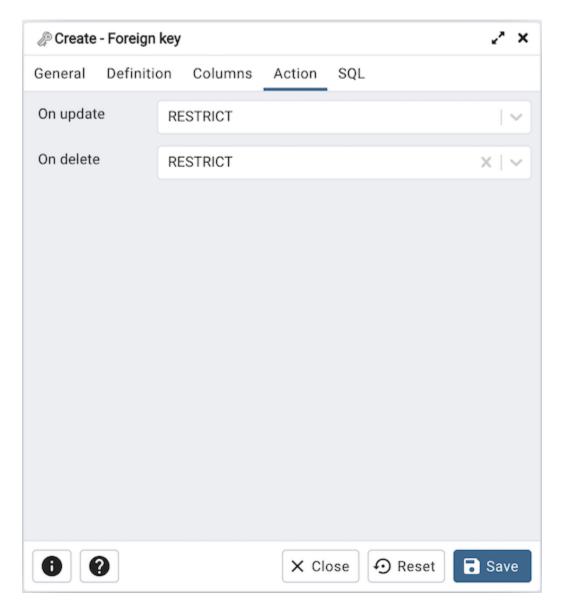
#### Use the fields in the *Columns* tab to specify one or more reference column(s).

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.

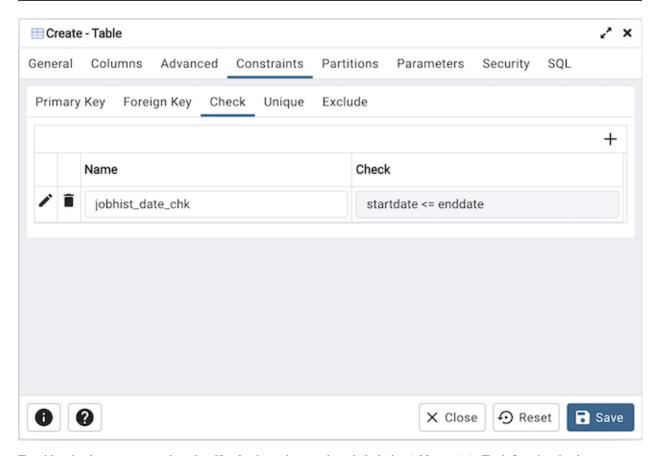
The supported actions are:

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.

continues on next page

Table 2 - continued from previous page

CASCADE	Delete any rows referencing the deleted row, or update the values of the referencing col-
	umn(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.

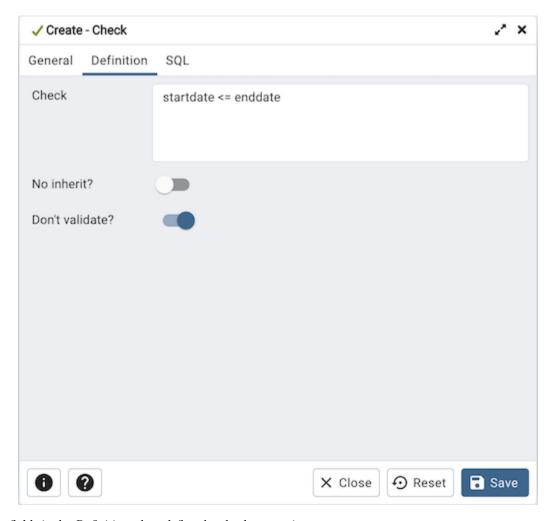


To add a check constraint, select the *Check* tab on the panel, and click the *Add* icon (+). To define the check constraint, click the *Edit* icon to the left of the *Trash* icon. A dialog similar to the *Check* dialog (accessed by right clicking on *Constraints* in the pgAdmin tree control) opens.

Use the fields in the *General* tab to identify the check constraint:

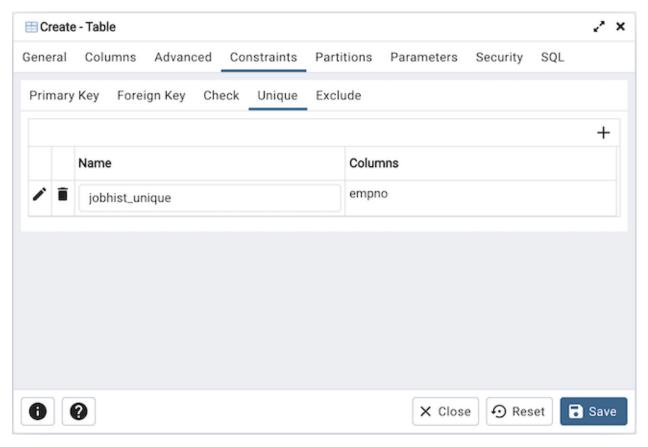
- Use the *Name* field to add a descriptive name for the check constraint. The name will be displayed in the *pgAdmin* tree control. With PostgreSQL 9.5 forward, when a table has multiple check constraints, they will be tested for each row in alphabetical order by name and after NOT NULL constraints.
- 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 that this constraint is not automatically inherited by a table's children. The default is *No*, meaning that the constraint will be inherited by any children.
- 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*.

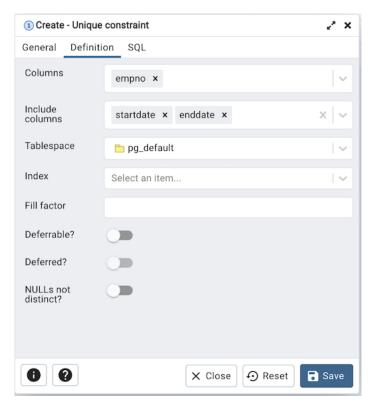


To add a unique constraint, select the *Unique* tab on the panel, and click the *Add* icon (+). To define the constraint, click the *Edit* icon to the left of the *Trash* icon. A dialog similar to the *Unique constraint* dialog (accessed by right clicking on *Constraints* in the *pgAdmin* tree control) opens.

Use the fields in the *General* tab to identify the unique constraint:

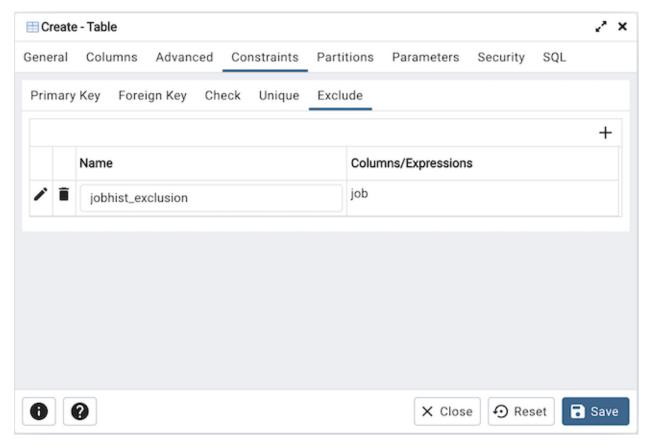
- Use the *Name* field to add a descriptive name for the unique constraint. The name will be displayed in the *pgAdmin* tree control.
- Provide notes about the unique constraint in the *Comment* field.

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*.

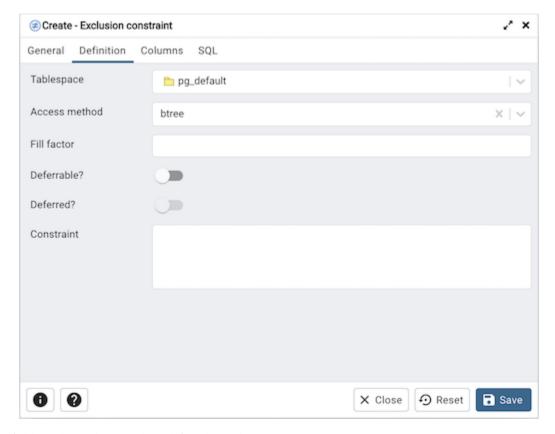


To add an exclusion constraint, select the *Exclude* tab on the panel, and click the *Add* icon (+). To define the constraint, click the *Edit* icon to the left of the *Trash* icon. A dialog similar to the *Exclusion constraint* dialog (accessed by right clicking on *Constraints* in the *pgAdmin* tree 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 *pgAdmin* tree control.
- Provide notes about the exclusion constraint in the *Comment* field.

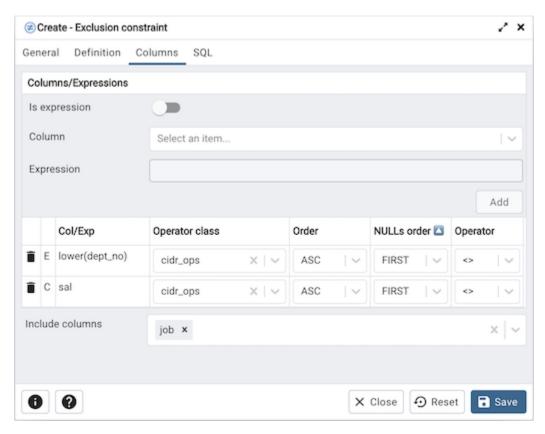
Click the Definition tab to continue.



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.

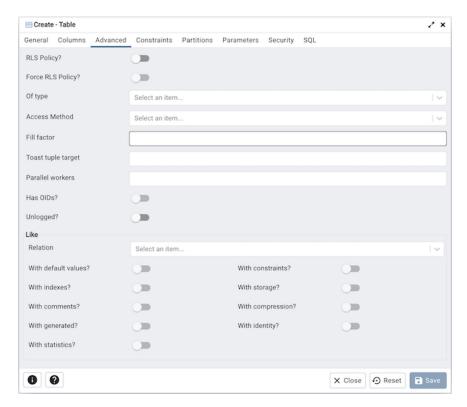
Click the Columns tab to continue.



Use the fields in the *Columns* tab to 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 drop-down list box next to Access Method to specify the table access method to use to store the contents for the new table; the method needs to be an access method of type TABLE. This field is optional. This option is available from v12 and above.
- 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 switch next to With default values? towards the right position to copy default values.
- Move the switch next to With constraints? towards the right position to copy table and column constraints.
- Move the switch next to With indexes? towards the right position to copy indexes.
- Move the switch next to With storage? towards the right position to copy storage settings.
- Move the switch next to With comments? towards the right position to copy comments.
- Move the switch next to *With compression?* towards the *right position* to copy compression method. This option is available only on PostgreSQL 14 and above.
- Move the switch next to *With generated?* towards the *right position* to copy generation expressions of copied column. This option is available only on PostgreSQL 12 and above.
- Move the switch next to With identity? towards the right position to copy any identity specifications of copied column.
- Move the switch next to With statistics? towards the right position to copy extended statistics.

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.

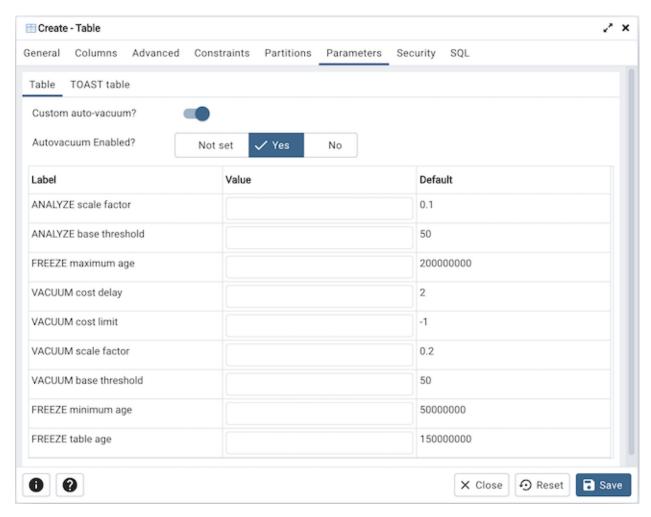
View of multi level Partitioned Table in object explorer:



- > 🗎 Columns
- > Constraints
- ▼ 

  □ Partitions (3)
  - > <u>m</u>2
  - > measurement\_y2007
  - Page 2008 Pag

Click the *Parameter* tab to continue.

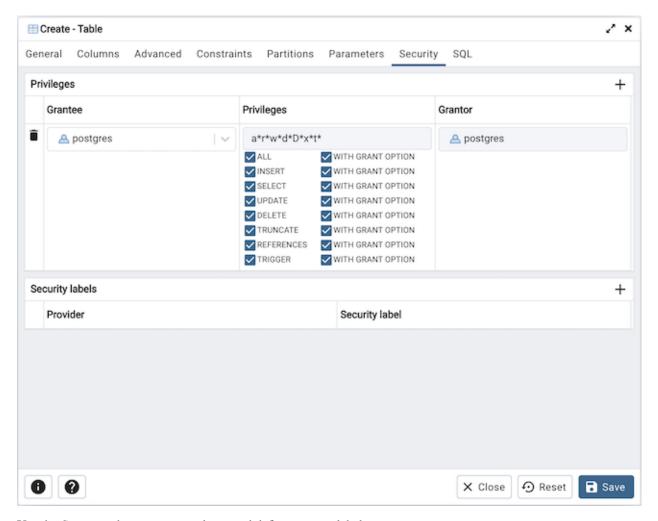


Use the tabs nested inside the *Parameter* tab to specify VACUUM and ANALYZE thresholds; use the *Table* tab and the *Toast Table* tab to customize values for the table and the associated toast table:

- 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 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.

#### **6.10.1 Example**

The following is an example of the sql command generated by user selections in the *Table* dialog:

```
E Create - Table
                                                                                                .^ ×
General Columns Advanced
                           Constraints Partitions
                                                  Parameters
                                                                       SQL
                                                              Security
 1 CREATE TABLE public.jobhist
2 (
 3
      empno numeric(4) NOT NULL,
      startdate time(0) with time zone NOT NULL,
 4
 5
      enddate time(0) with time zone,
      job character varying,
      sal numeric,
 8
      dept_no numeric(2),
 9
      CONSTRAINT jobhist_pk PRIMARY KEY (empno, startdate),
10
      CONSTRAINT jobhist_unique UNIQUE (empno),
      CONSTRAINT jobhist_ref_dept_fk FOREIGN KEY (dept_no)
11
12
          REFERENCES public.dept (deptno) MATCH SIMPLE
          ON UPDATE NO ACTION
13
14
          ON DELETE NO ACTION
15
          NOT VALID,
16
      CONSTRAINT jobhist_ref_emp_fk FOREIGN KEY (empno)
17
          REFERENCES public.emp (empno) MATCH SIMPLE
18
          ON UPDATE NO ACTION
          ON DELETE NO ACTION
19
20
          NOT VALID,
21
      CONSTRAINT jobhist_date_chk CHECK (startdate <= enddate),
      CONSTRAINT jobhist_exclusion EXCLUDE USING gist (
22
23
           job WITH <>)
24
25 )
26
      0
                                                                         X Close
                                                                                  Reset
                                                                                             ∃ Save
```

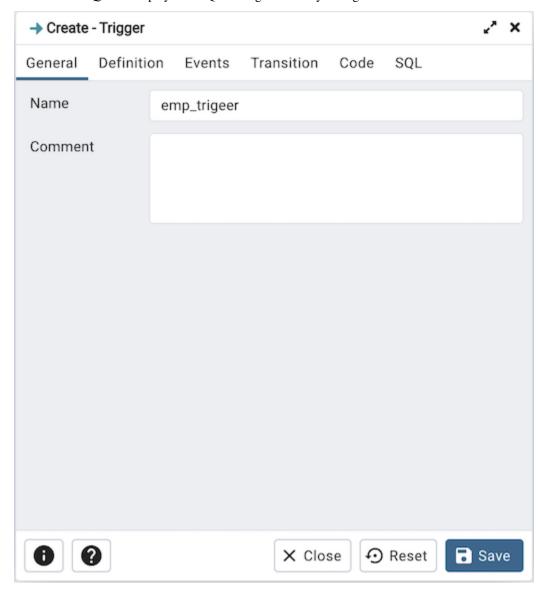
The example shown demonstrates creating a table named *jobhist*. It has six columns and a primary key constraint on the *empno and startdate* column.

- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

# 6.11 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.

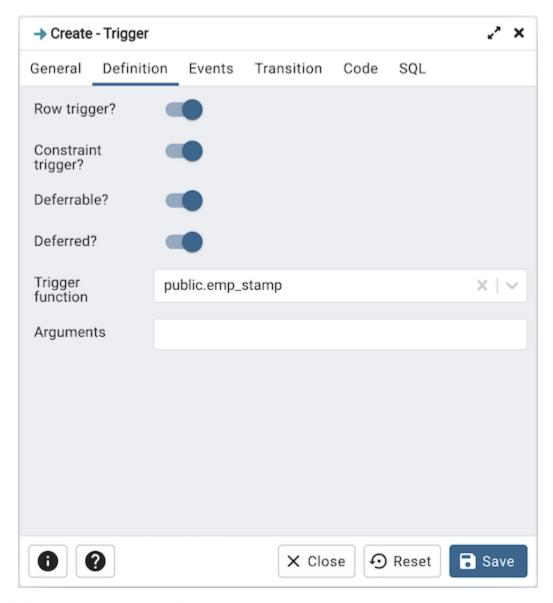
The *Trigger* dialog organizes the development of a trigger through the following dialog tabs: *General*, *Definition*, *Events*, and *Code*. The *SQL* tab displays the SQL code generated by dialog selections.



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 *pgAdmin* 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.

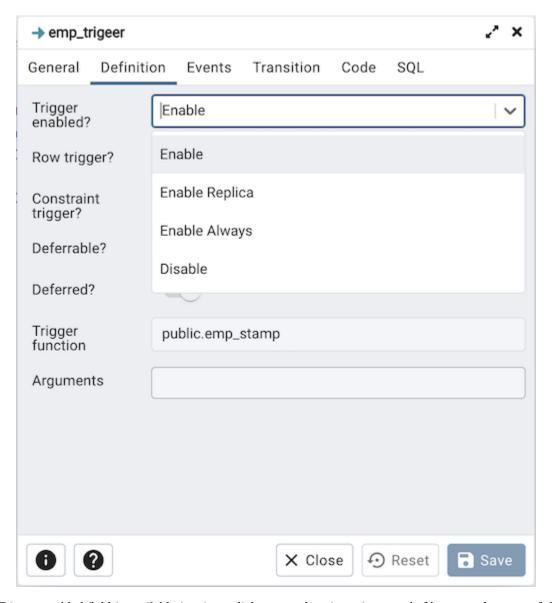
Click the *Definition* tab to continue.



Use the fields in the *Definition* tab to define the trigger:

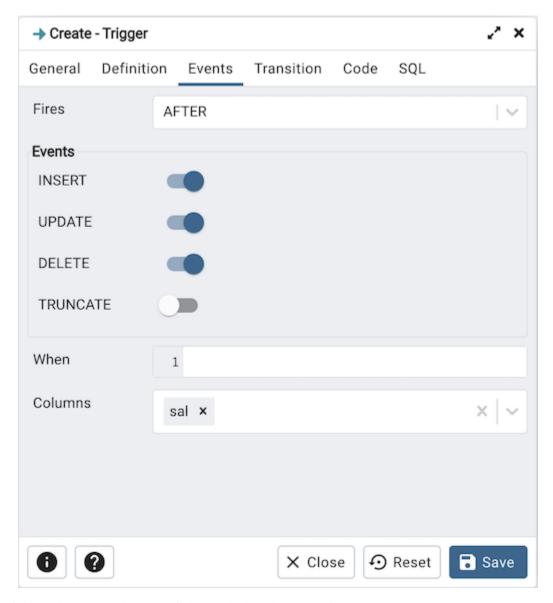
- Move the *Row trigger*? switch to the *No* position to disassociate the trigger from firing on each row in a table. The default is *Yes*.
- Move the Constraint trigger? switch to the Yes position to specify the trigger is a constraint trigger.
- If enabled, move the *Deferrable?* switch to the *Yes* 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.

6.11. Trigger Dialog 369



• *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.

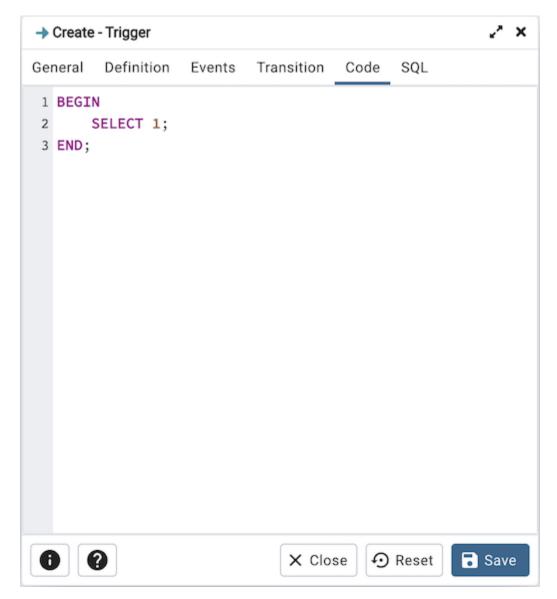


Use the fields in the *Events* tab to specify how and when the trigger fires:

- 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.

6.11. Trigger Dialog 371



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.

## **6.11.1 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 *emp\_trigger*.

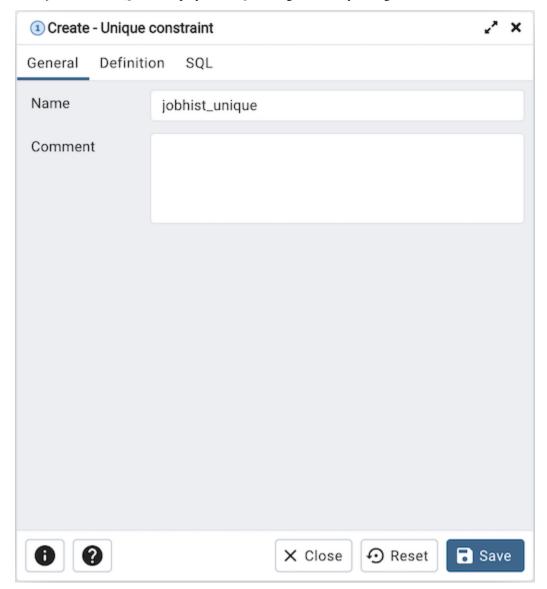
- Click the Info button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

6.11. Trigger Dialog 373

# **6.12 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.

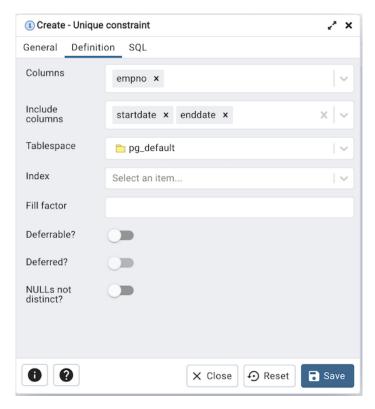
The *Unique constraint* dialog organizes the development of a unique constraint through the following dialog tabs: *General* and *Definition*. The *SQL* tab displays the SQL code generated by dialog selections.



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 *pgAdmin* tree control.

Click the *Definition* tab to continue.



Use the fields in the *Definition* tab to define the unique constraint:

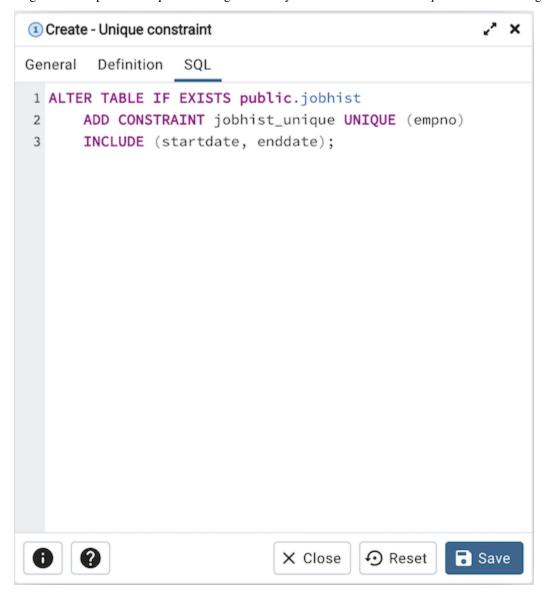
- Click inside the *Columns* field and select one or more column names from the drop-down listbox. To delete a selection, click the *x* to the left of the column name. The unique constraint should be different from the primary key constraint defined for the same table; the selected column(s) for the 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*.
- Move the *NULLs not distinct?* switch to the *Yes* position to treat null values as not distinct. The default is *No*. This option is available only on PostgreSQL 15 and above.

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.

#### **6.12.1 Example**

The following is an example of the sql command generated by user selections in the *Unique constraint* dialog:



The example shown demonstrates creating a unique constraint named *jobhist\_unique* on the *empno* column of the *jobhist* table.

- Click the *Info* button (i) to access online help.
- Click the Save button to save work.
- Click the *Close* button to exit without saving work.
- Click the *Reset* button to restore configuration parameters.

## **Management Basics**

pgAdmin provides point and click dialogs that help you perform server management functions. Dialogs simplify tasks such as managing named restore points, granting user privileges, and performing VACUUM, ANALYZE and REIN-DEX functions.

## 7.1 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*, *logical*, or *minimal*. 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.

# 7.2 Change Password Dialog

It is a good policy to routinely change your password to protect data, even in what you may consider a 'safe' environment. In the workplace, failure to apply an appropriate password policy could leave you in breach of Data Protection laws.

Please consider the following guidelines when selecting a password:

- Ensure that your password is an adequate length; 6 characters should be the absolute minimum number of characters in the password.
- The minimum password length is set by default to six characters. This value can be changed by setting the *PASSWORD\_LENGTH\_MIN* option to desired length in pgAdmin configuration; see *The config.py File* for more information.
- Ensure that your password is not open to dictionary attacks. Use a mixture of upper and lower case letters and numerics, and avoid words or names. Consider using the first letter from each word in a phrase that you will remember easily but is an unfamiliar acronym.
- Ensure that your password is changed regularly; at minimum, change it every ninety days.

The guidelines above should be considered a starting point: They are not a comprehensive list and they **will not guarantee security**.



Use the Change Password dialog to change your password:

- The name displayed in the *User* field is the role for which you are modifying the password; it is the role that is associated with the server connection that is highlighted in the tree control.
- Enter the password associated with the role in the *Current Password* field.
- Enter the desired password for in the New Password field.
- Re-enter the new password in the Confirm Password field.

Click the Save button to change your password.

Click the *Close* button to exit without changing your password.

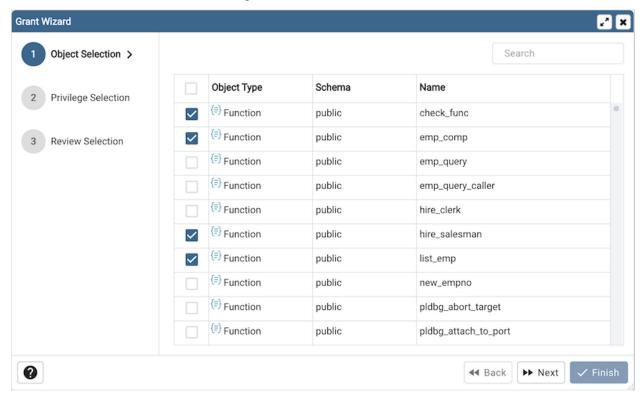
Click the *Reset* button to reset the values.

### 7.3 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*, *Privileges Selection* and *Review Selection*. The *Review Selection* window displays the SQL code generated by wizard selections.

To launch the *Grant Wizard* tool, select a database object in the *pgAdmin* tree control, then navigate through *Tools* on the menu bar to click on the *Grant Wizard* option.



Use the fields in the *Object Selection* 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 *X* button to close the wizard without modifying privileges.

7.3. Grant Wizard 379



Use the fields in the *Privileges Selection* 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 thier 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.

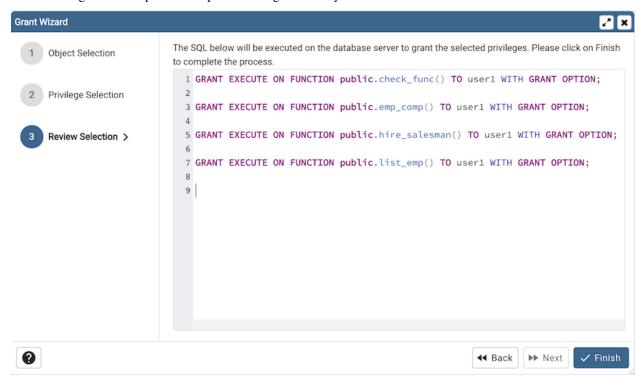
For more information about granting privileges on database objects, see the PostgreSQL core documentation.

Click the *Next* button to continue, the *Back* button to select or deselect additional database objects, or the *X* 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 *Review Selection* window (see an example below).

## 7.3.1 Example

The following is an example of the sql command generated by user selections in the *Grant Wizard* tool:

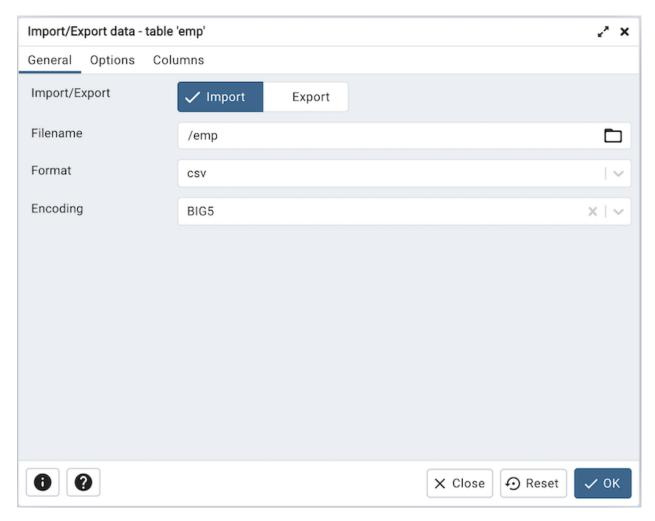


- Click the *Back* button to select or deselect additional database objects, roles and privileges.
- Click the *Finish* button to save selections and exit the wizard.

# 7.4 Import/Export Data Dialog

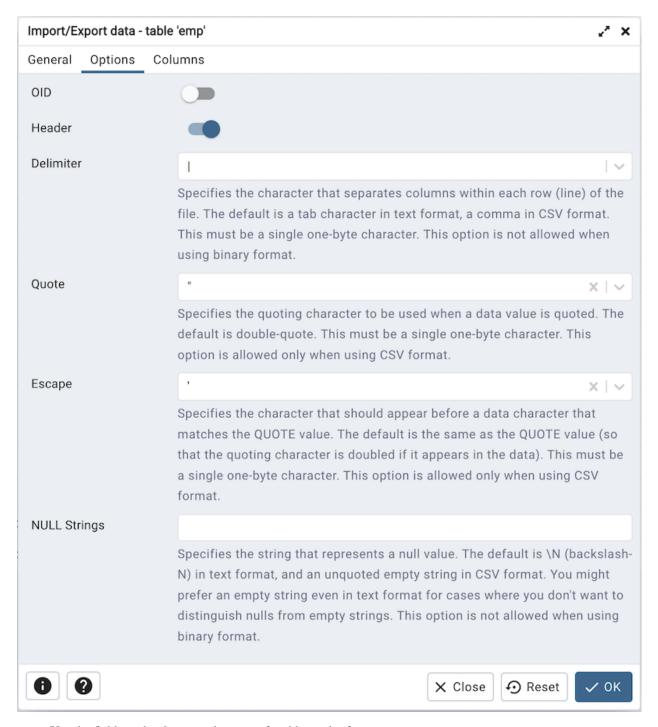
Use the Import/Export data dialog to copy data from a table to a file, or copy data from a file into a table.

The Import/Export data dialog organizes the import/export of data through the General, Options and Columns tabs.



Use the fields in the *General* tab to specify import and export preferences:

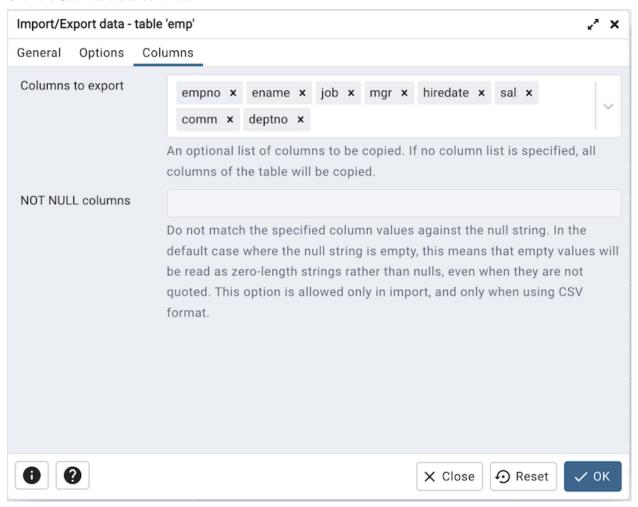
- Move the *Import/Export* switch to the *Import* position to specify that the server should import data to a table from a file. The default is *Import*.
- Enter the name of the source or target file in the *Filename* field. Optionally, select the *Browse* icon (ellipsis) to the right to navigate into 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 *Options* tab 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.
- Use the NULL Strings field to specify a string that will represent a null value within the source or target file.

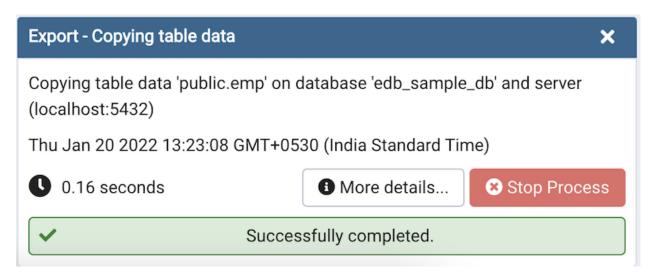
Click the *Columns* tab to continue.



Use the fields in the *Columns* tab to select the columns that will be imported or exported:

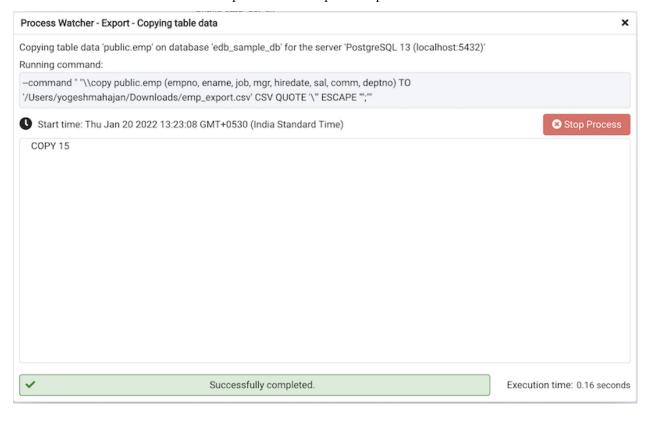
- Click inside the *Columns to export/import* field to deselect one or more columns from the drop-down listbox. To delete a selection, click the *x* to the left of the column name. Click an empty spot inside the field to access the drop-down list.
- If enabled, click inside the *NOT NULL columns* field 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. pgAdmin 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:



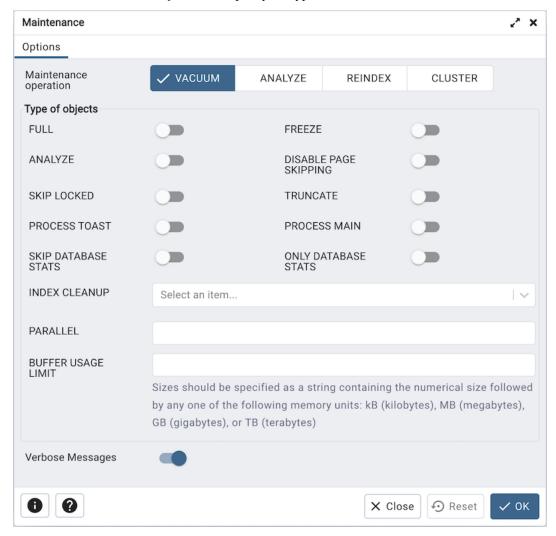
**Note:** If you are running *pgAdmin* in *Server Mode* you can click on the icon in the process watcher window to open the file location in the Storage Manager. You can use the *Storage Manager* to download the backup file on the client machine.

## 7.5 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.
  - Move the FREEZE switch to the Yes position to freeze data in a table when it will have no further updates.
  - Move the *ANALYZE* switch to the *Yes* position to issue ANALYZE commands whenever the content of a table has changed sufficiently.
  - Move the DISABLE PAGE SKIPPING switch to the Yes position to disables all page-skipping behavior.
  - Move the SKIP LOCKED switch to the Yes position to specifies that VACUUM should not wait for any
    conflicting locks to be released when beginning work on a relation. This option is available from v12

onwards.

- Move the TRUNCATE switch to the Yes position to specifies that VACUUM should attempt to truncate off
  any empty pages at the end of the table and allow the disk space for the truncated pages to be returned to
  the operating system. This option is available from v12 onwards.
- Move the PROCESS TOAST switch to the Yes position to specifies that VACUUM should attempt to process
  the corresponding TOAST table for each relation, if one exists. This option is available from v14 onwards.
- Move the PROCESS MAIN switch to the Yes position to specifies that VACUUM should attempt to process the main relation. This option is available from v16 onwards.
- Move the SKIP DATABASE STATS switch to the Yes position to specifies that VACUUM should skip updating the database-wide statistics about oldest unfrozen XIDs. This option is available from v16 onwards.
- Move the ONLY DATABASE STATS switch to the Yes position to specifies that VACUUM should do nothing
  except update the database-wide statistics about oldest unfrozen XIDs. This option is available from v16
  onwards.
- Use the INDEX CLEANUP field to force VACUUM to process indexes when there are more than zero dead tuples.
- Use the PARALLEL field to specify index vacuum and index cleanup phases of VACUUM in parallel using integer background workers. This option is available from v13 onwards.
- Use the BUFFER USAGE LIMIT field to specifies the Buffer Access Strategy ring buffer size for VACUUM.
   This size is used to calculate the number of shared buffers which will be reused as part of this strategy. This option is available from v16 onwards

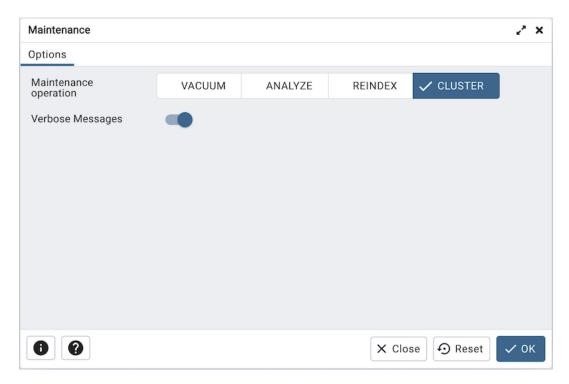


- Click *ANALYZE* to update the stored statistics used by the query planner. This enables the query optimizer to select the fastest query plan for optimal performance.
  - Move the SKIP LOCKED switch to the Yes position to specifies that ANALYZE should not wait for any
    conflicting locks to be released when beginning work on a relation. This option is available from v12
    onwards.

Use the BUFFER USAGE LIMIT field to specifies the Buffer Access Strategy ring buffer size for AN-ALYZE. This size is used to calculate the number of shared buffers which will be reused as part of this strategy. This option is available from v16 onwards



- 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.
  - Move the *SYSTEM* switch to the *Yes* position to recreate all indexes on system catalogs within the current database. This option is enabled only when database object is selected.
  - Move the *CONCURRENTLY* switch to the *Yes* position to rebuild the index without taking any locks that prevent concurrent inserts, updates, or deletes on the table. This option is available from v12 onwards.
  - Use the TABLESPACE field to specifies that indexes will be rebuilt on a new tablespace. This option is available from v14 onwards.



• 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*.

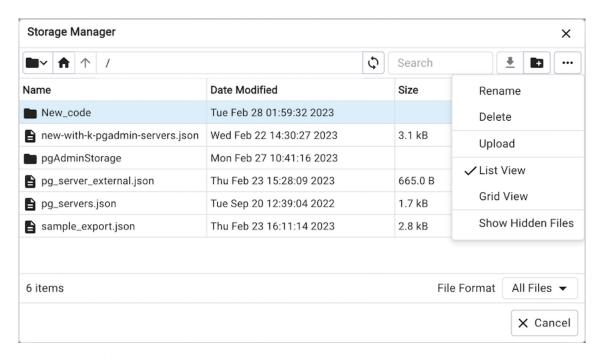
pgAdmin will run the maintenance process in background. You can view all the background process with there running status and logs on the *Processes* tab.

# 7.6 Storage Manager

Storage Manager is a feature that helps you manage your systems storage device. You can use Storage Manager to:

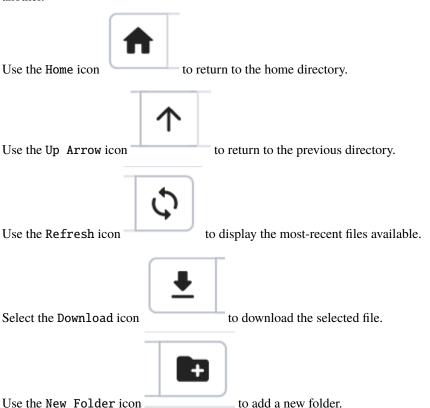
- Download, upload, or manage operating system files. To use this feature, *pgAdmin* must be running in *Server Mode* on your client machine.
- The shared storage option allows users to access the shared storages that are shared by admin users.
- Download backup or export files (custom, tar and plain text format) on a client machine.
- Download export dump files of tables.

You can access Storage Manager from the Tools Menu.



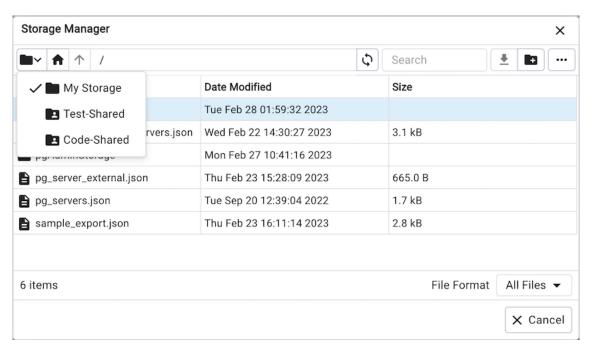
Use icons on the top of the *Storage Manager* window to manage storage:

Use the Folder icon to access shared storage. In order to enable shared storage, admins need to add the SHARED\_STORAGE variable to the config file. Users can access the shared storage with this and share files with one another.



Use the Format drop down list to select the format of the files to be displayed; choose from sql, csv, or All Files.

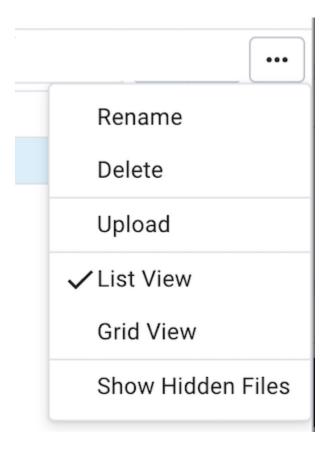
#### 7.6.1 Shared Storage



In the storage manager, My Storage is the pgAdmin user's storage directory, and other listed directories are shared storages set by the pgAdmin server administrator. Using these, pgAdmin users can have common storages to share files. pgAdmin server administrator can configure the shared storages using the *config file*. Storages can be marked as restricted to give read-only access to non-admin pgAdmin users.

**Note:** You must ensure the directories specified are writeable by the user that the web server processes will be running as, e.g. apache or www-data.\*

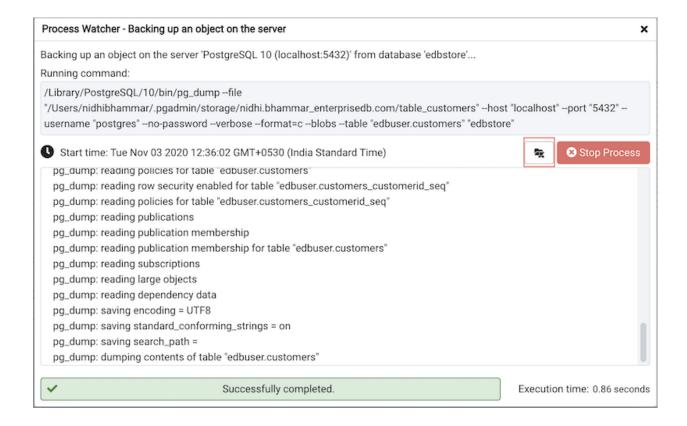
## 7.6.2 Other Options



Menu	Behavior
Rename	Click the <i>Rename</i> option to rename a file/folder.
Delete	Click the <i>Delete</i> option to rename a file/folder.
Upload	Click the <i>Upload</i> option to upload multiple files to the current folder.
List View	Click the <i>List View</i> option to to display all the files and folders in a list view.
Grid View	Click the <i>Grid View</i> option to to display all the files and folders in a grid view.
Show Hidden Files	Click the <i>Show Hidden Files</i> option to view hidden files and folders.

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.



# CHAPTER 8

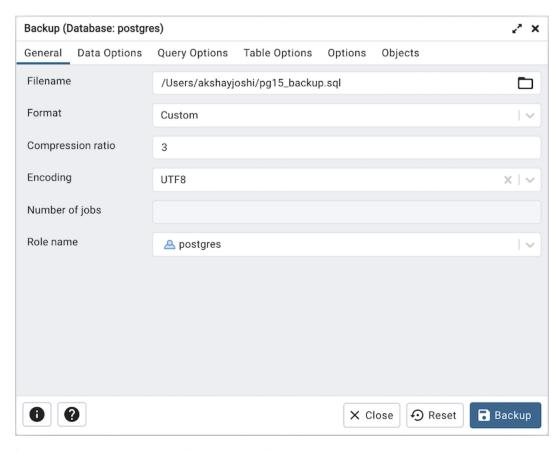
## Backup and Restore

A powerful, but user-friendly Backup and Restore tool provides an easy way to use pg\_dump, pg\_dumpall, and pg\_restore to take backups and create copies of databases or database objects for use in a development environment.

## 8.1 Backup Dialog

pgAdmin 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 pgAdmin 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.



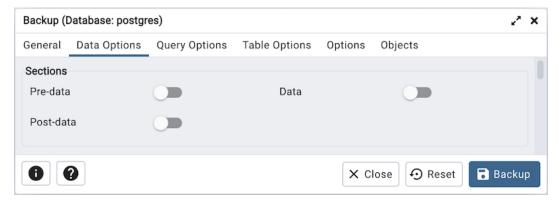
Use the fields in the *General* tab to specify parameters for the backup:

- 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

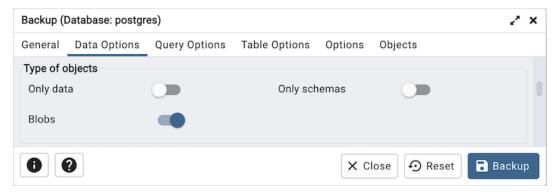
use no compression; specify a maximum compression value of 9. Please note that tar archives do not support compression.

- 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 *Rolename* to specify the role that owns the backup.

Click the *Data Options* tab to continue. Use the fields in the *Data Options* tab to provide options related to data or pgAdmin objects that correspond to  $pg\_dump$ .

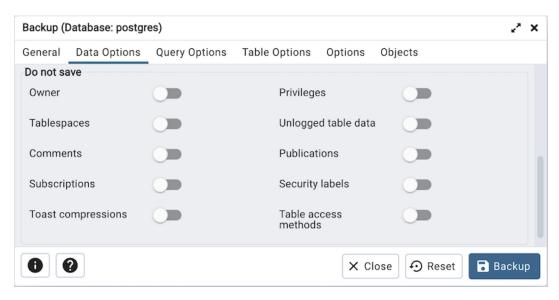


- Move switches in the **Sections** field box to select a portion of the object that will be backed up.
  - Move the switch next to *Pre-data* towards right position to include all data definition items not included in the data or post-data item lists.
  - Move the switch next to Data towards right position to backup actual table data, large-object contents, and sequence values.
  - Move the switch next to Post-data towards right position to include definitions of indexes, triggers, rules, and constraints other than validated check constraints.

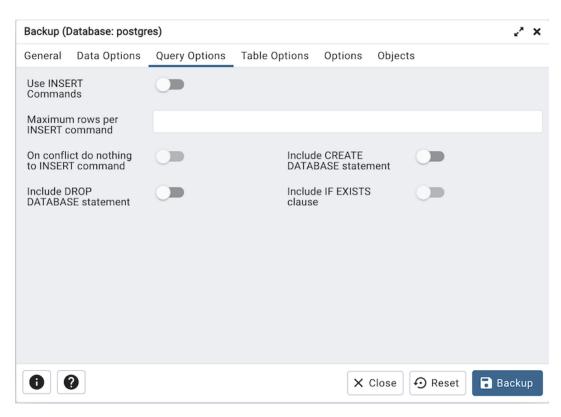


- Move switches in the **Type of objects** field box to specify details about the type of objects that will be backed up.
  - Move the switch next to Only data towards right position to limit the back up to data.
  - Move the switch next to *Only schemas* to limit the back up to schema-level database objects.
  - Move the switch next to *Blobs* towards left position to exclude large objects in the backup.

8.1. Backup Dialog 397



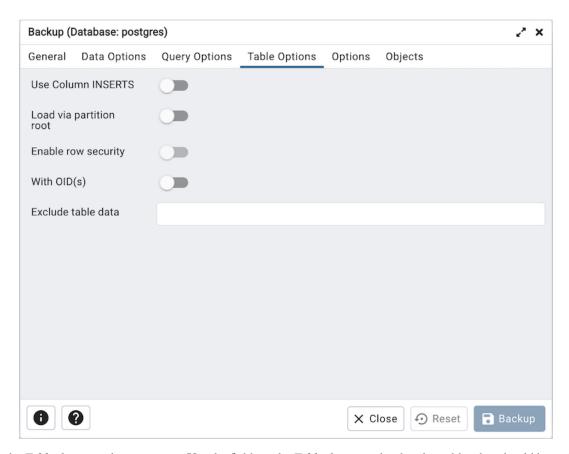
- Move switches in the **Do not save** field box to select the objects that will not be included in the backup.
  - Move the switch next to *Owner* towards right position to exclude commands that set object ownership.
  - Move the switch next to *Privileges* towards right position to exclude commands that create access privileges.
  - Move the switch next to *Tablespaces* towards right position to exclude tablespaces.
  - Move the switch next to Unlogged table data towards right position to exclude the contents of unlogged tables.
  - Move the switch next to *Comments* towards right position to exclude commands that set the comments.
     Note: This option is visible only for database server greater than or equal to 11.
  - Move the switch next to *Publications* towards right position to exclude publications.
  - Move the switch next to *Subscriptions* towards right position to exclude subscriptions.
  - Move the switch next to *Security labels* towards right position to exclude Security labels.
  - Move the switch next to *Toast compressions* towards right position to exclude Toast compressions. Note:
     This option is visible only for database server greater than or equal to 14.
  - Move the switch next to *Table access methods* towards right position to exclude Table access methods.
     Note: This option is visible only for database server greater than or equal to 15.



Click the *Query Options* tab to continue. Use these additional fields to specify the type of statements that should be included in the backup.

- Move the switch next to *Use INSERT commands* towards right 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.
- Use the *Maximum rows per INSERT command* field to controls the maximum number of rows per INSERT command. **Note:** This option is visible only for database server greater than or equal to 12.
- Move the switch next to On conflict do nothing to INSERT command towards right position to add ON CON-FLICT DO NOTHING to INSERT command. This option is not valid unless Use INSERT commands, Use Column INSERTS or Maximum rows per INSERT command is also specified. Note: This option is visible only for database server greater than or equal to 12.
- Move the switch next to *Include CREATE DATABASE statement* towards right 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* towards right 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 *Include IF EXISTS clause* towards right position to add an IF EXISTS clause to drop databases and other objects. This option is not valid unless *Include DROP DATABASE statement* is also set.

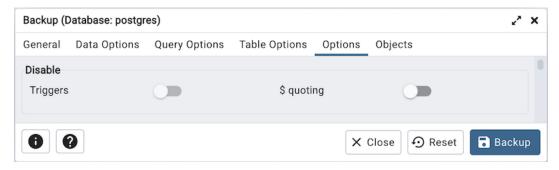
8.1. Backup Dialog 399



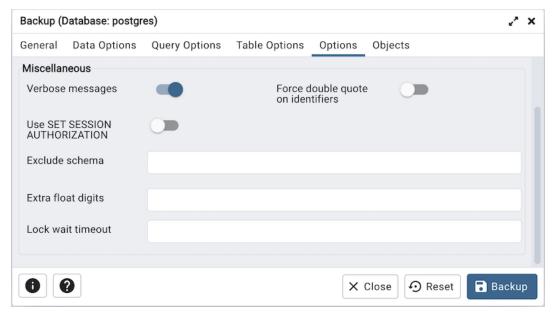
Click the *Table Options* tab to continue. Use the fields in the *Table Options* tab related to tables that should be included in the backup.

- Move the switch next to *Use Column INSERTS* towards right 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 *Load via partition root* towards right 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 the switch next to *Enable row security* towards right position to set row\_security to on instead, allowing the user to dump the parts of the contents of the table that they have access to. This option is relevant only when dumping the contents of a table which has row security.
- Move the switch next to With OIDs towards right position to include object identifiers as part of the table data for each table.
- Use the Exclude table data field to not dump data for any tables matching the table pattern.

Click the *Options* tab to continue. Use the fields in the *Options* tab to provide other backup options.



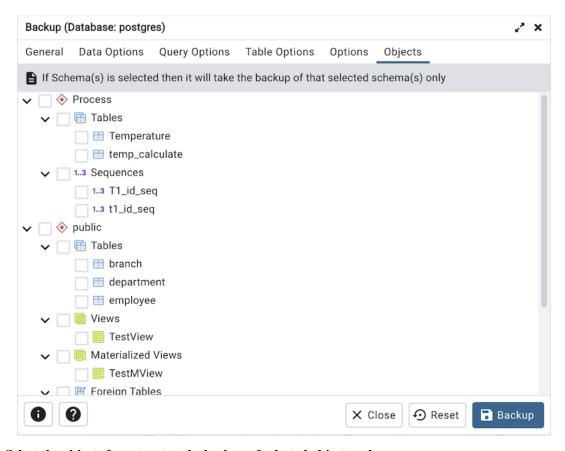
- Move switches in the **Disable** field box to specify the type of statements that should be excluded from the backup.
  - Move the switch next to *Triggers* (active when creating a data-only backup) towards right position to include commands that will disable triggers on the target table while the data is being loaded.
  - Move the switch next to \$ quoting towards right 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 Verbose messages towards left position to instruct pg\_dump to exclude verbose messages.
  - Move the switch next to Force double quotes on identifiers towards right position to force the quoting of all identifiers.
  - Move the switch next to *Use SET SESSION AUTHORIZATION* towards right position to include a statement
    that will use a SET SESSION AUTHORIZATION command to determine object ownership (instead of an
    ALTER OWNER command).
  - Use the *Exclude schema* field to not dump schemas whose name matches pattern.
  - Use the Extra float digits field to use the specified value when dumping floating-point data, instead of the maximum available precision.
  - Use the Lock wait timeout field to do not wait forever to acquire shared table locks at the beginning of the dump. Instead, fail if unable to lock a table within the specified timeout.

Click the Objects tab to continue.

8.1. Backup Dialog 401



- Select the objects from tree to take backup of selected objects only.
  - If Schema is selected then it will take the backup of that selected schema only.
  - If any Table, View, Materialized View, Sequences, or Foreign Table is selected then it will take the backup of those selected objects.

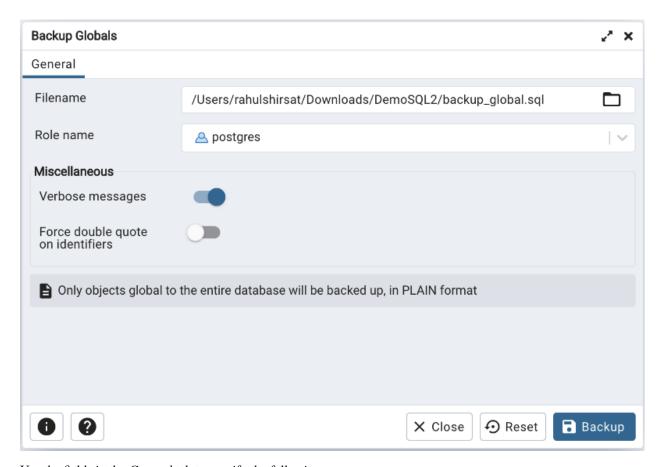
When you've specified the details that will be incorporated into the pg\_dump command:

- 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.

pgAdmin will run the backup process in background. You can view all the background process with there running status and logs on the *Processes* tab

# 8.2 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 pgAdmin *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.

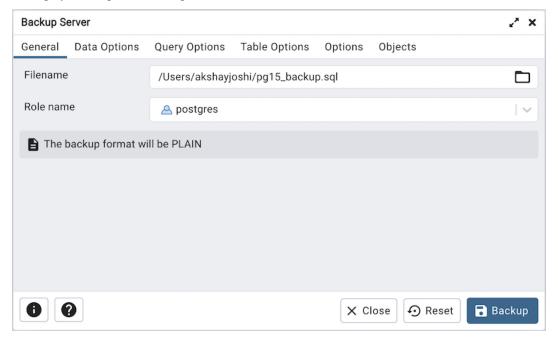
- Move the *Verbose messages* switch to the *No* position to exclude status messages from the backup. The default is *Yes*.
- Move the *Force double quote on identifiers* switch to the *Yes* position to name identifiers without changing case. The default is *No*.

Click the *Backup* button to build and execute a command based on your selections; click the *Cancel* button to exit without saving work.

pgAdmin will run the backup process in background. You can view all the background process with there running status and logs on the *Processes* tab

## 8.3 Backup Server Dialog

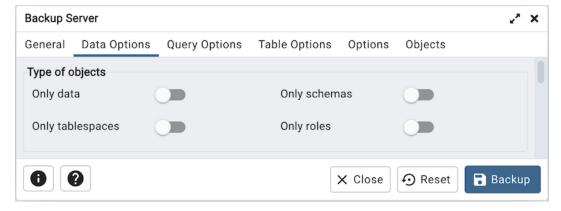
Use the *Backup Server* dialog to create a plain-text script that will recreate the selected server. You can use the pgAdmin *Query Tool* to play back a plain-text script, and recreate the server.



Use the fields in the *General* tab to specify the following:

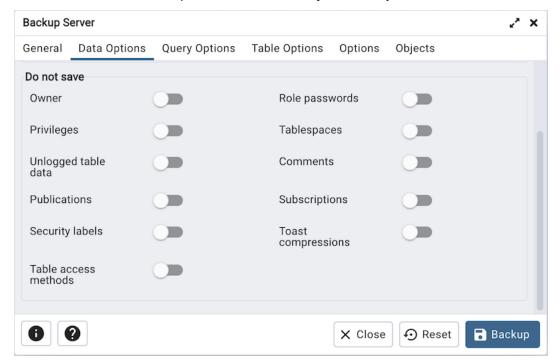
- Enter the name of the backup file in the *Filename* field. Optionally, select the *Browser* icon (ellipsis) to the right to navigate into a directory and select a file that will contain the archive.
- Use the *Encoding* drop-down listbox to select the character encoding method that should be used for 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.

Click the *Data Options* tab to continue. Use the fields in the *Data Options* tab to provide options related to data or pgAdmin objects that correspond to pg\_dumpall.

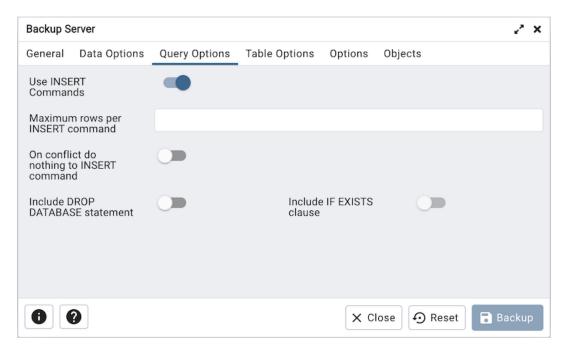


- Move switches in the **Type of objects** field box to specify details about the type of objects that will be backed up.
  - Move the switch next to Only data towards right position to limit the back up to data.

- Move the switch next to Only schemas to limit the back up to schema-level database objects.
- Move the switch next to *Only tablespaces* to limit the back up to tablespaces only.
- Move the switch next to Only roles to limit the back up to roles only.

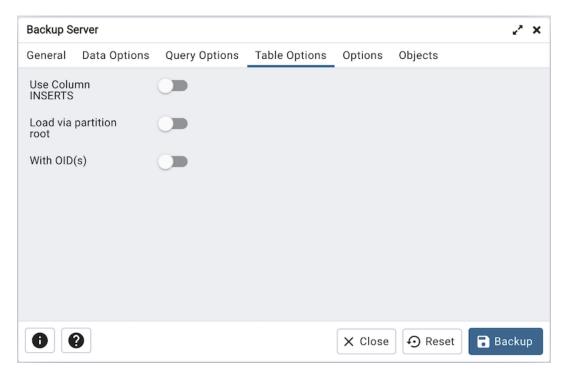


- Move switches in the **Do not save** field box to select the objects that will not be included in the backup.
  - Move the switch next to *Owner* towards right position to exclude commands that set object ownership.
  - Move the switch next to *Role passwords* towards right position to exclude passwords for roles.
  - Move the switch next to *Privileges* towards right position to exclude commands that create access privileges.
  - Move the switch next to *Tablespaces* towards right position to exclude tablespaces.
  - Move the switch next to Unlogged table data towards right position to exclude the contents of unlogged tables.
  - Move the switch next to *Comments* towards right position to exclude commands that set the comments.
     Note: This option is visible only for database server greater than or equal to 11.
  - Move the switch next to *Publications* towards right position to exclude publications.
  - Move the switch next to *Subscriptions* towards right position to exclude subscriptions.
  - Move the switch next to Security labels towards right position to exclude Security labels.
  - Move the switch next to *Toast compressions* towards right position to exclude Toast compressions. Note:
     This option is visible only for database server greater than or equal to 14.
  - Move the switch next to *Table access methods* towards right position to exclude Table access methods.
     Note: This option is visible only for database server greater than or equal to 15.



Click the *Query Options* tab to continue. Use these additional fields to specify the type of statements that should be included in the backup.

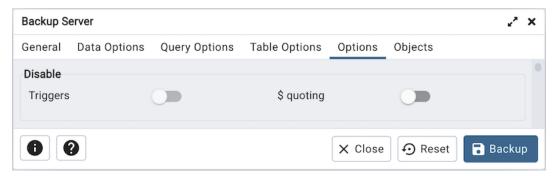
- Move the switch next to *Use INSERT commands* towards right position to dump the data in the form of INSERT statements rather than using a COPY command. Please note: this may make restoration from backup slow.
- Use the *Maximum rows per INSERT command* field to controls the maximum number of rows per INSERT command. **Note:** This option is visible only for database server greater than or equal to 12.
- Move the switch next to On conflict do nothing to INSERT command towards right position to add ON CON-FLICT DO NOTHING to INSERT command. This option is not valid unless Use INSERT commands, Use Column INSERTS or Maximum rows per INSERT command is also specified. Note: This option is visible only for database server greater than or equal to 12.
- Move the switch next to *Include DROP DATABASE statement* towards right 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 *Include IF EXISTS clause* towards right position to add an IF EXISTS clause to drop databases and other objects. This option is not valid unless *Include DROP DATABASE statement* is also set.



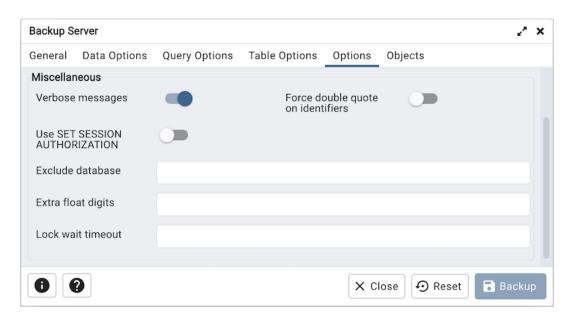
Click the *Table Options* tab to continue. Use the fields in the *Table Options* tab related to tables that should be included in the backup.

- Move the switch next to *Use Column INSERTS* towards right 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 *Load via partition root* towards right 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 the switch next to With OIDs towards right position to include object identifiers as part of the table data for each table.

Click the Options tab to continue. Use the fields in the Options tab to provide other backup options.



- 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 *Triggers* (active when creating a data-only backup) towards right position to include commands that will disable triggers on the target table while the data is being loaded.
  - Move the switch next to \$ quoting towards right 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 Verbose messages towards left position to instruct pg\_dumpall to exclude verbose messages.
  - Move the switch next to Force double quotes on identifiers towards right position to force the quoting of all identifiers.
  - Move the switch next to *Use SET SESSION AUTHORIZATION* towards right position to include a statement that will use a SET SESSION AUTHORIZATION command to determine object ownership (instead of an ALTER OWNER command).
  - Use the Exclude database field to not dump databases whose name matches pattern.
  - Use the Extra float digits field to use the specified value when dumping floating-point data, instead of the maximum available precision.
  - Use the Lock wait timeout field to do not wait forever to acquire shared table locks at the beginning of the dump. Instead, fail if unable to lock a table within the specified timeout.

When you've specified the details that will be incorporated into the pg\_dumpall command:

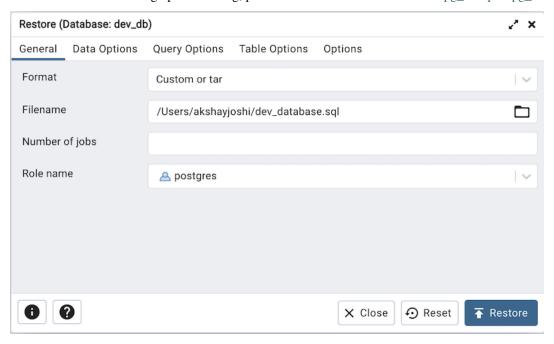
- Click the *Backup* button to build and execute a command that builds a backup based on your selections on the *Backup Server* dialog.
- Click the *Cancel* button to exit without saving work.

pgAdmin will run the backup process in background. You can view all the background process with there running status and logs on the *Processes* tab

## 8.4 Restore Dialog

The *Restore* dialog provides an easy way to use a Custom, tar, or Directory format backup taken with the pgAdmin *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.

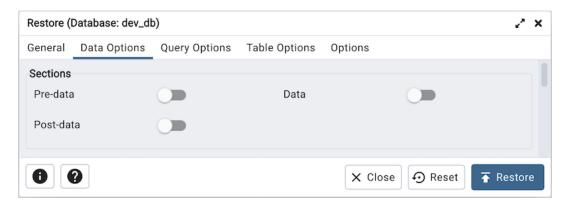


Use the fields on the General tab to specify general information about the restore process:

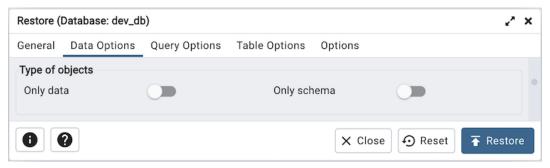
- Use the drop-down listbox in the *Format* field to select the format of your backup file.
  - Select Custom or tar to restore from a custom archive file to create a copy of the backed-up object.
  - Select *Directory* to restore from a compressed directory-format archive.
- Enter the complete path to the backup file in the *Filename* field. Optionally, select the *Browse* 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 *Data Options* tab to continue. Use the fields in the *Data Options* tab to provide options related to data or pgAdmin objects that correspond to pg\_restore.

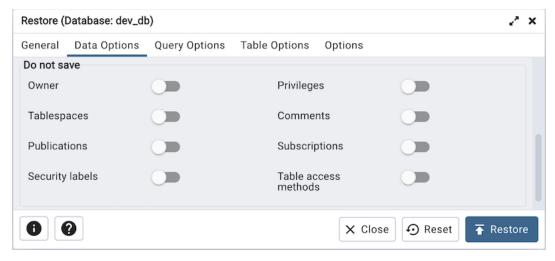
8.4. Restore Dialog 409



- Move switches in the **Sections** field box to specify the content that will be restored:
  - Move the switch next to *Pre-data* towards right position to restore all data definition items not included in the data or post-data item lists.
  - Move the switch next to Data towards right position to restore actual table data, large-object contents, and sequence values.
  - Move the switch next to *Post-data* towards right position position to restore definitions of indexes, triggers, rules, and constraints (other than validated check constraints).

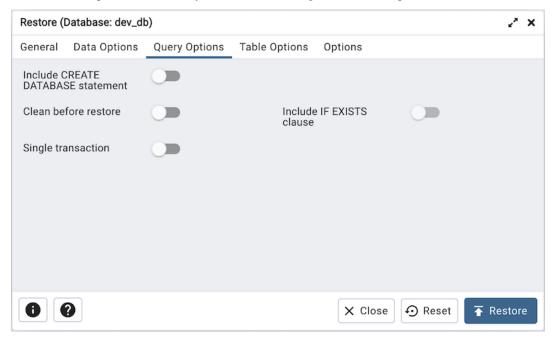


- Move switches in the **Type of objects** field box to specify the objects that will be restored:
  - Move the switch next to Only data towards right position to limit the restoration to data.
  - Move the switch next to Only schema to limit the restoration to schema-level database objects.



• Move switches in the **Do not save** box to specify which objects will not be restored:

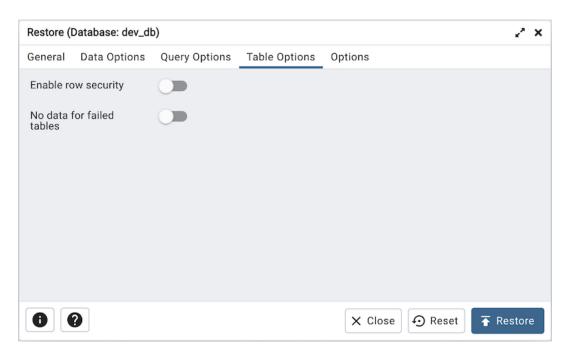
- Move the switch next to *Owner* towards right position to exclude commands that set object ownership.
- Move the switch next to *Privileges* towards right position to exclude commands that create access privileges.
- Move the switch next to *Tablespaces* towards right position to exclude tablespaces.
- Move the switch next to *Comments* towards right position to exclude commands that set the comments.
   Note: This option is visible only for database server greater than or equal to 11.
- Move the switch next to Publications towards right position to exclude publications.
- Move the switch next to Subscriptions towards right position to exclude subscriptions.
- Move the switch next to Security labels towards right position to exclude Security labels.
- Move the switch next to *Table access methods* towards right position to exclude Table access methods.
   Note: This option is visible only for database server greater than or equal to 15.



Click the *Query Options* tab to continue. Use these additional fields to specify the type of statements that should be included in the restore:

- Move the switch next to *Include CREATE DATABASE statement* towards right position to include a command that creates a new database before performing the restore.
- Move the switch next to *Clean before restore* towards right position to drop each existing database object (and data) before restoring.
- Move the switch next to *Include IF EXISTS clause* towards right position to add an IF EXISTS clause to drop databases and other objects. This option is not valid unless *Clean before restore* is also set.
- Move the switch next to *Single transaction* towards right 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*.

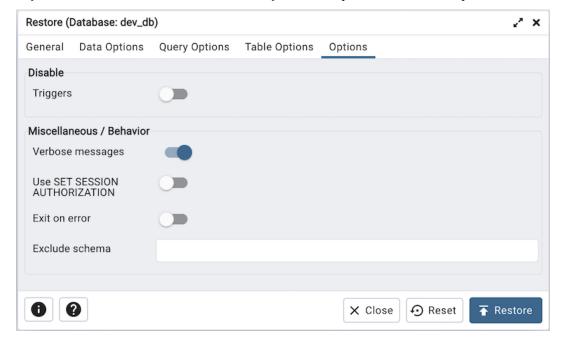
8.4. Restore Dialog 411



Click the *Table Options* tab to continue. Use the fields in the *Table Options* tab related to tables that should be included in the backup.

- Move the switch next to *Enable row security* towards right position to set row\_security to on instead, allowing the user to dump the parts of the contents of the table that they have access to. This option is relevant only when dumping the contents of a table which has row security.
- Move the switch next to *No data for failed tables* towards right position to ignore data that fails a trigger.

Click the *Options* tab to continue. Use the fields in the *Options* tab to provide other restore options.



- Move switches in the **Disable** box to specify the type of statements that should be excluded from the restore:
  - Move the switch next to *Triggers* (active when creating a data-only restore) towards right position to include commands that will disable triggers on the target table while the data is being loaded.

- Move switches in the **Miscellaneous/Behavior** box to specify miscellaneous restore options:
  - Move the switch next to *Verbose messages* towards left to instruct *pg\_restore* to exclude verbose messages.
  - Move the switch next to *Use SET SESSION AUTHORIZATION* towards right 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 towards right 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.
  - Use the *Exclude schema* field to not dump schemas whose name matches pattern.

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.

pgAdmin will run the restore process in background. You can view all the background process with there running status and logs on the *Processes* tab

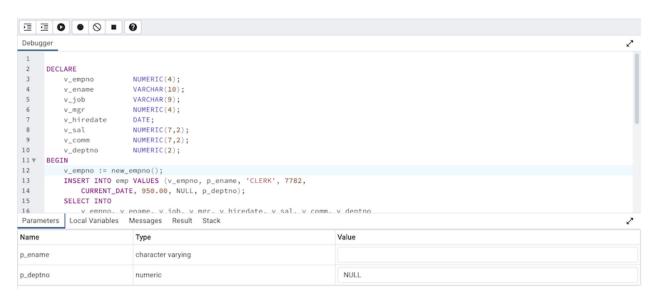
8.4. Restore Dialog 413

# CHAPTER 9

# **Developer Tools**

The pgAdmin *Tools* menu displays a list of powerful developer tools that you can use to execute and analyze complex SQL commands, manage data, and debug PL/SQL code.

## 9.1 Debugger



The debugger may be used to debug PL/pgSQL functions in PostgreSQL, as well as EDB-SPL functions, stored procedures and packages in EDB Postgres 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, for example:

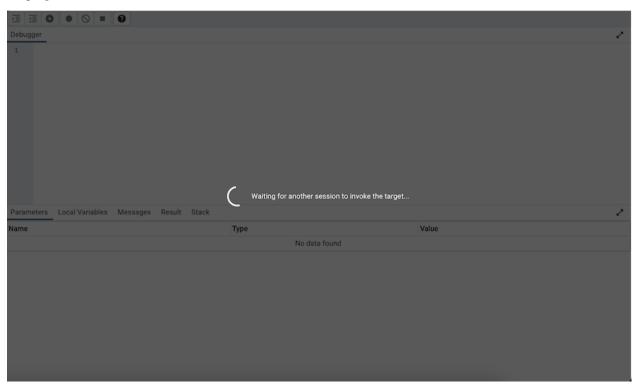
shared\_preload\_libraries = '\$libdir/plugin\_debugger'

After modifying the *shared\_preload\_libraries* parameter, restart the server to apply the changes.

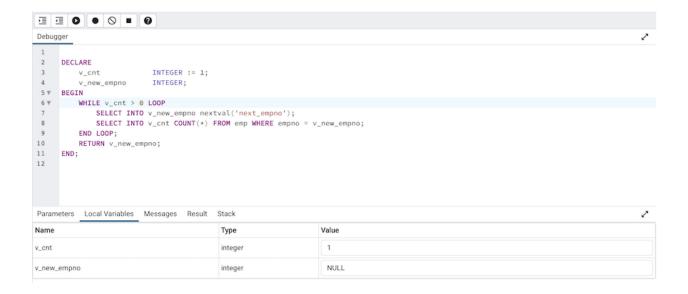
The debugger may be used for either in-context debugging or direct debugging of a target function or procedure. When you use the debugger for in-context debugging, you set a breakpoint at the first line of a program; when a session invokes the target, control is transferred to the debugger. When using direct debugging, the debugger prompts you for any parameters required by the target, and then allows you to step through the code.

#### 9.1.1 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.

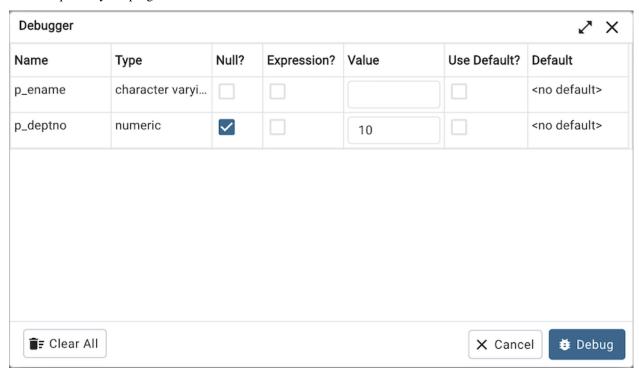


When another session invokes the target, the debugger will display the code, allowing you to add break points, or step through line-by-line. The other session is suspended until the debugging completes; then control is returned to the session.



## 9.1.2 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:



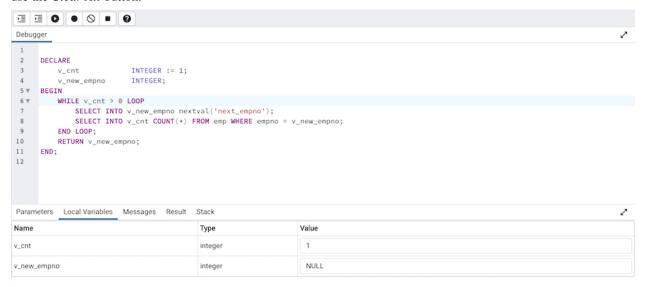
Use the fields on the *Debugger* dialog to provide a value for each parameter:

- 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.

9.1. Debugger 417

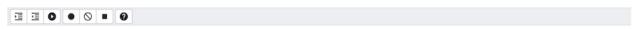
- Check the Expression? checkbox if the Value field contains an expression.
- Use the Value field to provide the parameter value that will be passed to the program. When entering parameter values, type the value into the appropriate cell on the grid, or, leave the cell empty to represent NULL, enter '(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 commadelimited 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.



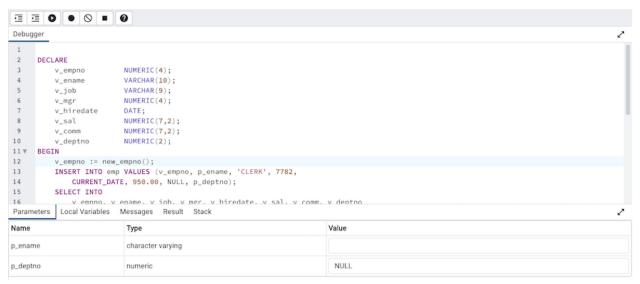
### 9.1.3 Using the Debugger

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 <i>Step into</i> icon to execute the currently highlighted line of code.
Step over	Click the <i>Step over</i> 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.
Con- tinue/Start	Click the <i>Continue/Start</i> icon to execute the highlighted code, and continue until the program encounters a breakpoint or completes.
Toggle breakpoint	Use the <i>Toggle breakpoint</i> icon to enable or disable a breakpoint (without removing the breakpoint).
Clear all break- points	Click the <i>Clear all breakpoints</i> icon to remove all breakpoints from the program.
Stop	Click the <i>Stop</i> icon to halt the execution of a program.
Help	Click the <i>Help</i> icon to open debugger documentation.

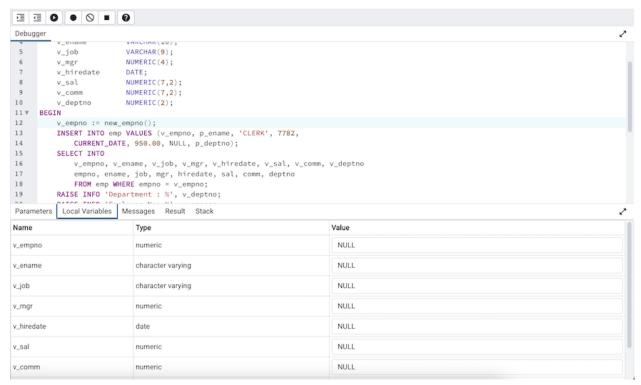
The top panel of the debugger window displays the program body; click in the grey margin next to a line number to add a breakpoint. The highlighted line in the top panel is the line that is about to execute.



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:



9.1. Debugger 419

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:



Select a caller to change focus to that stack frame and display the state of the caller in the upper panel.

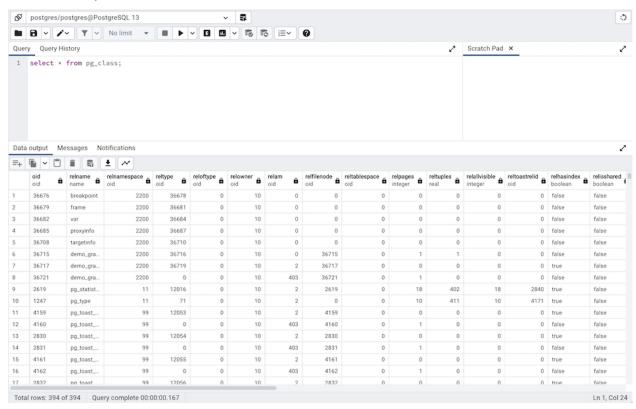
When the program completes, the *Results* tab displays the message returned by the server. If the program encounters an error, the *Messages* tab displays details:

```
Debugger
      DECLARE
                            NUMERIC(4);
          v_empno
                            VARCHAR(10);
          v_ename
                            VARCHAR(9);
          v_job
                            NUMERIC(4);
          v_mgr
          v_hiredate
                            DATE;
          v_sal
                            NUMERIC(7,2);
          v_comm
                            NUMERIC(7,2);
10
                           NUMERIC(2);
          v_deptno
11 ♥ BEGIN
12
          v_empno := new_empno();
          INSERT INTO emp VALUES (v_empno, p_ename, 'CLERK', 7782,
              CURRENT_DATE, 950.00, NULL, p_deptno);
15
          SELECT INTO
16 v emono, v ename, v iob, v mgr. v hiredate, v sal, v comm, v deptho
Parameters Local Variables Messages Result Stack
INFO: Inserting employee 8007
INFO: ..New salary: 950.00
INFO: User postgres added employee(s) on 2022-06-03
INFO: Department : <NULL&gt;
INFO: Employee No: 8007
                 : CLERK
INFO: Job
INFO: Manager
INFO: Hire Date : 2022-06-03
INFO: Commission: &lt:NULL&gt:
```

## 9.2 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 Object explorer 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.



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

9.2. Query Tool 421

notifications received from the server.

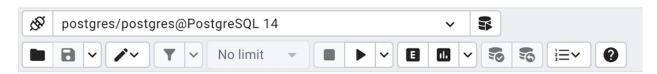
#### 9.2.1 Toolbar

The toolbar is described in the following subsections.

#### **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.

**Note:** The *Query Tool* and *View/Edit Data* tools are actually different operating modes of the same tool. Some controls will be disabled in either mode.



Hover over an icon in pgAdmin to display a tooltip that describes the icon's functionality.

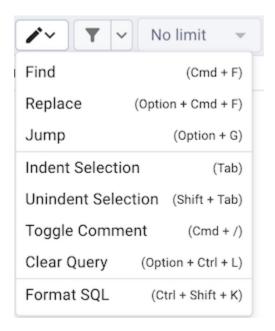
#### **File Options**

Icon	Behavior	Shortcut
Open File	Click the <i>Open File</i> icon to display a previously saved query in the SQL Editor.	Accesskey + O
Save File	<ul> <li>Click the <i>Save</i> icon to perform a quick-save of a previously saved query, or to access the <i>Save</i> menu:</li> <li>Select <i>Save</i> to save the selected content of the SQL Editor panel in a file.</li> <li>Select <i>Save As</i> to open a new browser dialog and specify a new location to which to save the selected content of the SQL Editor panel.</li> </ul>	Accesskey + S

#### **Filter/Limit Options**

Icon	Behavior	Shortcut
Filter	<ul> <li>Click the <i>Filter</i> icon to set filtering and sorting criteria for the data when in <i>View/Edit data mode</i>. Click the down arrow to access other filtering and sorting options: <ul> <li>In the <i>SQL Filter</i>, you can enter a SQL query as filtering criteria. In <i>Data Sorting</i>, you can select the column and specify the order for sorting.</li> <li>Click <i>Filter by Selection</i> to show only the rows containing the values in the selected cells.</li> <li>Click <i>Exclude by Selection</i> to show only the rows that do not contain the values in the selected cells.</li> <li>Click <i>Remove Sort/Filter</i> to remove any previously selected sort or filtering options.</li> </ul> </li> </ul>	Accesskey + F
Limit Selector	Select a value in the <i>Limit Selector</i> to limit the size of the dataset to a number of rows.	Accesskey + R

#### **Query Editing Options**



Icon	Behavior	Shortcut
Edit	Use the <i>Edit</i> 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.	Cmd+F
	Select Replace to locate and replace (with prompting) individual occurrences	Option+Cmd+F
	of the target.	(MAC)
		Ctrl+Shift+F
		(Others)
	Select Go to Line/Column to go to specified line number and column position	Cmd+L or Ctrl+L

continues on next page

9.2. Query Tool 423

Table 3 – continued from previous page

Icon	Behavior	Shortcut
	Select <i>Indent Selection</i> to indent the currently selected text.	Tab
	Select <i>Unindent Selection</i> to remove indentation from the currently selected text.	Shift+Tab
	Select <i>Toggle Comment</i> to comment/uncomment any lines that contain the selection in SQL style.	Cmd+/
	Select Clear Query to clear the query editor window.	Cmd+.
	Select Format SQL to format the selected SQL or all the SQL if none is selected	Shift+Cmd+K

#### **Query Execution**



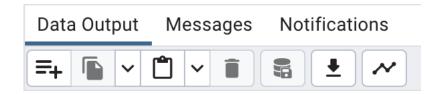
Icon	Behavior	Shortcut
Stop Execute script	<ul> <li>Click the <i>Stop</i> icon to cancel the execution of the currently running query.</li> <li>Click the <i>Execute script</i> icon to either execute or refresh the query highlighted in the SQL editor panel. Click the down arrow to access other execution options:</li> <li>Add a check next to <i>Auto rollback on error?</i> to instruct the server to automatically roll back a transaction if an error occurs during the transaction.</li> <li>Add a check next to <i>Auto commit?</i> 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.</li> </ul>	Accesskey + Q F5
Explain	Click the <i>Explain</i> icon to view an explanation plan for the current query. The result of the EXPLAIN is displayed graphically on the <i>Explain</i> tab of the output panel, and in text form on the <i>Data Output</i> tab.	F7

continues on next page

Table 4 – continued from previous page

Icon	Behavior	Shortcut
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.  • Select Settings to include the information on the configuration parameters.  • Select Wal to include the information on WAL record generation.	Shift+F7
Commit	Click the <i>Commit</i> icon to commit the transaction.	Shift+CTRL+M
Rollback	Click the <i>Rollback</i> icon to rollback the transaction.	Shift+CTRL+R
Macros	Click the <i>Macros</i> icon to manage the macros. You can create, edit or clear the macros through the <i>Manage Macros</i> option.	

## **Data Editing Options**



Click the Add row icon to add a new row  Click the Copy icon to copy the content with or without header:  Click the Copy icon to copy the content that is currently highlighted in the Data Output panel.  Click Copy with headers to copy the highlighted content along with the header.  Click the Paste icon to paste a previously copied row with or without serial/identity values:  Click the Paste icon to paste a previously copied row into a new row.  Click the Paste with SERIAL/IDENTITY values? if you want to paste the copied column values in the serial/identity columns.	Icon	Behavior	Shortcut
<ul> <li>Click the <i>Copy</i> icon to copy the content that is currently highlighted in the Data Output panel.</li> <li>Click <i>Copy</i> with headers to copy the highlighted content along with the header.</li> </ul> Paste <ul> <li>Click the Paste icon to paste a previously copied row with or without serial/identity values:</li> <li>Click the Paste icon to paste a previously copied row into a new row.</li> <li>Click the Paste with SERIAL/IDENTITY values? if you want to paste</li> </ul>	Add row	Click the Add row icon to add a new row	
rial/identity values:  • Click the <i>Paste</i> icon to paste a previously copied row into a new row.  • Click the <i>Paste with SERIAL/IDENTITY values?</i> if you want to paste	Сору	<ul> <li>Click the Copy icon to copy the content that is currently highlighted in the Data Output panel.</li> <li>Click Copy with headers to copy the highlighted content along with</li> </ul>	Accesskey + C
	Paste	<ul> <li>rial/identity values:</li> <li>Click the <i>Paste</i> icon to paste a previously copied row into a new row.</li> <li>Click the <i>Paste with SERIAL/IDENTITY values?</i> if you want to paste</li> </ul>	Accesskey + P

continues on next page

Table 5 – continued from previous page

Icon	Behavior	Shortcut
Delete	Click the <i>Delete</i> icon to mark the selected rows for deletion. These marked rows get deleted when you click the <i>Save Data Changes</i> icon.	Accesskey + D
Save Data Changes	Click the <i>Save Data Changes</i> icon to save data changes (insert, update, or delete) in the Data Output Panel to the server.	F6
Save results to file	Click the Save results to file icon to save the result set of the current query as a delimited text file (CSV, if the field separator is set to a comma). This button will only be enabled when a query has been executed and there are results in the data grid. You can specify the CSV/TXT settings in the Preference Dialogue under SQL Editor -> CSV/TXT output.	F8
Graph Visu- aliser	Use the Graph Visualiser button to generate graphs of the query results.	

#### **Status Bar**

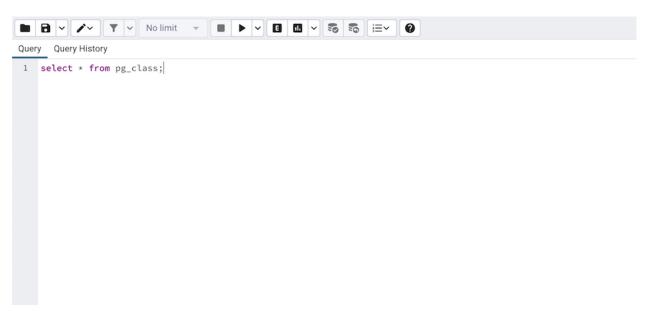
Total rows: 4 of 4	Query complete 00:00:00.090	Rows selected: 3	Changes staged: Added: 1 Updated: 1 Deleted: 1	Ln 2. Col 13
	4 , p		gg	

The status bar shows the following information:

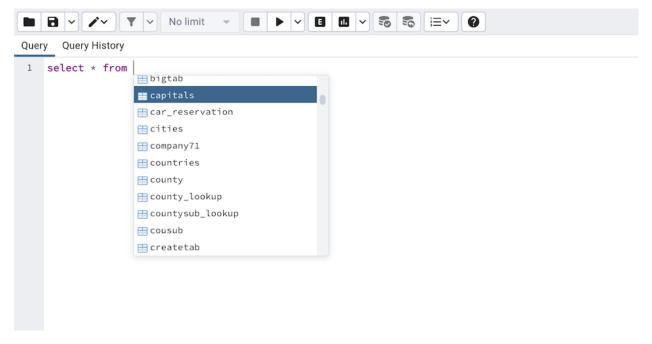
- **Total rows**: The total number of rows returned by the query.
- Query complete: The time is taken by the query to complete.
- **Rows selected**: The number of rows selected in the data output panel.
- Changes staged: This information showed the number of rows added, deleted, and updated.
- Ln: In the Query tab, it is the line number at which the cursor is positioned.
- Col: In the Query tab, it is the column number at which the cursor is positioned

## 9.2.2 The SQL Editor Panel

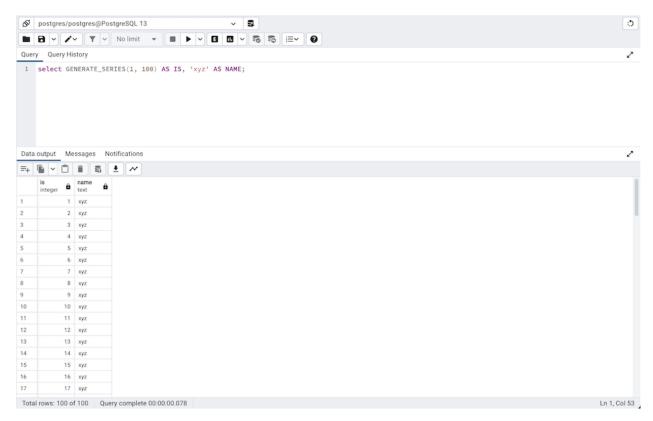
The *SQL editor* panel is a workspace where you can manually provide a query, copy a query from another source, or read a query from a file. The SQL editor features syntax coloring and autocompletion.



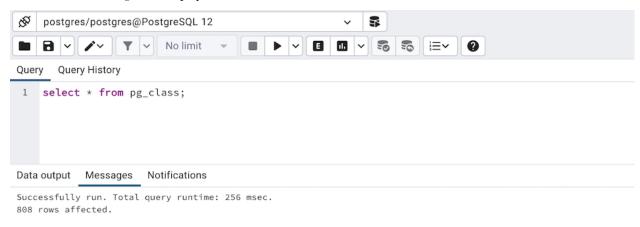
To use autocomplete, begin typing your query; when you would like the Query editor to suggest object names or commands that might be next in your query, press the Control+Space key combination. For example, type "SELECT \* FROM" (without quotes, but with a trailing space), and then press the Control+Space key combination to select from a popup menu of autocomplete options.



After entering a query, select the *Execute script* 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 script* 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.



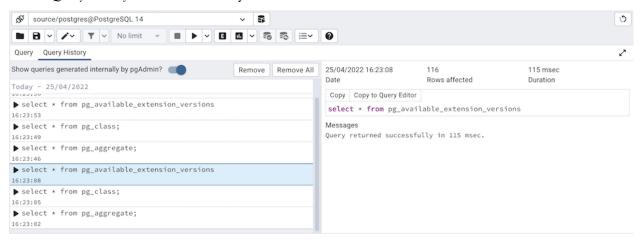
Options on the *Edit* menu offer functionality that helps with code formatting and commenting:

- 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 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.

## 9.2.3 Query History Panel

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



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).

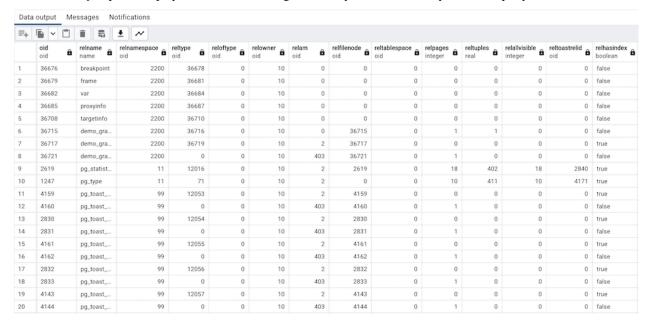
You can remove a single query by selecting it and clicking on the *Remove* button. If you would like to remove all of the histories from the *Query History* tab, then click on the *Remove All* button.

By using the *Copy* button, you can copy a particular query to the clipboard, and with the *Copy to Query Editor* button, you can copy a specific query to the Query Editor tab. During this operation, all existing content in the Query Editor is erased.

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 or config\_system.py (see the *config.py* documentation) by overriding the *MAX\_QUERY\_HIST\_STORED* value. See the *Deployment* section for more information.

## 9.2.4 The Data Output Panel

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



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

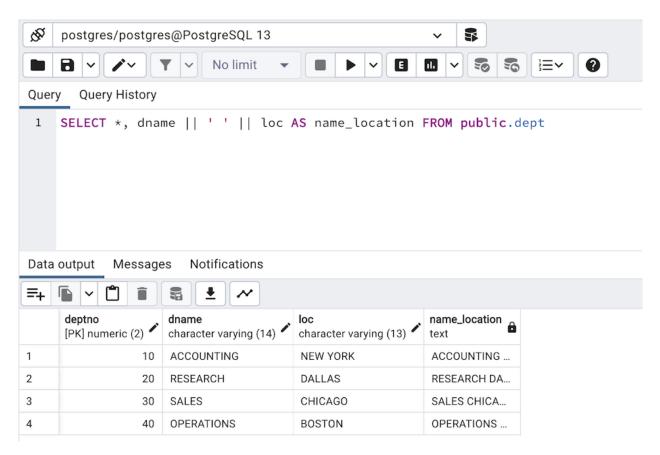
- Select and copy from the displayed result set.
- Use the *Execute script* options to retrieve query execution information and set query execution options.
- Use the Save results to file icon to save the content of the Data Output tab as a comma-delimited file.
- Edit the data in the result set of a 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.



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 tab to keep your previous results available.

## 9.2.5 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.

```
3
    postgres/postgres@PostgreSQL 16
                          No limit
                                                                                  0
Query
       Query History
                                                                ✓ Verbose
     CREATE EXTENSION IF NOT EXISTS postgis WITH SCHEMA
 1

✓ Costs

 2

✓ Buffers

 3
     CREATE TABLE capitals (
 4
         id integer NOT NULL,

✓ Timing

 5
         name character varying(256),
 6
         country_id integer,

✓ Summary

 7
          geom geometry (Point, 4326)

✓ Settings

 8
     );
 9
                                                               ✓ Wal
10
     CREATE TABLE countries (
11
          id integer NOT NULL,
12
         name character varying(256),
13
          geom geometry (MultiPolygon, 4326)
```

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

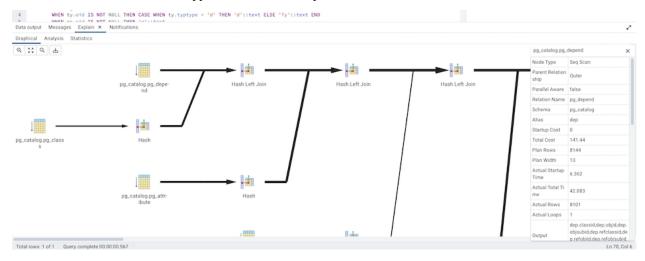
On successful generation of Explain plan, it will create three tabs/panels under the Explain panel.

#### 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 on 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.

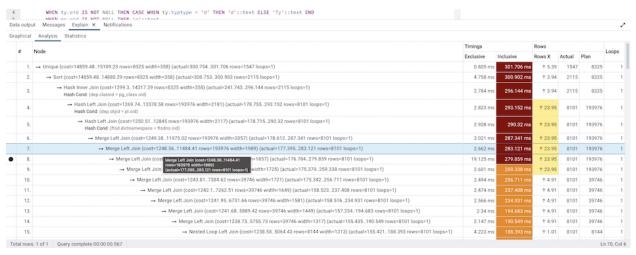
#### • Table

*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.

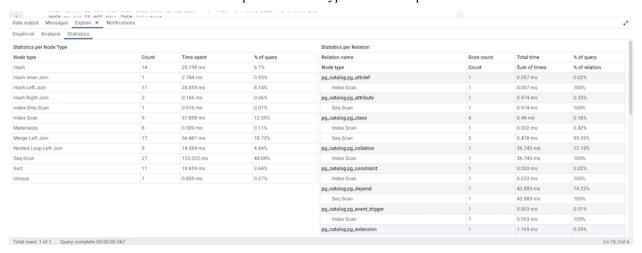
If percentage of the exclusive/inclusive timings of the total query time is: > 90 - Red color > 50 - Orange (between red and yellow) color > 10 - Yellow color

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



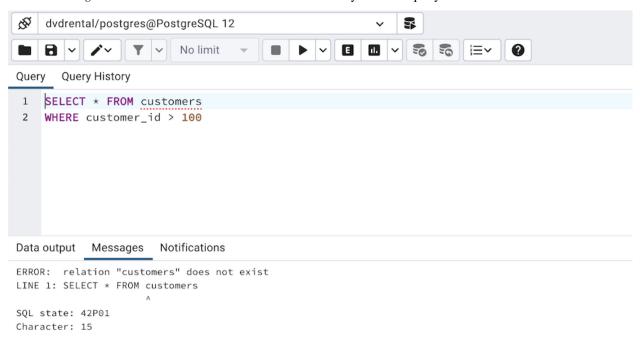
#### • Statistics

Statistics tab shows two tables: 1. Statistics per Plan Node Type 2. Statistics per Table

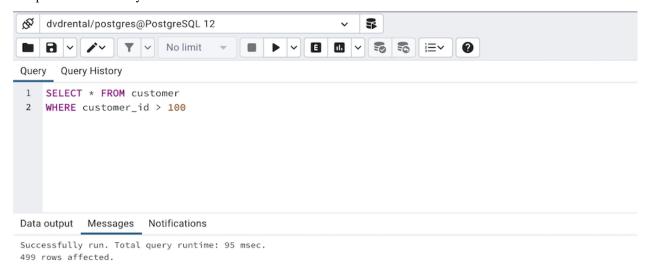


## 9.2.6 Messages Panel

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

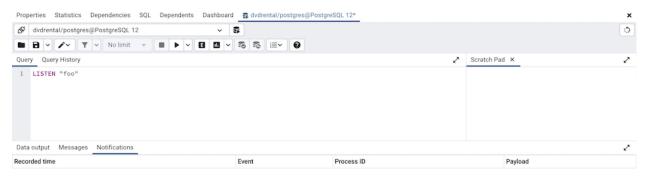


## 9.2.7 Notifications Panel

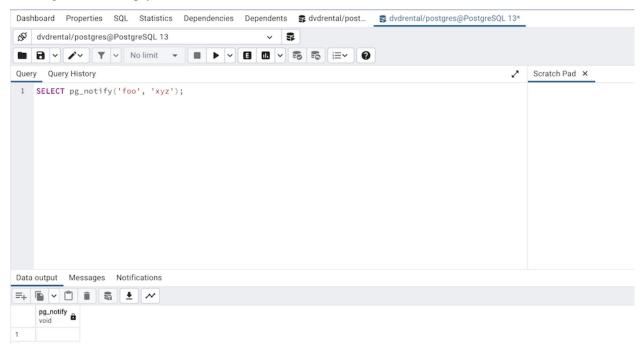
Use the *Notifications* tab to view the notifications using PostgreSQL *Listen/ Notify* feature. For more details see PostgreSQL documentation.

#### Example:

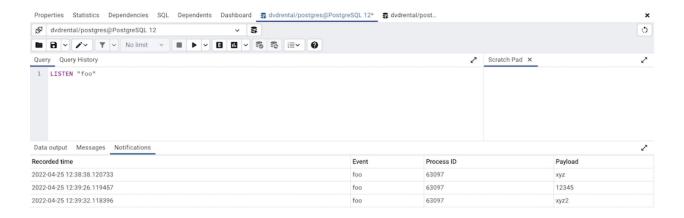
1. Execute LISTEN "foo" in first Query Tool session



2. In the another  $Query\ Tool$  session, execute Notify command or  $pg\_notify$  function to send the notification of the event together with the payload.

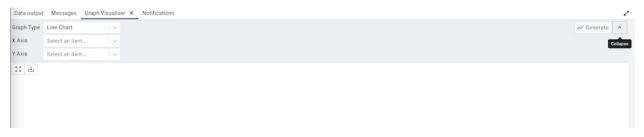


3. You can observe the *Notification* tab in the first *Query Tool* session where it shows the Recorded time, Event, Process ID, and the Payload of the particular channel.



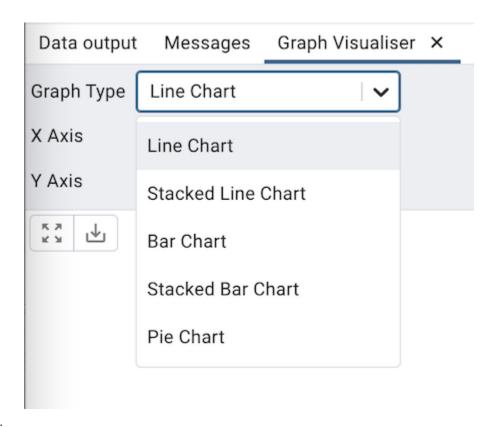
## 9.2.8 Graph Visualiser Panel

Click the Graph Visualiser button in the toolbar to generate the *Graphs* of the query results. The graph visualiser supports Line Charts, Stacked Line Charts, Bar Charts, Stacked Bar Charts, and Pie Charts.



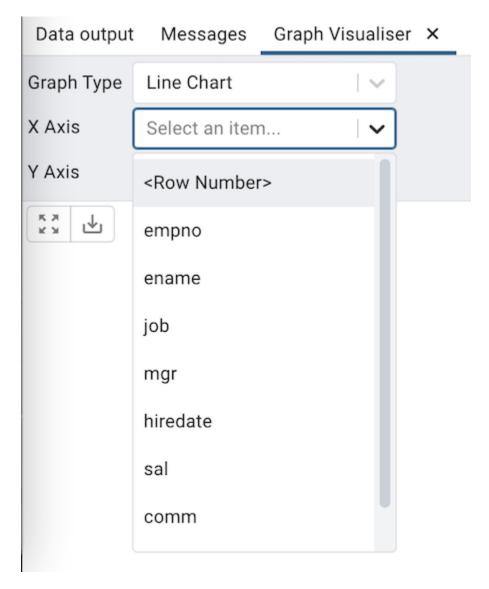
· Graph Type

Choose the type of the graph that you would like to generate.



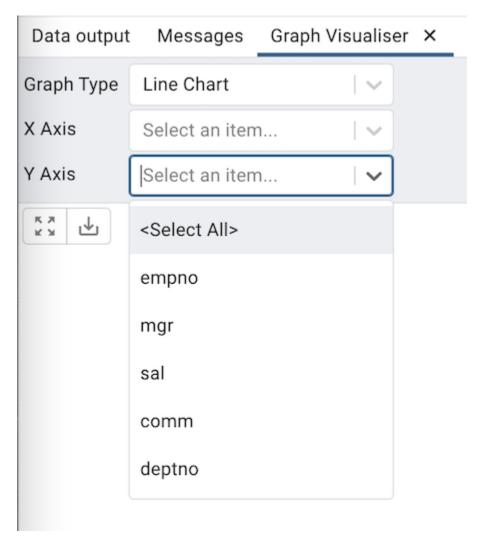
### • X Axis

Choose the column whose value you wish to display on X-axis from the X Axis dropdown. Select the < R N M M option to use the number of rows as labels on the X-axis.



### • Y Axis

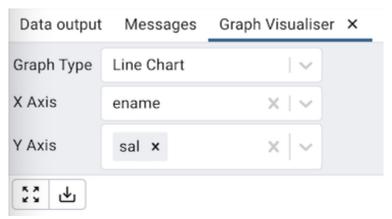
Choose the columns whose value you wish to display on Y-axis from the YAxis dropdown. Users can choose multiple columns. Choose the <Select All> option from the drop-down menu to select all the columns.



### • Download and Zoom button

Zooming is performed by clicking and selecting an area over the chart with the mouse. The *Zoom to original* button will bring you back to the original zoom level.

Click the *Download* button on the button bar to download the chart.



#### **Line Chart**

The *Line Chart* can be generated by selecting the 'Line Chart' from the Graph Type drop-down, selecting the X-axis and the Y-axis, and clicking on the 'Generate' button. Below is an example of a chart of employee names and their salaries.



Set *Use different data point styles?* option to true in the *Preferences Dialog*, to show data points in a different style on each graph lines.

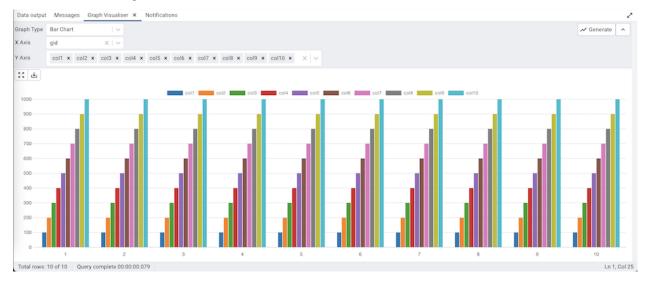
#### **Stacked Line Chart**

The *Stacked Line Chart* can be generated by selecting the 'Stacked Line Chart' from the Graph Type drop-down, selecting the X-axis and the Y-axis, and clicking on the 'Generate' button.



### **Bar Chart**

The *Bar Chart* can be generated by selecting the 'Bar Chart' from the Graph Type drop-down, selecting the X-axis and the Y-axis, and clicking on the 'Generate' button.



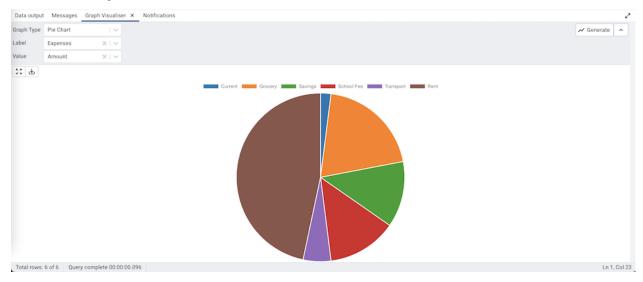
### **Stacked Bar Chart**

The *Stacked Bar Chart* can be generated by selecting the 'Stacked Bar Chart' from the Graph Type drop-down, selecting the X-axis and the Y-axis, and clicking on the 'Generate' button.



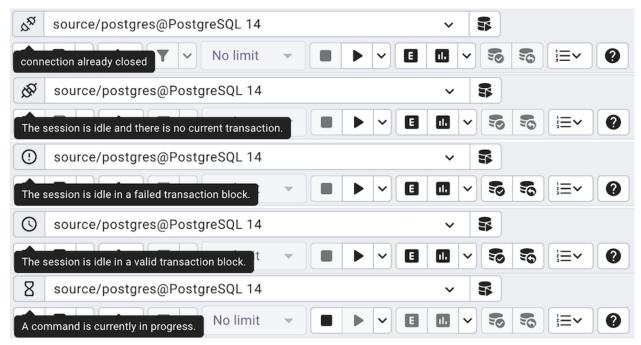
#### **Pie Chart**

The *Pie Chart* can be generated by selecting the 'Pie Chart' from the Graph Type drop-down, selecting the Label and Value, and clicking on the 'Generate' button.



#### 9.2.9 Connection Status

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



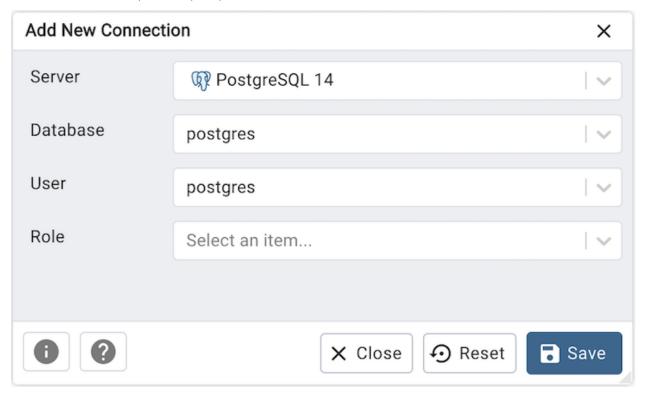
## 9.2.10 Change connection

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.



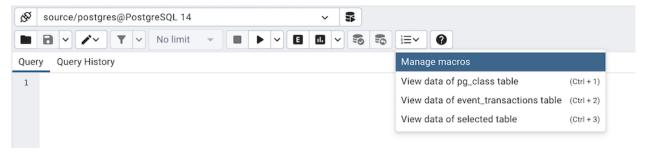
• Now select server, database, user, and role to connect and click on the 'Save' button.



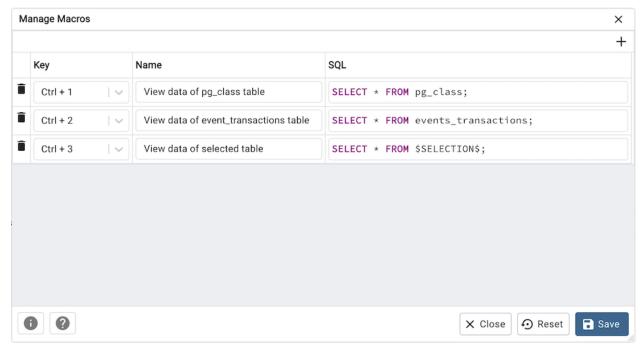
- A newly created connection will now get listed in the options.
- To connect, select the newly created connection from the dropdown list.

## 9.2.11 Macros

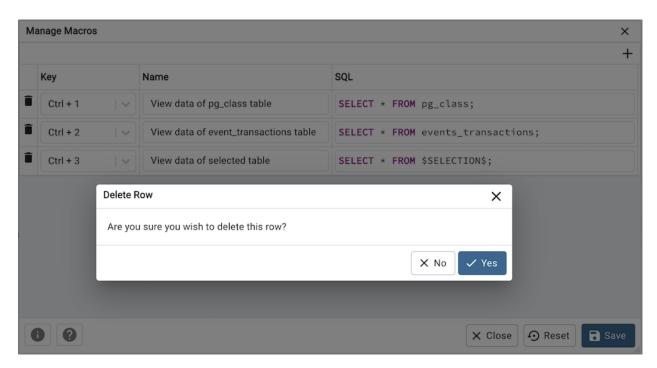
Query Tool Macros enable you to execute pre-defined SQL queries with a single key press. Pre-defined queries can contain the placeholder \$SELECTION\$. Upon macro execution, the placeholder will be replaced with the currently selected text in the Query Editor pane of the Query Tool.



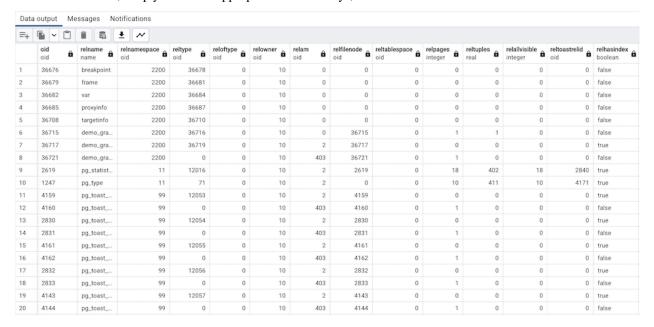
To create a macro, select the *Manage Macros* option from the *Macros* menu on the *Query Tool*. Select the key you wish to use, enter the name of the macro, and the query, optionally including the selection placeholder, and then click the *Save* button to store the macro.



To delete a macro, select the macro on the *Manage Macros* dialogue, and then click the *Delete* button. The server will prompt you for confirmation to delete the macro.

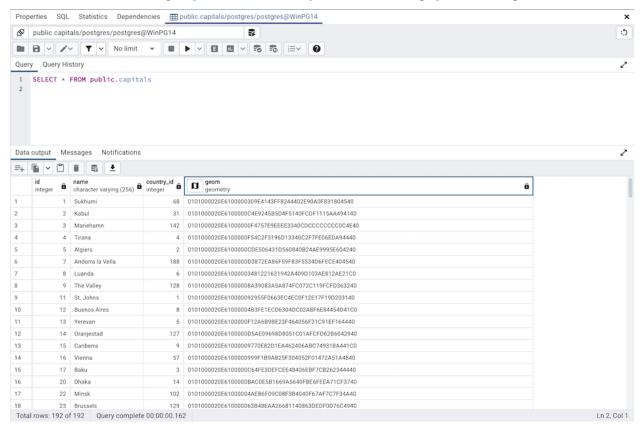


To execute a macro, simply select the appropriate shortcut keys, or select it from the *Macros* menu.



## 9.3 View/Edit Data

To view or modify data, right click on a table or view name in the *Object Explorer*. 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.



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.

### 9.3.1 The View/Edit Data Toolbar

The *Query Tool* and *View/Edit Data* tools are actually different operating modes of the same tool. Some controls will be disabled in either mode. Please see *The Query Tool Toolbar* for a description of the available controls.

#### 9.3.2 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 *Ok* 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.

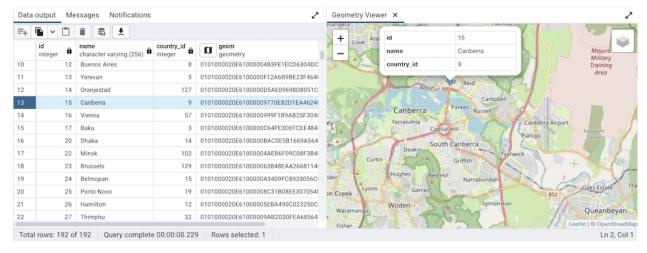
To write an empty string to the table, enter the special string "(two single quotes) in the field. If you want to write a string containing solely two single quotes to the table, you need to escape these quotes, by typing "

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 Data toolbar button.

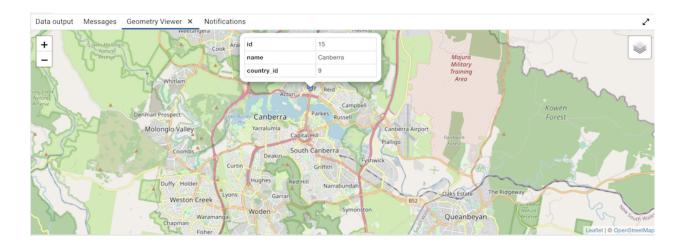
#### **Geometry Data Viewer**

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:



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:

9.3. View/Edit Data 447

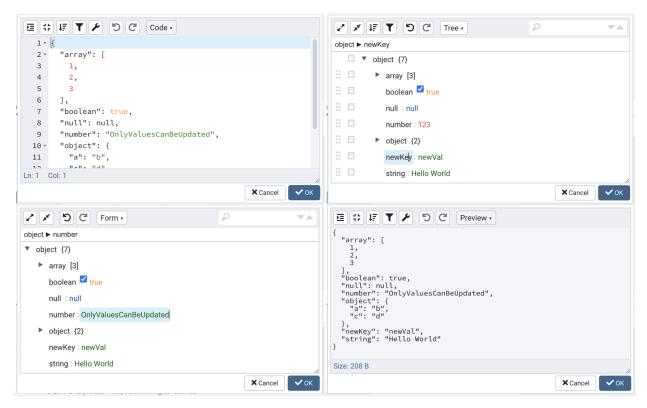


#### Note:

- 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.

#### **JSON Data Editor**

A built in json editor is provided for *JSON/JSONB Data*. Double clicking on json/jsonb data type cell in data grid will open JSON Editor. Editor provides different mode to view and edit json data.



*Code Mode*: Provides way to format & compact json data. Also provides ability to repair json data by fixing quotes and escape characters, removing comments and JSONP notation and turn JavaScript objects into JSON.

*Tree Mode*: Enabled to change, add, move, remove, and duplicate fields and values. Provides ability to searh & hilight data.

Form Mode: Allows only to edit values in json data there by providing ability to keep data structure unchanged while editing.

*Preview Mode*: Provides ability to check data before saving and also shows size of current json data. Format and compact json data as well.

Editor Toolbar

Different options are provided to manipulate json data.

Code/Preview mode:



Tree/Form mode:



Icon	Behavior	Available in mode
Format Json	Click to <i>Format Json</i> format json data with proper indentation.	Code, Preview
Compact Ison	Click to Compact Json get compacted json data.	Code, Preview

continues on next page

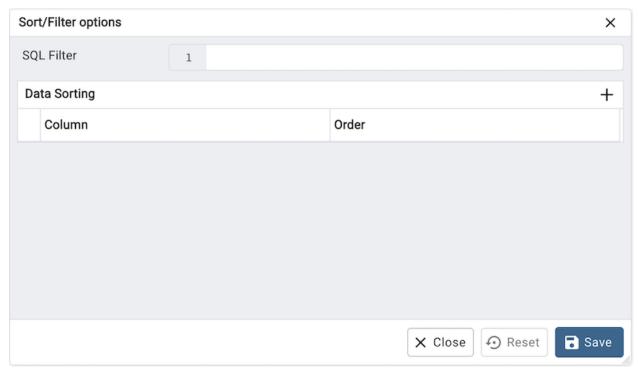
9.3. View/Edit Data 449

Table 6 – continued from previous page

Icon	Behavior	Available in mode
Sort	Click to <i>Sort</i> icon to set sorting criteria for the data using file values.	Code, Preview, Tree
Transform	Click to <i>Transform</i> to filter data using JSME query language.	Code, Preview, Tree
Undo	Click to <i>Undo</i> to undo last action performed on data.	Code, Preview, Tree, From
Redo	Click to Redo to repat last action performed on data.	Code, Preview, Tree, From
Mode	Click to <i>Mode</i> dropdown to change dipaly mode of editor. Different modes available are Code, Preview, Tree, From.	Code, Tree, From, Preview
Expand All	Click to Expand All to expand json data.	Tree, From
Collapse All	Click to <i>Redo</i> to collapse json data.	Tree, From
Search Box	Enter partial/complete string to search in data.	Tree, From

## 9.3.3 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 Save button to save work.
- Click the *Close* button to discard current changes and close the dialog.

#### 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.

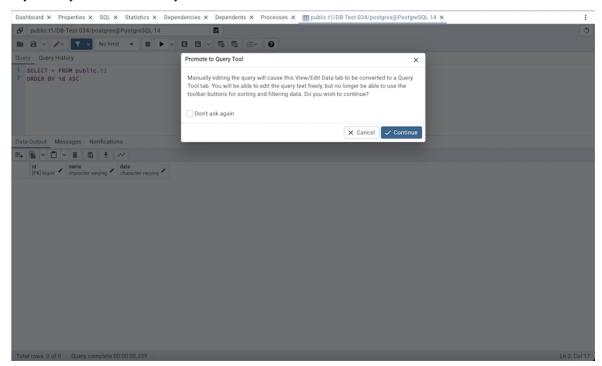
This allows you to specify an SQL Filter to limit the data displayed in the edit grid window:



9.3. View/Edit Data 451

## 9.3.4 Promote View/Edit Data to Query Tool

A View/Edit Data tab can be converted to a Query Tool Tab just by editing the query. Once you start editing, it will ask if you really want to move away from View/Edit.



You can disable the dialog by selecting the "Don't Ask again" checkbox. If you wish to resume the confirmation dialog, you can do it from "Prefrences -> Query Tool -> Editor -> Show View/Edit Data Promotion Warning?"

Once you chose to continue, you won't be able to use the features of View/Edit mode like the filter and sorting options, limit, etc. It is a one-way conversion. It will be a query tool now.

### 9.4 Schema Diff

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

The Schema Diff feature allows you to:

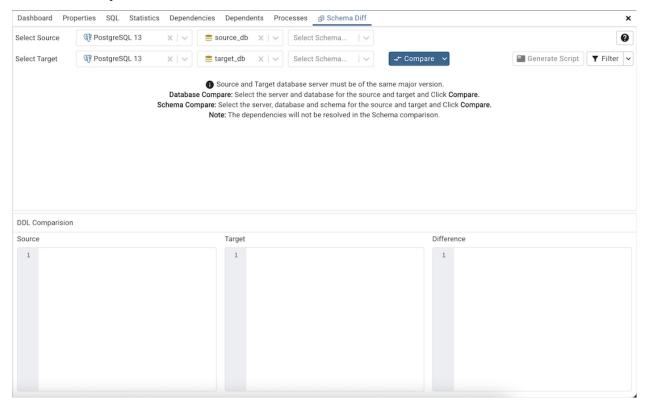
- Compare and synchronize the database objects (from source to target).
- Visualize the differences between database objects.
- List the differences in SQL statement for target database objects.
- Generate synchronization scripts.

#### 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.



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.

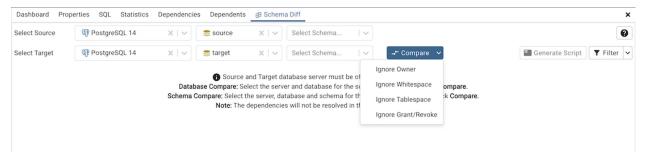
The Schema Diff panel is divided into two panels; an Object Comparison panel and a DDL Comparison panel.

9.4. Schema Diff 453

## 9.4.1 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 object explorer 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.

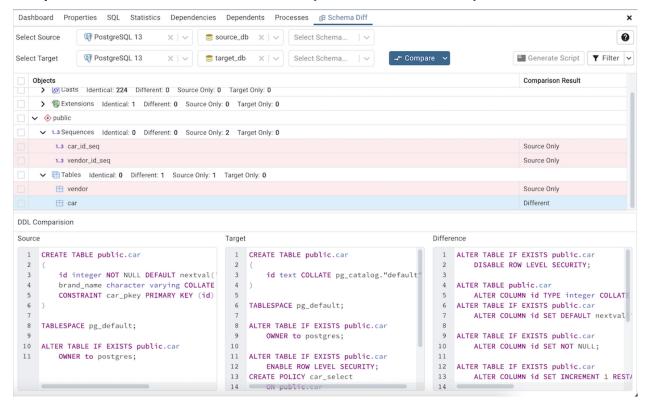
Next, select the databases that will be compared. The databases can be the same or different (and within the same server or from different servers).



Use the drop-down near to Compare button to ignore owner, whitespace, tablespace and grants.

- Ignore Owner Select to ignores the owner while comparing the objects.
- Ignore Whitespace Select to ignores the whitespace while comparing the string objects. Whitespace includes space, tabs, and CRLF.
- Ignore Tablespace Select to ignores the tablespace while comparing the objects.
- Ignore Grant/Revoke Select to ignores the grant and revoke command while comparing the objects.

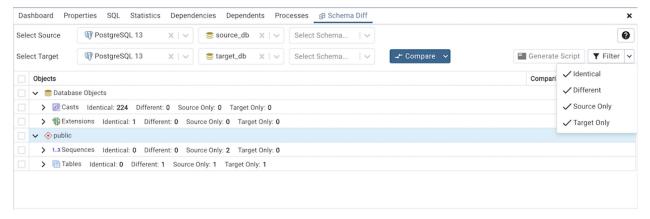
After you select servers, and databases, click on the Compare button to obtain the Comparison Result.



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.



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.

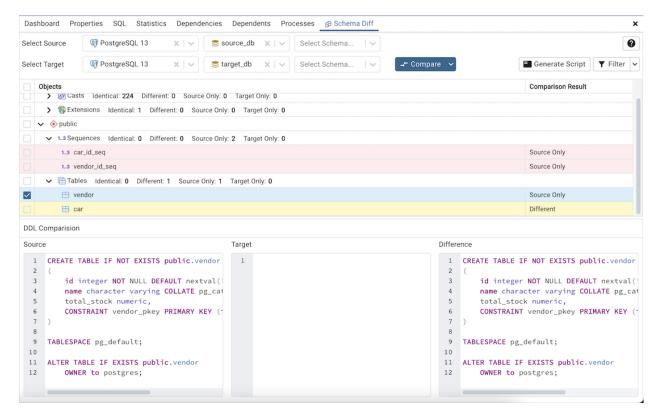
Click the *Help* icon to open Schema Diff documentation.

### 9.4.2 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.

9.4. Schema Diff 455



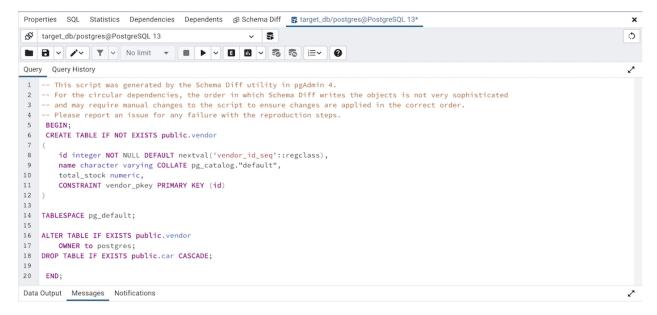
You can review the DDL statements of all the database objects to check for the differences in the SQL statements.

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.

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, pgAdmin 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.



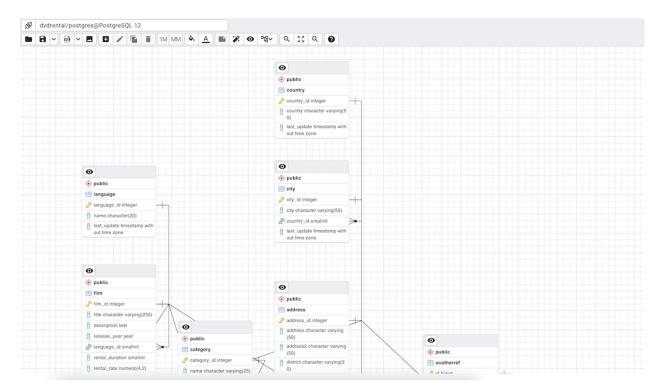
Apply the SQL Statement in the target database to synchronize the databases.

## 9.5 ERD Tool

The Entity-Relationship Diagram (ERD) tool is a database design tool that provides a graphical representation of database tables, columns, and inter-relationships. ERD can give sufficient information for the database administrator to follow when developing and maintaining the database. The ERD Tool allows you to:

- Design and visualize the database tables and their relationships.
- Add notes to the diagram.
- Auto-align the tables and links for cleaner visualization.
- Save the diagram and open it later to continue working on it.
- Generate ready to run SQL from the database design.
- Generate the database diagram for an existing database.
- Drag and drop tables from object explorer to the diagram.

9.5. ERD Tool 457



You can open multiple copies of the ERD tool in individual tabs simultaneously. You can also generate an ERD from a database, schema or a table.

- The ERD for database will fetch all the tables from all the schemas of the database and plot them with foreign key links.
- The ERD for schema will fetch all the tables from a schema and plot them with foreign key links. If any table refers to a table in another schema, then that link/foreign key will be removed.
- The ERD for table will fetch all the tables linked directly or indirectly to mentioned table. You can change the depth of traversal from *Preferences*.

#### 9.5.1 Toolbar

The *ERD Tool* toolbar uses context-sensitive icons that provide shortcuts to frequently performed tasks. The option is enabled for the highlighted icon and is disabled for the grayed-out icon.



Hover over an icon on Toolbar to display a tooltip that describes the icon's functionality.

## 9.5.2 File Options

Icon	Behavior	Shortcut
Open File	Click the <i>Open File</i> icon to load a previously saved diagram.	Ctrl + O
Save	Click the <i>Save</i> icon to perform a quick-save of a previously saved diagram, or to save the diagram to a file.	Ctrl + S
Save as	Click the <i>Save As</i> to open a new browser dialog and specify a new location to save the diagram. You need to click the down arrow beside the save button to see <i>Save As</i> .	Ctrl + Shift + S

## 9.5.3 Export Options

Icon	Behavior	Shortcut
Generate SQL	Click the <i>Generate SQL</i> icon to generate the DDL SQL for the diagram and open a query tool with the generated SQL ready for execution. You can select the option <i>With DROP Table</i> if you wish to have DROP Table DDL statements before each CREATE Table DDL. You can see the option by clicking the down arrow beside the SQL button.	
Download image	Click the <i>Download image</i> icon to save the ERD diagram in a image formate	Option + Ctrl + I

## 9.5.4 Editing Options

Icon	Behavior	Shortcut	
Add table	Click this button to add a new table to the diagram. On clicking, this will open a table dialog where you can put the table details.	Option/Alt Ctrl + A	+
Edit table	Click this button to edit a table on the diagram. On clicking, this will open a table dialog where you can change table details. This will enable when a table is selected.	Option/Alt Ctrl + E	+
Clone table	Click this button to clone the complete table structure, name it with a auto generated name and put it in the diagram.	Option/Alt Ctrl + C	+
Drop table/link	You can drop a table or link using this button. You need to select a table or link and click on this button to drop it.	Option/Alt Ctrl + D	+

## 9.5.5 Table Relationship Options

Icon	Behavior	Shortcut	
1M	Click this button to open a one-to-many relationship dialog to add a relationship between the two tables. The selected table becomes the referencing table and will have the <i>many</i> endpoint of the link.	Option/Alt Ctrl + O	+
MM	Click this button to open a many-to-many relationship dialog to add a relationship between the two tables. This option will create a new table based on the selected columns for the two relating tables and link them.	Option/Alt Ctrl + M	+

9.5. ERD Tool 459

# 9.5.6 Node Color Options

Icon	Behavior
Fill Color	Use Fill Color to change the background color of a table node. This is helpful if you want to identify a of group tables. Once set, all the newly added tables will take the same color.
Text Color	Use Text Color to change the text color of a table node based on the fill color to make text easily readable.

# 9.5.7 Utility Options

Icon	Behavior	Shortcut	
Add/Edit note	Click this button to make notes on tables nodes while designing the database.	Option/Alt Ctrl + N	+
Auto align	Click this button to auto align all tables and links to make it look more cleaner.	Option/Alt Ctrl + L	+
Show details	Click this button to toggle the column details visibility. It allows you to show few or more column details.	Option/Alt Shift + D	+
Cardinality No- tation	Change the cardinality notation format used to present relationship links. Options available are - Crow's Foot Notation and Chen Notation.		

## 9.5.8 Zoom Options

Icon	Behavior	Shortcut	
Zoom to fit	Click this button to zoom in/out automatically and fit all the tables to the view.	Option/Alt Shift + F	+
Zoom in	Click this button to zoom in the diagram.	Option/Alt Shift + "+"	+
Zoom out	Click this button to zoom out the diagram.	Option/Alt Shift + "-"	+

# 9.5.9 Table Dialog

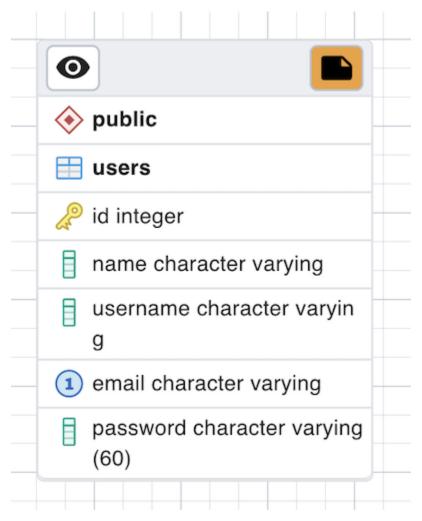


The table dialog allows you to:

- Change the table structure details.
- It can be used edit an existing table or add a new one.
- Refer *table dialog* for information on different fields.

9.5. ERD Tool 461

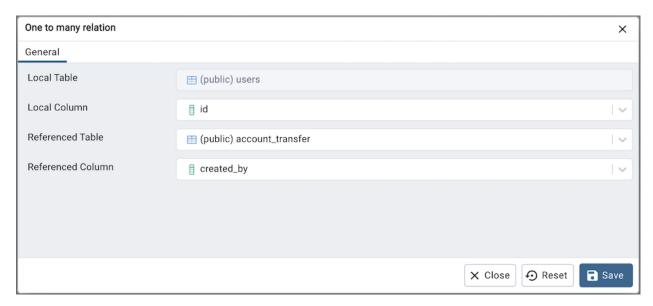
### 9.5.10 Table Node



The table node shows table details in a graphical representation:

- The top bar has a *details toggle button* that is used to toggle column details visibility. There is also a *note button* that is visible only if there is some note added. you can click on this button to quickly change the note.
- The first row shows the schema name of the table. Eg. public in above image.
- The second row shows the table name. Eg. users in above image.
- All other rows below the table name are the columns of the table along with data type. If the column is a primary key then it will have lock key icon eg. id is the primary key in above image. Otherwise, it will have column icon.
- you can click on the node and drag to move on the canvas.
- Upon double click on the table node or by clicking the edit button from the toolbar, the table dialog opens where you can change the table details. Refer *table dialog* for information on different fields.

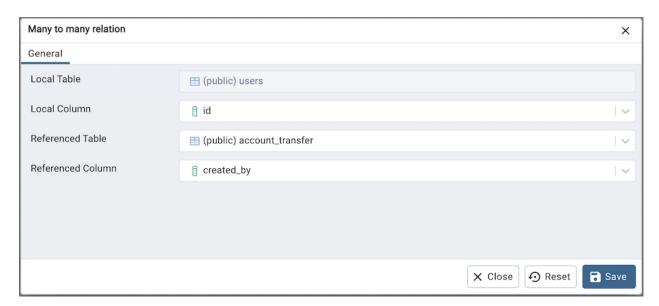
## 9.5.11 The One to Many Link Dialog



The one to many link dialog allows you to:

- Add a foreign key relationship between two tables.
- Local Table is the table that references a table and has the many end point.
- Local Column the column that references.
- Referenced Table is the table that is being referred and has the one end point.
- Referenced Column the column that is being referred.

### 9.5.12 The Many to Many Link Dialog



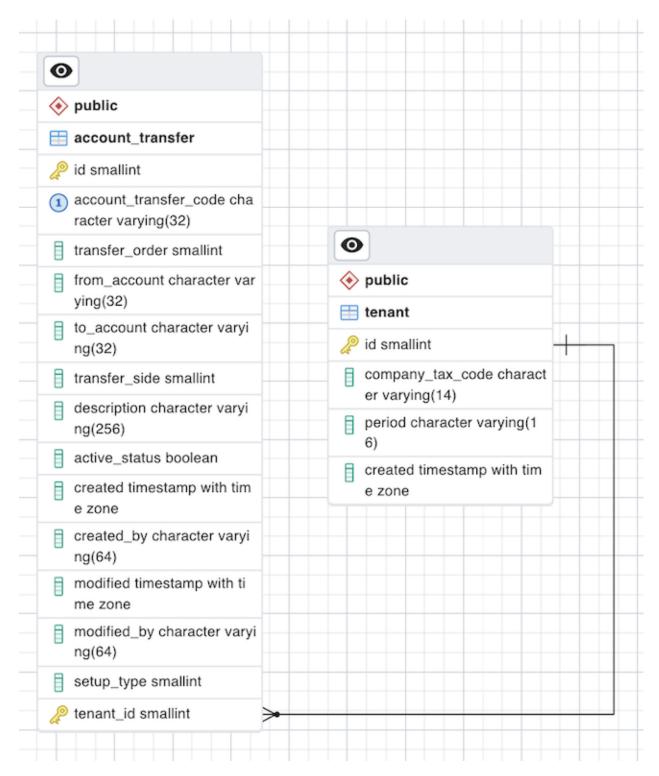
The many to many link dialog allows you to:

• Add a many to many relationship between two tables.

9.5. ERD Tool 463

- It creates a relationship tables having columns derived from the two tables and link them to the tables.
- *Left Table* is the first table that is to be linked. It will receive the *one* endpoint of the link with the new relation table.
- Left Column the column of the first table, that will always be a primary key.
- *Right Table* is the second table that is to be linked. It will receive the *one* endpoint of the link with the new relation table.
- Right Column the column of the second table, that will always be a primary key.

### 9.5.13 The Table Link



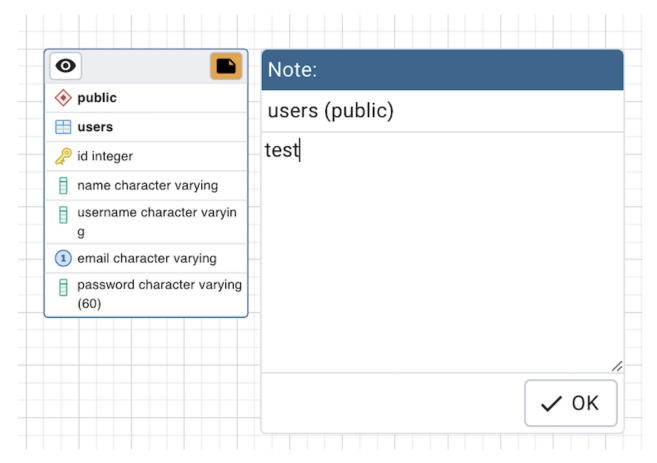
The table link shows relationship between tables:

- The single line endpoint of the link shows the column that is being referred.
- The three line endpoint of the link shows the column that refers.

9.5. ERD Tool 465

- If one of the columns that is being referred or that refers is removed from the table then the link will get dropped.
- you can click on the link and drag to move on the canvas.

#### 9.5.14 The Table Notes



- You can use the notes popup to mark some notes while designing the database.
- You open the pop-up using the toolbar note button.
- If some note is added to a table then it will have notes button on the table node. You can click on the button to check/update notes.

# 9.6 PSQL Tool

The PSQL tool allows users to connect to PostgreSQL or EDB Advanced server using the psql command line interface through their browser.

- Open the PSQL tool from the Tools or object explorer context menu, or use PSQL tool button at the top of the object explorer.
- PSQL will connect to the current connected database from the object explorer.

```
Properties
           SOL
                 >_ postgres/postgres@PostgreSQL 10
                                                                ×
psql (10.10)
Type "help" for help.
postgres=# select * from dept;
 deptno
             dname
                            loc
     20
           RESEARCH
                         DALLAS
     40
           OPERATIONS
                         BOSTON
     30
           SALES123
                         CHICAGO
           ACCOUNTING
                         NEW YORK
(4 rows)
postgres=#
```

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

**Note:** On the Windows platform, this feature is available on Windows 10 (1809 version), and Windows Server 2019 and onwards.

**Note:** The PSQL tool is always available when pgAdmin is running in Desktop mode, but is disabled by default in Server mode. This is because users can run arbitrary shell commands through psql which may be considered a security risk in some deployments. System Administrators can enable the use of the PSQL tool in the pgAdmin configuration by setting the *ENABLE\_PSQL* option to *True*; see *The config.py File* for more information.

9.6. PSQL Tool 467

# CHAPTER 10

### **Processes**

There are certain tasks which pgAdmin runs in the background. The processes running in the background can be viewed in the processes tab. It shows the process details of Backup, Restore, Maintenance, Import/Export and Cloud instance creation.



The columns of the processes table shows:

- The PID of the forked OS process.
- The *Type* of the task being performed.
- The Server name for which the task is.
- The *Object* can be a database, table, mview or anything which gives more info.
- The Start Time of the process, sorted descending by default.
- The current Status of the process. It can be Running, Finished, Failed, Terminated.

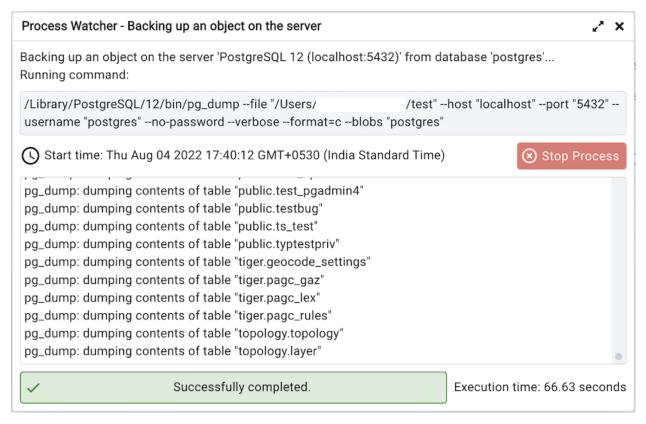
• The *Time Taken* to complete. It will keep updating if it is running.

There are two action buttons on each row:

- The Stop Process button allows you to kill a running process.
- The More details button allows you to open the process watcher which shows the process logs and other details.

You can also select the checkboxes and click on *Delete and Acknowledge* button on the top to clear the process info and logs.

### 10.1 Process Watcher



The Process Watcher logs all the activity associated with the process/task and provides additional information for troubleshooting Use the **Stop Process** button to stop the Backup process.

**Note:** If you are running *pgAdmin* in *Server Mode* 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.

# 10.2 Watch the demo

https://youtu.be/jjP6O\_xuHxE

10.2. Watch the demo 471

# CHAPTER 11

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 of pgAdmin. You can download pgAgent from the download area of the pgAdmin website.

# 11.1 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 pgAdmin tree control, the *Properties* tab of the main pgAdmin window will display details about the job, and the *Statistics* tab will display details about the job's execution.

### 11.1.1 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.

# 11.2 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 pgAdmin 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.

### 11.2.1 Database setup

Before using pgAdmin to manage pgAgent, you must create the pgAgent extension in the maintenance database registered with pgAdmin. 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;

#### 11.2.2 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 'pgadmin', connecting as the user 'postgres':

```
/path/to/pgagent hostaddr=127.0.0.1 dbname=postgres user=postgres
```

### 11.2.3 Service installation on Windows

**Note:** pgAgent is available in a pre-built installer if you use EnterpriseDB's PostgreSQL Installers. Use the Stack-Builder 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
```

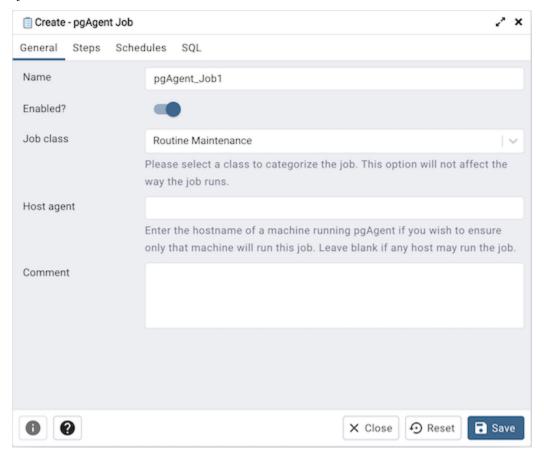
You can then start the service at the command line using *net start pgAgent*, or from the *Services* control panel applet. Any logging output or errors will be reported in the Application event log. The DEBUG mode may be used to run pgAgent from a command prompt. When run this way, log messages will output to the command window.

# 11.3 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 pgAdmin 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:

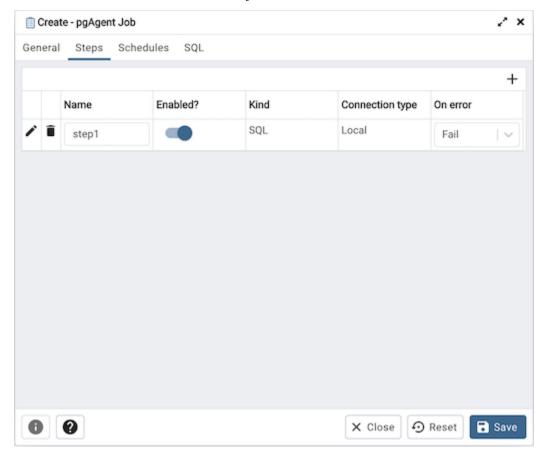
- Provide a name for the job in the Name field.
- Move the *Enabled* switch to the *Yes* position to enable a job, or *No* to disable a job.
- Use the *Job Class* drop-down to select a class (for job categorization).
- Use the *Host Agent* field to specify the name of a machine that is running pgAgent 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:

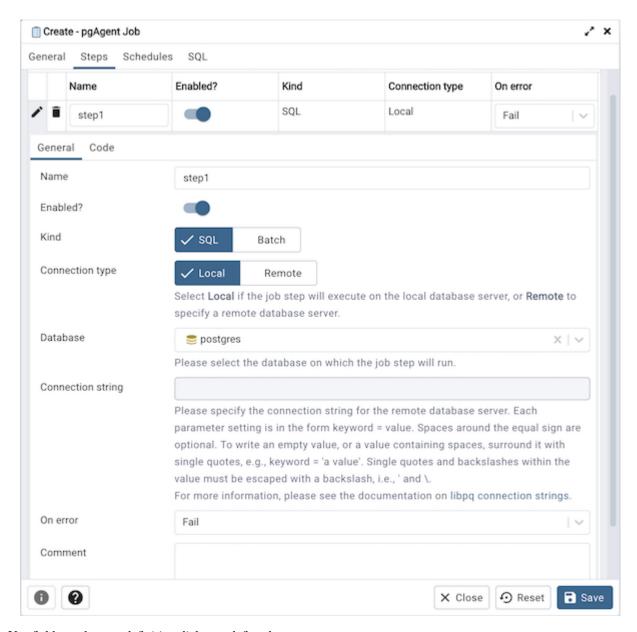
SELECT jagstation FROM pgagent.pga\_jobagent

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:

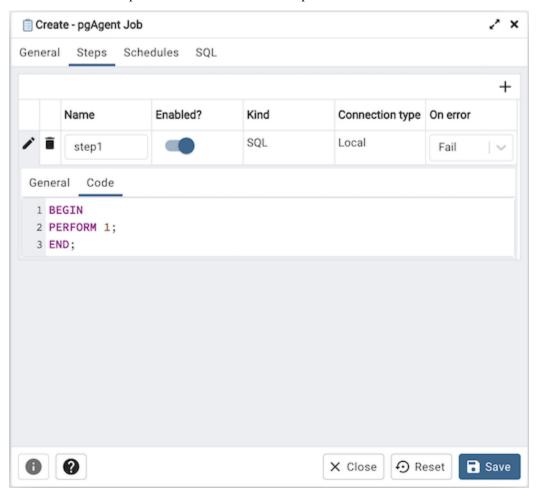
- Provide a name for the step in the Name field; please note that steps will be performed in alphanumeric order by name.
- Use the *Enabled* switch to include the step when executing the job (*True*) or to disable the step (*False*).
- Use the *Kind* switch to indicate if the job step invokes SQL code (*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.

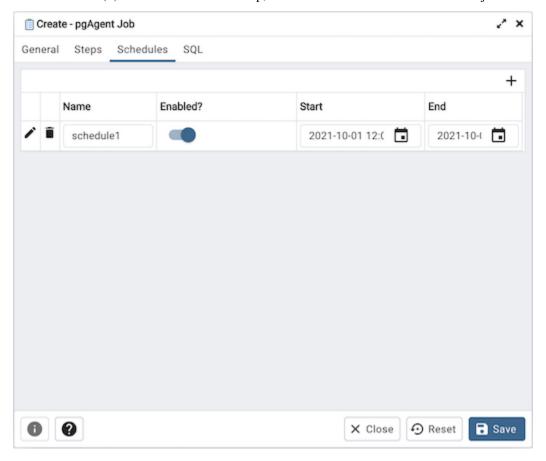
- Use the *Database* drop-down to select the database on which the job step will be performed.
- Use the *Connection string* field to specify a libpq-style connection string to the remote server on which the step will be performed. For more information about writing a connection string, please see the 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.



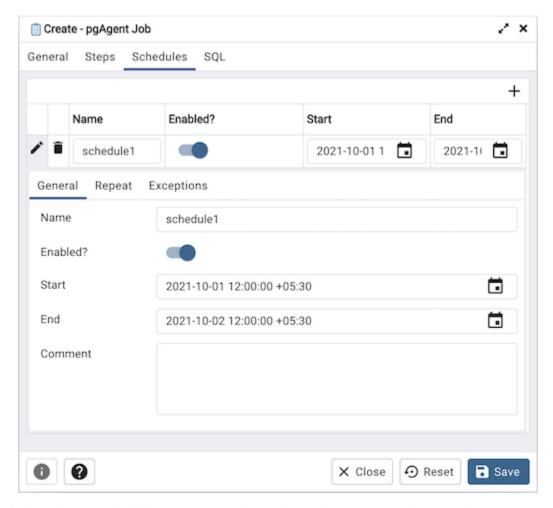
Use the context-sensitive field on the step definition dialog's *Code* tab to provide the SQL code or batch script that will be executed during the step:

- 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.



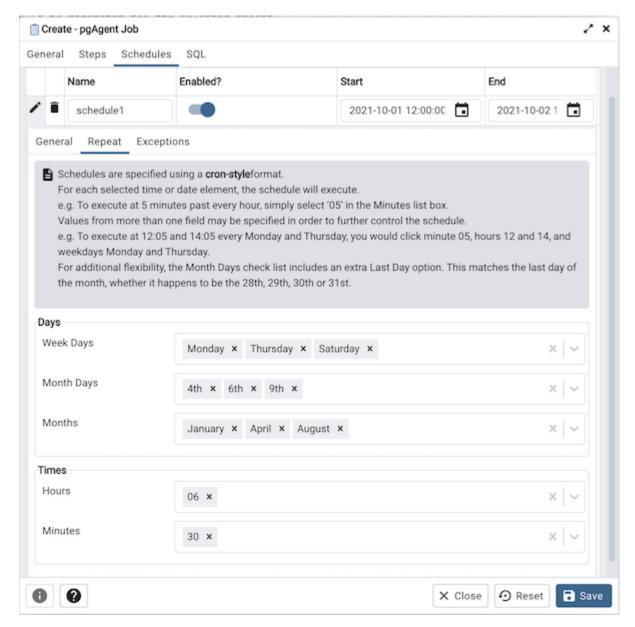
Click the Add icon (+) to add a schedule for the job; then click the compose icon (located at the left side of the header) to open the schedule definition dialog:



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.

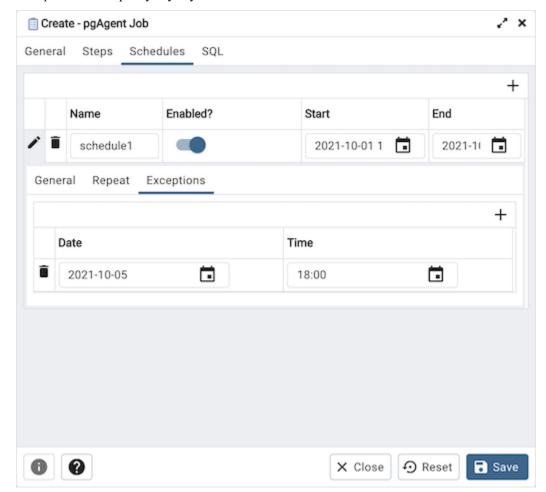
Click within a field to open a list of valid values for that field; click on a specific value to add that value to the list of selected values for the field. To clear the values from a field, click the X located at the right-side of the field.

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, irregardless 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.

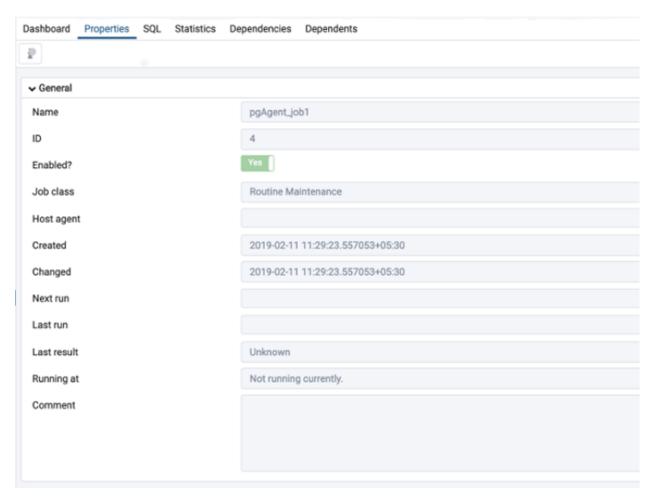
When you've finished defining the schedule, you can use the *SQL* tab to review the code that will create or modify your job.

```
2 ×
Create - pgAgent Job
General Steps Schedules
                         SQL
1 DO $$
2 DECLARE
       jid integer;
4
       scid integer;
5 BEGIN
6 -- Creating a new job
7 INSERT INTO pgagent.pga_job(
       jobjclid, jobname, jobdesc, jobhostagent, jobenabled
       1::integer, 'pgAgent_Job1'::text, ''::text, ''::text, true
10
11 ) RETURNING jobid INTO jid;
12
13 -- Steps
14 -- Inserting a step (jobid: NULL)
15 INSERT INTO pgagent.pga_jobstep (
16
      jstjobid, jstname, jstenabled, jstkind,
17
      jstconnstr, jstdbname, jstonerror,
18
      jstcode, jstdesc
19 ) VALUES (
20
       jid, 'stepl'::text, true, 's'::character(1),
21
       ''::text, 'postgres'::name, 'f'::character(1),
22
                                                                          3 Save
                                                     X Close

    Reset
```

Click the *Save* button to save the job definition, or *Close* 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 pgAdmin tree control of the server on which it was defined. The *Properties* tab in the main pgAdmin 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.

# pgAdmin Project Contributions

pgAdmin is an open-source project that invites you to get involved in the development process. For more information about contributing to the pgAdmin project, contact the developers on the pgAdmin mailing list pgadmin-hackers@postgresql.org to discuss any ideas you might have for enhancements or bug fixes.

In the following sections, you'll find detailed information about the development process used to develop, improve, and maintain the pgAdmin client.

# 12.1 Submitting Pull Requests

Before developing a feature or bug fix for pgAdmin you should always contact the developers on the mailing list pgadmin-hackers@postgresql.org to discuss your plans. This ensures that others know what you are doing and can then avoid duplicating your work, and in the case of large changes, gives the community the chance to discuss and refine your ideas before investing too much time writing code that may later be rejected.

You should always develop changes against a checkout of the source code from the GIT source code repository, and not a release tarball. This ensures that you're working with the latest code on the branch and makes it easier to generate Pull Requests correctly.

As of September 2022, the pgAdmin source repository can found at https://github.com/pgadmin-org/pgadmin4. A typical workflow for a relatively simple change might look as follows:

- 1. Visit the pgAdmin 4 project on GitHub, and click the *Fork* button to create your own development repository in your GitHub account.
- 2. Checkout a local copy of the source code with a command like:

```
$ git clone https://github.com/<your GitHub username>/pgadmin4.git
```

- 3. Develop and test your change in your local development environment.
- 4. Review your changes and check them thoroughly to ensure they meet the pgAdmin *Coding Standards*, and review them against the *Code Review Notes* to minimise the chances of them being rejected.

- 5. Once you're happy with your change, commit it with a suitable message. Include a one-line summary at the top, and if appropriate, further paragraphs following a blank line after the summary.
- 6. Push your changes to your fork of the repository.
- 7. Back in GitHub, create a new Pull Request against the pgadmin-org/pgadmin4 master branch.

**Note:** Each Pull Request should encompass a single bug fix or feature as a single commit. If necessary, you should squash multiple commits for larger changes or features into a single commit before submitting.

8. Your Pull Request will be reviewed by one or more members of the development team and either accepted or sent back with requested changes. In some cases it may be rejected if the change is not considered appropriate this is why it's important to discuss your work with the team before spending any significant time on it.

For more complex changes, you may wish to use a *Feature Branch* in your fork of the pgAdmin repository, and use that to create the Pull Request.

**Note:** This is a simple example of a workflow. You may choose to use other tools such as the GitHub CLI instead; documenting such tools and workflows is outside the scope of the pgAdmin documentation however.

### 12.2 Code Overview

The bulk of pgAdmin is a Python web application written using the Flask framework on the backend, and HTML5 with CSS3,ReactJS on the front end. A desktop runtime is also included for users that prefer a desktop application to a web application, which is written using NWjs (Node Webkit).

#### 12.2.1 **Runtime**

The runtime is based on NWjs which integrates a browser and the Python server creating a standalone application. The source code can be found in the **/runtime** directory in the source tree.

### 12.2.2 Web Application

The web application forms the bulk of pgAdmin and can be found in the /web directory in the source tree. The main file is **pgAdmin4.py** which can be used to run the built-in standalone web server, or as a WSGI application for production use.

#### Configuration

The core application configuration is found in **config.py**. This file includes all configurable settings for the application, along with descriptions of their use. It is essential that various settings are configured prior to deployment on a web server; these can be overridden in **config\_local.py** or **config\_system.py** (see the *config.py* documentation) to avoid modifying the main configuration file.

### **User Settings**

When running in desktop mode, pgAdmin has a single, default user account that is used for the desktop user. When running in server mode, there may be unlimited users who are required to login prior to using the application. pgAdmin utilised the **Flask-Security** module to manage application security and users, and provides options for self-service password reset and password changes etc.

Whether in desktop or server mode, each user's settings are stored in a SQLite OR external database which is also used to store the user accounts. This is initially created using the **setup.py** script which will create the database file and schema within it, and add the first user account (with administrative privileges) and a default server group for them. A **settings** table is also used to store user configuration settings in a key-value fashion. Although not required, setting keys (or names) are typically formatted using forward slashes to artificially namespace values, much like the pgAdmin 3 settings files on Linux or Mac.

Note that the local configuration must be setup prior to **setup.py** being run. The local configuration will determine how the script sets up the database, particularly with regard to desktop vs. server mode.

## 12.2.3 pgAdmin Core

The heart of pgAdmin is the **pgadmin** package. This contains the globally available HTML templates used by the Jinja engine, as well as any global static files such as images, Javascript and CSS files that are used in multiple modules.

The work of the package is handled in it's constructor, \_\_init\_\_.py. This is responsible for setting up logging and authentication, dynamically loading other modules, and a few other tasks.

### **12.2.4 Modules**

Units of functionality are added to pgAdmin through the addition of modules. Theses are Python object instance of classes, inherits the PgAdminModule class (a Flask Blueprint implementation), found in **web/pgadmin/utils.py**. It provide various hook points for other modules to utilise (primarily the default module - the browser).

To be recognised as a module, a Python package must be created. This must:

- 1) Be placed within the **web/pgadmin/** directory, and
- 2) Implements pgadmin.utils.PgAdminModule class
- 3) An instance variable (generally named **blueprint**) representing that particular class in that package.

Each module may define a **template** and **static** directory for the Blueprint that it implements. To avoid name collisions, templates should be stored under a directory within the specified template directory, named after the module itself. For example, the **browser** module stores it's templates in **web/pgadmin/browser/templates/browser/**. This does not apply to static files which may omit the second module name.

In addition to defining the Blueprint, the **views** module is typically responsible for defining all the views that will be rendered in response to client requests, we must provide a REST API url(s) for these views. These must include appropriate route and security decorators. Take a look at the NodeView class, which uses the same approach as Flask's MethodView, it can be found in **web/pgadmin/browser/utils.py**. This specific class is used by browser nodes for creating REST API url(s) for different operation on them. i.e. list, create, update, delete, fetch children, get statistics/reversed SQL/dependencies/dependents list for that node, etc. We can use the same class for other purpose too. You just need to inherit that class, and overload the member variables operations, parent\_ids, ids, node\_type, and then register it as node view with PgAdminModule instance.

Most pgAdmin modules will also implement the **hooks** provided by the PgAdminModule class. This is responsible for providing hook points to integrate the module into the rest of the application - for example, a hook might tell the caller what CSS files need to be included on the rendered page, or what menu options to include and what they should do.

12.2. Code Overview 489

Hook points need not exist if they are not required. It is the responsibility of the caller to ensure they are present before attempting to utilise them.

Hooks currently implemented are:

```
class MyModule(PgAdminModule):
    This is class implements the pgadmin.utils.PgAdminModule, and
    implements the hooks
   def get_own_stylesheets(self):
        Returns:
            list: the stylesheets used by this module, not including any
                  stylesheet needed by the submodules.
        return [url_for('static', 'css/mymodule.css')]
   def get_own_javascripts(self):
        Returns:
            list of dict:
            - contains the name (representation for this javascript
              module), path (url for it without .js suffix), deps (array of
              dependents), exports window object by the javascript module,
              and when (would you like to load this javascript), etc
              information for this module, not including any script needed
              by submodules.
        .....
       return [
            {
                'name': 'pgadmin.extension.mymodule',
                'path': url_for('static', filename='js/mymodule'),
                'exports': None,
                'when': 'server'
                }
            ]
   def get_own_menuitems(self):
        Returns:
            dict: the menuitems for this module, not including
                  any needed from the submodules.
        .....
        return {
            'help_items': [
                MenuItem(
                    name='mnu_mymodule_help',
                    priority=999,
                    # We need to create javascript, which registers itself
                    # as module
```

(continues on next page)

(continued from previous page)

pgAdmin Modules may include any additional Python modules that are required to fulfill their purpose, as required. They may also reference other dynamically loaded modules, but must use the defined hook points and fail gracefully in the event that a particular module is not present.

#### 12.2.5 Nodes

Nodes are very similar to modules, it represents an individual node or, collection object on the object explorer treeview. To recognised as a node module, a Python package (along with javascript modules) must be created. This must:

- 1) Be placed within the web/pgadmin/browser/ directory, and
- 2) Implements the BrowserPluginModule, and registers the node view, which exposes required the REST APIs
- 3) An instance of the class object

12.2. Code Overview 491

### 12.2.6 Front End

pgAdmin uses javascript extensively for the front-end implementation. It uses require.js to allow the lazy loading (or, say load only when required), ReactJS with CSS and MaterialUI for UI look and feel. We have divided each module in small chunks as much as possible. Not all javascript modules are required to be loaded (i.e. loading a javascript module for database will make sense only when a server node is loaded completely.) Please look at the javascript files node.js, browser.js, menu.js, panel.js, etc for better understanding of the code.

# 12.3 Coding Standards

pgAdmin uses multiple technologies and multiple languages, each of which have their own coding standards.

#### **12.3.1 General**

In all languages, indentations should be made with 4 spaces, and excessively long lines wrapped where appropriate to ensure they can be read on smaller displays (80 characters is used in many places, but this is not a required maximum size as it's quite wasteful on modern displays). Typically lines should not be longer than 120 characters.

Comments should be included in all code where required to explain its purpose or how it works if not obvious from a quick review of the code itself.

### 12.3.2 CSS 3

CSS3 is used for styling and layout throughout the application.

Most custom styling comes from individual modules which may advertise static stylesheets to be included in the module that is loading them via hooks.

Styling overrides (for example, to alter the look and feel) will typically be found in the **overrides.css** file in the main static file directory for the application.

Styling should never be applied inline in HTML, always through an external stylesheet, which should contain comments as appropriate to explain the usage or purpose for the style.

Styles should be specified clearly, one per line. For example:

```
/* iFrames should have no border */
iframe {
    border-width: 0;
}

/* Ensure the codemirror editor displays full height gutters when resized */
.CodeMirror, .CodeMirror-gutters {
    height: 100% !important;
}
```

All stylesheets must be CSS3 compliant.

### 12.3.3 HTML 5

HTML 5 is used for page structure throughout the application, in most cases being rendered from templates by the Jinja2 template engine in Flask.

All HTML must be HTML 5 compliant.

### 12.3.4 Javascript

Client-side code is written in Javascript using ReactJS and various plugins. Whilst much of the code is rendered from static files, there is also code that is rendered from templates using Jinja2 (often to inject the users settings) or constructed on the fly from module hooks.

A typical Javascript function might be formatted like this (this snipped is from a template):

```
// Delete a server group
function delete_server_group(item) {
    alertify.confirm(
        'Delete server group?',
        'Are you sure you wish to delete the server group "{0}"?'.replace('{0}', tree.

    getLabel(item)),
        function() {
            var id = tree.getId(item)
            $.post("{{ url_for('NODE-server-group.delete') }}", { id: id })
                 .done(function(data) {
                    if (data.success == 0) {
                         report_error(data.errormsg, data.info);
                    } else {
                        var next = tree.next(item);
                        var prev = tree.prev(item);
                        tree.remove(item);
                        if (next.length) {
                             tree.select(next);
                        } else if (prev.length) {
                             tree.select(prev);
                        }
                    }
                }
            )
        },
        null
    )
}
```

Note the use of a descriptive function name, using the underscore character to separate words in all lower case, and short but descriptive lower case variable names.

Note: From version 3.0 onwards, new or refactored code should be written using ES6 features and conventions.

### 12.3.5 Python

Python is used for the backend web server. All code must be compatible with Python 2.7 and should include PyDoc comments whilst following the official Python coding standards defined in PEP 8. An example function along with the required file header is shown below:

```
# pgAdmin 4 - PostgreSQL Tools
# Copyright (C) 2013 - 2024, The pgAdmin Development Team
# This software is released under the PostgreSQL Licence
"""Integration hooks for server groups."""
from flask import render_template, url_for
from flask.ext.security import current_user
from pgadmin.settings.settings_model import db, ServerGroup
def get_nodes():
   """Return a JSON document listing the server groups for the user"""
   groups = ServerGroup.query.filter_by(user_id=current_user.id)
   value = ''
   for group in groups:
      value += '{"id":%d,"label":"%s","icon":"icon-server-group","inode":true},' \
             % (group.id, group.name)
   value = value[:-1]
   return value
```

# 12.4 Code Snippets

This document contains code for some of the important classes, listed as below:

- PgAdminModule
- NodeView
- BaseDriver
- BaseConnection

### 12.4.1 PgAdminModule

PgAdminModule is inherited from Flask.Blueprint module. This module defines a set of methods, properties and attributes, that every module should implement.

```
class PgAdminModule(Blueprint):
   Base class for every PgAdmin Module.
   This class defines a set of method and attributes that
    every module should implement.
   def __init__(self, name, import_name, **kwargs):
       kwargs.setdefault('url_prefix', '/' + name)
        kwargs.setdefault('template_folder', 'templates')
        kwargs.setdefault('static_folder', 'static')
        self.submodules = []
        self.parentmodules = []
        super().__init__(name, import_name, **kwargs)
   def register_preferences(self):
        # To be implemented by child classes
       pass
   def register(self, app, options):
        Override the default register function to automagically register
        sub-modules at once.
        super().register(app, options)
       def create_module_preference():
            # Create preference for each module by default
            if hasattr(self, 'LABEL'):
                self.preference = Preferences(self.name, self.LABEL)
            else:
                self.preference = Preferences(self.name, None)
            self.register_preferences()
        # Create and register the module preference object and preferences for
        # it just before starting app
        app.register_before_app_start(create_module_preference)
        for module in self.submodules:
            module.parentmodules.append(self)
            if app.blueprints.get(module.name) is None:
                app.register_blueprint(module)
                app.register_logout_hook(module)
```

(continues on next page)

(continued from previous page)

```
def get_own_messages(self):
    Returns:
        dict: the i18n messages used by this module, not including any
            messages needed by the submodules.
    return dict()
def get_own_menuitems(self):
    Returns:
        dict: the menuitems for this module, not including
            any needed from the submodules.
   return defaultdict(list)
def get_exposed_url_endpoints(self):
    Returns:
        list: a list of url endpoints exposed to the client.
   return []
@property
def messages(self):
   res = self.get_own_messages()
    for module in self.submodules:
        res.update(module.messages)
    return res
@property
def menu_items(self):
   menu_items = self.get_own_menuitems()
    for module in self.submodules:
        for key, value in module.menu_items.items():
            menu_items[key].extend(value)
    menu_items = dict((key, sorted(value, key=attrgetter('priority')))
                      for key, value in menu_items.items())
   return menu_items
@property
def exposed_endpoints(self):
    res = self.get_exposed_url_endpoints()
    for module in self.submodules:
        res += module.exposed_endpoints
    return res
```

#### 12.4.2 NodeView

The NodeView class exposes basic REST APIs for different operations used by the pgAdmin Browser. The basic idea has been taken from Flask's MethodView class. Because we need a lot more operations (not, just CRUD), we can not use it directly.

```
class NodeView(View, metaclass=type(MethodView)):
   A PostgreSQL Object has so many operaions/functions apart from CRUD
   (Create, Read, Update, Delete):
   i.e.
   - Reversed Engineered SQL
   - Modified Query for parameter while editing object attributes
     i.e. ALTER TABLE ...
   - Statistics of the objects
   - List of dependents
   - List of dependencies
   - Listing of the children object types for the certain node
     It will used by the browser tree to get the children nodes
   This class can be inherited to achieve the diffrent routes for each of the
   object types/collections.
      OPERATION |
                              URL
                                             | HTTP Method |
                                                              Method
               | /obj/[Parent URL]/ | GET
                                                         | list
   List
   Properties | /obj/[Parent URL]/id
                                            | GET
                                                          | properties
                                                          | create
                                             | POST
   Create
                 | /obj/[Parent URL]/
                 //obj/[Parent URL]/id
                                            | DELETE
   Delete
                                                          | delete
                / /obj/[Parent URL]/id
                                             | PUT
   Update
                                                          | update
   SQL (Reversed | /sql/[Parent URL]/id
                                             | GET
                                                          | sql
   Engineering)
   SQL (Modified | /msql/[Parent URL]/id
                                             | GET
                                                           | modified_sql
   Properties)
   Statistics | /stats/[Parent URL]/id | GET
                                                          | statistics
   Dependencies | /dependency/[Parent URL]/id | GET
                                                          | dependencies
   Dependents | /dependent/[Parent URL]/id | GET
                                                          | dependents
   Nodes
                 | /nodes/[Parent URL]/
                                             | GET
                                                           | nodes
   Current Node | /nodes/[Parent URL]/id
                                             | GET
                                                           | node
   Children
                 | /children/[Parent URL]/id | GET
                                                           | children
   NOTE:
   Parent URL can be seen as the path to identify the particular node.
   In order to identify the TABLE object, we need server -> database -> schema
   information.
   operations = dict({
```

```
'obj': [
        {'get': 'properties', 'delete': 'delete', 'put': 'update'},
        {'get': 'list', 'post': 'create'}
    ],
    'nodes': [{'get': 'node'}, {'get': 'nodes'}],
    'sql': [{'get': 'sql'}],
    'msql': [{'get': 'modified_sql'}],
    'stats': [{'get': 'statistics'}],
    'dependency': [{'get': 'dependencies'}],
    'dependent': [{'get': 'dependents'}],
    'children': [{'get': 'children'}]
})
@classmethod
def generate_ops(cls):
    cmds = []
    for op in cls.operations:
        idx = 0
        for ops in cls.operations[op]:
            meths = []
            for meth in ops:
                meths.append(meth.upper())
            if len(meths) > 0:
                cmds.append({
                    'cmd': op, 'req': (idx == 0),
                    'with_id': (idx != 2), 'methods': meths
                })
            idx += 1
    return cmds
# Inherited class needs to modify these parameters
node_type = None
# Inherited class needs to modify these parameters
node_label = None
# This must be an array object with attributes (type and id)
parent_ids = []
# This must be an array object with attributes (type and id)
ids = \Pi
@classmethod
def get_node_urls(cls):
    assert cls.node_type is not None, \
        "Please set the node_type for this class ({0})".format(
            str(cls.__class__.__name__))
    common_url = '/'
    for p in cls.parent_ids:
        common_url += '<{0}:{1}>/'.format(str(p['type']), str(p['id']))
    id_url = None
    for p in cls.ids:
        id_url = '{0}<{1}:{2}>'.format(
            common_url if not id_url else id_url,
```

```
p['type'], p['id'])
    return id_url, common_url
def __init__(self, **kwargs):
    self.cmd = kwargs['cmd']
# Check the existance of all the required arguments from parent_ids
# and return combination of has parent arguments, and has id arguments
def check_args(self, **kwargs):
   has_id = has_args = True
    for p in self.parent_ids:
        if p['id'] not in kwargs:
            has_args = False
            break
    for p in self.ids:
        if p['id'] not in kwargs:
            has_id = False
            break
    return has_args, has_id and has_args
def dispatch_request(self, *args, **kwargs):
    http_method = flask.request.method.lower()
    if http_method == 'head':
        http_method = 'get'
    assert self.cmd in self.operations, \
        'Unimplemented command ({0}) for {1}'.format(
            self.cmd.
            str(self.__class__.__name__)
        )
   has_args, has_id = self.check_args(**kwargs)
    assert (
        self.cmd in self.operations and
        (has_id and len(self.operations[self.cmd]) > 0 and
         http_method in self.operations[self.cmd][0]) or
        (not has_id and len(self.operations[self.cmd]) > 1 and
         http_method in self.operations[self.cmd][1]) or
        (len(self.operations[self.cmd]) > 2 and
         http_method in self.operations[self.cmd][2])
    ), \
        'Unimplemented method (\{0\}) for command (\{1\}), which \{2\} '
        'an id'.format(http_method,
                       self.cmd,
                       'requires' if has_id else 'does not require')
   meth = None
    if has_id:
        meth = self.operations[self.cmd][0][http_method]
```

```
elif has_args and http_method in self.operations[self.cmd][1]:
        meth = self.operations[self.cmd][1][http_method]
    else:
        meth = self.operations[self.cmd][2][http_method]
   method = getattr(self, meth, None)
    if method is None:
        return make_json_response(
            status=406,
            success=0,
            errormsg=gettext(
                'Unimplemented method ({0}) for this url ({1})').format(
                    meth, flask.request.path
            )
        )
    return method(*args, **kwargs)
@classmethod
def register_node_view(cls, blueprint):
    cls.blueprint = blueprint
    id_url, url = cls.get_node_urls()
    commands = cls.generate_ops()
    for c in commands:
        cmd = c['cmd'].replace('.', '-')
        if c['with_id']:
            blueprint.add_url_rule(
                '/{0}{1}'.format(
                    c['cmd'], id_url if c['req'] else url
                ),
                view_func=cls.as_view(
                    '{0}{1}'.format(
                        cmd, '_id' if c['req'] else ''
                    ),
                    cmd=c['cmd']
                methods=c['methods']
            )
        else:
            blueprint.add_url_rule(
                '/{0}'.format(c['cmd']),
                view_func=cls.as_view(
                    cmd, cmd=c['cmd']
                methods=c['methods']
            )
def children(self, *args, **kwargs):
    """Build a list of treeview nodes from the child nodes."""
```

```
children = self.get_children_nodes(*args, **kwargs)
    # Return sorted nodes based on label
    return make_json_response(
        data=sorted(
            children, key=lambda c: c['label']
        )
    )
def get_children_nodes(self, *args, **kwargs):
    Returns the list of children nodes for the current nodes. Override this
    function for special cases only.
    :param args:
    :param kwargs: Parameters to generate the correct set of tree node.
    :return: List of the children nodes
    children = []
    for module in self.blueprint.submodules:
        children.extend(module.get_nodes(*args, **kwargs))
    return children
```

#### 12.4.3 BaseDriver

```
class BaseDriver(metaclass=DriverRegistry):
   class BaseDriver():
   This is a base class for different server types.
   Inherit this class to implement different type of database driver
   implementation.
   (For PostgreSQL/EDB Postgres Advanced Server, we will be using psycopg)
   Abstract Properties:
    _____
    * Version (string):
       Current version string for the database server
   * libpq_version (string):
       Current version string for the used libpq library
   Abstract Methods:
   * get_connection(*args, **kwargs)
   - It should return a Connection class object, which may/may not be
     connected to the database server.
```

```
* release_connection(*args, **kwargs)
- Implement the connection release logic
* gc()
- Implement this function to release the connections assigned in the
  session, which has not been pinged from more than the idle timeout
 configuration.
@property
@abstractmethod
def version(cls):
   pass
@property
@abstractmethod
def libpq_version(cls):
   pass
@abstractmethod
def get_connection(self, *args, **kwargs):
   pass
@abstractmethod
def release_connection(self, *args, **kwargs):
   pass
@abstractmethod
def gc_timeout(self):
   pass
```

#### 12.4.4 BaseConnection

implementation.

- \* execute\_scalar(query, params, formatted\_exception\_msg)
  - Implement this method to execute the given query and returns single datum result.
- \* execute\_async(query, params, formatted\_exception\_msg)
  - Implement this method to execute the given query asynchronously and returns result.
- \* execute\_void(query, params, formatted\_exception\_msg)
  - Implement this method to execute the given query with no result.
- \* execute\_2darray(query, params, formatted\_exception\_msg)
  - Implement this method to execute the given query and returns the result as a 2 dimensional array.
- \* execute\_dict(query, params, formatted\_exception\_msg)
  - Implement this method to execute the given query and returns the result as an array of dict (column name -> value) format.
- \* def async\_fetchmany\_2darray(records=-1, formatted\_exception\_msg=False):
- Implement this method to retrieve result of asynchronous connection and polling with no\_result flag set to True.
   This returns the result as a 2 dimensional array.
   If records is -1 then fetchmany will behave as fetchall.
- \* connected()
  - Implement this method to get the status of the connection. It should return True for connected, otherwise False
- \* reset()
  - Implement this method to reconnect the database server (if possible)
- \* transaction\_status()
  - Implement this method to get the transaction status for this connection. Range of return values different for each driver type.
- \* ping()
  - Implement this method to ping the server. There are times, a connection has been lost, but the connection driver does not know about it. This can be helpful to figure out the actual reason for query failure.
- \* \_release()
  - Implement this method to release the connection object. This should not be directly called using the connection object itself.
  - NOTE: Please use BaseDriver.release\_connection(...) for releasing the connection object for better memory management, and connection pool management.
- \* \_wait(conn)

```
- Implement this method to wait for asynchronous connection to finish the
    execution, hence - it must be a blocking call.
* _wait_timeout(conn, time)
  - Implement this method to wait for asynchronous connection with timeout.
    This must be a non blocking call.
* poll(formatted_exception_msg, no_result)
  - Implement this method to poll the data of query running on asynchronous
    connection.
* cancel_transaction(conn_id, did=None)
  - Implement this method to cancel the running transaction.
* messages()
  - Implement this method to return the list of the messages/notices from
    the database server.
* rows_affected()
  - Implement this method to get the rows affected by the last command
    executed on the server.
.....
ASYNC_OK = 1
ASYNC\_READ\_TIMEOUT = 2
ASYNC_WRITE_TIMEOUT = 3
ASYNC_NOT_CONNECTED = 4
ASYNC\_EXECUTION\_ABORTED = 5
ASYNC\_TIMEOUT = 0.2
ASYNC_WAIT_TIMEOUT = 2
ASYNC_NOTICE_MAXLENGTH = 100000
@abstractmethod
def connect(self, **kwargs):
    pass
@abstractmethod
def execute_scalar(self, query, params=None,
                   formatted_exception_msg=False):
   pass
@abstractmethod
def execute_async(self, query, params=None,
                  formatted_exception_msg=True):
   pass
@abstractmethod
def execute_void(self, query, params=None,
                 formatted_exception_msg=False):
    pass
@abstractmethod
```

```
def execute_2darray(self, query, params=None,
                    formatted_exception_msg=False):
    pass
@abstractmethod
def execute_dict(self, query, params=None,
                 formatted_exception_msg=False):
   pass
@abstractmethod
def async_fetchmany_2darray(self, records=-1,
                            formatted_exception_msg=False):
   pass
@abstractmethod
def connected(self):
   pass
@abstractmethod
def reset(self):
   pass
@abstractmethod
def transaction_status(self):
   pass
@abstractmethod
def ping(self):
   pass
@abstractmethod
def _release(self):
   pass
@abstractmethod
def _wait(self, conn):
   pass
@abstractmethod
def _wait_timeout(self, conn, time):
   pass
@abstractmethod
def poll(self, formatted_exception_msg=True, no_result=False):
@abstractmethod
def status_message(self):
   pass
@abstractmethod
def rows_affected(self):
```

```
pass
@abstractmethod
def cancel_transaction(self, conn_id, did=None):
    pass
```

#### 12.5 Code Review Notes

This document lists a number of standard items that will be checked during the review process for any changes submitted for inclusion in pgAdmin.

- Ensure all code follows the pgAdmin Coding Standards.
- Ensure all code has unit test coverage and API/feature test coverage where appropriate.
- Copyright years must be correct and properly formatted (to make it easy to make bulk updates every year). The start date should always be 2013, and the end year the current year, e.g.

```
Copyright (C) 2013 - 2024, The pgAdmin Development Team
```

- Ensure there's a blank line immediately following any copyright headers.
- Include PyDoc comments for functions, classes and modules. Node modules should be """Implements the XXXX node""".
- Ensure that any generated SQL does not have any leading or trailing blank lines and consistently uses 4 space indents for nice formatting.
- Don't special-case any Slony objects. pgAdmin 4 will have no direct knowledge of Slony, unlike pgAdmin 3.
- If you copy/paste modules, please ensure any comments are properly updated.
- Read all comments, and ensure they make sense and provide useful commentary on the code.
- Ensure that field labels both use PostgreSQL parlance, but also are descriptive. A good example is the "Init" field on an FTS Template Init is the PG term, but adding the word "Function" after it makes it much more descriptive.
- Re-use code whereever possible, but factor it out into a suitably central location don't copy and paste it unless modifications are required!
- Format code nicely to make it readable. Break up logical chunks of code with blank lines, and comment well to describe what different sections of code are for or pertain to.
- Ensure that form validation works correctly and is consistent with other dialogues in the way errors are displayed.
- On dialogues with Schema or Owner fields, pre-set the default values to the current schema/user as appropriate. In general, if there are common or sensible default values available, put them in the fields for the user.
- 1 Pull Request == 1 feature. If you need to fix/update existing infrastructure in your change, it's usually easier if it's in a separate Pull Request. Pull Requests containing multiple new features or unrelated changes are likely to be rejected.
- Ensure the change is fully functional, and works! If you wish to send a work in progress (WIP) change that is not intended for commit, instead of submitting a Pull Request, send either as a link to a repository fork or a patch to the pgadmin-hackers@postgresql.org mailing list, clearly stating that it's a WIP, and noting what does or does not yet work.

# 12.6 Translations

pgAdmin supports multiple languages using the Flask-Babel Python module. A list of supported languages is included in the **web/config.py** configuration file and must be updated whenever languages are added or removed with ISO 639-1 (two letter) language codes. The codes are named **\$LANG** in this document.

# 12.6.1 Translation Marking

Strings can be marked for translation in either Python code (using **gettext**()) or Jinja templates (using \_()). Here are some examples that show how this is achieved.

Python:

```
errormsg = gettext('No server group name was specified')
```

Jinja:

```
<input type="submit" value="{{ _('Change Password') }}">
```

```
<title>{{ _('%(appname)s Password Change', appname=config.APP_NAME) }}</title>
```

```
define(['sources/gettext', ...], function(gettext, ...){
    ...
    var alert = alertify.prompt(
        gettext('Password Change'),
        gettext('New password for %(userName)s', {userName: 'jsmith' }),
    ...
    )
})
```

# 12.6.2 Updating and Merging

Whenever new strings are added to the application, the template catalogue (web/pgadmin/messages.pot) and the existing translation catalogues (web/pgadmin/translations/\$LANG/LC\_MESSAGES/messages.po) must be updated and compiled. This can be achieved using the following commands from the web directory in the Python virtual environment for pgAdmin:

```
(pgadmin4) user$ pybabel extract -F babel.cfg -o pgadmin/messages.pot pgadmin
```

Once the template has been updated it needs to be merged into the existing message catalogues:

```
(pgadmin4) user$ pybabel update -i pgadmin/messages.pot -d pgadmin/translations
```

Finally, the message catalogues can be compiled for use:

```
(pgadmin4) user$ pybabel compile -d pgadmin/translations
```

12.6. Translations 507

# 12.6.3 Adding a New Language

Adding a new language is simple. First, add the language name and identifier to web/config.py:

```
# Languages we support in the UI
LANGUAGES = {
    'en': 'English',
    'zh': 'Chinese (Simplified)',
    'de': 'German',
    'pl': 'Polish'
}
```

Then, create the new message catalogue from the **web** directory in the source tree in the Python virtual environment for pgAdmin:

```
(pgadmin4) user$ pybabel init -i pgadmin/messages.pot -d pgadmin/translations -l $LANG
```

# CHAPTER 13

# Release Notes

pgAdmin release notes provide information on the features and improvements in each release. This page includes release notes for major releases and minor (bugfix) releases. Select your version from the list below to see the release notes for it.

# 13.1 Version 8.4

Release date: 2024-03-07

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v8.3.

# 13.1.1 Supported Database Servers

PostgreSQL: 12, 13, 14, 15, and 16

**EDB Advanced Server**: 12, 13, 14, 15, and 16

# 13.1.2 Bundled PostgreSQL Utilities

psql, pg\_dump, pg\_dumpall, pg\_restore: 16.1

#### 13.1.3 New features

Issue #6058 - Allow preferences customization using a configuration file.

Issue #7138 - Add support for JSON log format.

Issue #7204 - Add –yes option for skipping the confirmation prompt while deleting the user via CLI for scripting purpose.

# 13.1.4 Housekeeping

Issue #7097 - Upgrade CodeMirror from version 5 to 6.

Issue #7148 - Added documentation for Dashboard's System Statistics tab.

Issue #7187 - Separate the application name, branding & version information from the configuration file.

Issue #7234 - Upgrade python packages cryptography to 42.0.x and Flask-Security-Too to 5.3.x.

# **13.1.5 Bug fixes**

Issue #6792 - Fix multiple issues where PasswordExecCommand was not working in server mode and PasswordExecCommand was not loaded when importing servers.

Issue #6808 - Fix the tabbed panel backward/forward shortcut for tabs.

Issue #7027 - Fixed an issue where dependencies and dependents were not showing if a composite type is used as an attribute in another composite type.

Issue #7164 - Fix an issue where constraint check control is enabled in the edit table dialog.

Issue #7165 - Fix an issue where the scripts created by generate script of Schema diff for Table with sequence was not working earlier.

Issue #7193 - Ensure that the OAuth2 session is logged out when users log out from pgAdmin.

Issue #7217 - Remove role related checks on the UI dashboard when terminating session/query and let PostgreSQL take care of it.

Issue #7225 - Fix an issue where type column in dependents/dependencies tab is not showing correct label.

#### 13.2 Version 8.3

Release date: 2024-02-09

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v8.2.

#### 13.2.1 Supported Database Servers

**PostgreSQL**: 12, 13, 14, 15, and 16

**EDB Advanced Server**: 12, 13, 14, 15, and 16

# 13.2.2 Bundled PostgreSQL Utilities

psql, pg\_dump, pg\_dumpall, pg\_restore: 16.0

#### 13.2.3 New features

Issue #4419 - Allow drag-n-drop columns collection tree node as comma separated columns.

Issue #6380 - Added support to rename columns in Views.

Issue #6392 - Added BYPASSRLS|NOBYPASSRLS option while creating a Role.

Issue #6450 - Added support for column storage syntax while creating table.

Issue #6557 - Use COOKIE\_DEFAULT\_PATH or SCRIPT\_NAME in session cookie path.

Issue #6792 - Added configurable parameter to enable support for PasswordExecCommand in server mode.

# 13.2.4 Housekeeping

# **13.2.5 Bug fixes**

Issue #5083 - Fixed an issue where format sql was messing up operator. Included many other feature changes, more details here

Issue #6785 - Fixed an issue where formatting inserts empty lines in specific case.

Issue #7053 - Add support for selecting a schema in the backup database dialog with no tables, mviews, views or foreign tables.

Issue #7055 - Fixed a UI border issue on the dependencies tab for columns with icon.

Issue #7073 - Fixed an issue where multiple errors were showing if user does not have connect privileges.

Issue #7085 - Fixed an issue where group membership information was displayed incorrectly.

Issue #7113 - Ensure that the correct SQL is generated when changing the column data type to "char".

Issue #7153 - Fixed an issue pgAdmin fails to launch due to inconsistent PATH variable name on windows.

Issue #7166 - Fix a backend server error when creating a named restore point.

Issue #7175 - Fix wrong default for FORCE\_LOWER in the FLASK\_GRAVATAR module.

# 13.3 Version 8.2

Release date: 2024-01-11

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v8.1.

#### 13.3.1 Supported Database Servers

**PostgreSQL**: 12, 13, 14, 15, and 16

**EDB Advanced Server**: 12, 13, 14, 15, and 16

13.3. Version 8.2 511

# 13.3.2 Bundled PostgreSQL Utilities

psql, pg\_dump, pg\_dumpall, pg\_restore: 16.0

#### 13.3.3 New features

Issue #2483 - Administer pgAdmin Users and Preferences Using the Command Line Interface (CLI).

Issue #5908 - Allow users to convert View/Edit table into a Query tool to enable editing the SQL generated.

Issue #6085 - Added copy server support, allowing the duplication of existing servers with the option to make certain modifications.

Issue #7016 - Added keep-alive support for SSH sessions when connecting to a PostgreSQL server via an SSH tunnel.

# 13.3.4 Housekeeping

Issue #6926 - Ensure that eventlet's subprocess should be used following the resolution of an issue with Python 3.12 by eventlet.

# **13.3.5** Bug fixes

Issue #6193 - Fixed an issue where query tool title did not change after "Save As" until any new change is made.

Issue #6781 - Fixed an issue where export servers was not adding extension if not specified.

Issue #6815 - Fixed an issue where pgAdmin imports servers to the wrong accounts for the external authentication.

Issue #7002 - Fixed an issue where an error occurred in the SQL tab when using an extended index(pgroonga).

Issue #7041 - Fixed an issue where changes done to a node using edit dialog are not reflecting on the properties tab if the properties tab is active.

Issue #7059 - Fixed an issue where DB Restrictions were not visible on the server dialog.

Issue #7061 - Ensure that the 'Dbo' schema is displayed as a regular schema rather than a system catalog schema.

Issue #7062 - Introduce LDAP configuration parameter LDAP\_IGNORE\_MALFORMED\_SCHEMA to ignore fetching schema from the LDAP server.

Issue #7064 - Fixed an error-'amname' when generating ERD for database containing partition tables.

Issue #7066 - Fixed an issue where object explorer last tree state was not saving.

Issue #7070 - Fixed an issue where pgAgent job schedule dialog is not opening for edit.

Issue #7078 - Fixed an issue where user is not able to cancel or terminate active queries from dashboard.

Issue #7082 - Fixed browser autocomplete related issues on pgAdmin authentication related pages.

Issue #7091 - Fixed an issue where auto commit/rollback setting not persisting across query tool connection change.

Issue #7104 - Fixed an issue where Schema Diff not generating difference for missing columns.

# 13.4 Version 8.1

Release date: 2023-12-14

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v8.0.

# 13.4.1 Supported Database Servers

**PostgreSQL**: 12, 13, 14, 15, and 16

**EDB Advanced Server**: 12, 13, 14, 15, and 16

# 13.4.2 Bundled PostgreSQL Utilities

psql, pg\_dump, pg\_dumpall, pg\_restore: 16.0

#### 13.4.3 New features

Issue #4580 - Add support for generating ERD for a schema.

Issue #6854 - Add support for creating a function with custom return type.

# 13.4.4 Housekeeping

Issue #6991 - Fixed several accessibility-related issues for enhanced usability.

# **13.4.5 Bug fixes**

Issue #5471 - Ensure focus is not changed to ssh tunnel password input when user explicitly focus on server password input.

Issue #6095 - Provide a way to bypass the SSL cert verification for OAuth2 provider.

Issue #6488 - Fixed an issue where database name was missing in an error message if name contains any special characters.

Issue #6717 - Ensure that indexes created by constraints are visible in the object explorer when "Show system objects" is enabled.

Issue #6803 - Fixed an issue where reading process logs throws an error when DATA\_DIR is moved to a networked drive.

Issue #6814 - Remove the 'Close Window' submenu specifically for OSX to prevent unintended closure of the entire application.

Issue #6842 - Rename all references of 'Execute query' to 'Execute script' to be more relevant.

Issue #6887 - Fixed an issue where syntax error was not highlighting in query tool.

Issue #6921 - Fixed an issue where on entering full screen, the option label is not changed to 'Exit Full Screen' in desktop mode.

Issue #6950 - Ensure that the Authentication Source in the drop-down of the UserManagement dialog aligns with the entries specified for AUTHENTICATION\_SOURCES in the configuration file.

Issue #6958 - Reverse engineer serial columns when generating ERD for database/table.

Issue #6964 - Fixed an issue where the Schema was not visible in the dropdown for table properties or when creating a new table.

Issue #6968 - Fixed an issue where option key was not registering in PSQL tool.

13.4. Version 8.1 513

Issue #6984 - Fixed an issue where the Vacuum option INDEX\_CLEANUP have an incorrect value ('AUTO') for database versions < 14.

Issue #6989 - Fixed an issue where the pgAdmin page went blank when clicking the delete button in the User Management dialog.

Issue #7000 - Ensure that correct timezone is set for Docker deployments.

Issue #7011 - Fixed an issue where all rows and filter rows buttons of object explorer toolbar were disabled for views and other supported nodes.

Issue #7017 - Fixed an issue where schema diff tool is not loading preferences on start.

#### 13.5 Version 8.0

Release date: 2023-11-23

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v7.8.

# 13.5.1 Supported Database Servers

PostgreSQL: 12, 13, 14, 15, and 16

**EDB Advanced Server**: 12, 13, 14, 15, and 16

# 13.5.2 Bundled PostgreSQL Utilities

psql, pg\_dump, pg\_dumpall, pg\_restore: 16.0

#### 13.5.3 New features

Issue #2821 - Have close buttons on individual panel tabs instead of common.

Issue #4733 - Allow closing all the tabs, including SQL and Properties.

Issue #5394 - Changes in the context menu on panel tabs - Add close, close all and close others menu items.

#### 13.5.4 Housekeeping

Issue #6441 - Update app bundle built to use notarytool instead of altool.

Issue #6479 - Replace the current layout library wcDocker with ReactJS based rc-dock.

Issue #6850 - Upgrade Flask, Werkzeug and other modules depends on the latest version of Flask.

# **13.5.5 Bug fixes**

Issue #2986 - Fix an issue where the scroll position of panels was not remembered on Firefox.

Issue #5770 - Add DROP SQL for foreign keys in SQL generated by ERD when using WITH DROP option.

Issue #5807 - Fixed an issue where psql was not taking the role used to connect in server properties.

Issue #6017 - Fixed an issue where Geometry Viewer renders geometry incorrectly after trying to view 3D or non-4326 SRID geometry.

Issue #6459 - Fix the sorting of size on the statistics panel.

Issue #6487 - Fixed restoration of query tool database connection after dropping and re-creating the database with the same name.

Issue #6602 - Fix an issue where the default server-group is being deleted if the load-server json file contains no servers.

Issue #6720 - Fix an issue of the incorrect format (no indent) of SQL stored functions/procedures.

Issue #6769 - Server config information in the about dialog should be only visible to admin users.

Issue #6784 - Fixed an issue where Schema Diff does not work when the user language is set to any language other than English in Preferences.

Issue #6817 - Fixed the query generated when creating subscription where copy\_data parameter was missing.

Issue #6820 - Ensure backup/restore/maintenance works with invalid pgpass file parameter.

Issue #6835 - Fix an issue where OAUTH\_ADDITIONAL\_CLAIMS does not recognise AzureAD with > 150 groups.

Issue #6840 - Fixed circular import error occurring while deploying cloud PostgreSQL instance from pgAdmin.

Issue #6874 - Fix an issue where the browser window stuck on spinning with an Oauth user without email.

Issue #6875 - Fix an issue where import/export data is not working for shared servers.

Issue #6877 - Remove the max length of 255 from password exec command in server configuration dialog.

Issue #6884 - Remove gettext from empty strings in the title of the storage graph.

Issue #6902 - Fixed an issue where the change server password is not working in desktop mode.

Issue #6904 - Fix a crash issue occurring when debugging a function with arguments when using pgAdmin with external config database.

Issue #6920 - Fix an issue in ERD tool where SQL generated is missing columns for the table.

Issue #6934 - Clear the password field in the config database on clear saved server password.

Issue #6962 - Fixed an issue where the data type of the array type was not visible for the column in the Foreign Table dialog.

#### 13.6 Version 7.8

Release date: 2023-10-19

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v7.7.

13.6. Version 7.8 515

#### 13.6.1 Supported Database Servers

PostgreSQL: 12, 13, 14, 15, and 16

EDB Advanced Server: 12, 13, 14 and 15

# 13.6.2 Bundled PostgreSQL Utilities

psql, pg\_dump, pg\_dumpall, pg\_restore: 16.0

#### 13.6.3 New features

Issue #640 - Add support for foreign table's new functionality for PG 11 and above.

Issue #6229 - Allow setting custom username for shared servers, with default as username of server being shared.

Issue #6373 - Add 'GENERATED ALWAYS AS..' option while creating column constraints for Foreign Table

Issue #6797 - GUI representation of the system's activity using the 'system\_stats' extension.

Issue #6802 - Added 'load balance hosts' connection string parameter for PG 16 and above.

# 13.6.4 Housekeeping

Issue #6782 - Use PG16 as the default PostgreSQL version.

# **13.6.5 Bug fixes**

Issue #4995 - Fixed an issue in ERD tool where the downloaded images have a few links cut.

Issue #5749 - Fixed an issue where user was not able to assign new/old columns as primary key once column with primary key is deleted.

Issue #6285 - Add support for setting prepare threshold in server connection.

Issue #6482 - Fixed an issue where the wrong message "Current database has been moved or renamed" is displayed when debugging any function.

Issue #6538 - Fixed an issue where Processes tab displays wrong server name in some scenario.

Issue #6579 - Fix an issue where global/native keyboard shortcuts are not working when any cell of data output grid has focus.

Issue #6666 - Fixed query history slowness issue by storing query only for those having certain threshold max length.

Issue #6674 - Fix an issue where foreign table column name becomes "none" if the user changes any column data type.

Issue #6718 - Pin the cryptography version to fix PyO3 modules initialisation error.

Issue #6790 - Ensure that the backup works properly for PG 16 on the latest docker image.

Issue #6799 - Fixed an issue where the user is unable to select objects on the backup dialog due to tree flickering.

Issue #6836 - Fixed an issue where non-super PostgreSQL users are not able to terminate their own connections from dashboard.

Issue #6851 - Fix an issue where scale in columns is not allowed to have value as 0 or below.

Issue #6858 - Fix an issue in graphical explain plan where query tool crashes when the plan has parallel workers details and sort node is clicked for details.

Issue #6865 - Fix an issue where user login is not working if username/email contains single quote in server mode.

#### 13.7 Version 7.7

Release date: 2023-09-21

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v7.6.

# 13.7.1 Supported Database Servers

**PostgreSQL**: 11, 12, 13, 14 and 15

**EDB Advanced Server**: 11, 12, 13, 14 and 15

# 13.7.2 Bundled PostgreSQL Utilities

psql, pg\_dump, pg\_dumpall, pg\_restore: 15.4

#### 13.7.3 New features

Issue #642 - Added support to select/deselect objects in the Backup dialog.

Issue #4805 - Added all the new options of the 'WITH' clause in the subscription dialog.

Issue #6378 - Added USING method while creating the table.

Issue #6379 - Added compression method option while creating a column.

Issue #6383 - Added Strategy, Locale Provider, ICU Locale, ICU Rules, and OID options while creating a database.

Issue #6400 - Added USING method while creating the materialized view.

Issue #6736 - Add support for additional ID token claim checks for OAuth 2 authentication.

# 13.7.4 Housekeeping

Issue #2411 - Added the 'data type' column in the properties tab of the Columns collection node.

# **13.7.5 Bug fixes**

Issue #6274 - Fix an issue where user is not able to change the password when SMTP is not configured.

Issue #6704 - Ensure user is redirected to login page after failed login.

Issue #6712 - Ensure that Materialized view size fields in "Statistics" should be human-readable.

Issue #6730 - Fix an issue where changing the password shows success but the new password is not working.

Issue #6738 - Fix an issue where login form doesn't appear if internal auth source is removed.

Issue #6764 - Fix a security related issue where an authenticated user can run remote command using validate binary path API (CVE-2023-5002).

13.7. Version 7.7 517

#### 13.8 Version 7.6

Release date: 2023-08-24

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v7.5.

#### 13.8.1 Supported Database Servers

**PostgreSQL**: 11, 12, 13, 14 and 15

**EDB Advanced Server**: 11, 12, 13, 14 and 15

# 13.8.2 Bundled PostgreSQL Utilities

psql, pg\_dump, pg\_dumpall, pg\_restore: 15.3

#### 13.8.3 New features

Issue #2595 - Added Expression to CREATE INDEX.

Issue #3942 - Added cascade option while creating an extension.

Issue #5759 - Added 'Ignore Grants' option in the schema diff tool.

Issue #6004 - Added 'Ignore Tablespace' option in the schema diff tool.

Issue #6375 - Added support for ALTER INDEX column statistics.

Issue #6376 - Added unlogged option while creating a sequence.

Issue #6377 - Added all like options while creating a table.

Issue #6381 - Added support for SYSTEM, CONCURRENTLY and TABLESPACE options in

REINDEX.

Issue #6382 - Added WAL option to EXPLAIN ANALYZE command.

Issue #6397 - Added new/missing options to the VACUUM command.

Issue #6415 - Added SKIP\_LOCKED and BUFFER\_USAGE\_LIMIT option to Analyze command.

Issue #6448 - Add support for TRUNCATE trigger in foreign table.

Issue #6595 - Ensure that Schema Diff comparison results should be displayed in the sorted order.

# 13.8.4 Housekeeping

Issue #3702 - Generate software bill of materials as part of the package builds.

Issue #6588 - Added support for PostgreSQL and EPAS 16 to ensure it works without any errors.

# **13.8.5 Bug fixes**

Issue #5454 - Fix incorrect redirection URL after authentication by removing fixed value set to SCRIPT\_NAME environment variable in pgAdmin4.wsgi file.

Issue #6208 - Allow changing the POOL\_SIZE and MAX\_OVERFLOW config values of the pgAdmin config DB connection pool.

Issue #6252 - Fix an issue where query tool on shared server is throwing error if the pgAdmin config DB is external.

Issue #6420 - Fix the query tool issue where raise Notice from func/proc or code blocks are no longer displayed live.

Issue #6500 - Fix the issue where query tool window turns blank if the user tries to generate a graph on the result.

Issue #6624 - Fix an issue where changing MFA\_SUPPORTED\_METHODS breaks the MFA validation.

Issue #6630 - Fix an issue where pgAdmin 7.5 fails to render table SQL with extension loaded index method.

Issue #6639 - Fix an issue where cycle syntax was not added in SQL when creating new sequence from  $\Pi$ 

Issue #6651 - Fix an issue where the SET directive is excluded from the function header in the schema diff tool.

Issue #6660 - Fix a query tool error 'pgAdminThread' object has no attribute 'native\_id'.

Issue #6664 - Ensure keyboard shortcut for query execution is disabled when query execution is in progress.

# 13.9 Version 7.5

Release date: 2023-07-27

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v7.4.

# 13.9.1 Supported Database Servers

**PostgreSQL**: 11, 12, 13, 14 and 15

**EDB Advanced Server**: 11, 12, 13, 14 and 15

# 13.9.2 Bundled PostgreSQL Utilities

psql, pg\_dump, pg\_dumpall, pg\_restore: 15.3

#### 13.9.3 New features

Issue #6369 - Added support to detach partitions using concurrently and finalize.

Issue #6374 - Added all supported index storage parameters while creating an index.

Issue #6416 - Added new/missing parameters to pg\_dumpall (Backup Server).

Issue #6417 - Added new/missing parameters to pg\_dump (Backup Objects).

Issue #6562 - Added new/missing parameters to pg\_restore.

# 13.9.4 Housekeeping

Issue #6295 - Remove Bootstrap and jQuery from authentication pages and rewrite them in ReactJS.

Issue #6323 - Enable cluster deployment with gp3 volume for AWS & BigAnimal cloud providers.

Issue #6423 - Clarify the LICENSE file to indicate that it is the PostgreSQL Licence.

Issue #6532 - Remove unsupported PostgreSQL versions from the container.

13.9. Version 7.5 519

# 13.9.5 Bug fixes

Issue #6163 - Fix an issue where queries can't complete execution.

Issue #6165 - Fixed an issue where Import Export not working when using pgpassfile.

Issue #6317 - Fix an issue where queries longer than 1 minute get stuck - Container 7.1

Issue #6356 - Fix an issue where queries get stuck with auto-completion enabled.

Issue #6364 - Fixed Query Tool/ PSQL tool tab title not getting updated on database rename.

Issue #6406 - Ensure user gets proper error if incorrect credentials are entered while authenticating AWS.

Issue #6489 - Fix an issue where the edit server fails in desktop mode if the server password is not stored.

Issue #6499 - Ensure that Backup, Restore, and Maintenance should work properly when pgpass file is used.

Issue #6501 - Fix the query tool auto-complete issue on the server reconnection.

Issue #6502 - Fix the query tool restore connection issue.

Issue #6509 - Fix the reconnecton issue if the PostgreSQL server is restarted from the backend.

Issue #6514 - Fix the connection and stability issues since v7, possibly related to background schema changes.

Issue #6515 - Fixed an issue where the query tool is unable to execute a query on Postgres 10 and below versions.

Issue #6524 - Fix the lost connection error in v7.4.

Issue #6531 - Fixed an issue where pgAdmin failed to setup role with hyphens in name.

Issue #6537 - Fixed an issue where filters are not working and query history shows empty queries.

Issue #6544 - Fix an issue where adding a sub-folder inside a folder is not working as expected in File Manager.

Issue #6556 - Fix an error 'list' object has no attribute 'strip' while attempting to populate auto-complete manually the first time.

Issue #6558 - Fixed an issue where ERD Tool can't load the saved pgerd file from Shared Storage.

Issue #6582 - Fix an issue where inserting more than 10 rows does not work correctly in View Data; only parts end up in the table.

# 13.10 Version 7.4

Release date: 2023-06-29

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v7.3.

# 13.10.1 Supported Database Servers

**PostgreSQL**: 11, 12, 13, 14 and 15

**EDB Advanced Server**: 11, 12, 13, 14 and 15

# 13.10.2 Bundled PostgreSQL Utilities

psql, pg\_dump, pg\_dumpall, pg\_restore: 15.3

#### 13.10.3 New features

Issue #6367 - Added support to drop databases using the 'WITH (FORCE)' option.

Issue #6368 - Add "[ NULLS [ NOT ] DISTINCT ]" option while creating an Index.

Issue #6370 - Added 'OR REPLACE' clause while creating trigger.

Issue #6371 - Added security\_invoker option while creating a view.

# 13.10.4 Housekeeping

# 13.10.5 Bug fixes

Issue #5306 - Fix an issue where object explorer tree crashes occasionally.

Issue #6065 - Ensure that query tool shortcuts are working properly.

Issue #6258 - Add Password exec command and Expiration time to server export JSON and also allow them to import.

Issue #6266 - When opening pgAdmin the layout should be auto reset if it is corrupted. Reset layout menu should work if layout is corrupted while using pgAdmin.

Issue #6291 - Fix an issue where loading more rows indicator will not disappear if connection is lost.

Issue #6340 - Fix an encoding error when connecting through Pgbouncer.

Issue #6352 - Fix an issue where explain plan details is showing HTML escaped characters.

Issue #6354 - Fixed an issue where queries with temporary tables in the same transaction is not working.

Issue #6363 - Fixed an issue where preview images for themes were not loading.

Issue #6420 - Fix raise notice from func/proc or code blocks are no longer displayed live.

Issue #6431 - Fix an issue where PSQL is not working if the database name have quotes or double quotes.

Issue #6435 - Fix an issue where all the menus are enabled when pgAdmin is opened and no object is selected in the object explorer.

# 13.11 Version 7.3

Release date: 2023-06-06

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v7.2.

13.11. Version 7.3 521

#### 13.11.1 Supported Database Servers

**PostgreSQL**: 11, 12, 13, 14 and 15

**EDB Advanced Server**: 11, 12, 13, 14 and 15

# 13.11.2 Bundled PostgreSQL Utilities

psql, pg\_dump, pg\_dumpall, pg\_restore: 15.3

#### 13.11.3 New features

#### 13.11.4 Housekeeping

# 13.11.5 Bug fixes

Issue #6136 - Fix an issue where editing a database object de-selects it on the browser tree.

Issue #6341 - Ensure that SSH Tunnel should work properly after upgrading to 7.2 from 7.1

Issue #6342 - Fixed an issue where pgadmin is unable to take the defined role.

Issue #6345 - Fixed an issue where Foreign Key with 3 or more columns are shown in the wrong order in SQL and Properties

SQL and Properties.

Issue #6353 - Ensure that the master password dialog should not be visible if the parameter MASTER\_PASSWORD\_REQUIRED is set to False.

#### 13.12 Version 7.2

Release date: 2023-06-01

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v7.1.

#### 13.12.1 Supported Database Servers

PostgreSQL: 11, 12, 13, 14 and 15

**EDB Advanced Server**: 11, 12, 13, 14 and 15

#### 13.12.2 Bundled PostgreSQL Utilities

psql, pg\_dump, pg\_dumpall, pg\_restore: 15.2

#### 13.12.3 New features

Issue #3831 - Add Option to only show active connections on Dashboard.

Issue #4769 - Allow pgAdmin to retrive master password from external script/program.

Issue #5048 - Add an option to hide/show empty object collection nodes.

Issue #5123 - Added support to use standard OS secret store to save server/ssh tunnel passwords instead of master password in pgAdmin desktop mode.

Issue #5868 - Implement new PostgreSQL 15 features in publication dialog and SQL.

# 13.12.4 Housekeeping

# 13.12.5 Bug fixes

Issue #5789 - Fixed an issue where Foreign Key columns are shown in the wrong order in SQL and Properties.

Issue #5817 - Ensure that a new row should be added on top in the User Management dialog.

Issue #5926 - Fixed an issue where REVOKE ALL DDL in table SQL was added only for one role.

Issue #6003 - Indicate the user if all the server's children nodes are hidden from the preferences setting.

Issue #6026 - Tools menu should be toggled for "pause replay of wal" and "resume replay of wal".

Issue #6043 - Make the 'Connect to server' dialog a modal dialog.

Issue #6080 - pgAdmin icon not showing on taskbar on Windows 10.

Issue #6127 - Fixed an issue where properties were not visible for FTS Parsers, FTS Templates, MViews, and Rules in Catalog objects.

Issue #6147 - Heartbeat is getting logged, though no server is connected in pgAdmin.

Issue #6204 - Ensure that name can't be empty in edit mode for Primary Key and Index.

Issue #6221 - Fix circular reference error for the multirange data types in the query tool.

Issue #6247 - Fixed an issue where PSQL tool prompts for password if using password exec command.

Issue #6253 - Fix an issue in the register server when setting the role, an arbitrary SQL query can be fired.

Issue #6267 - Ensure the user is able to log in if the specified OAUTH2\_USERNAME\_CLAIM is present in the OAuth2 profile.

Issue #6278 - Use dependent instead of dependant in the message.

Issue #6279 - Fix incorrect number of foreign tables displayed. Show column comments in RE-SQL.

Issue #6280 - View SQL tab not quoting column comments.

Issue #6281 - VarChar Field Sizes are missing from Query's Grid header.

Issue #6331 - Separate multiple Blocking PIDs with delimiter on Dashboard.

# 13.13 Version 7.1

Release date: 2023-05-04

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v7.0.

13.13. Version 7.1 523

#### 13.13.1 Supported Database Servers

**PostgreSQL**: 11, 12, 13, 14 and 15

**EDB Advanced Server**: 11, 12, 13, 14 and 15

# 13.13.2 Bundled PostgreSQL Utilities

psql, pg\_dump, pg\_dumpall, pg\_restore: 15.2

#### 13.13.3 New features

Issue #2078 - Show object breadcrumbs path along with its comment on object hover.

Issue #3275 - Allow on demand record count setting to be changed per user using preferences.

Issue #3316 - Added support to show statistics for materialized views.

Issue #3318 - Added support to create an unnamed index.

# 13.13.4 Housekeeping

# 13.13.5 Bug fixes

Issue #5044 - Fixed an issue where query tool hangs for some time when multiple columns are selected to open geometry viewer.

Issue #5777 - Fixed an issue where the browser tree state is not remembered when reopening pgAdmin.

Issue #5820 - Fixed an issue where collation was set to none if we remove it while creating partitioned table.

Issue #5922 - Ensure that when pasting a row in query tool grid, default value is used for autogenerated/serial columns.

Issue #6059 - Show proper message if the debugger is stopped by the user.

Issue #6075 - Ensure that the save button is enabled when registering a new server fails due to an API error.

Issue #6077 - The search object, query tool, and psql tool menu should not be displayed for pgAgent.

Issue #6120 - Fixed error occurring while logging out from pgAdmin keeping a query tool opened.

Issue #6128 - Fix a SQL error occurring on roles dependents SQL.

Issue #6130 - Ensure to quote the primary key value if needed while deleting rows from the table.

Issue #6137 - Fixed error occurring while dumping the servers from CLI.

Issue #6138 - Throw an appropriate error when a table for which View/Edit data is open, is deleted, and query is executed.

Issue #6151 - Ensure that internal users are able to login when auth sources are [ldap, internal].

Issue #6158 - Fixed an issue with the properties tab not getting updated if the user updates the selected node.

Issue #6159 - Ensure that the ERD tool should work with the external database after moving to psycopg3.

Issue #6180 - Fixed an issue where PSQL is not working if the PGUSER env variable is set and service file is used for server connection details.

Issue #6183 - Mac crash when you press Command-Shift-C.

Issue #6189 - Fix an issue in View SQL when column level privileges are set with multiple roles.

Issue #6191 - Fixed an issue where Reset Password is not working after upgrading to latest flask-security-too.

Issue #6234 - Usernames/roles with special character are not double quoted when assigning privileges.

# 13.14 Version 7.0

Release date: 2023-04-13

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v6.21.

# 13.14.1 Supported Database Servers

PostgreSQL: 11, 12, 13, 14 and 15

**EDB Advanced Server**: 11, 12, 13, 14 and 15

# 13.14.2 Bundled PostgreSQL Utilities

psql, pg\_dump, pg\_dumpall, pg\_restore: 15.2

#### 13.14.3 New features

Issue #3298 - Auto expand row edit form when a new row is added for Primary Key, Foreign Key, Unique Constraint and Exclusion Constraint.

Issue #5014 - Added support for mounting shared storage in server mode.

Issue #5022 - Add a note on top of keyboard shortcuts preferences to show the Accesskey of the browser.

Issue #5750 - Added capability to deploy PostgreSQL servers on Google Cloud.

Issue #5805 - Added support of BigAnimal v3 API.

Issue #5854 - On pressing Ctrl+C on a tree object, copy the fully qualified name to clipboard.

Issue #5855 - Added option to create unique constraint with nulls not distinct.

# 13.14.4 Housekeeping

Issue #4734 - Rename the "Properties..." context menu option of the objects to "Edit Object..." and the "Browser" tree to "Object Explorer".

Issue #5011 - Upgrade from psycopg2 to psycopg3.

Issue #5701 - Remove Bootstrap and jQuery usage.

Issue #5830 - Add .ts and .tsx files under linter and fix linter issues.

Issue #5901 - Update SQLAlchemy, Flask, Flask-SQLAlchemy, and other packages to current versions.

13.14. Version 7.0 525

# 13.14.5 Bug fixes

Issue #4423 - Fixed an issue where list of tables is not displayed in a SQL\_ASCII database.

Issue #4784 - Handle errors occurring during decoding UTF-8 encoded query result data which contains ascii characters.

Issue #4884 - Fixed an issue where it is not possible to import csv data to tables having columns with german umlauts in their name.

Issue #4891 - Fixed 'rawunicodeescape' codec can't decode issue in a SQL\_ASCII database.

Issue #5504 - Fixed an issue where incorrect view of text[] fields in query and table results when use other then UTF8 (win1251) codepage and symbols.

Issue #5735 - Show appropriate error message when master password is not set instead of 'Crypt key missing'.

Issue #5775 - Display the 'No menu available for this object' message if the selected tree node does not have any options.

Issue #5824 - Ensure that the user's storage directory is created when the users are created, as well as for those users whose directories have not yet been created.

Issue #5833 - Fixed an issue where user MFA entry was not getting delete after deleting a user.

Issue #5834 - Fixed issue where pgAgent jobs were not getting dropped from properties tab.

Issue #5874 - Make "using" and "with check" fields a textarea in the RLS policy.

Issue #5894 - Use fetch instead of axios to close connections in SQLEditor, ERD, Schema Diff and Debugger to ensure it completes.

Issue #5904 - Fixed an issue where the count query should not be triggered when the estimated count is less than zero.

Issue #5907 - Validate user inputs provided in configuration files before starting pgadmin server.

Issue #5916 - Fix an issue in search objects where objects were unable to locate occasionally.

Issue #5919 - While restoring the database connections due to lost server connection, ensure that the databases which were previously connected are only reconnected.

Issue #5921 - Ensure that when pasting rows the rows are added right below the selected rows for copying.

Issue #5929 - Dashboard graph Y-axis width should increase with label.

Issue #5934 - Ensure that default values are set only for insert statement if user does not provide any values, in case of updating existing values to blank it should be set to null.

Issue #5941 - Fixed an issue where migration on external database is not working.

Issue #5943 - Use http for SVG namespace URLs which were changed to https for SonarQube fixes.

Issue #5952 - Ensure that the schema diff tool should not allow comparison between Postgres Server and EDB Postgres Advanced Server.

Issue #5953 - Fixed error while executing continue in debugging session after some time of debug execution.

Issue #5955 - Fix an issue where query tool is stuck when running query after discarding changed data.

Issue #5958 - Fix an issue where new dashboard graphs are partially following theme colors.

Issue #5959 - Fix an issue where Backup, Restore, and Maintenance not working if connection timeout is set in the server dialog.

Issue #6018 - Change the foreground color of the code mirror text for Dark Theme.

Issue #6093 - Fix the dependents SQL of Roles which is throwing a type casting error on PostgreSQL 15.

Issue #6100 - Fixed the LDAP authentication issue for the simultaneous login attempts.(CVE-2023-1907)

Issue #6109 - Fixed asyncio random task error messages in Query tool.

Issue #6122 - Fixed CSV export from Query Tool results does not include all columns for multiple CTEs.

# 13.15 Version 6.21

Release date: 2023-03-09

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v6.20.

#### 13.15.1 Supported Database Servers

**PostgreSQL**: 10, 11, 12, 13, 14 and 15

**EDB Advanced Server**: 10, 11, 12, 13, 14 and 15

# 13.15.2 Bundled PostgreSQL Utilities

psql, pg\_dump, pg\_dumpall, pg\_restore: 15.1

#### 13.15.3 New features

Issue #5832 - Allow changing cardinality notation in ERD to use Chen notation.

Issue #5842 - Add additional logging for successful logins and user creation.

# 13.15.4 Housekeeping

# 13.15.5 Bug fixes

Issue #5269 - Ensure that the schema diff tool should pick up the change in the column grants.

Issue #5685 - Ensure that Grant column permission to a view is visible in the SQL tab.

Issue #5747 - Ensure that content on the DDL comparison panel should get refreshed on selecting the object using the up and down arrow keys.

Issue #5756 - Fix for query tool prompting for unsaved changes even if no changes have been made.

Issue #5758 - Fixed an issue where lock layout menu was not in sync with preferences.

Issue #5760 - Fixed an issue where query was not pasted to editor after trojan source warning.

Issue #5764 - Fix an issue where the maintenance dialog for Materialized View gives an error.

Issue #5773 - Fixed an issue where Clear Saved Password should be disabled if the password is already cleared.

Issue #5790 - Fixed an issue where the user can't create trigger AFTER UPDATE OF.

Issue #5803 - Fix an issue where query tool is stripping spaces in grid cell.

Issue #5810 - Fix an issue where sequence owner is remove on sequence edit.

Issue #5822 - Do not allow to save invalid JSON in query tool JSON editor.

Issue #5847 - Fixed an issue where pgAdmin failed to connect when the Postgres password included special characters.

Issue #5870 - Ensure that the database migration does not fail with a NoSuchTableError exception.

Issue #5872 - Handle MERGE operation in query tool explain introduced in PostgreSQL 15.

Issue #5889 - Fixed an issue where the database server is not connected using a service file.

Issue #5909 - Fixed an issue where the file name in the query tool tab was not changing if another file is opened.

13.15. Version 6.21 527

### 13.16 Version 6.20

Release date: 2023-02-09

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v6.19.

# 13.16.1 Supported Database Servers

PostgreSQL: 10, 11, 12, 13, 14 and 15

**EDB Advanced Server**: 10, 11, 12, 13, 14 and 15

# 13.16.2 Bundled PostgreSQL Utilities

psql, pg\_dump, pg\_dumpall, pg\_restore: 15.1

#### 13.16.3 New features

Issue #4728 - Added support for setting PostgreSQL connection parameters.

# 13.16.4 Housekeeping

Issue #4320 - Added bundled PG utilities in the release note.

Issue #5525 - Upgrade Flask-Migrate to 4.x.

Issue #5723 - Improve performance by removing signal-based zoom-in, zoom-out, etc functionality from the runtime environment.

Issue #5794 - Use uplot for Dashboard graphs to reduce CPU usage.

# 13.16.5 Bug fixes

Issue #5532 - Fixed an issue where the client cert location was not stored on the shared servers.

Issue #5567 - Fix orphan database connections resulting in an inability to connect to databases.

Issue #5702 - Fix an issue where role is used as username for newly added servers when opening query tool.

Issue #5705 - Ensure that all parts of the application recommend and enforce the same length of passwords.

Issue #5732 - Fixed an issue where Kerberos authentication to the server is not imported/exported.

Issue #5733 - Ensure that the system columns should not visible in the import/export data.

Issue #5746 - Increase the length of the value column of the setting table.

Issue #5748 - Fixed console error while attaching the partition.

Issue #5751 - Fix failing import servers CLI due to vulnerability fix.

Issue #5761 - Fix an issue where drag and drop object names is not working.

Issue #5763 - Ensure that keyboard hotkey to open query tool and search object should work properly.

Issue #5781 - Fixed an issue where Query history is not getting loaded with external database.

Issue #5796 - Ensure nested menu items are shown in quick search result.

# 13.17 Version 6.19

Release date: 2023-01-17

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v6.18.

# 13.17.1 Supported Database Servers

PostgreSQL: 10, 11, 12, 13, 14 and 15

**EDB Advanced Server**: 10, 11, 12, 13, 14 and 15

#### 13.17.2 New features

Issue #5569 - Added support of AWS provider for BigAnimal cloud deployment.

# 13.17.3 Housekeeping

Issue #5563 - Allow YouTube video demo links to be added to appropriate pgAdmin documentation.

Issue #5615 - Rewrite pgAdmin main menu bar to use React.

# 13.17.4 Bug fixes

Issue #5487 - Fixed an issue where incorrect password used with shared server.

Issue #5541 - Ensure the browser tree does not freeze while rendering 10k+ nodes/objects.

Issue #5542 - Fixed an issue updating the schema node de-select the node in the tree if only one schema is present in the collection node.

Issue #5559 - Fixed tree flickering issue on scroll.

Issue #5577 - Fixed an issue where the default value of string for columns should wrap in quotes in the create script.

Issue #5586 - Fix the webserver and internal authentication setup issue.

Issue #5613 - Ensure the appbundle has correct permissions so that pgAdmin can be accessed by users other than owner.

Issue #5622 - Fixed an issue where the ignore owner flag is not working for some cases in the Schema Diff.

Issue #5626 - Fixed an issue where actions performed on the tree node should update the context menu options.

Issue #5627 - Ensure that the submenus under the trigger's context menu are enabled/disabled correctly.

Issue #5640 - Update boto3 & botocore to the latest version.

Issue #5641 - Fixed an issue where Geometry viewer does not show popup when columns are less than 3.

Issue #5647 - Fixed an issue where row count notification was disappearing automatically.

Issue #5661 - Fix select dropdown border issue.

Issue #5666 - Fixed a missing "jwks\_uri" in metadata error that occurred when logging in with an oAuth2 provider like Azure or Google.

Issue #5675 - Fixed an issue where rename panel was losing focus when trying to add name if input box is empty.

Issue #5734 - Ensure that the authenticated users can't access each other's directories and files by providing relative paths.(CVE-2023-0241)

13.17. Version 6.19 529

# 13.18 Version 6.18

Release date: 2022-12-19

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v6.17.

# 13.18.1 Supported Database Servers

PostgreSQL: 10, 11, 12, 13, 14 and 15

**EDB Advanced Server**: 10, 11, 12, 13, 14 and 15

#### 13.18.2 New features

Issue #4088 - Enhancements to the ERD when selecting a relationship.

Issue #5503 - Added native menu support in desktop mode.

# 13.18.3 Housekeeping

# 13.18.4 Bug fixes

Issue #5453 - Fixed an issue where Transaction IDs were not found in session in the Query Tool.

Issue #5470 - Fixed an issue where tablespace was missing on partition tables in SQL.

Issue #5536 - Fixed an issue where properties tab was refreshing on tab change even if the selected node is same.

Issue #5551 - Fixed an issue with auto-complete not working as expected with double quotes.

Issue #5564 - Ensure that table statistics are sorted by size.

Issue #5598 - Fixed an issue while updating server node info removes the clear saved password menu.

Issue #5603 - Fixed an issue where master password was not set correctly with external config database.

Issue #5606 - Fixed an error in the collation create script for PG-15.

Issue #5629 - Fixed BigAnimal authentication aborted issue.

Issue #5637 - Ensure that the BigAnimal displays PG version 11-14 for Oracle compatible databases type.

Issue #5645 - Fixed a typo.

# 13.19 Version 6.17

Release date: 2022-12-02

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v6.16.

# 13.19.1 Supported Database Servers

PostgreSQL: 10, 11, 12, 13, 14 and 15

**EDB Advanced Server**: 10, 11, 12, 13, 14 and 15

#### 13.19.2 New features

# 13.19.3 Housekeeping

Issue #5147 - Update the BigAnimal API version to V2.

Issue #5493 - Remove all traces of Backbone and Underscore.

# 13.19.4 Bug fixes

Issue #5488 - Fixed an issue where the wrong schema is displayed for a foreign key in the schema diff tool.

Issue #5495 - Ensure that the query history date format in Desktop mode matches the format of the locale of the pgadmin server.

Issue #5505 - Fixed an issue where the CSV file would not download if the CSV quote character length exceeded 1.

Issue #5513 - Ensure that DATA\_DIR dependent folders/files are automatically created inside the specified DATA\_DIR if they are not specified separately in the configuration file.

Issue #5539 - Improved error message to make it easier for users to understand.

Issue #5548 - Fixed an issue where editor position was wrong when editing data from result grid.

Issue #5575 - Ensure the query tool is launched successfully for the servers registered with the PostgreSQL service.

Issue #5593 - Ensure that only authorized and authenticated users can validate binary paths (CVE-2022-4223).

#### 13.20 Version 6.16

Release date: 2022-11-18

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v6.15.

# 13.20.1 Supported Database Servers

PostgreSQL: 10, 11, 12, 13, 14 and 15

**EDB Advanced Server**: 10, 11, 12, 13, 14 and 15

13.20. Version 6.16 531

#### 13.20.2 New features

- Issue #1832 Added support for storing configurations of pgAdmin in an external database.
- Issue #4756 Added the ability to generate ERDs for tables.
- Issue #5468 Add the possibility to configure the Oauth2 claim which is used for the pgAdmin username.

#### 13.20.3 Housekeeping

# 13.20.4 Bug fixes

Issue #2174 - Ensure that the browser tree should auto scroll to the selected node when expanding the server node.

Issue #4841 - Use SocketIO instead of REST for schema diff compare.

Issue #5066 - Ensure that users can use custom characters as CSV field separators/CSV quotes when downloading query results.

Issue #5058 - Ensure that the save button should be disabled by default on the Sort/Filter dialog in the query tool.

Issue #5098 - Fix an issue where the save button is enabled when the table properties dialog is opened.

Issue #5122 - Ensure that the spinner should be visible on the browser tree on node refresh.

Issue #5149 - Ensure the Generate ERD option is hidden if the connection to the database is not allowed.

Issue #5206 - Reposition the select dropdown when the browser is resized.

Issue #5281 - Ensure that autocomplete works properly with objects starting with double quotes.

Issue #5344 - Ensure that pgAdmin routes should have the SCRIPT\_NAME prefix.

Issue #5424 - Ensure that the appropriate permissions are set on the key file before trying an SSL connection with the server in server mode.

Issue #5429 - Fixed an issue where parameters for roles were not visible.

Issue #5452 - The container deployment document should include the server json file format.

Issue #5455 - Fixed an issue where the dependents tab wasn't working for PG 15.

Issue #5458 - Ensure that the browser path column in the search object shows the complete path.

Issue #5463 - Fixed an issue where the result grid was not working properly while trying to edit data by hitting Enter key.

Issue #5465 - Fixed an issue where the screen was freezing while closing the wcDocker panel.

Issue #5473 - Fixed an issue where AutoComplete was not working correctly due to incorrect regex.

Issue #5475 - Fixed an issue where the 'Confirm on close or refresh' setting was ignored when closing the query/ERD tool opened in the new tab.

Issue #5507 - Fixed an issue where pgadmin does not respect reverse proxy any more.

Issue #5521 - Fixed SocketIO calls when pgAdmin 4 server is running from a sub directory.

Issue #5522 - Ensure that the load file paths are children of the storage directory.

Issue #5533 - Use the shared server username when opening query tool.

Issue #5535 - Fixed an issue where the 'save\_password' column threw an error for the shared server when using an external database.

Issue #5537 - Ensure that the correct error message in ERD for permission denied should be shown.

# 13.21 Version 6.15

Release date: 2022-10-20

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v6.14.

### 13.21.1 Supported Database Servers

**PostgreSQL**: 10, 11, 12, 13, 14 and 15

**EDB Advanced Server**: 10, 11, 12, 13, 14 and 15

### 13.21.2 New features

Issue #3491 - Added support for IAM token based authentication for AWS RDS or Azure DB.

Issue #4392 - Added support to specify the background fill color to the table nodes in the ERD tool.

Issue #4994 - Allow reordering table columns using drag and drop in ERD Tool.

Issue #4997 - Add option to generate SQL with DROP table DDL in ERD Tool.

Issue #5304 - Added high availability options to AWS deployment.

Issue #5390 - Expose the Gunicorn limit\_request\_line parameter in the container, with the default set to the maximum 8190.

# 13.21.3 Housekeeping

Issue #5065 - Use SocketIO instead of REST for fetching database tables data in ERD.

Issue #5293 - Ensure that the tooltips are consistent throughout the entire application.

Issue #5357 - Remove Python's 'Six' package completely.

# 13.21.4 Bug fixes

Issue #5101 - Ensure consistent orderings for ACLS when comparing schemas in the schema diff.

Issue #5132 - Ensure that the result grid column should take width as pre preferences setting on first execution.

Issue #5133 - Fixed an exception occur while taking backup and SSL certificates/keys are not found in the specified path.

Issue #5145 - Fixed intermittent error shown while OAuth2 login.

Issue #5167 - Ensure that the path to the psqlrc file is correct when multiple users open the PSQL tool at the same time.

Issue #5188 - Ensure that the continue/start button should be disabled if the user stops the Debugger for the procedures.

Issue #5210 - Ensure that the query tool creates a new tab with the appropriate user when pressing Alt+Shift+Q.

Issue #5212 - Added the close button for all the notifications of the notistack.

Issue #5249 - Added the ability to display the selected text from the query tool in the find/replace box.

Issue #5261 - Ensure that the search filter should be cleared when a new row is added to the user management.

Issue #5262 - Ensure that the user management dialog should not allow the same email addresses with different letter casings when creating users.

13.21. Version 6.15 533

Issue #5277 - Fixed XSS vulnerability issues.

Issue #5296 - Ensure that the scroll position should be preserved for the result set in the query tool on tab change.

Issue #5308 - Ensure that the default value for a column should be used if it is made empty.

Issue #5327 - Fixed an issue where user was unable to select privileges in Safari.

Issue #5332 - Fixed console warning shown while updating database node from browser tree.

Issue #5338 - Fixed an issue where the prompt is not visible when clicking on the 'save results to file' button on the large data.

Issue #5352 - Fixed error occurring while LDAP authentication for a user with multiple email attributes.

Issue #5364 - Fixed an issue where notifications disappeared quickly.

Issue #5367 - Ensure that the correct value should be returned if an exception occurs while decoding the password.

Issue #5368 - Fixed the issue while downloading the file from the file manager.

Issue #5386 - Ensure that the login form is hidden if the authentication source is OAuth2 or Kerberos.

Issue #5397 - Fixed an issue where the password recovery link was not working.

Issue #5402 - Ensure that scroll bar on browser tree should be visible on windows resize.

Issue #5405 - Fixed the cross-site scripting vulnerability.

Issue #5427 - Fixed an issue where filtered rows were not working.

# 13.22 Version 6.14

Release date: 2022-09-22

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v6.13.

# 13.22.1 Supported Database Servers

PostgreSQL: 10, 11, 12, 13 and 14

**EDB Advanced Server**: 10, 11, 12, 13 and 14

#### 13.22.2 New features

# 13.22.3 Housekeeping

Issue #4059 - Port schema diff to React. (RM #6133)

Issue #4060 - Remove Backgrid and Backform. (RM #6134)

Issue #5035 - Port the remaining components of the ERD Tool to React. (RM #7343)

Issue #5120 - Updated keyboard shortcut documentation. (RM #7446)

Issue #5260 - Remove Alertify from pgAdmin completely. (RM #7619)

Issue #5263 - Port search object dialog to React. (RM #7622)

# 13.22.4 Bug fixes

Issue #5144 - Ensure that if BigAnimal authentication is aborted, API calls should be stopped. (RM #7472)

Issue #5209 - Fixed an issue where pgAdmin failed to start due to bin path migration. (RM #7557)

Issue #5230 - Fixed an issue where backup does not work due to parameter 'preexec\_fn' no longer being supported. (RM #7580)

Issue #5250 - Ensure that the browser tree should be refreshed after changing the ownership. (RM #7607)

Issue #5274 - Fixed the error message displayed when clicking the cloud server for which deployment is in progress. (RM #7636)

Issue #5275 - Fixed an issue where the wrong SQL displayed in difference if the user create an RLS policy on the table without a column. (RM #7637)

Issue #5282 - Ensure that the dump servers functionality works from setup.py. (RM #7644)

Issue #5284 - Ensure that the Import/Export server menu option is visible. (RM #7646)

Issue #5286 - Fixed API test case for change password in the server mode. (RM #7648)

Issue #5287 - Fixed an issue with the non-visibility of columns added prior to import/export data. (RM #7649)

Issue #5292 - Fixed an issue where textarea of the JSON Editor does not resize with dialog. (RM #7656)

Issue #5299 - Fixed ModuleNotFoundError when running setup.py to load/dump servers. (RM #7663)

Issue #5323 - Replace the language selection 'Brazilian' with 'Portuguese (Brazilian). (RM #7693)

Issue #5325 - Fixed an issue where server names with special characters are not displayed correctly in the process tab. (RM #7695)

Issue #5333 - Fixed an issue where ERD throws an error if variable is added to the column. (RM #7709)

Issue #5342 - Fixed an error while saving changes to the ERD table.

Issue #5343 - Fixes a redirect vulnerability when the user opens the pgAdmin URL.

### 13.23 Version 6.13

Release date: 2022-08-25

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v6.12.

### 13.23.1 Supported Database Servers

PostgreSQL: 10, 11, 12, 13 and 14

**EDB Advanced Server**: 10, 11, 12, 13 and 14

### 13.23.2 New features

Issue #3709 - Added support to show all background processes in separate panel.

Issue #7387 - Added support to create triggers from existing trigger functions in EPAS.

13.23. Version 6.13 535

## 13.23.3 Housekeeping

Issue #7344 - Port Role Reassign dialog to React.

Issue #7345 - Port User Management dialog to React.

Issue #7404 - Port process watcher to React.

Issue #7462 - Remove the SQL files for the unsupported versions of the database server.

Issue #7567 - Port About dialog to React.

Issue #7568 - Port change user password and 2FA dialog to React.

Issue #7590 - Port change ownership dialog to React.

Issue #7595 - Update the container base image to Alpine 3.16 (with Python 3.10.5).

Issue #7602 - Fixed improper parsing of HTTP requests in Pallets Werkzeug v2.1.0 and below (CVE-2022-29361).

# 13.23.4 Bug fixes

Issue #7452 - Ensure that an error is thrown if clipboard access is not provided and change the copy rows shortcut.

Issue #7468 - Fixed an issue where the History tab is getting blank and showing an error after some queries are executed.

Issue #7481 - Fixed an issue where OWNED BY was incorrectly set to NONE when adding user privileges on the sequence.

Issue #7497 - Fixed an issue with the error message being displayed at the right place for Azure deployments.

Issue #7521 - Fixed an issue where the Query Editor loses focus when saving a query (Alt+s).

Issue #7527 - Fixed API test cases for Postgres 14.4.

Issue #7540 - Ensure that rename panel should work on view/edit panels.

Issue #7563 - Fixed an issue where autocomplete is not working after clearing the query editor.

Issue #7573 - Ensure that autocomplete does not appear when navigating code using arrow keys.

Issue #7575 - Fixed an issue where Alt-Shift-Q didn't work after creating a new query.

Issue #7579 - Fixed an issue where copy and pasting a row in the results grid doesn't set the default for boolean.

Issue #7586 - Fixed an issue with rendering geometry when selecting a complete column.

Issue #7587 - Ensure that the children of information\_schema and pg\_catalog node should be displayed.

Issue #7591 - Fixed column "none" does not exist issue, while comparing schema objects.

Issue #7596 - Fixed an issue where schema diff did not pick up the change in RLS policy.

Issue #7608 - Fixed an issue where the cloud deployment wizard creates the cluster with the High Availability even if that option is not selected.

Issue #7611 - Ensure that schema diff maintains view ownership when view definitions are modified.

Issue #7614 - Fixed crypt key is missing issue when logout from the pgAdmin.

Issue #7616 - Ensure that the next button should be disabled if the password did not match for Azure deployment.

Issue #7617 - Fixed an issue where Azure cloud deployment failed.

Issue #7625 - Fixed Spanish translations typo.

Issue #7630 - Ensure that If the trigger function definition is changed, drop and recreate the trigger in the schema diff.

Issue #7632 - Fixed an issue where a user could not authenticate using Azure CLI on OSX.

Issue #7633 - Ensure that the autofocus is on the input control for the master password and server password dialogs.

Issue #7641 - Pin Flask-SocketIO <= v5.2.0. The latest version does not support Werkzeug in production environments

# 13.24 Version 6.12

Release date: 2022-07-28

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v6.11.

### 13.24.1 New features

Issue #4488 - Added option to trigger autocomplete on key press in the query tool.

Issue #4607 - Allow users to delete files/folders from the storage manager.

Issue #7389 - Allow users to search within the file/storage manager.

Issue #7486 - Added support for visualizing the graphs using Stacked Line, Bar, and Stacked Bar charts in the query tool.

Issue #7487 - Added support for visualise the graph using a Pie chart in the query tool.

# 13.24.2 Housekeeping

Issue #7313 - Port the file/storage manager to React.

Issue #7341 - Port change password dialog to React.

Issue #7342 - Port Master Password dialog to React.

Issue #7492 - Removing dynamic module loading and replacing it with static loading.

Issue #7546 - Port named restore point dialog to React.

# 13.24.3 Bug fixes

Issue #7428 - Preserve the settings set by the user in the import/export data dialog.

Issue #7471 - Ensure that the splash screen can be moved.

Issue #7508 - Fixed an issue where comments on indexes are not displayed.

Issue #7512 - Ensure that notices should not disappear from the messages tab.

Issue #7517 - Enable the start debugging button once execution is completed.

Issue #7518 - Ensure that dashboard graph API is not called after the panel has been closed.

Issue #7519 - Ensure that geometry should be shown for all the selected cells.

Issue #7520 - Fixed the JSON editor issue of hiding the first record.

Issue #7522 - Added support for Azure PostgreSQL deployment in server mode.

Issue #7523 - Fixed typo error for Statistics on the table header.

Issue #7524 - Fixed an issue where new folders cannot be created in the save dialog.

13.24. Version 6.12 537

# 13.25 Version 6.11

Release date: 2022-06-30

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v6.10.

#### 13.25.1 New features

Issue #2647 - Added mouse over indication for breakpoint area in the Debugger.

Issue #2648 - Added search text option to the Debugger panel.

Issue #7178 - Added capability to deploy PostgreSQL servers on Microsoft Azure.

Issue #7332 - Added support for passing password using Docker Secret to Docker images.

Issue #7351 - Added the option 'Show template databases?' to display template databases regardless of the setting of 'Show system objects?'.

Issue #7485 - Added support for visualise the graph using a Line chart in the query tool.

# 13.25.2 Housekeeping

Issue #6132 - Port Debugger to React.

Issue #7315 - Updates documentation for the Traefik v2 container deployment.

Issue #7411 - Update pgcli to latest release 3.4.1.

Issue #7469 - Upgrade Chartjs to the latest 3.8.0.

# 13.25.3 Bug fixes

Issue #7423 - Fixed an issue where there is no setting to turn off notifications in the Query Tool.

Issue #7440 - Fixed an issue where passwords entered in the 'Connect To Server' dialog were truncated.

Issue #7441 - Ensure that the Query Editor should be focused when switching between query tool tabs.

Issue #7443 - Fixed and issue where 'Use spaces' not working in the query tool.

Issue #7453 - Fixed an issue where the Database restriction is not working.

Issue #7460 - Fixed an issue where pgAdmin stuck while creating a new index.

Issue #7461 - Fixed an issue where the connection wasn't being closed when the user switched to a new connection and closed the query tool.

Issue #7468 - Skip the history records if the JSON info can't be parsed instead of showing 'No history'.

Issue #7502 - Fixed an issue where an error message is displayed when creating the new database.

Issue #7506 - Fixed permission denied error when deploying PostgreSQL in Azure using Docker.

### 13.26 Version 6.10

Release date: 2022-06-02

This release contains a number of bug fixes and new features since the release of pgAdmin 4 v6.9.

### 13.26.1 New features

Issue #7364 - Added the ability to resize columns on dashboard tables.

# 13.26.2 Housekeeping

Issue #7283 - PG 15 compatibility issues fixed.

Issue #7337 - Port connect server password dialog to React.

# 13.26.3 Bug fixes

Issue #6962 - Fixed the browser tree overlapping nodes and expansion issue.

Issue #7002 - Added the ability to detect and warn users about bidirectional Unicode characters.

Issue #7347 - Ensure that when Authentication buttons are disabled their text is visible in the Dark and High contrast theme.

Issue #7368 - Ensure that unwanted APIs should not be getting called for BigAnimal.

Issue #7372 - Tell Docker to always pull the latest base images when building containers.

Issue #7373 - Fixed an issue with geometry window zoom mouse scroll not working.

Issue #7374 - Fixed an issue when switching between connections in the Query Tool dropdown, the background and foreground colors should be changed.

Issue #7376 - Fixed an issue where a popup for unsaved changes appears when clicking on the open file button for a blank query editor.

Issue #7380 - Added support for multi-cell selection in the query tool grid.

Issue #7383 - Fixed an issue where Preferences are not saved when the dialog is maximized.

Issue #7388 - Fixed an issue where an error message fills the entire window if the query is long.

Issue #7393 - Ensure that the editor position should not get changed once it is opened.

Issue #7394 - Fixed an issue where geometry is not visible when a single cell is selected.

Issue #7399 - Added missing toggle case keyboard shortcuts to the query tool.

Issue #7402 - Ensure that Dashboard graphs should be refreshed on changing the node from the browser tree.

Issue #7403 - Fixed an issue where comments on domain constraints were not visible when selecting a domain node.

Issue #7405 - Ensure that null values are accepted for the numeric columns in react-data-grid.

Issue #7408 - Fixed an issue when a table has a column with an array type with length or precision, the column type is not selected while editing the table.

# 13.27 Version 6.9

Release date: 2022-05-12

This release contains a number of bug fixes and new features since the release of pgAdmin4 6.8.

13.27. Version 6.9 539

### 13.27.1 New features

Issue #3253 - Added status bar to the Query Tool.

Issue #3989 - Ensure that row numbers should be visible in view when scrolling horizontally.

Issue #6830 - Relocate GIS Viewer Button to the Left Side of the Results Table.

Issue #7179 - Added capability to deploy PostgreSQL servers on EDB BigAnimal.

Issue #7282 - Added options 'Ignore owner' and 'Ignore whitespace' to the schema diff panel.

Issue #7325 - Added support for Azure AD OAUTH2 authentication.

### 13.27.2 Housekeeping

Issue #6131 - Port query tool to React.

Issue #6746 - Improve the Kerberos Documentation.

Issue #7255 - Ensure the database and schema restriction controls are not shown as a drop-down.

Issue #7340 - Port data filter dialog to React.

# 13.27.3 Bug fixes

Issue #6725 - Fixed an issue where the Query tool opens on minimum size if the user opens multiple query tool Window quickly.

Issue #6958 - Only set permissions on the storage directory upon creation.

Issue #7026 - Fixed an issue where the Browser panel is not completely viewable.

Issue #7168 - Improvement to the Geometry Viewer popup to change the size of the result tables when column names are quite long.

Issue #7187 - Fixed an issue where the downloaded ERD diagram was 0 bytes.

Issue #7188 - Fixed an issue where the connection bar is not visible.

Issue #7231 - Don't strip binaries when packaging them in the server RPM as this might break cpython modules.

Issue #7252 - Ensure that Columns should always be visible in the import/export dialog.

Issue #7260 - Fixed an issue where an Empty message popup after running a query.

Issue #7262 - Ensure that Autocomplete should work after changing the connection.

Issue #7294 - Fixed an issue where the copy and paste row does not work if the first column contains no data

Issue #7296 - Ensure that after deleting multiple objects from the properties panel, the browser tree should be refreshed.

Issue #7299 - Fixed sorting issue in the statistics panel.

Issue #7305 - Fixed an issue where the Dashboard Server Activity was showing old queries as active.

Issue #7307 - Fixed an issue where the table showed duplicate columns when creating multiple sequences on the same column.

Issue #7308 - Ensure that sorting should be preserved on refresh for Server Activity.

Issue #7322 - Fixed an issue while creating a new database throwing an error that failed to retrieve data.

Issue #7333 - Fixed an issue where the drag and drop table in ERD throws an error.

Issue #7339 - Ensure that the Dashboard column sort order should be remembered when the refresh button is clicked.

# 13.28 Version 6.8

Release date: 2022-04-07

This release contains a number of bug fixes and new features since the release of pgAdmin4 6.7.

#### 13.28.1 New features

Issue #7215 - Added transaction start time to Server activity sessions view.

Issue #7249 - Added support for unique keys in ERD.

Issue #7257 - Support running the container under OpenShift with alternate UIDs.

# 13.28.2 Housekeeping

Issue #7132 - Port Properties panel for collection node, Dashboard, and SQL panel in React.

Issue #7149 - Port preferences dialog to React.

# 13.28.3 Bug fixes

Issue #4256 - Fixed an issue where SQL for revoke statements are not shown for databases.

Issue #5836 - Adds a new LDAP authentication configuration parameter that indicates the case sensitivity of the LDAP schema/server.

Issue #6960 - Ensure that the master password dialog is popped up if the crypt key is missing.

Issue #7059 - Fixed an issue where the error is shown on logout when the authentication source is oauth2.

Issue #7176 - Fixed an issue where the browser tree state was not preserved correctly.

Issue #7197 - Fixed an issue where foreign key relationships do not update when the primary key is modified.

Issue #7216 - Ensure that the values of certain fields are prettified in the statistics tab for collection nodes.

Issue #7221 - Ensure objects depending on extensions are not displayed in Schema Diff.

Issue #7238 - Fixed an issue where foreign key is not removed even if the referred table is removed in ERD.

Issue #7239 - Fixed an issue where the newly added table is not visible under the Tables node on refresh.

Issue #7261 - Correct typo in the documentation.

Issue #7263 - Fixed schema diff issue where function's difference DDL was showing incorrectly when arguments had default values with commas.

Issue #7264 - Ensure that the correct user should be selected in the new connection dialog.

Issue #7265 - Fixed schema diff issue in which the option 'null' doesn't appear in the DDL statement for the foreign table.

Issue #7267 - Fixed an issue where unexpected error messages are displayed when users change the language via preferences.

Issue #7269 - Ensure that pgAdmin4 should work with latest jinja2 version.

Issue #7275 - Fixed 'Cannot read properties of undefined' error while creating the table via the ERD tool.

13.28. Version 6.8 541

# 13.29 Version 6.7

Release date: 2022-03-14

This release contains a number of bug fixes and new features since the release of pgAdmin4 6.6.

#### **Note: Security Release**

Please note that this release includes a security update to fix an issue where a user could upload files to directories outside of their storage directory, when using pgAdmin running in server mode.

Users running pgAdmin in server mode, including the standard container based distribution, should upgrade to this release as soon as possible.

This issue does not affect users running in desktop mode.

# 13.29.1 Bug fixes

Issue #7220 - Fixed a schema diff issue where difference SQL isn't generated when foreign key values for a table differ.

Issue #7228 - Fixed a schema diff issue where string separator '\_\$PGADMIN\$\_' is visible for identical user mappings.

Issue #7230 - Fixed an issue where pgAdmin 4 took ~75 seconds to display the 'Starting pgAdmin' text on the splash screen.

Issue #7233 - Ensure that upload paths are children of the storage directory (CVE-2022-0959).

# 13.30 Version 6.6

Release date: 2022-03-10

This release contains a number of bug fixes and new features since the release of pgAdmin4 6.5.

#### 13.30.1 New features

Issue #7177 - Added capability to deploy PostgreSQL servers on Amazon RDS.

# 13.30.2 Housekeeping

Issue #7180 - Rename the menu 'Disconnect Database' to 'Disconnect from database'.

# 13.30.3 Bug fixes

- Issue #6956 Fixed a schema diff issue in which user mappings were not compared correctly.
- Issue #6991 Fixed an issue where pgadmin cannot connect to LDAP when STARTTLS is required before bind.
- Issue #6999 Fixed an issue where a warning is flashed every time for an email address when authentication sources are internal and ldap.
- Issue #7105 Fixed an issue where the parent partition table was not displayed during autocomplete.
- Issue #7124 Fixed the schema diff issue where tables have different column positions and a column has a default value
- Issue #7152 Added comments column for the functions collection node.
- Issue #7172 Allow users to scroll and enter input when there is a validation error.
- Issue #7173 Fixed an issue where the User Management dialog is not opening.
- Issue #7181 Ensure that the user should be able to add new server with unix socket connection.
- Issue #7186 Fixes an issue where the connect server/database menu was not updated correctly.
- Issue #7202 Ensure that Flask-Security-Too is using the latest version.

### 13.31 Version 6.5

Release date: 2022-02-11

This release contains a number of bug fixes and new features since the release of pgAdmin4 6.4.

#### 13.31.1 New features

Issue #7139 - Added support to open SQL help, Dialog help, and online help in an external web browser.

### 13.31.2 Housekeeping

- Issue #7016 Port Dependent, dependencies, statistics panel to React.
- Issue #7017 Port Import/Export dialog to React.
- Issue #7163 Rename the menu 'Disconnect Server' to 'Disconnect from server'.

### 13.31.3 Bug fixes

- Issue #6916 Added flag in runtime to disable GPU hardware acceleration.
- Issue #7035 Fixed an issue where connections keep open to (closed) connections on the initial connection to the database server.
- Issue #7085 Ensure that Partitioned tables should be visible correctly when creating multiple partition levels.
- Issue #7086 Correct documentation for 'Add named restore point'.
- Issue #7100 Fixed an issue where the Browser tree gets disappears when scrolling sequences.
- Issue #7109 Make the size blank for all the directories in the file select dialog.
- Issue #7110 Ensure that cursor should be focused on the first options of the Utility dialogs.
- Issue #7118 Ensure that JSON files should be downloaded properly from the storage manager.
- Issue #7123 Fixed an issue where restore generates incorrect options for the schema.
- Issue #7126 Fixed an issue where the F2 Function key removes browser panel contents.
- Issue #7127 Added validation for Hostname in the server dialog.

13.31. Version 6.5 543

- Issue #7135 Enforce the minimum Windows version that the installer will run on.
- Issue #7136 Fixed an issue where the query tool is displaying an incorrect label.
- Issue #7142 Fixed an issue where a warning message was shown after database creation/modification.
- Issue #7145 Ensure that owner should be ignored while comparing extensions.
- Issue #7146 Fixed event trigger comparing issue in Schema Diff tool.
- Issue #7150 Fixed an issue when uploading a CSV throwing an error in the Desktop mode.
- Issue #7151 Fixed value error in the restore dialog.
- Issue #7154 Ensure that the layout should not be reset if a query tool is opened and pgAdmin is restarted.

# 13.32 Version 6.4

Release date: 2022-01-13

This release contains a number of bug fixes and new features since the release of pgAdmin4 6.3.

## 13.32.1 New features

Issue #4803 - Added support to import/export server groups and servers from GUI.

# 13.32.2 Housekeeping

- Issue #7018 Port Restore dialog to React.
- Issue #7019 Port Maintenance dialog to React.

# 13.32.3 Bug fixes

- Issue #6745 Fixed an issue where Tablespace is created though an error is shown on the dialog.
- Issue #7003 Fixed an issue where Explain Analyze shows negative exclusive time.
- Issue #7034 Fixed an issue where Columns with default value not showing when adding a new row.
- Issue #7075 Ensure that help should be visible properly for Procedures.
- Issue #7077 Fixed an issue where the Owner is not displayed in the reverse engineering SQL for Procedures.
- Issue #7078 Fixed an issue where an operation error message pop up showing the database object's name incorrectly.
- Issue #7081 Fixed an issue in SQL generation for PostgreSQL-14 functions.
- Issue #7093 Fixed an issue where SubPlans may overlap other nodes & make them inaccessible in Graphical EXPLAIN View.
- Issue #7096 Ensure that Truncate and Reset statistics should work.
- Issue #7102 Fixed a schema diff issue where generated script adds unwanted line endings for

Functions/Procedures/Trigger Functions.

# 13.33 Version 6.3

Release date: 2021-12-16

This release contains a number of bug fixes and new features since the release of pgAdmin4 6.2.

#### 13.33.1 New features

Issue #6543 - Added support for Two-factor authentication for improving security.

Issue #6872 - Include GSSAPI support in the PostgreSQL libraries and utilities on macOS.

Issue #7039 - Added support to disable the auto-discovery of the database servers.

# 13.33.2 Housekeeping

Issue #6088 - Replace Flask-BabelEx with Flask-Babel.

Issue #6984 - Port Backup Global, Backup Server, and Backup object dialog in React.

Issue #7004 - Replaced alertifyjs notifiers with React-based notistack.

Issue #7010 - Upgrade Flask to version 2.

Issue #7053 - Replace Alertify alert and confirm with React-based model dialog.

# 13.33.3 Bug fixes

Issue #6840 - Fixed an issue where tooltip data are not displaying on downloaded graphical explain plan.

Issue #6877 - Fixed schema diff owner related issue.

Issue #6906 - Fixed an issue where referenced table drop-down should be disabled in foreign key -> columns after one row is added.

Issue #6955 - Ensure that sort order should be maintained when renaming a server group.

Issue #6957 - Fixed schema diff related some issues.

Issue #6963 - Ensure that the user should be allowed to set the schema of an extension while creating it.

Issue #6978 - Increase the width of the scrollbars.

Issue #6986 - Fixed an issue where the user can't debug function with timestamp parameter.

Issue #6989 - Fixed an issue where the Change Password menu option is missing for internal authentication source when more than one authentication source is defined.

Issue #7005 - Fixed an issue where On-demand rows throw an error when any row cell is edited and saved it then scroll to get more rows.

Issue #7006 - Ensure that Python 3.10 and the latest eventlet dependency should not break the application.

Issue #7013 - Fix an RPM build issue that could lead to a conflict with python3 at installation.

Issue #7015 - Fixed an issue where the error is thrown while creating a new server using Add New Server from the dashboard while tree item is not selected.

Issue #7020 - Ensure that statue message should not hide the last line of messages when running a long query.

Issue #7024 - Fixed an issue where reverse engineering SQL is wrong for Aggregate.

Issue #7029 - Correct the SQL definition for function/procedure with the Atomic keyword in PG14.

Issue #7031 - Fixed an issue where SQLite database definition is wrong because the USER\_ID FK references the table user\_old which is not available.

Issue #7040 - Add "section" to the Debian package control files.

Issue #7044 - Update the dropzone version to 5.9.3 and Flask-SQLAlchemy to 2.5.\*.

Issue #7046 - Fixed some accessibility issues.

13.33. Version 6.3 545

- Issue #7048 Fixed unhashable type issue while opening the about dialog.
- Issue #7064 Ensure that the Owner should not be disabled while creating the procedure.
- Issue #7071 Fixed an issue where confirmation pop-up is hidden behind Reassign/Drop Owned Dialog.

## 13.34 Version 6.2

Release date: 2021-11-18

This release contains a number of bug fixes and new features since the release of pgAdmin4 6.1.

#### 13.34.1 New features

Issue #3834 - Added support of Aggregate and Operator node in view-only mode.

Issue #6953 - Ensure that users should be able to modify the REMOTE\_USER environment variable as per their environment by introducing the new config parameter WEBSERVER\_REMOTE\_USER.

# 13.34.2 Housekeeping

# 13.34.3 Bug fixes

Issue #5427 - Fixed pgAdmin freezing issue by providing the error message for the operation that can't perform due to lock on the particular table.

Issue #6780 - Ensure that columns should be merged if the newly added column is present in the parent table.

Issue #6809 - Fixed an issue where pgAdmin is not opening properly.

Issue #6832 - Ensure that internal authentication when combined with other authentication providers, the order of internal source should not matter while picking up the provider.

Issue #6845 - Ensure that inherit table icon should be visible properly in the tree view.

Issue #6859 - Fixed an issue where properties panel is not updated when any object is added from the browser tree.

Issue #6896 - Ensure that the user should be able to navigate browser tree objects using arrow keys from keyboard.

Issue #6905 - Fixed an issue where database nodes are not getting loaded behind a reverse proxy with SSL.

Issue #6925 - Fixed SQL syntax error if select "Custom auto-vacuum" option and not set Autovacuum option to Yes or No.

Issue #6939 - Fixed an issue where older server group name displayed in the confirmation pop-up when the user removes server group.

Issue #6944 - Fixed an issue where JSON editor preview colours have inappropriate contrast in dark mode.

Issue #6945 - Fixed JSON Editor scrolling issue in code mode.

Issue #6940 - Fixed an issue where user details are not shown when the non-admin user tries to connect to the shared server.

Issue #6949 - Ensure that dialog should be opened when clicking on Reassign/Drop owned menu.

Issue #6954 - Ensure that changing themes should work on Windows when system high contrast mode is enabled.

Issue #6972 - Ensure that the Binary path for PG14 should be visible in the preferences.

Issue #6974 - Added operators and aggregates in search objects.

Issue #6976 - Fixed an issue where textarea should be allowed to resize and have more than 255 chars.

Issue #6981 - Fixed an issue where SQL for index shows the same column multiple times.

Issue #6988 - Reset the layout if pgAdmin4 detects the layout is in an inconsistent state.

# 13.35 Version 6.1

Release date: 2021-10-21

This release contains a number of bug fixes and new features since the release of pgAdmin4 6.0.

#### 13.35.1 New features

Issue #4596 - Added support for indent guides in the browser tree.

Issue #6081 - Added support for advanced table fields like the foreign key, primary key in the ERD tool.

Issue #6241 - Added support to allow tables to be dragged to ERD Tool.

Issue #6529 - Added index creation when generating SQL in the ERD tool.

Issue #6657 - Added support for authentication via the webserver (REMOTE\_USER).

Issue #6794 - Added support to enable/disable rules.

# 13.35.2 Housekeeping

# 13.35.3 Bug fixes

Issue #6719 - Fixed OAuth2 integration redirect issue.

Issue #6754 - Ensure that query highlighting color in the query tool should be less intensive.

Issue #6776 - Changed the label 'Inherits Tables?' to 'Is inherited?' as it misleading in the properties panel.

Issue #6790 - Fixed an issue where the user is unable to create an index with concurrently keyword.

Issue #6797 - Remove an extra blank line at the start of the SQL for function, procedure, and trigger function.

Issue #6802 - Fixed the issue of editing triggers for advanced servers.

Issue #6828 - Fixed an issue where the tree is not scrolling to the object selected from the search result.

Issue #6858 - Fixed object delete issue from the properties tab for the collection nodes.

Issue #6876 - Ensure that the Dashboard should get updated after connecting to the server.

Issue #6881 - Fixed an issue where the browser tree doesn't show all contents on changing resolution.

Issue #6882 - Ensure that columns should be displayed in the order of creation instead of alphabetical order in the browser tree.

Issue #6890 - Fixed background colour issue in the browser tree.

Issue #6891 - Added support for composite foreign keys in the ERD tool.

Issue #6900 - Fixed an issue where exclusion constraint cannot be created from table dialog if the access method name is changed once.

Issue #6905 - Fixed an issue where the users are unable to load the databases behind an HTTP reverse proxy.

Issue #6908 - Fixed an issue where each click to refresh the collection node, the number of objects decreasing by tens or more.

Issue #6912 - Fixed browser tree sort order regression issue.

Issue #6915 - Fixed an issue where the blank string is stored instead of NULL in the server table of SQLite database.

Issue #6928 - Ensure that the master password should be prompt when MASTER\_PASSWORD\_REQUIRED is set to True and AUTHENTICATION\_SOURCES is webserver.

Issue #6929 - Ensure that only the table node should be allowed to drop on the ERD tool.

Issue #6930 - Fixed an issue where the existing server group is disappeared on rename it.

Issue #6935 - Fixed an issue where the wrong SQL is generated when deleting and renaming table columns together.

13.35. Version 6.1 547

# 13.36 Version 6.0

Release date: 2021-10-07

This release contains a number of bug fixes and new features since the release of pgAdmin4 5.7.

#### 13.36.1 New features

Issue #4211 - Added support for OWNED BY Clause for sequences.

# 13.36.2 Housekeeping

Issue #5741 - Revisit all the CREATE and DROP DDL's to add appropriate 'IF EXISTS', 'CASCADE' and 'CREATE OR REPLACE'.

Issue #6129 - Port browser tree to React.

Issue #6588 - Port object nodes and properties dialogs to React.

Issue #6687 - Port Grant Wizard to react.

Issue #6692 - Remove GPDB support completely.

# 13.36.3 Bug fixes

Issue #2097 - Fixed an issue where grant wizard is unresponsive if the database size is huge.

Issue #2546 - Added support to create the Partitioned table using COLLATE and opclass.

Issue #3827 - Ensure that in the Query History tab, query details should be scrollable.

Issue #6712 - Fixed an issue where collapse and expand arrows mismatch in case of nested IF.

Issue #6713 - Fixed an issue where the last message is not visible in the Debugger.

Issue #6723 - Updated query error row selection color as per dark theme style guide.

Issue #6724 - Fixed an issue where the drop cascade button enables for Databases.

Issue #6736 - Fixed an issue where Refresh view options are not working for materialized view.

Issue #6755 - Fixed keyerror issue in schema diff for 'attnum' and 'edit\_types' parameter.

Issue #6759 - Ensure that RLS names should not be editable in the collection node of properties tab.

Issue #6798 - Fixed an issue where Execute button of the query tool gets disabled once we change anything in the data grid.

Issue #6834 - Ensure that SQL help should work for EPAS servers.

# 13.37 Version 5.7

Release date: 2021-09-09

This release contains a number of bug fixes and new features since the release of pgAdmin4 5.6.

### 13.37.1 New features

- Issue #2538 Added support for the truncate table with restart identity.
- Issue #4264 Make code folding case insensitive in the code mirror.
- Issue #4629 Added database and server information on the Maintenance process watcher dialog.
- Issue #6495 Allow the referenced table to be the same as the local table in one to many relationship for ERD Tool.
- Issue #6625 Make closing tabs to be smarter by focusing on the appropriate tab when the user closed a tab.
- Issue #6691 Set PSQLRC and PSQL\_HISTORY env vars to apt. user storage path in the server mode.

# 13.37.2 Housekeeping

# 13.37.3 Bug fixes

- Issue #4567 Fixed an issue where privileges were revoked using SQL query on objects like tables that do not correctly show in SQL tab.
- Issue #4815 Fixed an issue where the user can not paste the updated table header in safari 12 and 13 browsers.
- Issue #5849 Ensure that trigger function SQL should have 'create or replace function' instead of 'create function' only.
- Issue #6419 Fixed blank screen issue on windows and also made changes to use NWjs manifest for remembering window size.
- Issue #6531 Fixed the export image issue where relation lines are over the nodes.
- Issue #6544 Fixed width limitation issue in PSQL tool window.
- Issue #6564 Fixed an issue where columns with sequences get altered unnecessarily with a schema diff tool.
- Issue #6570 Ensure that the lock panel should not be blocked for larger records.
- Issue #6572 Partially fixes the data output panel display issue.
- Issue #6620 Fixed an issue where whitespace in function bodies was not applied while generating the script using Schema Diff.
- Issue #6627 Introduced OAUTH2\_SCOPE variable for the Oauth2 scope configuration.
- Issue #6641 Enables pgAdmin to retrieve user permissions in case of nested roles which helps to terminate the session for AWS RDS.
- Issue #6663 Fixed no attribute '\_asdict' error when connecting the database server.
- Issue #6668 Fixed errors related to HTML tags shown in the error message for JSON editor.
- Issue #6671 Fixed UnboundLocalError where local variable 'user\_id' referenced before assignment.
- Issue #6682 Renamed 'Auto rollback?' to 'Auto rollback on error?'.
- Issue #6684 Fixed the JSON editor issue of hiding the first record.
- Issue #6685 Ensure that deleting a database should not automatically connect to the next database.
- Issue #6704 Ensure that pgAdmin should not fail at login due to a special character in the hostname.
- Issue #6710 Fixed an issue where multiple query tool tabs getting closed for the single close event.

13.37. Version 5.7 549

# 13.38 Version 5.6

Release date: 2021-08-12

This release contains a number of bug fixes and new features since the release of pgAdmin4 5.5.

#### 13.38.1 New features

Issue #4904 - Added support to copy SQL from main window to query tool.

Issue #5198 - Added support for formatted JSON viewer/editor when interacting with data in a JSON column.

# 13.38.2 Housekeeping

Issue #6622 - Rename the "Resize by data?" to "Columns sized by" and disabled the 'Maximum column width' button if 'Columns sized by' is set to 'Column data'.

# 13.38.3 Bug fixes

Issue #6337 - Ensure that the login account should be locked after N number of attempts. N is configurable using the 'MAX\_LOGIN\_ATTEMPTS' parameter.

Issue #6369 - Fixed CSRF errors for stale sessions by increasing the session expiration time for desktop mode.

Issue #6448 - Fixed an issue in the search object when searching in 'all types' or 'subscription' if the user doesn't have access to the subscription.

Issue #6574 - Fixed an issue where paste is not working through Right-Click option on PSQL.

Issue #6580 - Fixed TypeError 'NoneType' object is not sub scriptable.

Issue #6586 - Fixed incorrect tablespace options in the drop-down for move objects dialog.

Issue #6618 - Fixed an issue where the titles in query tabs are different.

Issue #6619 - Fixed incorrect binary path issue when the user deletes the binary path from the preferences.

Issue #6643 - Ensure that all the required options should be loaded when the Range data type is selected while creating a custom data type.

Issue #6650 - Fixed dashboard server activity issue when active since parameter is None.

Issue #6664 - Fixed an issue where even if the user is locked, he can reset the password and can login into pgAdmin.

# 13.39 Version 5.5

Release date: 2021-07-15

This release contains a number of bug fixes and new features since the release of pgAdmin4 5.4.

### 13.39.1 New features

Issue #1975 - Highlighted long running queries on the dashboards.

Issue #3893 - Added support for Reassign/Drop Owned for login roles.

Issue #3920 - Do not block the query editor window when running a query.

Issue #5940 - Added support for OAuth 2 authentication.

Issue #6559 - Added option to provide maximum width of the column when 'Resize by data?' option in the preferences is set to True.

# 13.39.2 Housekeeping

# 13.39.3 Bug fixes

Issue #4189 - Ensure that the Data Output panel can be snapped back after it is detached.

Issue #6388 - Fixed replace keyboard shortcut issue in the query tool on the normal keyboard layout.

Issue #6398 - Fixed an issue where detaching the query editor panel gives a blank white panel.

Issue #6427 - Remove leading whitespace and replace it with '[...] ' in the Query Tool data grid so cells don't look empty.

Issue #6448 - Fixed an issue in the search object when searching in 'all types' or 'subscription' if the user doesn't have access to the subscription.

Issue #6489 - Fixed an issue where Execute/Refresh button should not be disabled when we run the empty query.

Issue #6505 - Fixed an issue where the New Connection Drop Down has lost default maintenance database, auto-select, and tab-through functionality.

Issue #6536 - Fixed directory selection issue with the folder dialog.

Issue #6541 - Ensure that setting 'Open in new browser tab' should be visible, it should not be based on the value of 'ENABLE\_PSQL'.

Issue #6547 - Fixed copy/paste issues for PSQL tool terminal.

Issue #6550 - Disable email deliverability check that was introduced in flask-security-too by default to maintain backwards compatibility.

Issue #6555 - Fixed Czech translation string for 'Login' keyword.

Issue #6557 - Fixed an issue where incorrect column name listed in the properties of Index.

### 13.40 Version 5.4

Release date: 2021-06-17

This release contains a number of bug fixes and new features since the release of pgAdmin4 5.3.

13.40. Version 5.4 551

### 13.40.1 New features

- Issue #1561 Added browse button to select the binary path in the Preferences.
- Issue #1591 Added Grant Wizard option under Package node.
- Issue #2341 Added support to launch PSQL for the connected database server.
- Issue #4064 Added window maximize/restore functionality for properties dialog.
- Issue #5370 Added support to set the binary path for the different database server versions.
- Issue #6231 Added OS, Browser, Configuration details in the About dialog.
- Issue #6395 Added support for rotating the pgAdmin log file on the basis of size and age.
- Issue #6524 Support non-admin installation on Windows.

# 13.40.2 Housekeeping

- Issue #4622 Added RESQL/MSQL test cases for Table and its child nodes.
- Issue #6225 Updated Flask-Security-Too to the latest v4.
- Issue #6460 Added a mechanism to detect a corrupt/broken config database file.

# 13.40.3 Bug fixes

- Issue #4203 Fixed the issue of renaming the database by another user.
- Issue #6404 Ensure that the Query Tool connection string should not be changed as per the 'Query Tool tab title'.
- Issue #6466 Ensure that the user should be able to add members in Login/Role group while creating it.
- Issue #6469 Ensure that the calendar control should be disabled in the properties panel for Role.
- Issue #6473 Disable browser password saving in the runtime.
- Issue #6478 Fixed duplicate SQL issue for tables with more than one partition.
- Issue #6482 Fixed an issue where the Foreground Color property of server dialog does not work.
- Issue #6513 Fixed an issue where pgAdmin does not open after password reset in server mode.
- Issue #6520 Fixed an issue where a decimal number is appended for character varying fields while downloading the data in CSV format.

# 13.41 Version 5.3

Release date: 2021-05-20

This release contains a number of bug fixes and new features since the release of pgAdmin4 5.2.

#### 13.41.1 New features

Issue #5954 - Added support to set auto width of columns by content size in the data output window.

Issue #6158 - Added support to connect PostgreSQL servers via Kerberos authentication.

Issue #6397 - Added "IF NOT EXISTS" clause while creating tables and partition tables which is convenient while using the ERD tool.

# 13.41.2 Housekeeping

# 13.41.3 Bug fixes

- Issue #4436 Fixed an issue where drag and drop object is not correct in codemirror for properties dialog.
- Issue #5477 Added support for cache bust webpack chunk files.
- Issue #5555 Fixed an issue where data is displayed in the wrong order when executing the query repeatedly.
- Issue #5776 Ensure that while connecting to the server using SSPI login, it should not prompt for the password.
- Issue #6329 Fixed an issue where the wrong SQL is showing for the child partition tables.
- Issue #6341 Fixed an issue where CSV download quotes the numeric columns.
- Issue #6355 Ensure that pgAdmin should not allow opening external files that are dragged into it.
- Issue #6377 Fixed an issue where schema diff does not create DROP DEFAULT statement for columns.
- Issue #6385 Ensure that Backup and Restore should work on shared servers.
- Issue #6392 Fixed an issue where the filter 'Include/Exclude By Selection' not working for null values.
- Issue #6399 Ensure that the user should not be able to add duplicate panels.
- Issue #6407 Added support for the creation of Nested Table and Varying Array Type for Advanced Server.
- Issue #6408 Fixed ModuleNotFoundError when running setup.py from outside of the root.
- Issue #6409 Fixed an issue where the current debug line is not visible in the 'Dark' theme.
- Issue #6413 Fixed an issue where duplicate columns are visible in the browser tree, which is owned by two sequences.
- Issue #6414 Fixed an issue where the Help message not displaying correctly on Login/Group role.
- Issue #6416 Added comment column in the properties panel for View and Materialized View collection node.
- Issue #6417 Fixed an issue where query editor is not being closed if the user clicks on the 'Don't Save' button.
- Issue #6420 Ensure that pgAdmin4 shut down completely on the Quit command.
- Issue #6443 Fixed an issue where file dialog showing incorrect files for the selected file types.
- Issue #6444 Fixed an issue where the user is not warned if Kerberos ticket expiration is less than 30 min while initiating a global backup.

Issue #6445 - Ensure that proper identification should be there when the server is connected using Kerberos or without Kerberos.

# 13.42 Version 5.2

Release date: 2021-04-22

This release contains a number of bug fixes and new features since the release of pgAdmin4 5.1.

#### 13.42.1 New features

### 13.42.2 Housekeeping

Issue #5319 - Improve code coverage and API test cases for Server module.

13.42. Version 5.2 553

# 13.42.3 Bug fixes

Issue #4001 - Updated docs and screenshots to cover the Notifications tab on the Query Tool.

Issue #5519 - Ensure that the query tool tab should be closed after server disconnection when auto-commit/auto-rollback is set to false.

Issue #5908 - Fixed an issue where shortcut keys are not working with manage macro.

Issue #6076 - Fixed an issue where correct error not thrown while importing servers and JSON file has incorrect/insufficient keys.

Issue #6082 - Ensure that the user should not be to change the connection when a long query is running.

Issue #6107 - Fixed flickering issue of the input box on check constraints.

Issue #6161 - Fixed an issue where the cursor shifts its focus to the wrong window for all the query tool related model dialogs.

Issue #6220 - Corrected the syntax for 'CREATE TRIGGER', use 'EXECUTE FUNCTION' instead of 'EXECUTE PROCEDURE' from v11 onwards.

Issue #6274 - Ensure that the strings in the LDAP auth module are translatable.

Issue #6293 - Fixed an issue where the procedure creation is failed when providing the Volatility option.

Issue #6306 - Fixed an issue while selecting the row which was deleted just before the selection operation.

Issue #6325 - Ensure that the file format for the storage manager should be 'All files' and for other dialogs, it should remember the last selected format.

Issue #6327 - Ensure that while comparing domains check function dependencies should be considered in schema diff.

Issue #6333 - Fixed sizing issue of help dialog for Query Tool and ERD Tool when open in the new browser tab.

Issue #6334 - Fixed SQL panel black screen issue when detaching it in runtime.

Issue #6338 - Added missing dependency 'xdg-utils' for the desktop packages in RPM and Debian.

Issue #6344 - Fixed cannot unpack non-iterable response object error when selecting any partition.

Issue #6356 - Mark the Apache HTTPD config file as such in the web DEB and RPM packages.

Issue #6367 - Fixed an issue where the Save button is enabled by default when open the table's properties dialog on PG 9.5.

Issue #6375 - Fixed an issue where users are unable to see data of the partitions using the View/Edit data option.

Issue #6376 - Fixed an issue where a connection warning should be displayed on the user clicks on explain or explain analyze and the database server is disconnected from the browser tree.

Issue #6379 - Fixed an issue where foreign data wrapper properties are not visible if the host option contains two host addresses.

### 13.43 Version 5.1

Release date: 2021-03-25

This release contains a number of bug fixes and new features since the release of pgAdmin4 5.0.

### 13.43.1 New features

- Issue #5404 Show the login roles that are members of a group role be shown when examining a group role.
- Issue #6212 Make the container distribution a multi-arch build with x86\_64 and Arm64 support.
- Issue #6268 Make 'kerberos' an optional feature in the Python wheel, to avoid the need to install MIT Kerberos on the system by default.
- Issue #6270 Added '-replace' option in Import server to replace the list of servers with the newly imported one.
- Issue #6271 Added zoom scaling options with keyboard shortcuts in runtime.

# 13.43.2 Housekeeping

- Issue #3976 Use schema qualification while accessing the catalog objects.
- Issue #6176 Make the 'Save Data Changes' icon to be more intuitive.

# 13.43.3 Bug fixes

- Issue #4014 Fixed alignment issue under preferences for the German language.
- Issue #4020 Fixed color issue on the statistics tab for collection node in the safari browser.
- Issue #4438 Fixed an issue where adding/updating records fails if the table name contains percent sign.
- Issue #4784 Ensure that autovacuum and analyze scale factors should be editable with more than two decimals.
- Issue #4847 Fixed an issue where % displayed twice in explain analyze for query and table.
- Issue #4849 Rename text 'table' with 'relation' in the statistic tab for explain analyze.
- Issue #4959 Fixed an issue where the properties tab for collection nodes is unresponsive after switching the tabs.
- Issue #5073 Fixed an issue where the Save button is enabled for functions/procedures by default when open the properties dialog.
- Issue #5119 Fixed an issue where hanging symlinks in a directory cause select file dialog to break.
- Issue #5467 Allow underscores in the Windows installation path.
- Issue #5628 Remove the "launch now" option in the Windows installer, as UAC could cause it to run as an elevated user.
- Issue #5810 Ensure that cell content being auto selected when editing the cell data.
- Issue #5869 Ensure that SQL formatter should not add extra tabs and format the SQL correctly.
- Issue #6018 Fixed encoding issue when database encoding set to SQL\_ASCII and name of the column is in ASCII character.
- Issue #6159 Ensure that the user should be able to kill the session from Dashboard if the user has a 'pg\_signal\_backend' role.
- Issue #6206 Ensure that the view/edit data panel should not be opened for unsupported nodes using the keyboard shortcut.
- Issue #6227 Ensure PGADMIN\_DEFAULT\_EMAIL looks sane when initialising a container deployment.
- Issue #6228 Improve the web setup script for Linux to make the platform detection more robust and overrideable.
- Issue #6233 Ensure that SQL formatter should not use tab size if 'Use spaces?' set to false.
- Issue #6253 Fixed an issue where the user is unable to create a subscription if the host/IP address for connection is 127.0.0.1.
- Issue #6259 Ensure that proper error message should be shown on the properties and statistics tab in case of insufficient privileges for a subscription.
- Issue #6260 Fixed an issue where the 'Create Slot' option is disabled in case of the same IP/host provided but the port is different.
- Issue #6269 Ensure the Python interpreter used by the runtime ignores user site-packages.

13.43. Version 5.1 555

- Issue #6272 Fixed an issue where the user is not able to change the connection in Query Tool when any SQL file is opened.
- Issue #6279 Ensure that the venv activation scripts have the correct path in them on Linux.
- Issue #6281 Fixed an issue where schema diff showing wrong SQL when comparing triggers with different when clause.
- Issue #6286 Ensure that the template database should be visible while creating the database.
- Issue #6292 Fixed string index out of range error where the dependent tab is in focus and selecting any publication or table.
- Issue #6294 Fixed an issue where the dependent tab throwing an error when selecting any login/group role.
- Issue #6307 Fixed an issue where the incorrect values visible in the dependents tab for publication.
- Issue #6312 Fixed an issue where copy/paste rows in view data paste the wrong value for boolean type.
- Issue #6316 Ensure that the primary key should be visible properly in the table dialog.
- Issue #6317 Ensure that toggle buttons are accessible by most screen readers.
- Issue #6322 Fixed an issue where the top menu disappears when entering into the full screen for minimum screen resolution.
- Issue #6323 Ensure that the grantor name should be visible properly for the security tab in the table dialog.

# 13.44 Version 5.0

Release date: 2021-02-25

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.30.

#### 13.44.1 New features

- Issue #5091 Make Statistics, Dependencies, Dependants tabs closable and the user can add them back using the 'Add panel' option.
- Issue #5912 Added support for Logical Replication.
- Issue #5967 Implemented runtime using NWjs to open pgAdmin4 in a standalone window instead of the system tray and web browser.
- Issue #6148 Added Quick Search functionality for menu items and help articles.
- Issue #6153 Added publication and subscription support in Schema Diff.
- Issue #6170 Ensure logs are not stored in the container, and only sent to the console.

### 13.44.2 Housekeeping

- Issue #5017 Use cheroot as the default production server for pgAdmin4.
- Issue #6145 Documentation of Logical Replication.
- Issue #6195 Documentation of runtime with NWjs.
- Issue #6196 Documentation of Quick Search support.
- Issue #6207 Updated the JS dependencies to the latest.

# 13.44.3 Bug fixes

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 #6045 - Fixed autocomplete issue where it is not showing any suggestions if the schema name contains escape characters.

Issue #6087 - Fixed an issue where the dependencies tab showing multiple owners for the objects having shared dependencies.

Issue #6117 - Fixed an issue where the user is unable to update column-level privileges from the security tab.

Issue #6143 - Fixed an issue where shared server entries not getting deleted from SQLite database if the user gets deleted.

Issue #6157 - Fixed an issue where strike-through is not visible for rows selected for deletion after scrolling.

Issue #6163 - Fixed an issue where Zoom to fit button only works if the diagram is larger than the canvas.

Issue #6164 - Ensure that the diagram should not vanish entirely if zooming out too far in ERD.

Issue #6177 - Fixed an issue while downloading ERD images in Safari and Firefox.

Issue #6178 - Fixed an issue where the user unable to change the background color for a server.

Issue #6179 - Fixed an issue where Generate SQL displayed twice in the ERD tool.

Issue #6180 - Updated missing documentation for the 'Download Image' option in ERD.

Issue #6187 - Limit the upgrade check to run once per day.

Issue #6193 - Ensure that ERD throws a warning before closing unsaved changes if open in a new tab.

Issue #6197 - Fixed an issue where the ERD image is not properly downloaded.

Issue #6201 - Added SSL support for creating a subscription.

Issue #6208 - Fixed an issue where utility(Backup, Maintenance, ...) jobs are failing when the log level is set to DEBUG.

Issue #6230 - Fixed an issue where the user is not able to create the subscription.

Issue #6250 - Ensure DEB/RPM packages depend on the same version of each other.

### 13.45 Version 4.30

Release date: 2021-01-28

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.29.

#### 13.45.1 New features

Issue #1802 - Added ERD Diagram support with basic table fields, primary key, foreign key, and DDL SQL generation.

Issue #5457 - Added support for Kerberos authentication, using SPNEGO to forward the Kerberos tickets through a browser.

Issue #6147 - Documentation of Kerberos support.

Issue #6152 - Documentation of ERD Diagram support.

Issue #6160 - Add a container option (PGADMIN DISABLE POSTFIX) to disable the Postfix server.

13.45. Version 4.30 557

# 13.45.2 Housekeeping

- Issue #5338 Improve code coverage and API test cases for pgAgent.
- Issue #6052 Added connected pgAdmin user and connection name in the log file.
- Issue #6079 Updated mimetype from 'text/javascript' to 'application/javascript' as 'text/javascript' is obsolete.
- Issue #6162 Include PostgreSQL 13 utilities in the container.

# 13.45.3 Bug fixes

- 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 #5829 Fixed incorrect log information for AUTHENTICATION\_SOURCES.
- Issue #5875 Ensure that the 'template1' database should not be visible after pg\_upgrade.
- 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 SOL.
- 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 #5997 Updated Flask-BabelEx to the latest.
- 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 #6075 Fixed an issue where Non-admin user is unable to view shared server created using service.
- Issue #6077 Fixed accessibility issues in various dialogs.
- Issue #6084 Fixed TypeError exception in schema diff when selected any identical object.
- Issue #6096 Updated deployment documentation, refer correctly to uWSGI where Gunicorn had been referenced.
- 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.

# 13.46 Version 4.29

Release date: 2020-12-10

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.28.

#### 13.46.1 New features

# 13.46.2 Housekeeping

Issue #5328 - Improve code coverage and API test cases for Foreign Tables.

Issue #5337 - Improve code coverage and API test cases for Views and Materialized Views.

Issue #5343 - Improve code coverage and API test cases for Debugger.

Issue #6062 - Ensure that code coverage should cover class and function declarations.

# 13.46.3 Bug fixes

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 #5943 - Ensure that folder rename should work properly in Storage Manager.

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 #5982 - Fixed documentation issue where JSON is not valid.

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.

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 #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 #6022 - Fixed an issue where shared servers import is failing.

Issue #6072 - Fixed DLL load failed while importing bcrypt.

# 13.47 Version 4.28

Release date: 2020-11-12

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.27.

13.46. Version 4.29 559

### 13.47.1 New features

- Issue #3318 Added support to download utility files at the client-side.
- 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 #5891 Added support to compare schemas and databases in schema diff.

# 13.47.2 Housekeeping

Issue #5938 - Documentation of Storage Manager.

# 13.47.3 Bug fixes

- Issue #4639 Ensure that some fields should be disabled for the trigger in edit mode.
- Issue #5736 Fixed an issue where the validation error message is shown twice.
- Issue #5760 Ensure that non-superuser should be able to debug the function.
- Issue #5842 Ensure that query history should be listed by date/time in descending order.
- Issue #5858 Ensure that search object functionality works with case insensitive string.
- 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 #5919 Added security related enhancements.
- Issue #5923 Fixed an issue where non-closeable tabs are getting closed.
- 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.

### 13.48 Version 4.27

Release date: 2020-10-15

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.26.

#### 13.48.1 New features

- Issue #1402 Added Macro support.
- Issue #2519 Added support to view trigger function under the respective trigger node.
- Issue #3794 Allow user to change the database connection from an open query tool tab.
- Issue #5200 Added support to ignore the owner while comparing objects in the Schema Diff tool.
- Issue #5857 Added documentation for Macro support.

# 13.48.2 Housekeeping

- Issue #5330 Improve code coverage and API test cases for Functions.
- Issue #5395 Added RESQL/MSQL test cases for Functions.
- Issue #5497 Merged the latest code of 'pgcli' used for the autocomplete feature.

# 13.48.3 Bug fixes

- 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 #5417 Fixed and improve API test cases for the schema diff tool.
- Issue #5739 Ensure that the import/export feature should work with SSH Tunnel.
- 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 #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 #5839 Ensure that multiple extensions can be dropped from the properties tab.
- Issue #5841 Fixed an issue where the server is not able to connect using the service.
- Issue #5843 Fixed an issue where the 'PARALLEL UNSAFE' option is missing from reverse engineering SQL of function/procedure.
- 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.
- 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 #5861 Ensure that the 'Remove Server' option should be visible in the context menu.
- Issue #5867 Fixed an issue where some properties are not being updated correctly for the shared server.
- 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.

### 13.49 Version 4.26

Release date: 2020-09-17

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.25.

13.49. Version 4.26 561

### 13.49.1 New features

- Issue #2042 Added SQL Formatter support in Query Tool.
- Issue #4059 Added a new button to the query tool toolbar to open a new query tool window.
- Issue #4979 Added shared server support for admin users.
- Issue #5772 Warn the user when connecting to a server that is older than pgAdmin supports.

# 13.49.2 Housekeeping

- Issue #5332 Improve code coverage and API test cases for Columns and Constraints (Index, Foreign Key, Check, Exclusion).
- Issue #5344 Improve code coverage and API test cases for Grant Wizard.
- Issue #5774 Improve code coverage and API test cases for Tables.
- Issue #5792 Added documentation for shared server support.

# 13.49.3 Bug fixes

- Issue #4216 Ensure that schema names starting with 'pg' should be visible in browser tree when standard\_conforming\_strings is set to off.
- Issue #5426 Adjusted the height of jobstep code block to use maximum space.
- Issue #5652 Modified the 'Commit' and 'Rollback' query tool button icons.
- Issue #5722 Ensure that the user should be able to drop the database even if it is connected.
- Issue #5732 Fixed some accessibility issues.
- Issue #5734 Update the description of GIN and GiST indexes in the documentation.
- 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 #5751 Enable the 'Configure' and 'View log' menu option when the server taking longer than usual time to start.
- 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 #5773 Fixed an issue where the application ignores the fixed port configuration value.
- Issue #5775 Ensure that 'setup-web.sh' should work in Debian 10.
- 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 #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.

# 13.50 Version 4.25

Release date: 2020-08-20

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.24.

#### 13.50.1 New features

- Issue #3904 Replace charting library Flotr2 with ChartJS using React.
- Issue #5126 Modified schema diff tool to compare two databases instead of two schemas.
- Issue #5610 Add a -yes command line option to setup-web.sh to allow non-interactive use.

# 13.50.2 Housekeeping

- 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 #5336 Improve code coverage and API test cases for Types.
- Issue #5700 Remove old Python 2 compatibility code.
- Issue #5731 Upgrade font awesome from v4 to v5.

# 13.50.3 Bug fixes

- 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 #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 #5429 Ensure that the Dictionaries drop-down shows all the dictionaries in the FTS configuration dialog.
- Issue #5490 Make the runtime configuration dialog non-modal.
- Issue #5526 Fixed an issue where copying and pasting a cell with multiple line data will result in multiple rows.
- Issue #5567 Fixed an issue where conversion of bytea to the binary string results in an error.
- 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 #5664 Fixed an issue where 'ALTER VIEW' statement is missing when the user sets the default value of a column for View.
- Issue #5670 Fixed an issue where the error message does not have a close button on utility dialogs.
- Issue #5689 Added the 'ORDER BY' clause for the privileges type to fix schema diff issue.
- Issue #5708 Correct TLS certificate filename in the container deployment docs.
- 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 disconnecting the database server.
- Issue #5724 Clarify some of the differences when running in server mode in the docs.

13.50. Version 4.25 563

Issue #5730 - Resolve schema diff dependencies by selecting the appropriate node automatically and maintain the order in the generated script.

### 13.51 Version 4.24

Release date: 2020-07-23

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.23.

#### **13.51.1** New features

- Issue #5235 Support configuration files that are external to the application installation.
- Issue #5484 Added support for LDAP authentication with different DN by setting the dedicated user for the LDAP connection.
- Issue #5583 Added support for schema level restriction.
- Issue #5601 Added RLS Policy support in Schema Diff.
- Issue #5622 Added support for permissive/restricted policy type while creating RLS Policy.
- Issue #5650 Added support for LDAP anonymous binding.
- Issue #5653 Added High Contrast theme support.

# 13.51.2 Housekeeping

- Issue #5323 Improve code coverage and API test cases for Foreign Data Wrapper.
- 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.

## 13.51.3 Bug fixes

- 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 #4235 Fixed tab indent issue on a selection of lines is deleting the content when 'use spaces == true' in the preferences.
- Issue #5137 Fixed save button enable issue when focusing in and out of numeric input field.
- Issue #5287 Fixed dark theme-related CSS and modify the color codes.
- Issue #5414 Use QStandardPaths::AppLocalDataLocation in the runtime to determine where to store runtime logs.
- Issue #5463 Fixed an issue where CSV download quotes numeric columns.
- Issue #5470 Fixed backgrid row hover issue where on hover background color is set for edit and delete cell only.
- Issue #5530 Ensure that the referenced table should be displayed on foreign key constraints.
- Issue #5554 Replace the runtime themes with ones that don't have sizing issues.
- Issue #5569 Fixed reverse engineered SQL for partitions when storage parameters are specified.
- Issue #5577 Include LICENSE and DEPENDENCIES [inventory] files in official packages.
- 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 #5630 Fixed an issue where installation of pgadmin4 not working on 32-bit Windows.

- 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 #5662 Fixed accessibility issue where few dialogs are not rendering properly when we zoomed in browser window 200% and screen resolution is low.
- Issue #5666 Added missing dependencies/dependent and corrected some wrongly identified.
- Issue #5673 Fixed an issue where fetching the schema throws an error if the database is not connected in Schema Diff.
- Issue #5675 Fixed CSRF errors when pgAdmin 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.

# 13.52 Version 4.23

Release date: 2020-06-25

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.22.

#### 13.52.1 New features

- 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.

### 13.52.2 Housekeeping

- Issue #5325 Improve code coverage and API test cases for Collations.
- Issue #5574 Cleanup Windows build scripts and ensure Windows x64 builds will work.
- Issue #5581 Documentation of Row Level Security Policies.

# 13.52.3 Bug fixes

- 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 #3787 Disabled the Stop process button after clicking it and added a message 'Terminating the process...' to notify the user.
- Issue #4226 Fixed an issue where select all checkbox only selects the first 50 tables.
- Issue #5416 Ensure that the query tool panel gets closed when clicking on the 'Don't Save' button.
- Issue #5465 Fixed an issue where the Edge browser version is showing wrong and warning message gets displayed.
- Issue #5492 Fixed an issue where the search object is unable to locate inherited tables and constraint filters are not working.
- Issue #5507 Fixed connection and version number detection issue when the database server is upgraded.
- Issue #5521 Fixed an issue when dumping servers from a desktop pgAdmin app by providing an option

'-sqlite-path'.

13.52. Version 4.23 565

- Issue #5539 Fixed typo in exception keyword.
- Issue #5584 Fixed an issue where two identical tables showing different by schema diff tool.
- Issue #5592 Ensure that pgadmin should be able to connect to the server which has password more than 1000 characters.

### 13.53 Version 4.22

Release date: 2020-05-28

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.21.

#### 13.53.1 New features

- Issue #5452 Added connected pgAdmin user and connection name in the log file.
- Issue #5489 Show the startup log as well as the server log in the runtime's log viewer.

# 13.53.2 Housekeeping

- Issue #5255 Implement Selenium Grid to run multiple tests across different browsers, operating systems, and machines in parallel.
- 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 #5443 Remove support for Python 2.
- Issue #5444 Cleanup Python detection in the runtime project file.
- Issue #5455 Refactor pgAdmin4.py so it can be imported and is a lot more readable.
- Issue #5493 Search object UI improvements.
- Issue #5525 Cleanup and refactor the macOS build scripts.
- Issue #5552 Update dependencies in the Docker container.
- Issue #5553 Remove PG 9.4 utilities from the Docker container as it's now out of support.

# 13.53.3 Bug fixes

- Issue #3694 Gracefully informed the user that the database is already connected when they click on "Connect Database...".
- 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 job should be worked properly for indexes under a materialized view.
- Issue #4279 Ensure that file browse "home" button should point to \$HOME rather than /.
- 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 #5052 Fixed internal server error when clicking on Triggers -> 'Enable All' for partitions.
- Issue #5398 Fixed generated SQL issue for auto vacuum options.
- 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 #5466 Correct ipv4 "all interfaces" address in the container docs, per Frank Limpert.
- Issue #5469 Fixed an issue where select2 hover is inconsistent for the SSL field in create server dialog.
- Issue #5473 Fixed post-login redirect location when running in server mode under a non-default root.
- Issue #5480 Fixed an issue where the background job creation fails if there is only a version-specific python binary available in PATH.
- Issue #5481 Fixed data truncation issue when updating the data of type character with length.
- Issue #5487 Fixed an issue where if LDAP\_SEARCH\_BASE\_DN is not set then, the value for LDAP\_BASE\_DN will be considered.
- Issue #5496 Fixed an issue where clicking on Select All button, not selecting all the options in pgAgent job scheduler.
- Issue #5503 Clarify and correct the docs on enabling the pl/debugger plugin on the server.
- Issue #5510 Fixed Unicode decode error 'utf-8' codec can't decode byte.

### 13.54 Version 4.21

Release date: 2020-04-30

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.20.

#### 13.54.1 New features

- Issue #2172 Added search object functionality.
- Issue #2186 Added LDAP authentication support.
- Issue #4636 Added job step and job schedule disable icons to identify it quickly within the browser tree.
- Issue #5181 Added support for parameter toast\_tuple\_target and parallel\_workers of the table.
- 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 #5348 Documentation of LDAP authentication support.
- Issue #5353 Added an option to prevent a browser tab being opened at startup.
- Issue #5399 Warn the user if an unsupported, deprecated or unknown browser is detected.

13.54. Version 4.21 567

## 13.54.2 Housekeeping

- Issue #4620 Add Reverse Engineered and Modified SQL tests for procedures.
- Issue #4623 Add Reverse Engineered and Modified SQL tests for pgAgent jobs.

### 13.54.3 Bug fixes

- 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 #4206 Ensure that the grant wizard should be closed on pressing the ESC key.
- Issue #4292 Added dark mode support for the configuration dialog on Windows/macOS runtime.
- 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 #4545 Fixed an issue wherein grant wizard the last object is not selectable.
- Issue #4573 Ensure that if the delimiter is set other than comma then download the file as '.txt' file.
- 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 #4864 Make the configuration window in runtime to auto-resize.
- 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 #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 #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 #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 #5210 Ensure that text larger than underlying field size should not be truncated automatically.
- Issue #5213 Fixed an issue when the user performs refresh on a large size materialized view.

- Issue #5227 Fixed an issue where user cannot be added if many users are already exists.
- 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 #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 #5351 Fixed compilation warnings while building pgAdmin.
- 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 pgAdmin4 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 #5402 Fixed an issue where the checkbox is not visible on Configuration dialog in runtime for the dark theme.
- Issue #5409 Fixed validation issue in Synonyms node.
- Issue #5410 Fixed an issue while removing the package body showing wrong modified SQL.
- Issue #5415 Ensure that the query tool context menu should work on the collection nodes.
- Issue #5419 Ensure that the user should not be able to change the authentication source.
- Issue #5420 Ensure error should be handled properly when LDAP user is created with the same name.
- Issue #5430 Added title to the login page.
- Issue #5432 Fixed an issue where an internal user is not created if the authentication source is set to internal and ldap.
- Issue #5439 Fixed an issue where the user is not able to create a server if login with an LDAP account.
- Issue #5441 Fixed an issue where the search object not able to locate pg\_toast\_\* tables in the pg\_toast schema.
- Issue #5447 Fixed failed to fetch utility error when click on refresh(any option) materialized view.

#### 13.55 Version 4.20

Release date: 2020-04-02

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.19.

13.55. Version 4.20 569

#### 13.55.1 New features

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.

# 13.55.2 Housekeeping

Issue #5271 - Enhance the color of switch control for both light and dark theme.

Issue #5284 - Added and fixed gettext usage for better translation coverage.

# 13.55.3 Bug fixes

Issue #4237 - Fix an issue where the user can not change the value of DateTime picker control using keyboard.

Issue #4608 - Fixed some accessibility issues in the dialogs.

Issue #4942 - Fixed chrome driver download utility issue for Ubuntu.

Issue #5128 - Change some colors and opacity to comply with WCAG color contrast standards.

Issue #5143 - Fix an accessibility issue to maximize the panel for all alertify dialog.

Issue #5221 - Improve logic to get the DDL statements as a part of the comparison.

Issue #5241 - Fixed tab key navigation issue for Grant Wizard.

Issue #5279 - Fixed Unicode character issue causing error on Python2 environment.

Issue #5292 - Fixed focus color issue for Alertify dialog buttons.

# 13.56 Version 4.19

Release date: 2020-03-05

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.18.

#### 13.56.1 New features

Issue #5154 - Added accessibility support in AlertifyJS.

Issue #5170 - Added Czech language support.

Issue #5179 - Added Python 3.8 support.

# 13.56.2 Housekeeping

Issue #5088 - Improve code coverage and API test cases for the Event Trigger module.

Issue #5133 - Improvements in the UI for both default and dark themes.

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.

# 13.56.3 Bug fixes

- Issue #4955 Changed the color of selected and hovered item for Select2 dropdown.
- Issue #4996 Improve the style of the highlighted code after query execution for Dark mode.
- 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 #5107 Set proper focus on tab navigation for file manager dialog.
- 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 pgAdmin.
- Issue #5184 Fixed Firefox monospaced issue by updating the font to the latest version.
- 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.

## 13.57 Version 4.18

Release date: 2020-02-06

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.17.

#### 13.57.1 New features

- Issue #2554 Added support for a multi-level partitioned table.
- Issue #3452 Added a Schema Diff tool to compare two schemas and generate a 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 #4770 Added labels and titles after parsing and validating all the pgAdmin4 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.

#### 13.57.2 Housekeeping

- 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 #5071 Improve the test framework to run for multiple classes defined in a single file.
- Issue #5072 Updated wcDocker package which includes aria-label accessibility improvements.
- Issue #5096 Replace node-sass with sass for SCSS compilation.

13.57. Version 4.18 571

## 13.57.3 Bug fixes

Issue #3812 - Ensure that path file name should not disappear when changing ext from the dropdown in file explorer dialog.

Issue #4410 - Fixed an issue while editing char[] or character varying[] column from View/Edit data throwing an error.

Issue #4511 - Fixed an issue where Grant wizard unable to handle multiple objects when the query string parameter exceeds its limit.

Issue #4601 - Added tab navigation on close buttons for all the panels and create/properties dialog.

Issue #4827 - Fix column resizable issue in the file explorer dialog.

Issue #5000 - Logout the pgAdmin session when no user activity of mouse move, click or keypress.

Issue #5025 - Fix an issue where setting STORAGE\_DIR to empty should show all the volumes on Windows in server mode.

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 #5076 - Ensure Postfix starts in the container, now it runs as non-root by default.

Issue #5116 - Fixed an issue where Save Password control disappears after clicking on it while creating a server.

## 13.58 Version 4.17

Release date: 2020-01-09

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.16.

#### 13.58.1 New features

Issue #4764 - Allow screen-reader to read relationship attributes in nested elements.

Issue #5060 - Ensure all binaries are securely signed and linked with the hardened runtime in the macOS bundle

#### 13.58.2 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 #5048 - Added code coverage tool for pgAdmin.

# 13.58.3 Bug fixes

Issue #4198 - Fix syntax highlighting in code mirror for backslash and escape constant.

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 #4633 - Added support to view multilevel partitioned tables.

Issue #4842 - Ensure that constraints, indexes, rules, triggers, and compound triggers should be created on partitions.

Issue #4943 - Added more information to the 'Database connected/disconnected' message.

Issue #4950 - Ensure that the user should be able to select/modify tablespace for the partitioned table on v12 and above.

Issue #4999 - Rename some internal environment variables that could conflict with Kubernetes.

Issue #5004 - Fix vulnerability issues reported by 'yarn audit'. Replace the deprecated uglifyjs-webpack-plugin with a terser-webpack-plugin.

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 #5013 - Add a note to the documentation about the use of non-privileged ports on filesystems that don't support extended attributes when running the container.

Issue #5047 - Added tab navigation for tabs under explain panel in query tool.

Issue #5068 - Fix an issue where the table is not created with autovacuum\_enabled and toast.autovacuum\_enabled for PG/EPAS 12.

## 13.59 Version 4.16

Release date: 2019-12-12

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.15.

**Warning:** Warning: This release includes a change to the container distribution to run pgAdmin as a non-root user. Those users of the container who are running with mapped storage directories may need to change the ownership on the host machine, for example:

sudo chown -R 5050:5050 <host\_directory>

#### 13.59.1 New features

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 #4939 - Run pgAdmin in the container as a non-root user (pgadmin, UID: 5050)

Issue #4944 - Allow Gunicorn logs in the container to be directed to a file specified through

GUNICORN\_ACCESS\_LOGFILE.

Issue #4990 - Changed the open query tool and data filter icons.

13.59. Version 4.16 573

## 13.59.2 Housekeeping

- Issue #4696 Add Reverse Engineered and Modified SQL tests for Materialized Views.
- Issue #4807 Refactored code of table and it's child nodes.
- Issue #4938 Refactored code of columns node.

#### 13.59.3 Bug fixes

- Issue #3538 Fix issue where the Reset button does not get enabled till all the mandatory fields are provided in the dialog.
- Issue #4220 Fix scrolling issue in 'Users' dialog.
- Issue #4516 Remove the sorting of table headers with no labels.
- 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 #4724 Fix network disconnect issue while establishing the connection via SSH Tunnel and it impossible to expand the Servers node.
- 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 #4792 Ensure that the superuser should be able to create database, as the superuser overrides all the access restrictions.
- Issue #4818 Fix server connection drops out issue in query tool.
- Issue #4836 Updated the json file name from 'servers.json' to 'pgadmin4/servers.json' in the container deployment section of the documentation.
- 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 #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 #4926 Fix VPN network disconnect issue where pgAdmin4 hangs on expanding the Servers node.
- Issue #4930 Fix main window tab navigation accessibility issue.
- Issue #4933 Ensure that the Servers collection node should expand independently of server connections.
- Issue #4934 Fix the help button link on the User Management dialog.
- 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 pgAdmin4 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 #4978 Fix pgAdmin4 failed to start issue after upgrading to version 4.15.
- 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.

# 13.60 Version 4.15

Release date: 2019-11-14

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.14.

#### 13.60.1 New features

- 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/resetting changes on a Properties dialog.
- Issue #4348 Added support for custom theme creation and selection.

# 13.60.2 Housekeeping

# 13.60.3 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 #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 #4173 Fix an issue where a black arrow-kind image is displaying at the background of browser tree images.
- 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 #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 #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 #4779 Updated documentation for the query tool toolbar buttons.
- Issue #4835 Fixed an issue where psql of v12 throwing "symbol not found" error while running Maintenance and Import/Export.
- Issue #4845 Fixed potential error in the properties dialog for the Code tab.

13.60. Version 4.15 575

- Issue #4850 Fixed an issue where Datetimepicker control opens when clicking on the label.
- Issue #4895 Fixed potential issue in reset function for nested objects.
- 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 #4924 Fixed docker container exit issue occurs due to change in Gunicorn's latest version.

# 13.61 Version 4.14

Release date: 2019-10-17

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.13.

#### 13.61.1 New features

- Issue #3009 Added Copy with headers functionality when copy data from Query Tool/View Data.
- Issue #4778 Implemented the Query Plan Analyser.
- Issue #4823 Include PostgreSQL 12 binaries in the container.

## 13.61.2 Housekeeping

- Issue #4472 Add Reverse Engineered and Modified SQL tests for Synonyms.
- Issue #4628 Add Reverse Engineered and Modified SQL tests for Unique Constraints.
- Issue #4701 Optimize Webpack to improve overall performance.

# 13.61.3 Bug fixes

- Issue #3386 Ensure backup a partition table should not backup the whole database.
- Issue #4199 Ensure that 'ENTER' key in the data filter should not run the query.
- 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 #4755 Ensure that pgAdmin should work behind reverse proxy if the inbuilt server is used as it is.
- Issue #4756 Fix issue where pgAdmin does not load completely if loaded in an iframe.
- Issue #4760 Ensure the search path should not be quoted for Database.
- Issue #4768 Ensure pgAdmin should work behind reverse proxy on a non standard port.
- Issue #4769 Fix query tool open issue on Internet Explorer.
- Issue #4777 Fix issue where query history is not visible in the query history tab.
- 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 #4817 Ensure the MAC OSX app should be notarized for Catalina.

# 13.62 Version 4.13

Release date: 2019-09-19

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.12.

#### 13.62.1 New features

Issue #2828 - Added Gather Merge, Named Tuple Store Scan and Table Function Scan icon for explain module.

Issue #4553 - Don't wait for the database connection before rendering the Query Tool UI, for improved UX.

Issue #4651 - Allow configuration options to be set from the environment in the container distribution.

Issue #4667 - Ensure editable and read-only columns in Query Tool should be identified by icons and tooltips in the column header.

Issue #4691 - Add an Italian translation.

Issue #4752 - Refactor Dockerfile to avoid needing to run supporting scripts (i.e. 'docker build .' will work) and minimise layers.

## 13.62.2 Housekeeping

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 #4690 - Add Modified SQL tests for Resource Group.

#### 13.62.3 Bug fixes

Issue #2706 - Added ProjectSet icon for explain module.

Issue #3778 - Ensure Boolean columns should be editable using keyboard keys.

Issue #3936 - Further code refactoring to stabilise the Feature Tests.

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 #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 #4419 - Fix a debugger error when using Python 2.7.

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 #4577 - Fix an error that could be seen when click on any system column of a table.

Issue #4584 - Unescape HTML entities in database names in the Query Tool title bar.

13.62. Version 4.13 577

- 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 #4642 Ensure port and username should not be mandatory when a service is provided.
- 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 #4657 Fix PGADMIN\_SERVER\_JSON\_FILE environment variable support in the container.
- 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 #4736 Fix query tool and view data issue with the Italian language.
- Issue #4742 Ensure Primary Key should be created with Index.
- Issue #4750 Fix query history exception for Python 3.6.

## 13.63 Version 4.12

Release date: 2019-08-22

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.11.

#### 13.63.1 New features

- Issue #4144 Add support of Compound Triggers for EPAS 12+.
- 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 #4549 Allow a banner to be displayed on the login and other related pages showing custom text.
- 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.

# 13.63.2 Housekeeping

- 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.

## 13.63.3 Bug fixes

- Issue #3605 Fix issue where Deleting N number of rows makes first N number of rows disable.
- Issue #4179 Fix generation of reverse engineered SQL for tables with Greenplum 5.x.
- Issue #4229 Update wcDocker to allow the browser's context menu to be used except in tab strips and panel headers.
- Issue #4401 Ensure type names are properly encoded in the results grid.
- Issue #4414 Fix generation of reverse engineered SQL for partition table, partitions were shown as a child of indexes.
- 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 #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 #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 #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.

# 13.64 Version 4.11

Release date: 2019-07-25

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.10.

13.64. Version 4.11 579

#### 13.64.1 New features

- Issue #1760 Add support for editing of resultsets in the Query Tool, if the data can be identified as updatable.
- Issue #4318 Set the mouse cursor appropriately based on the layout lock state.
- Issue #4335 Add EXPLAIN options for SETTINGS and SUMMARY.

## 13.64.2 Housekeeping

- Issue #4415 Add Reverse Engineered SQL tests for Roles and Resource Groups.
- Issue #4441 Add Reverse Engineered SQL tests for FDWs.
- 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.
- Issue #4475 Add Reverse Engineered SQL tests for Constraints.

# 13.64.3 Bug fixes

- Issue #3919 Allow keyboard navigation of all controls on subnode grids.
- Issue #3996 Fix dropping of pgAgent schedules through the Job properties.
- Issue #4224 Prevent flickering of large tooltips on the Graphical EXPLAIN canvas.
- 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 #4395 EXPLAIN options should be Query Tool instance-specific.
- 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 #4429 Ensure drag/drop from the treeview works as expected on Firefox.
- Issue #4437 Fix table icon issue when updating any existing field.
- Issue #4442 Ensure browser should not be started by Selenium when feature tests are excluded from a test run.
- 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 #4450 Fix reverse engineered sql for Foreign Data Wrapper created on EPAS server in redwood mode.
- 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.

# 13.65 Version 4.10

Release date: 2019-07-04

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.9.

#### 13.65.1 New features

Issue #4139 - Allow some objects to be dragged/dropped into the Query Tool to insert their signature into the query text

Issue #4400 - Allow the path to /pgadmin4/servers.json to be overridden in the container distribution.

## 13.65.2 Bug fixes

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 #4421 - Ensure the version comparision should be correct for windows installer.

## 13.66 Version 4.9

Release date: 2019-06-27

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.8.

#### 13.66.1 New features

Issue #3174 - Visually distinguish simple tables from tables that are inherited and from which other tables are inherited.

## 13.66.2 Housekeeping

Issue #4202 - Add a framework for testing reversed engineered SQL and CRUD API endpoints.

## 13.66.3 Bug fixes

Issue #3994 - Fix issue where the dependencies tab for inherited tables/foreign keys shows partial text.

Issue #4036 - Allow editing of data where a primary key column includes a % sign in the value.

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" tabsets such as the Query Tool and Debugger.

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 #4283 - Initial support for PostgreSQL 12.

Issue #4288 - Initial support for PostgreSQL 12.

13.65. Version 4.10 581

- Issue #4290 Initial support for PostgreSQL 12.
- Issue #4255 Prevent the geometry viewer grabbing key presses when not in focus under Firefox, IE and Edge.
- Issue #4306 Prevent the "Please login to access this page" message displaying multiple times.
- Issue #4310 Ensure that the Return key can be used to submit the Master Password dialogue.
- Issue #4317 Ensure that browser auto-fill doesn't cause Help pages to be opened unexpectedly.
- 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 #4357 Fix connection restoration issue when pgAdmin server is restarted and the page is refreshed.
- 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.

## 13.67 Version 4.8

Release date: 2019-06-03

This release contains a number of bug fixes and new features since the release of pgAdmin4 4.7.

#### 13.67.1 New features

Issue #2653 - Allow the UI layout to be fully locked or to prevent docking changes.

# 13.67.2 Bug fixes

- Issue #4169 Omit the geometry viewer in the Query Tool from layout saving.
- Issue #4307 Improve the performance of explain plan by embedding the images only when downloading it.
- Issue #4308 Fix the issue of accessing the SQL for Views and Materialized Views. Regression of pluralisation of folder names.

## 13.68 Version 4.7

Release date: 2019-05-30

This release contains a number of bug fixes since the release of pgAdmin4 4.6.

# 13.68.1 Bug fixes

- Issue #3377 In server mode, update all the saved server credentials when user password is changed.
- Issue #3885 Fix the responsive layout of the main menu bar.
- Issue #4162 Fix syntax error when adding more than one column to the existing table.
- Issue #4164 Fix file browser path issue which occurs when client is on Windows and server is on Mac/Linux.
- Issue #4184 Added Master Password to increase the security of saved passwords.
- Issue #4194 Fix accessibility issue for menu navigation.
- Issue #4208 Update the UI logo.
- Issue #4217 Fixed CSRF security vulnerability issue, per Alvin Lindstam
- Issue #4218 Properly assign dropdownParent in Select2 controls.
- Issue #4219 Ensure popper.js is installed when needed.
- Issue #4227 Fixed Tab key navigation for Maintenance dialog.
- Issue #4244 Fix Tab key issue for Toggle switch controls and button on the dialog footer in Safari browser.
- Issue #4245 Ensure that element should get highlighted when they get focus on using Tab key.
- Issue #4246 Fixed console error when subnode control is used in panels.
- Issue #4261 Stop using application/x-javascript as a mime type and use the RFC-compliant application/javascript instead.
- Issue #4262 Fixed error on displaying table properties of a table partitioned by list having a default partition.
- Issue #4263 Fix handling of JSON in the Query Tool with NULL elements.
- Issue #4269 Fix navigation of switch cells in grids.
- Issue #4275 Clarify wording for the NO INHERIT option on constraints, per Michel Feinstein.
- Issue #4276 Relax the permission check on the directory containing the config database, as it may fail in some environments such as OpenShift.
- Issue #4278 Prevent Backgrid Password cells from losing focus if the browser opens an autocomplete list.
- Issue #4284 Fix syntax error when creating a table with a serial column.

13.68. Version 4.7 583

## 13.69 Version 4.6

Release date: 2019-05-02

This release contains a number of new features and fixes reported since the release of pgAdmin4 4.5

#### 13.69.1 Features

Issue #4165 - Depend on psycopg2-binary in the Python wheel, rather than psycopg2.

# 13.69.2 Bug fixes

Issue #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.

Issue #3327 - Ensure that newly added row in backgrid should be visible.

Issue #3582 - Ensure that JSON strings as comments should be added properly for all the objects.

Issue #3605 - Fix an issue where Deleting N number of rows makes first N number of rows disable.

Issue #3938 - Added support for Default Partition.

Issue #4087 - Fix an issue where 'GRANT UPDATE' sql should be displayed for default sequence privileges.

Issue #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.

Issue #4104 - Ensure that record should be add/edited for root partition table with primary keys.

Issue #4121 - Fixed alignment issue of columns in definition section of Index node.

Issue #4134 - Fixed 'Location cannot be empty' error when open Tablespace properties.

Issue #4138 - Fix an issue where the dropdown becomes misaligned/displaced.

Issue #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.

Issue #4160 - Fixed 'Increment value cannot be empty' error for existing tables.

Issue #4161 - Ensure that parameters of procedures for EPAS server 10 and below should be set/reset properly.

Issue #4163 - Prevent duplicate columns being included in reverse engineered SQL for tables.

Issue #4182 - Ensure sanity of the permissions on the storage and session directories and the config database.

# 13.70 Version 4.5

Release date: 2019-04-10

This release contains a number of new features and fixes reported since the release of pgAdmin4 4.4.

## 13.70.1 Bug fixes

Issue #2214 - Fixed 'Change Password' issue for SCRAM authentication.

Issue #3656 - Ensure that two consecutive SELECT statements should work properly.

Issue #4131 - Relabel the Save button on the datagrid text editor to avoid confusion with the actual Save button that updates the database.

Issue #4142 - Added recommended ESLinter checks.

Issue #4143 - Ensure that pgAdmin4 should work properly with psycopg2 v2.8

## 13.71 Version 4.4

Release date: 2019-04-04

This release contains a number of new features and fixes reported since the release of pgAdmin4 4.3.

**Warning:** This release includes a bug fix (Issue #3887) which will rename the per-user storage directories for existing users when running in server mode. Previously, saved SQL queries were stored under the *STORAGE\_DIR* in a sub-directory named after the username part of the user's email address. From this version onwards, the full email address is used, with the @ replaced with an underscore. For example, in v.4.3 with *STORAGE\_DIR* set to /var/lib/pgadmin4 user files may be stored in:

/var/lib/pgadmin4/storage/username/

With the fix, that directory will be renamed (or created for new users) as:

/var/lib/pgadmin4/storage/username\_example.com/

#### 13.71.1 Features

Issue #2001 - Add support for reverse proxied setups with Gunicorn, and document Gunicorn, uWSGI & NGINX configurations.

Issue #4017 - Make the Query Tool history persistent across sessions.

Issue #4018 - Remove the large and unnecessary dependency on React and 87 other related libraries.

Issue #4030 - Add support for IDENTITY columns.

Issue #4075 - Add an ePub doc build target.

13.70. Version 4.5 585

## 13.71.2 Bug fixes

- Issue #1269 Fix naming inconsistency for the column and FTS parser modules.
- Issue #2627 Include inherited column comments and defaults in reverse engineered table SQL.
- Issue #3104 Improve a couple of German translations.
- Issue #3887 Use the user's full email address (not just the username part) as the basis for the storage directory name.
- Issue #3968 Update wcDocker to fix the issue where the Scratch Pad grows in size if the results panel is resized.
- Issue #3995 Avoid 'bogus varno' message from Postgres when viewing the SQL for a table with triggers.
- Issue #4019 Update all Python and JavaScript dependencies.
- Issue #4037 Include comment SQL for inherited columns in reverse engineered table SQL.
- Issue #4050 Make the WHEN field a CodeMirror control on the Event Trigger dialogue.
- Issue #4052 Fix the online help button on the resource group dialogue.
- Issue #4053 Enable the online help button on the index dialogue.
- Issue #4054 Handle resultsets with zero columns correctly in the Query Tool.
- Issue #4058 Include inherited columns in SELECT scripts.
- Issue #4060 Fix the latexpdf doc build.
- Issue #4062 Fix handling of numeric arrays in View/Edit Data.
- Issue #4063 Enlarge the grab handles for resizing dialogs etc.
- Issue #4069 Append the file suffix to filenames when needed in the File Create dialogue.
- Issue #4071 Ensure that Firefox prompts for a filename/location when downloading query results as a CSV file.
- Issue #4073 Change the CodeMirror active line background colour to \$color-danger-lighter so it doesn't conflict with the selection colour.
- Issue #4081 Fix the RE-SQL syntax for roles with a VALID UNTIL clause.
- Issue #4082 Prevent an empty error message being shown when "downloading" a CREATE script using the CSV download.
- Issue #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.
- Issue #4085 Display errors during CSV download from the Query Tool in the UI rather than putting them in the CSV file.
- Issue #4090 Improve the German translation for Backup Server.
- Issue #4096 Ensure the toolbar buttons are properly reset following a CSV download in the Query Tool.
- Issue #4099 Fix SQL help for EPAS 10+, and refactor the URL generation code into a testable function.
- Issue #4100 Ensure sequences can be created with increment, start, minimum and maximum options set.
- Issue #4105 Fix an issue where JSON data would not be rendered in the Query Tool.
- Issue #4109 Ensure View/Materialized View node should be visible after updating any property.
- Issue #4110 Fix custom autovacuum configuration for Materialized Views.

# 13.72 Version 4.3

Release date: 2019-03-07

This release contains a number of new features and fixes reported since the release of pgAdmin4 4.2

#### 13.72.1 Features

Issue #1825 - Install a script to start pgAdmin (pgadmin4) from the command line when installed from the Python wheel.

Issue #2233 - Add a "scratch pad" to the Query Tool to hold text snippets whilst editing.

Issue #2418 - Add Commit and Rollback buttons to the Query Tool.

Issue #3439 - Allow X-FRAME-OPTIONS to be set for security. Default to SAMEORIGIN.

Issue #3559 - Automatically expand child nodes as well as the selected node on the treeview if there is only one.

Issue #3886 - Include multiple versions of the PG utilties in containers.

Issue #3991 - Update Alpine Linux version in the docker container.

Issue #4034 - Support double-click on Query Tool result grid column resize handles to auto-size to the content.

# 13.72.2 Bug fixes

Bug #3096 - Ensure size stats are prettified on the statistics tab when the UI language is not English.

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 #3544 - Make the Query Tool tab titles more concise and useful.

Bug #3587 - Fix support for bigint's in JSONB data.

Bug #3583 - Update CodeMirror to 5.43.0 to resolve issues with auto-indent.

Bug #3600 - Ensure JSON data isn't modified in-flight by psycopg2 when using View/Edit data.

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 #3873 - Fix context sub-menu alignment on Safari.

Bug #3890 - Update documentation screenshots as per new design.

Bug #3906 - Fix alignment of Close and Maximize button of Grant Wizard.

Bug #3911 - Add full support and testsfor all PG server side encodings.

Bug #3912 - Fix editing of table data with a JSON primary key.

Bug #3933 - Ignore exceptions in the logger.

Bug #3942 - Close connections gracefully when the user logs out of pgAdmin.

Bug #3946 - Fix alignment of checkbox to drop multiple schedules of pgAgent job.

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 #3963 - Fix alignment of import/export toggle switch.

Bug #3970 - Prevent an error when closing the Sort/Filter dialogue with an empty filter string.

Bug #3974 - Fix alignment of Connection type toggle switch of pgagent.

Bug #3981 - Fix the query to set bytea\_output so that read-only standbys don't consider it a write query.

Bug #3982 - Add full support and testsfor all PG server side encodings.

13.72. Version 4.3 587

- Bug #3985 Don't embed docs and external sites in iframes, to allow the external sites to set X-FRAME-OPTIONS = DENY for security.
- Bug #3992 Add full support and testsfor all PG server side encodings.
- 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 #4022 Fix an issue where importing servers fails if a group already exists for a different user.

#### 13.73 Version 4.2

Release date: 2019-02-07

This release contains a number of fixes reported since the release of pgAdmin4 4.1

## 13.73.1 Bug fixes

- Issue #3051 Replace Bootstrap switch with Bootstrap4 toggle to improve the performance.
- Issue #3272 Replace the PyCrypto module with the cryptography module.
- Issue #3453 Fixed SQL for foreign table options.
- Issue #3475 Fixed execution time to show Hours part for long running queries in Query Tool.
- Issue #3608 Messages tab of Query Tool should be clear on subsequent execution of table/view using View/Edit Data
- Issue #3609 Clear drop-down menu should be disabled for View/Edit Data.
- Issue #3664 Fixed Statistics panel hang issue for 1000+ tables.
- Issue #3693 Proper error should be thrown when server group is created with existing name.
- Issue #3695 Ensure long string should be wrap in alertify dialogs.
- Issue #3697 Ensure that output of the query should be displayed even if Data Output window is detached from the Query Tool.
- Issue #3740 Inline edbspl trigger functions should not be visible in Grant Wizard.
- Issue #3774 Proper SQL should be generated when create function with return type as custom type argument.
- Issue #3800 Ensure that database restriction of server dialog should work with special characters.
- Issue #3811 Ensure that Backup/Restore button should work on single click.
- Issue #3837 Fixed SQL for when clause while creating Trigger.
- Issue #3838 Proper SQL should be generated when creating/changing column with custom type argument.
- Issue #3840 Ensure that file format combo box value should be retained when hidden files checkbox is toggled.
- Issue #3846 Proper SQL should be generated when create procedure with custom type arguments.
- Issue #3849 Ensure that browser should warn before close or refresh.
- Issue #3850 Fixed EXEC script for procedures.
- Issue #3853 Proper SQL should be generated when create domain of type interval with precision.
- Issue #3858 Drop-down should be closed when click on any other toolbar button.
- Issue #3862 Fixed keyboard navigation for dialog tabs.
- Issue #3865 Increase frames splitter mouse hover area to make it easier to resize.
- Issue #3871 Fixed alignment of tree arrow icons for Internet Explorer.
- Issue #3872 Ensure object names in external process dialogues are properly escaped.
- Issue #3891 Correct order of Save and Cancel button for json/jsonb editing.

- Issue #3897 Data should be updated properly for FTS Configurations, FTS Dictionaries, FTS Parsers and FTS Templates.
- Issue #3899 Fixed unable to drop multiple Rules and Foreign Tables from properties tab.
- Issue #3903 Fixed Query Tool Initialization Error.
- Issue #3908 Fixed keyboard navigation for Select2 and Privilege cell in Backgrid.
- Issue #3916 Correct schema should be displayed in Materialized View dialog.
- Issue #3927 Fixed debugger issue for procedure inside package for EPAS servers.
- Issue #3929 Fix alignment of help messages in properties panels.
- Issue #3932 Fix alignment of submenu for Internet Explorer.
- Issue #3935 Ensure that grant wizard should list down functions for EPAS server running with no-redwood-compat mode.
- Issue #3941 Dashboard graph optimization.
- Issue #3954 Remove Python 2.6 code that's now obsolete.
- Issue #3955 Expose the bind address in the Docker container via PGADMIN\_BIND\_ADDRESS.
- Issue #3961 Exclude HTTPExceptions from the all\_exception\_handler as they should be returned as-is.

#### 13.74 Version 4.1

Release date: 2019-01-15

This release contains a number of fixes reported since the release of pgAdmin4 4.0

# 13.74.1 Bug fixes

- Issue #3505 Fix SQL generated for tables with inherited columns.
- Issue #3575 Ensure the context menu works after a server is renamed.
- Issue #3836 Fix ordering of VACUUM options which changed in PG11.
- Issue #3842 Don't show system catalogs in the schemas property list unless show system objects is enabled.
- Issue #3861 Fix help for the backup/restore dialogues.
- Issue #3866 Ensure that last row of table data should be visible and user will be able to add new row.
- Issue #3877 Make the browser more robust in the face of multibyte characters in SQL\_ASCII databases.

13.74. Version 4.1 589

## 13.75 Version 4.0

Release date: 2019-01-10

This release contains a number of features and fixes reported since the release of pgAdmin4 3.6

#### 13.75.1 Features

- Issue #3589 Allow query plans to be downloaded as an SVG file.
- Issue #3692 New UI design.
- Issue #3801 Allow servers to be pre-loaded into container deployments.

# 13.75.2 Bug fixes

- Issue #3083 Increase the size of the resize handle of the edit grid text pop-out.
- Issue #3354 Fix handling of array types as inputs to the debugger.
- Issue #3433 Fix an issue that could cause the Query Tool to fail to render.
- Issue #3549 Display event trigger functions correctly on EPAS.
- Issue #3559 Further improvements to treeview restoration.
- Issue #3599 Run Postfix in the container build so passwords can be reset etc.
- Issue #3619 Add titles to the code areas of the Query Tool and Debugger to ensure that panels can be re-docked within them.
- Issue #3679 Fix a webpack issue that could cause the Query Tool to fail to render.
- Issue #3702 Ensure we display the relation name (and not the OID) in the locks table wherever possible.
- Issue #3711 Fix an encoding issue in the Query Tool.
- Issue #3726 Include the WHERE clause on EXCLUDE constraints in RE-SQL.
- Issue #3753 Fix an issue when user define Cast from smallint->text is created.
- Issue #3757 Hide Radio buttons that should not be shown on the maintenance dialogue.
- Issue #3780 Ensure that null values handled properly in CSV download.
- Issue #3796 Tweak the wording on the Grant Wizard.
- Issue #3797 Prevent attempts to bulk-drop schema objects.
- Issue #3798 Ensure the browser toolbar buttons work in languages other than English.
- Issue #3805 Allow horizontal sizing of the edit grid text pop-out.
- Issue #3809 Ensure auto complete should works when first identifier in the FROM clause needs quoting.
- Issue #3810 Ensure auto complete should works for columns from a schema-qualified table.
- Issue #3821 Ensure identifiers are properly displayed in the plan viewer.
- Issue #3830 Make the setup process more robust against aborted executions.
- Issue #3856 Fixed an issue while creating export job.

# 13.76 Version 3.6

Release date: 2018-11-29

This release contains a number of features and fixes reported since the release of pgAdmin4 3.5

#### 13.76.1 Features

Issue #1513 - Add support for dropping multiple objects at once from the collection Properties panel.

Issue #3772 - Add the ability to import and export server definitions from a config database.

## 13.76.2 Bug fixes

Issue #3016 - Ensure previous notices are not removed from the Messages tab in the Query Tool if an error occurs during query execution.

Issue #3029 - Allow the selection order to be preserved in the Select2 control to fix column ordering in data Import/Export.

Issue #3629 - Allow use of 0 (integer) and empty strings as parameters in the debugger.

Issue #3723 - Properly report errors when debugging cannot be started.

Issue #3734 - Prevent the debugger controls being pressed again before previous processing is complete.

Issue #3736 - Fix toggle breakpoints buttons in the debugger.

Issue #3742 - Fix changes to the NOT NULL and default value options in the Table Dialogue.

Issue #3746 - Fix dropping of multiple functions/procedures at once.

## 13.77 Version 3.5

Release date: 2018-11-01

This release contains a number of features and fixes reported since the release of pgAdmin4 3.4

#### **13.77.1 Features**

Issue #1253 - Save the treeview state periodically, and restore it automatically when reconnecting.

Issue #3562 - Migrate from Bootstrap 3 to Bootstrap 4.

# 13.77.2 Bug fixes

Issue #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.

Issue #3638 - Fix syntax error when creating new pgAgent schedules with a start date/time and exception.

Issue #3674 - Cleanup session files periodically.

Issue #3660 - Rename the 'SQL Editor' section of the Preferences to 'Query Tool' as it applies to the whole tool, not just the editor.

Issue #3676 - Fix CREATE Script functionality for EDB-Wrapped functions.

Issue #3700 - Fix connection garbage collector.

Issue #3703 - Purge connections from the cache on logout.

13.76. Version 3.6 591

Issue #3722 - Ensure that utility existence check should work for schema and other child objects while taking Backup/Restore.

Issue #3730 - Fixed fatal error while launching the pgAdmin4 3.5. Update the version of the Flask to 0.12.4 for release.

# 13.78 Version 3.4

Release date: 2018-10-04

This release contains a number of features and fixes reported since the release of pgAdmin4 3.3

#### 13.78.1 Features

Issue #2927 - Move all CSS into SCSS files for consistency and ease of colour maintenance etc.

Issue #3514 - Add optional data point markers and mouse-over tooltips to display values on graphs.

Issue #3564 - Add shortcuts for View Data and the Query tool to the Browser header bar.

## 13.78.2 Bug fixes

Issue #3464 - Ensure the runtime can startup properly if there are wide characters in the logfile path on Windows.

Issue #3551 - Fix handling of backslashes in the edit grid.

Issue #3576 - Ensure queries are no longer executed when dashboards are closed.

Issue #3596 - Fix support for the CLOB datatype in EPAS.

Issue #3607 - Fix logic around validation and highlighting of Sort/Filter in the Query Tool.

Issue #3630 - Ensure auto-complete works for objects in schemas other than public and pg\_catalog.

Issue #3657 - Ensure changes to Query Tool settings from the Preferences dialogue are applied before executing queries.

Issue #3658 - Swap the Schema and Schemas icons and Catalog and Catalogs icons that had been used the wrong way around.

# 13.79 Version 3.3

Release date: 2018-09-06

This release contains a number of features and fixes reported since the release of pgAdmin4 3.2

#### 13.79.1 Features

Issue #1407 - Add a geometry viewer that can render PostGIS data on a blank canvas or various map sources.

Issue #3503 - Added new backup/restore options for PostgreSQL 11. Added dump options for 'pg\_dumpall'.

Issue #3553 - Add a Spanish translation.

# 13.79.2 Bug fixes

- Issue #3136 Stabilise feature tests for continuous running on CI systems.
- Issue #3191 Fixed debugger execution issues.
- Issue #3313 Ensure 'select all' and 'unselect all' working properly for pgAgent schedule.
- Issue #3325 Fix sort/filter dialog issue where it incorrectly requires ASC/DESC.
- Issue #3347 Ensure backup should work with '-data-only' and '-schema-only' for any format.
- Issue #3407 Fix keyboard shortcuts layout in the preferences panel.
- Issue #3420 Merge pgcli code with version 1.10.3, which is used for auto complete feature.
- Issue #3461 Ensure that refreshing a node also updates the Property list.
- Issue #3525 Ensure that refresh button on dashboard should refresh the table.
- Issue #3528 Handle connection errors properly in the Query Tool.
- Issue #3547 Make session implementation thread safe
- Issue #3548 Ensure external table node should be visible only for GPDB.
- Issue #3554 Fix auto scrolling issue in debugger on step in and step out.
- Issue #3558 Fix sort/filter dialog editing issue.
- Issue #3561 Ensure sort/filter dialog should display proper message after losing database connection.
- Issue #3578 Ensure sql for Role should be visible in SQL panel for GPDB.
- Issue #3579 When building the Windows installer, copy system Python packages before installing dependencies to ensure we don't end up with older versions than intended.
- Issue #3604 Correct the documentation of View/Edit data.

# 13.80 Version 3.2

Release date: 2018-08-09

This release contains a number of features and fixes reported since the release of pgAdmin4 3.1

#### 13.80.1 Features

- Issue #2136 Added version number for URL's to ensure that files are only cached on a per-version basis.
- Issue #2214 Add support for SCRAM password changes (requires psycopg2 >= 2.8).
- Issue #3074 Add support for reset saved password.
- Issue #3397 Add support for Trigger and JIT stats in the graphical query plan viewer.
- Issue #3412 Add support for primary key, foreign key, unique key, indexes and triggers on partitioned tables for PG/EPAS 11.
- Issue #3506 Allow the user to specify a fixed port number in the runtime to aid cookie whitelisting etc.
- Issue #3510 Add a menu option to the runtime to copy the appserver URL to the clipboard.

13.80. Version 3.2 593

## 13.80.2 Bug fixes

- Issue #3185 Fix the upgrade check on macOS.
- Issue #3191 Fix a number of debugger execution issues.
- Issue #3294 Infrastructure (and changes to the Query Tool, Dashboards and Debugger) for realtime preference handling.
- Issue #3309 Fix Directory format support for backups.
- Issue #3316 Support running on systems without a system tray.
- Issue #3319 Cleanup and fix handling of Query Tool Cancel button status.
- Issue #3363 Fix restoring of restore options for sections.
- Issue #3371 Don't create a session when the /misc/ping test endpoint is called.
- Issue #3446 Various procedure/function related fixes for EPAS/PG 11.
- Issue #3448 Exclude system columns in Import/Export.
- Issue #3457 Fix debugging of procedures in EPAS packages.
- Issue #3458 pgAdmin4 should work with python 3.7.
- Issue #3468 Support SSH tunneling with keys that don't have a passphrase.
- Issue #3471 Ensure the SSH tunnel port number is honoured.
- Issue #3511 Add support to save and clear SSH Tunnel password.
- Issue #3526 COST statement should not be automatically duplicated after creating trigger function.
- Issue #3527 View Data->Filtered Rows dialog should be displayed.

## 13.81 Version 3.1

Release date: 2018-06-28

This release contains a number of features and fixes reported since the release of pgAdmin4 3.0

#### 13.81.1 Features

- Issue #1447 Add support for SSH tunneled connections
- Issue #2686 Add an option to auto-complete keywords in upper case
- Issue #3204 Add support for LISTEN/NOTIFY in the Query Tool
- Issue #3273 Allow sorting in the file dialogue
- Issue #3362 Function and procedure support for PG11
- Issue #3388 Allow the connection timeout to be configured on a per-server basis

## 13.81.2 Bug fixes

- Issue #1220 Backup and Restore should not be started if database name contains "=" symbol
- Issue #1221 Maintenance should not be started if database name contains "=" symbol
- Issue #3179 Fix an error generating SQL for trigger functions
- Issue #3238 Standardise the error handling for parsing of JSON response messages from the server
- Issue #3250 Fix handling of SQL\_ASCII data in the Query Tool
- Issue #3257 Catch errors when trying to EXPLAIN an invalid query
- Issue #3277 Ensure server cleanup on exit only happens if the server actually started up
- Issue #3284 F5 key should work to refresh Browser tree

- Issue #3289 Fix handling of SQL\_ASCII data in the Query Tool
- Issue #3290 Close button added to the alertify message box, which pops up in case of backend error
- Issue #3295 Ensure the debugger gets focus when loaded so shortcut keys work as expected
- Issue #3298 Fixed Query Tool keyboard issue where arrow keys were not behaving as expected for execute options dropdown
- Issue #3303 Fix a Japanese translation error that could prevent the server starting up
- Issue #3306 Fixed display SQL of table with index for Greenplum database
- Issue #3307 Allow connections to servers with port numbers < 1024 which may be seen in container environments
- Issue #3308 Fixed issue where icon for Partitioned tables was the same as Non Partitioned tables for Greenplum database
- Issue #3310 Fixed layout of the alertify error message in the Query Tool
- Issue #3324 Fix the template loader to work reliably under Windows (fixing external tables under Greenplum)
- Issue #3333 Ensure the runtime core application is setup before trying to access any settings
- Issue #3342 Set SESSION\_COOKIE\_SAMESITE='Lax' per Flask recommendation to prevents sending cookies
- with CSRF-prone requests from external sites, such as submitting a form
- Issue #3353 Handle errors properly if they occur when renaming a database
- Issue #3356 Include the schema name on RE-SQL for packages
- Issue #3374 Fix autocomplete
- Issue #3392 Fix IPv6 support in the container build
- Issue #3409 Avoid an exception on GreenPlum when retrieving RE-SQL on a table
- Issue #3411 Fix a French translation error that could prevent the server starting up
- Issue #3431 Fix the RE-SQL generation for GreenPlum external tables

# 13.82 Version 3.0

Release date: 2018-03-22

This release contains a number of features and fixes reported since the release of pgAdmin4 2.1

#### 13.82.1 Features

- Issue #1894 Allow sorting when viewing/editing data
- Issue #1978 Add the ability to enable/disable UI animations
- Issue #2895 Add keyboard navigation options for the main browser windows
- Issue #2896 Add keyboard navigation in Query tool module via Tab/Shift-Tab key
- Issue #2897 Support keyboard navigation in the debugger
- Issue #2898 Support tab navigation in dialogs
- Issue #2899 Add configurable shortcut keys for various common options in the main window
- Issue #2901 Configurable shortcuts in the Debugger
- Issue #2904 Ensure clickable images/buttons have appropriate tooltips for screen readers
- Issue #2950 Add a marker (/pga4dash/) to the dashboard queries to allow them to be more easily filtered from server logs
- Issue #2951 Allow dashboard tables and charts to be enabled/disabled
- Issue #3004 Support server and database statistics on Greenplum
- Issue #3036 Display partitions in Greenplum
- Issue #3044 Display functions in Greenplum
- Issue #3086 Rewrite the runtime as a tray-based server which can launch a web browser

13.82. Version 3.0 595

- Issue #3097 Support EXPLAIN on Greenplum
- Issue #3098 Unvendorize REACT so no longer required in our source tree
- Issue #3107 Hide tablespace node on GPDB
- Issue #3140 Add support for connecting using pg\_service.conf files
- Issue #3168 Support for external tables in GPDB
- Issue #3182 Update Jasmine to v3
- Issue #3184 Add a French translation
- Issue #3195 Pass the service name to external processes
- Issue #3246 Update container build to use Alpine Linux and Gunicorn instead of CentOS/Apache

In addition, various changes were made for PEP8 compliance

# 13.82.2 Bug fixes

- Issue #1173 Add a comment to the existing node
- Issue #1925 Fix issue resizing column widths not resizable in Query Tool after first query
- Issue #2104 Runtime update display file version and copyright year under installers properties
- Issue #2249 Application no longer hangs after reload in runtime
- Issue #2251 Runtime fixed OSX html scroll direction ignored in MacOS setup
- Issue #2309 Allow text selection/copying from disabled CodeMirror instances
- Issue #2480 Runtime update fix to Context Menus on Mac that do not work
- Issue #2578 Runtime update fix to HTML access keys that don't work
- Issue #2581 Fix keyboard shortcut for text selection
- Issue #2677 Update Elephant icon for pgAdmin4 on Windows
- Issue #2776 Fix unreadable font via Remote Desktop
- Issue #2777 Fix spacing issue on server tree
- Issue #2783 Runtime update fixed blank screen on Windows Desktop
- Issue #2906 Correct display issues on HiDPI screens
- Issue #2961 Issues when creating a pgAgent Schedule
- Issue #2963 Fix unicode handling in the external process tools and show the complete command in the process viewer
- Issue #2980 Copy text from the Query tool into the clipboard adds invisible characters
- Issue #2981 Support keyboard navigation in the debugger
- Issue #2983 Fix intermittent specified\_version\_number ValueError issue on restart
- Issue #2985 Fix drag and drop issues
- Issue #2998 Don't listen on port 443 if TLS is not enabled when launching the container
- Issue #3001 Runtime update fix scrolling with mouse wheel on mac pgAdmin 4.2.1
- Issue #3002 Fix block indent/outdent with configurable width
- Issue #3003 Runtime update fix copy to clipboard
- Issue #3005 Runtime update fix unable to select tabs in pgAdmin 4.2.1
- Issue #3013 Fix a minor UI issue on dashboard while displaying subnode control in Backgrid
- Issue #3014 Fix validation of sequence parameters
- Issue #3015 Support Properties on Greenplum databases
- Issue #3016 Ensure debug messages are available in "messages" window when error occurs
- Issue #3021 Update scan and index scan EXPLAIN icons for greater clarity
- Issue #3027 Ensure we capture notices raised by queries

- Issue #3031 Runtime issue causing double and single quotes not to work
- Issue #3039 Runtime issue causing wrong row counts on count column
- Issue #3042 Runtime issue causing empty dialog box when refreshing
- Issue #3043 Runtime issue causing word sizing in macOS High Sierra
- Issue #3045 Runtime issue causing copy cells issues copying cells for key binding
- Issue #3046 Fix connection status indicator on IE/FF
- Issue #3050 Correct display of RE-SQL for partitioned tables in Greenplum
- Issue #3052 Don't include sizes on primitive data types that shouldn't have them when modifying columns
- Issue #3054 Ensure the user can use keyboard shortcuts after using button controls such as Cancel, Open and Save
- Issue #3057 Update the regression tests to fix issues with Python 3.5 and PG 9.2
- Issue #3058 Fix on-click handling of treeview nodes that wasn't refreshing SQL/Dependencies/Dependents in some circumstances
- Issue #3059 Fix table statistics for Greenplum
- Issue #3060 Fix quoting of function names in RE-SQL
- Issue #3066 Ensure column names on indexes on views are properly quoted in RE-SQL
- Issue #3067 Prevent the filter dialog CodeMirror from overflowing onto the button bar of the dialog
- Issue #3072 Add a (configurable) limit to the number of pgAgent job history rows displayed on the statistics tab
- Issue #3073 Ensure the pgAgent job start/end time grid fields synchronise with the subnode control and validate correctly
- Issue #3075 Runtime issue causing Select, Update, and Insert script generation for a table fails to load
- Issue #3077 Remove dependency on standards\_conforming\_strings being enabled
- Issue #3079 Fix handling of tie/datetime array types when adding columns to a table
- Issue #3080 Fix alignment issues in keyboard shortcut options
- Issue #3081 Add missing reverse-engineered SQL header and drop statement for sequences
- Issue #3090 Ensure message severity is decoded when necessary by the driver
- Issue #3094 Ensure all messages are retrieved from the server in the Query Tool
- Issue #3099 Fix creation of tables and columns in GPDB
- Issue #3105 Ensure we can properly update rows with upper-case primary key columns
- Issue #3135 Insert rows correctly when a table has OIDs and a Primary Key in uppercase
- Issue #3122 Ensure SSL options are pushed down to external tools like pg\_dump
- Issue #3129 Handle opening of non-UTF8 compatible files
- Issue #3137 Allow copying of SQL from the dashboard tables
- Issue #3138 Fix tablespace tests for Python 3.x
- Issue #3150 Fix function reserve SQL for GPDB
- Issue #3157 Fix unicode handling in the external process tools and show the complete command in the process viewer
- Issue #3171 Runtime issue causing inability to scroll in File Selector with trackpad on OSX
- Issue #3176 Disable function statistics on Greenplum
- Issue #3180 Ensure Indexes are displayed on PG 10 tables
- Issue #3190 Skip tests where appropriate on GPDB
- Issue #3196 Ensure the file manager properly escapes file & directory names
- Issue #3197 Appropriately set the cookie path
- Issue #3200 Ensure the host parameter is correctly pickup up from the service file
- Issue #3219 Update required ChromeDriver version for current versions of Chrome
- Issue #3226 Move the field error indicators in front of the affected fields so they don't obscure spinners or drop downs etc.
- Issue #3244 Show more granular timing info in the Query Tool history panel

13.82. Version 3.0 597

Issue #3248 - Ensure Alertify dialogues are modal to prevent them being closed by mis-click

# 13.83 Version 2.1

Release date: 2018-01-11

This release contains a number of features and fixes reported since the release of pgAdmin4 2.0

#### 13.83.1 Features

- Issue #1383 Allow connections to be coloured in the treeview and Query Tool
- Issue #1489 Improve user interface for selection query in Data Filter window
- Issue #2368 Improve data entry in Query Tool
- Issue #2781 Allow configuration of CSV and clipboard formatting of query results
- Issue #2802 Allow connections to be coloured in the treeview and Query Tool.
- Issue #2810 Allow files to be opened by double clicking on them within Query Tool
- Issue #2845 Make the "Save Changes" prompts in the Query Tool optional
- Issue #2849 Add support for editing data in tables with OIDs but no primary keys and updates the editor to retrieve all row values on save, thus immediately showing default values and allowing subsequent editing without a refresh

# 13.83.2 Bug fixes

- Issue #1365 Prevent the Windows installer accepting paths containing invalid characters
- Issue #1366 Fix /NOICONS switch in the windows installer
- Issue #1436 Fix issue with debugger which is failing for sub procedure on PPAS 9.6
- Issue #1749 Fixes in pgAgent module including; 1) allowing start date earlier than end date when scheduling job, 2)

Datetime picker not displaying in grid and 3) validation error not displaying propertly for Datetime control

- Issue #2094 Display relevant error messages when access is denied creating a schema
- Issue #2098 Cleanup some inconsistent error dialog titles
- Issue #2258 Fix handling of DATERANGE[] type
- Issue #2278 Display long names appropriately in dialogue headers
- Issue #2443 Confirm with the user before exiting the runtime
- Issue #2524 Fix debugging of self-referencing functions
- Issue #2566 Fix the Pause/Resume Replay of WAL files for PostgreSQL 10
- Issue #2624 Ensure the switch animation is consistent on the table dialogue and avoid displaying an error incorrectly
- Issue #2651 Ensure estimated rows are included correctly in CREATE script for functions
- Issue #2679 Getting started links does not open second time if User open any URL and Click on Close button with cross bar
- Issue #2705 User can add expirty date on Windows
- Issue #2715 Ensure we can download large files and keep the user informed about progress
- Issue #2720 Ensure password changes are successful if authenticating using a pgpass file
- Issue #2726 Ensure the auto-complete selection list can display longer names
- Issue #2738 Ensure line numbers form CodeMirror don't appear on top of menus
- Issue #2748 Format JSON/JSONB nicely when displaying it in the grid editor pop-up
- Issue #2760 When selecting an SSL cert or key, update only the expected path in the UI, not all of them

- Issue #2765 Do not decrypt the password when the password is 'None'. This should avoid the common but harmless exception "ValueError: IV must be 16 bytes long while decrypting the password."
- Issue #2768 Only allow specification of a pgpass file if libpq >= 10
- Issue #2769 Correct keyboard shortcut. Don't un-comment code with alt+. in the Query Tool. It's only supposed to respond to ctrl/cmd+
- Issue #2772 Remove external links from Panel's context menu
- Issue #2778 Ensure the datatype cache is updated when a domain is added
- Issue #2779 Ensure column collation isn't lost when changing field size
- Issue #2780 Ensure auto-indent honours the spaces/tabs config setting
- Issue #2782 Re-hash the way that we handle rendering of special types such as arrays
- Issue #2787 Quote the owner name when creating types
- Issue #2806 Attempt to decode database errors based on lc\_messages
- Issue #2811 Display process output as it happens
- Issue #2820 Logs available when executing backup and restore
- Issue #2821 Attempt to decode database errors based on lc\_messages
- Issue #2822 Re-hash the way that we handle rendering of special types such as arrays.
- Issue #2824 Fix a number of graphical explain rendering issues
- Issue #2836 Fix counted rows display in table properties
- Issue #2842 Fix a number of graphical explain rendering issues
- Issue #2846 Add an option to manually count rows in tables to render the properties
- Issue #2854 Fix utility output capture encoding
- Issue #2859 Allow form validation messages to be close in case the eclipse anything on the form
- Issue #2866 Ensure we don't show the full path on the server when using virtual filesystem roots in server mode for SSL certs
- Issue #2875 Ensure the scroll location is retains in the Query Tool data grid if the user changes tab and then returns
- Issue #2877 Remove the artificial limit of 4000 characters from text areas
- Issue #2880 Honour whitespace properly in the data grid
- Issue #2881 Fix support for time without timezone
- Issue #2886 Resolve issue where Insert failed when tried with default primary key value
- Issue #2891 Allow changing of the users password without leaving the app
- Issue #2892 Refuse password changes (and tell the user) if the notification email cannot be sent
- Issue #2908 Fix bundle creation on Windows which was failing due to rn line endings in code mirror
- Issue #2918 Add missing init.py to backports.csv when building the MSVC windows build
- Issue #2920 Push HTTPD logs to container stdout/stderr as appropriate
- Issue #2921 Fixes in pgAgent module including; 1) allowing start date earlier than end date when scheduling job, 2)
- Datetime picker not displaying in grid and 3) validation error not displaying propertly for Datetime control
- Issue #2922 Don't login the user with every request in desktop mode. Just do it once
- Issue #2923 Prevent the user pressing the select button in the file manager when it is supposed to be disabled
- Issue #2924 Cleanup the layout of the filter data dialogue
- Issue #2928 Prevent multiple connections to new slow-to-respond servers being initiated in error
- Issue #2934 Fix a reference before assignment error in the file dialogue
- Issue #2937 Prevent attempts to select directories as files in the file dialogue
- Issue #2945 Ensure invalid options can't be selected on triggers on views
- Issue #2949 Display complete SQL for FTS dictionaries
- Issue #2952 Don't try to render security URLs in desktop mode
- Issue #2954 Allow selection of validation error text
- Issue #2974 Clear the messages tab when running EXPLAIN/EXPLAIN ANALYZE

13.83. Version 2.1 599

Issue #2993 - Fix view data for views/mat views

## 13.84 Version 2.0

Release date: 2017-10-05

This release contains a number of features and fixes reported since the release of pgAdmin4 1.6

#### **13.84.1 Features**

- Issue #1918 Add a field to the Server Dialogue allowing users to specify a subset of databases they'd like to see in the treeview
- Issue #2135 Significantly speed up loading of the application
- Issue #2556 Allow for slow vs. fast connection failures
- Issue #2579 Default the file browser view to list, and make it configurable
- Issue #2597 Allow queries to be cancelled from the dashboard and display additional info in the subnode control
- Issue #2649 Support use of SSL certificates for authentication
- Issue #2650 Support use of pgpass files
- Issue #2662 Ship with pre-configured paths that can work in both Server and Desktop modes out of the box
- Issue #2689 Update icons with new designs and remove from menus to de-clutter the UI

# 13.84.2 Bug fixes

- Issue #1165 Prevent continual polling for graph data on the dashboard if the server is disconnected
- Issue #1697 Update CodeMirror version
- Issue #2043 Properly handle trigger functions with parameters
- Issue #2074 Make \$ quoting consistent
- Issue #2080 Fix issue where Browser hangs/crashes when loading data (using sql editor) from table which contains large blob data
- Issue #2153 Fix handline of large file uploads and properly show any errors that may occur
- Issue #2168 Update CodeMirror version
- Issue #2170 Support SSL in the regression tests
- Issue #2324 Fix PostGIS Datatypes in SQL tab, Create / Update dialogues for Table, Column, Foreign Table and Type node
- Issue #2447 Update CodeMirror version
- Issue #2452 Install pgadmin4-v1 1.5 on Centos7
- Issue #2501 Fix collation tests on Windows, replace use of default 'POSIX' collation with 'C' collation for testing
- Issue #2541 Fix issues using special keys on MacOS
- Issue #2544 Correct malformed query generated when using custom type
- Issue #2551 Show tablespace on partitions
- Issue #2555 Fix issue in Query Tool where messages were not displaying from functions/procedures properly
- Issue #2557 Tidy up tab styling
- Issue #2558 Prevent the tab bar being hidden when detached tabs are being closed
- Issue #2559 Stop tool buttons from changing their styling unexpectedly
- Issue #2560 Fix View 'CREATE Script' Problem
- Issue #2562 Update CodeMirror version

- Issue #2563 Fix paths under non-standard virtual directories
- Issue #2566 Fix Pause/Resume Replay of WAL files for PostgreSQL 10
- Issue #2567 Use the proper database connection to fetch the default priviledges in the properties tab of the database
- Issue #2582 Unset compression ratio if it is an empty string in Backup module
- Issue #2586 Cleanup feature tests
- Issue #2590 Allow navigation of query history using the arrow keys
- Issue #2592 Stop Flask from initialising service twice in Debug mode
- Issue #2593 Ensure babel-polyfill is loaded in older qWebKits
- Issue #2594 Fix disconnection of new databases
- Issue #2596 Define the proper NODE\_ENV environment during running the webpack
- Issue #2606 Ensure role names are escaped in the membership control
- Issue #2616 Domain create dialog do not open and Font size issue in Security label control
- Issue #2617 Add missing pgagent file in webpack.config.js
- Issue #2619 Fix quoting of index column names on tables
- Issue #2620 Set database name to blank(") when job type is set to batch, while creating pgAgent job
- Issue #2631 Change mapping of cell from 'numeric' to 'integer' for integer control as numeric cell has been removed from the code
- Issue #2633 Fix pgAgent job step issues
- Issue #2634 Add New Server through Quick links
- Issue #2637 Fix Copy so it still works after query results have been copied
- Issue #2641 User management issues styling and inability to edit users properly
- Issue #2644 Fix alertify notification messages where checkmark box disconnected from frame
- Issue #2646 Fix the path reference of load-node.gif which was referencing to vendor directory
- Issue #2654 Update datetime picker
- Issue #2655 Fix connection string validation for pgAgent jobs
- Issue #2656 Change Datetimepicker to expand from bottom in pgAgent so calendar does not get hidden
- Issue #2657 Fix syntax error while saving changes for start/end time, weekdays, month, hours, minutes while updating the pgAgent Job
- Issue #2659 Fix issue where unable to add/update variables for columns of a table
- Issue #2660 Not able to select rows in History Tab
- Issue #2668 Fix RE-SQL for triggers with a single arg
- Issue #2670 Improve datamodel validations for default Validator if user (developer) does not implement validate function in datamodel
- Issue #2671 Fix array data type formating for bigint, real, float, double precision
- Issue #2681 Reset Query Tool options before running tests
- Issue #2684 Fix layout of password prompt dialogue
- Issue #2691 View data option is missing from pgAdmin4 2.0 version
- Issue #2692 Base type is missing for Domain on pgAdmin4
- Issue #2693 User list is not available on User mapping pgAdmin4
- Issue #2698 User can not create function due to missing return type
- Issue #2699 Filtered Rows issue on pgAdmin4
- Issue #2700 Cancel button is visible after query executed successfully
- Issue #2707 Disable trigger button does not work on pgAdmin4
- Issue #2708 Tablespace name should displayed instead of %s(new\_tablespace)s with Move Objects to another tablespace
- Issue #2709 Display user relations in schema prefixed by 'pg'
- Issue #2713 Fix an exception seen sometimes when the server is restarted

13.84. Version 2.0 601

Issue #2742 - Ensure using an alternate role to connect to a database doesn't cause an error when checking recovery state.

#### 13.85 Version 1.6

Release date: 2017-07-13

This release contains a number of features and fixes reported since the release of pgAdmin4 1.5

#### 13.85.1 Features

- Issue #1344 Allow the Query Tool, Debugger and web browser tabs to be moved to different monitors as desired
- Issue #1533 Set focus on the first enabled field when a dialogue is opened
- Issue #1535 Teach dialogues about Escape to cancel, Enter to Save/OK, and F1 for help
- Issue #1971 Retain column sizing in the Query Tool results grid when the same query is re-run multiple times in a row
- Issue #1972 Prompt the user to save dirty queries rather than discard them for a more natural workflow
- Issue #2137 On-demand loading for the Query Tool results
- Issue #2191 Add support for the hostaddr connection parameter. This helps us play nicely with Kerberos/SSPI and friends
- Issue #2282 Overhaul the query history tab to allow browsing of the history and full query text
- Issue #2379 Support inserting multiple new rows into a table without clicking Save for each row
- Issue #2485 Add a shortcut to reset the zoom level in the runtime
- Issue #2506 Allow the user to close the dashboard panel
- Issue #2513 Add preferences to enable brace matching and brace closing in the SQL editors

# 13.85.2 Bug fixes

- Issue #1126 Various FTS dictionary cleanups
- Issue #1229 Fix default values and SQL formatting for event triggers
- Issue #1466 Prevent attempts to debug procedures with variadic arguments
- Issue #1525 Make \$ quoting consistent
- Issue #1575 Properly display security labels on EPAS 9.2+
- Issue #1795 Fix validation for external and range types
- Issue #1813 List packages in PPAS 9.2-9.4 when creating synonyms
- Issue #1831 Fix server stats display for EPAS 9.2, where inet needs casting to text for concatenation
- Issue #1851 Reverse engineer SQL for table-returning functions correctly
- Issue #1860 Ensure default values are honoured when adding/editing columns
- Issue #1888 Fix various issues with pgAgent job steps and schedules
- Issue #1889 Fix various issues with pgAgent job steps and schedules
- Issue #1890 Fix various issues with pgAgent job steps and schedules
- Issue #1920 Ensure saved passwords are effective immediately, not just following a restart when first saved
- Issue #1928 Fix the handling of double precision[] type
- Issue #1934 Fix import/export to work as expected with TSV data
- Issue #1999 Handle warning correctly when saving query results to an unmounted USB drive

- Issue #2013 Increase the default size of the Grant Wizard to enable it to properly display privileges at the default size on smaller displays
- Issue #2014 To fix unexpected behaviour displayed if user stops debugging on package/procedure fire\_emp
- Issue #2043 Properly handle trigger functions with parameters
- Issue #2078 Refresh the SQL editor view on resize to ensure the contents are re-rendered for the new viewport
- Issue #2086 Allow editing of the WITH ADMIN option of role membership
- Issue #2113 Correct the validation logic when modifying indexes/exclusion constraints
- Issue #2116 Enable dialogue help buttons on Language and Foreign Table dialogues
- Issue #2142 Fix canceling of Grant Wizard on Windows
- Issue #2155 Fix removal of sizes from column definitions
- Issue #2162 Allow non-superusers to debug their own functions and prevent them from setting global breakpoints
- Issue #2242 Fix an issue in NodeAjaxControl caching with cache-node field and add cache-node field in Trigger & Event trigger node so that whenever the user creates new Trigger Function we get new data from server in NodeAjaxControl
- Issue #2280 Handle procedure flags (IMMUTABLE STRICT SECURITY DEFINER PARALLEL RESTRICTED) properly in RE-SQL on EPAS
- Issue #2324 Fix the PostGIS Datatypes in SQL tab, Create / Update dialogues for Table, Column, Foreign Table and Type node
- Issue #2344 Fix issue with ctrl-c / ctrl-v not working in Query Tool
- Issue #2348 Fix issue when resizing columns in Query Too/View Data where all row/colums will select/deselect
- Issue #2355 Properly refresh the parent node when renaming children
- Issue #2357 Cache statistics more reliably
- Issue #2381 Fix the RE-SQL for for views to properly qualify trigger function names
- Issue #2386 Display and allow toggling of trigger enable/disable status from the trigger dialogue
- Issue #2398 Bypass the proxy server for local addresses on Windows
- Issue #2400 Cleanup handling of default/null values when data editing
- Issue #2414 Improve error handling in cases where the user tries to rename or create a server group that would duplicate an existing group
- Issue #2417 Order columns in multi-column pkeys correctly
- Issue #2422 Fix RE-SQL for rules which got the table name wrong in the header and DROP statement
- Issue #2425 Handle composite primary keys correctly when deleting rows in the Edit Grid
- Issue #2426 Allow creation of ENUM types with no members
- Issue #2427 Add numerous missing checks to ensure objects really exist when we think they do
- Issue #2435 Pass the database ID to the Query Tool when using the Script options
- Issue #2436 Ensure the last placeholder is included when generating UPDATE scripts for tables
- Issue #2448 Ensure that boolean checkboxes cycle values in the correct order
- Issue #2450 Fix error on the stats tab with PG10. Also, rename the 10.0\_plus template directory to 10\_plus to match the new versioning
- Issue #2461 Allow users to remove default values from columns properly
- Issue #2468 Fix issue where function create script won't compile
- Issue #2470 Fix an intermittent error seen during result polling
- Issue #2476 Improvements to the Query Results grid including improvements to the UI and allow copy/paste from sets of rows, columns or arbitrary blocks of cells
- Issue #2477 Ensure text editors render in an appropriate place on the results grid
- Issue #2479 No need for the menu icon to link to the homepage, as pgAdmin is a SPA
- Issue #2482 Use a more sensible name for Query Tool tabs
- Issue #2486 Ensure the feature tests use the correct test settings database
- Issue #2487 Maintain a client-side cache of preference values, populated using an async call

13.85. Version 1.6 603

- Issue #2489 Fix clipboard handling with large datasets
- Issue #2492 Ensure the initial password is properly hashed during setup in web mode
- Issue #2498 Properly handle bytea[], and 'infinity'::real/real[]
- Issue #2502 Properly handle bytea[], and 'infinity'::real/real[]
- Issue #2503 Handle missing/dropped synonyms gracefully
- Issue #2504 Update MatView and pgAgent modules to work with recent integer/numeric changes
- Issue #2507 Ensure revoked public privileges are displayed in the RE-SQL for functions
- Issue #2518 Fix encoding issue when saving servers
- Issue #2522 Improve speed of Select All in the results grid
- Issue #2527 Fix deletion of table rows with the column definition having NOT NULL TRUE and HAS NO

#### **DEFAULT VALUE**

- Issue #2528 Allow breakpoints to be set on triggers on views
- Issue #2529 Resolve a number of issues with domains and domain constraints
- Issue #2532 Refresh nodes correctly when there is a single child that is updated
- Issue #2534 Fix handling of CREATE TABLE OF <type>
- Issue #2535 Fix clear history functionality
- Issue #2540 Ensure the save password option is enabled when creating a server

## 13.86 Version 1.5

Release date: 2017-05-19

This release contains a number of features and fixes reported since the release of pgAdmin4 1.4.

#### 13.86.1 Features

Issue #2216 - Allow column or row selection in the Query Tool

# 13.86.2 Bug fixes

- Issue #2225 Hide menu options for creating objects, if the object type is set to hidden. Includes Jasmine tests
- Issue #2253 Fix various issues in CSV file download feature
- Issue #2257 Improve handling of nulls and default values in the data editor
- Issue #2271 Don't change the trigger icon back to "enabled" when the trigger is updated when it's disabled
- Issue #2284 Allow creation of tables with pure numeric names
- Issue #2292 Only reconnect to databases that were previously connected
- Issue #2314 Fix various issues in CSV file download feature
- Issue #2315 Fix sorting of sizes on the statistics views by sorting raw values and prettifying on the client side.

Includes Jasmine tests for the prettyfying function

- Issue #2318 Order foreign table columns correctly
- Issue #2331 Fix binary search algorithm so new treeview nodes are added in the correct position
- Issue #2336 Update inode info when refreshing treeview nodes.
- Issue #2339 Ensure the treeview can be scrolled horizontally
- Issue #2350 Fix handling of default parameters ordering in functions
- Issue #2354 Fix the Backup module where it was not working if user changes its preference language other than English

- Issue #2356 Ensure errors thrown when deleting rows in the Query Tool in edit mode are shown properly
- Issue #2360 Fix various issues in CSV file download feature
- Issue #2369 Support loading files with Unicode BOMs
- Issue #2377 Update psycopg2 version for PostgreSQL 10 compatibility
- Issue #2379 Make various improvements to the NULL/DEFAULT handling in the data editor
- Issue #2405 Ensure object names are properly escaped for external process management
- Issue #2410 Fix PostgreSQL 10.0 compatibility issues

## 13.87 Version 1.4

Release date: 2017-04-13

This release contains a number of features and fixes reported since the release of pgAdmin4 1.3.

#### 13.87.1 Features

- Issue #2232 Add the ability to gray-out/disable the "Save Password" option when creating a connection to a server
- Issue #2261 Display table DDL for Greenplum in SQL tab
- Issue #2320 Added German translation

## 13.87.2 Bug fixes

- Issue #2077 Add missing "Run Now" option for pgAdmin jobs
- Issue #2105 Fix validation on the table dialogue so the Save button isn't enabled if the name is removed and autovac custom settings are enabled
- Issue #2145 Resolve the issue for restoring the table from the backup
- Issue #2187 Ensure the web/ directory is cleared before upgrading Windows installations
- Issue #2190 Allow users to select UI language at login or from Preferences rather than unpredictable behaviour from browsers
- Issue #2226 Show tooltips for disabled buttons to help user learning
- Issue #2241 Fix numeric control validation in nested schemas
- Issue #2243 Fix dropping of databases with Unicode names
- Issue #2244 Prevent an error being displayed if the user views data on a table with no columns
- Issue #2246 Add missing braces to reverse engineered SQL header block for Functions
- Issue #2258 Fix handling of DATERANGE[] type
- Issue #2264 Resolve error message ExtDeprecationWarning displayed on new pgAdmin4 setup for Python 3.4 on ubuntu 14.04 Linux 64
- Issue #2265 Resolved import/Export issue for a table
- Issue #2274 Properly handle truncated table names
- Issue #2277 Resolved various file-system encoding/decoding related cases
- Issue #2281 Ensure menus are updated after disconnecting a server
- Issue #2283 Check if cell is in multiselect mode before setting default selection of multiple values
- Issue #2287 Properly handle EXPLAIN queries entered directly by the user in the Query Tool
- Issue #2291 Fix error highlighting in the Query Tool
- Issue #2299 Fix usage of QString
- Issue #2303 Fix ascending/descending sort order in backgrid while clicking on the headers

13.87. Version 1.4 605

- Issue #2304 Resolve the issue for restoring the table from the backup
- Issue #2305 Resolve the issue where Generic function qtLiteral was not adapting values properly when they contain non ascii characters
- Issue #2310 Fix Dialog Help where Query Tool/Debugger opens in new browser tab
- Issue #2319 Resolve issue where Click on pgAdmin4 logo leads to unauthorized error
- Issue #2321 Improved functionality of browser tree when adding new nodes if parent collection node has not loaded
- Issue #2330 Ensure the Query Tool displays but does not render HTML returned by the server in the results grid

#### 13.88 Version 1.3

Release date: 2017-03-10

This release contains a number of features and fixes reported since the release of pgAdmin4 1.2.

#### 13.88.1 Features

- Issue #2036 Query tool efficiency SlickGrid result set format efficiency
- Issue #2038 Query tool efficiency Incremental back off when polling
- Issue #2163 Make syntax highlighting more visible
- Issue #2210 Build a universal Python wheel instead of per-python-version ones
- Issue #2215 Improve visibility of syntax highlighting colours

# 13.88.2 Bug fixes

- Issue #1796 Add missing "Run Now" option for pgAdmin jobs
- Issue #1797 Resolve encoding issues with DATA\_DIR
- Issue #1914 Resolved error utf8' codec can't decode byte
- Issue #1983 Fix bug in Sql query contains Arabic Charaters
- Issue #2089 Add PARALLEL SAFE|UNSAFE|RESTRICTED support
- Issue #2115 Fix exclusion constraint reverse engineered SQL
- Issue #2119 Fix display of long integers and decimals
- Issue #2126 Correct node labels in Preferences for EDB functions and procedures
- Issue #2151 Display un-sized varlen column types correctly in the Query Tool
- Issue #2154 Fix display of long integers and decimals
- Issue #2159 Resolve issue where Query editor is not working with Python2.6
- Issue #2160 Various encoding fixes to allow 'ascii' codec to decode byte 0xc3 in position 66: ordinal not in range(128)
- Issue #2166 Resolved import/Export issue for a table
- Issue #2173 Resolved issues where Sequences API test cases are not working in PG9.2 and PPAS9.2
- Issue #2174 Resolved various file-system encoding/decoding related cases
- Issue #2185 Removed sorting columns on the treeview
- Issue #2192 Fix startup complete tests to ensure we properly poll the server for completed startup
- Issue #2198 Fix function arguments when generating create SQL
- Issue #2200 Properly handle event trigger functions in different schemas
- Issue #2201 Fix renaming of check constraints when the table name is changed at the same time
- Issue #2202 Fix issue where Dependents query fails due to non ascii characters

- Issue #2204 Fixed issue where pgadmin 4 jobs not showing any activity
- Issue #2205 Fix display of boolean nulls in the Query Tool
- Issue #2208 Ensure primary key column names are quoted in View Data mode of the Query Tool
- Issue #2212 Ensure servers are deleted when their parent group is deleted
- Issue #2213 Enable right click on browser tree
- Issue #2218 Show the correct indeterminate state when editing new boolean values
- Issue #2228 Authenticate the runtime to the server
- Issue #2230 Prevent the Slonik logo obscuring the login dialogue on small displays in server mode

## 13.89 Version 1.2

Release date: 2017-02-10

This release contains a number of features and fixes reported since the release of pgAdmin4 1.1.

#### 13.89.1 Features

- Issue #1375 Migrate the runtime to QtWebEngine from QtWebKit
- Issue #1765 Find and replace functionality with regexp and group replacement
- Issue #1789 Column width of data output panel should fit to data (as pgAdmin III)
- Issue #1790 [Web] Support setting a field's value to "null"
- Issue #1848 macOS applundle is missing postgresql binaries for import etc.
- Issue #1910 Remember last used directory in the file manager
- Issue #1911 Direct path navigation in the file manager
- Issue #1922 Improve handling of corrupt configuration databases
- Issue #1963 Add a Chinese (Simplified) translation
- Issue #1964 Create a docs tarball along with the source tarball
- Issue #2025 Allow the SQL Editors to word-wrap
- Issue #2124 Create a template loader to simplify SQL template location, and remove duplicate templates

#### 13.89.2 Bug fixes

- Issue #1227 Display improved error message for Debugger listener starting error and reset between executions
- Issue #1267 Fix issue where MINIFY\_HTML doesn't work with the docs
- Issue #1364 Ensure dialogue control buttons are consistent
- Issue #1394 Fix Table dialogue column specification issues
- Issue #1432 Enhanced OSX File Browser
- Issue #1585 Cannot save scripts to the network
- Issue #1599 Ensure the grant wizard works with objects with special characters in the name
- Issue #1603 Fix quoting of objects names for external utilities.
- Issue #1679 Re-engineer the background process executor to avoid using sqlite as some builds of components it relies on do not support working in forked children
- Issue #1680 Render column headers at the correct width in the Query Tool under Firefox
- Issue #1729 Improve display of role options
- Issue #1730 Improve the display of role membership on both the properties panel and role dialogue
- Issue #1745 Ensure breakpoints are cleared properly when working with Debugger

13.89. Version 1.2 607

- Issue #1747 Add newly created triggers to the treeview
- Issue #1780 Properly size the SQL Editor gutter as the width of the line numbers increases
- Issue #1792 List files and folders alphabetically
- Issue #1800 Handle the template property on databases appropriately
- Issue #1801 Handle databases with datallowconn == false
- Issue #1807 Properly detect when files have changed in the Query Tool and set flag accordingly
- Issue #1830 Fix a SQL error when reverse-engineering ROLE SQL on EPAS servers
- Issue #1832 Prevent attempts to access what may be an empty list in Dependancies tab
- Issue #1840 Enable/disable NULLs and ASC/DESC options for index columns and exclusion constraints appropriately
- Issue #1842 Show index columns in the correct order in RE-SQL
- Issue~#1855 Ensure~dialogue~panels~show~their~errors~themselves,~and~not~in~the~properties~panel~when~creating~Trigger~Function
- Issue #1865 Properly schema qualify domains when reverse engineering SQL
- Issue #1874 Add file resources to the windows runtime
- Issue #1893 Fix refreshing of Unique constraints
- Issue #1896 Use the correct OID for retrieving properties of freshly created exclusion constraints
- Issue #1899 Properly quote role names when specifying function ownership
- Issue #1909 Handle startup errors more gracefully in the runtime
- Issue #1912 Properly format arguments passed by triggers to functions
- Issue #1919 Ensure all changes to rows are stored in the data editor
- Issue #1924 Ensure the check\_option is only set when editing views when appropriate
- Issue #1936 Don't strip rn from "Download as CSV" batches of rows, as it leads to malformed data
- Issue #1937 Generate mSQL for new schemas correctly
- Issue #1938 Fix sorting of numerics in the statistics grids
- Issue #1939 Updated dynamic default for the window size (90% x 90%)
- Issue #1949 Ensure trigger function names are schema qualified in trigger RE-SQL
- Issue #1951 Fix issue where nnable to browse table columns when oid values exceed max int
- Issue #1953 Add display messages and notices received in the Query Tool
- Issue #1961 Fix upgrade check on Python 3
- Issue #1962 Ensure treeview collection nodes are translated in the UI
- Issue #1967 Store layout changes on each adjustment
- Issue #1976 Prevent users selecting elements of the UI that shouldn't be selectable
- Issue #1979 Deal with Function arguments correctly in the properties dialogue
- Issue #1986 Fix various encoding issues with multibyte paths and filenames resulting in empty file save
- Issue #1992 Quote identifiers correctly in auto-complete
- Issue #1994 Update to show modifications in edit grid
- Issue #2000 Allow setting of effective\_io\_concurrency on tablespaces in 9.6+
- Issue #2005 Fix various mis-spellings of VACUUM
- Issue #2006 Fix error when modifying table name or set schema on tables with postgis geometry column
- Issue #2007 Correctly sort rows by the pkey when viewing first/last 100
- Issue #2009 Reset the column list properly if the access method is changed on an index to ensure error handling works correctly
- Issue #2012 Prevent attempts to create server groups with no name
- Issue #2015 Enable trigger option when user tries to change Row trigger value through properties section
- Issue #2024 Properly handle setting comments and other options on databases with allowconn = False
- Issue #2026 Improve detection of the pldbgapi extension and functions before allowing debugging

- Issue #2027 Fix inconsistent table styling
- Issue #2028 Fix display of double scrollbars on the grant wizard
- Issue #2032 Fix time formatting on dashboards
- Issue #2033 Show icons for unique and exclusion constraints in the dependency/dependents panels
- Issue #2045 Update copyright year on doc page
- Issue #2046 Fix error when setting up regression on Windows for pgadmin4
- Issue #2047 Ensure dialogues cannot be moved under the navbar
- Issue #2061 Enable/disable NULLs and ASC/DESC options for index columns and exclusion constraints
- appropriately
- Issue #2065 Improve display of columns of exclusion contraints and foreign keys in the properties lists
- Issue #2069 Correct tablespace displayed in table properties
- Issue #2076 Handle sized time/timestamp columns correctly
- Issue #2109 Update copyright year
- Issue #2110 Handle saved directories that no longer exist gracefully
- Issue #2112 Enable comments on Initial database through right Click
- Issue #2133 Fix display of graphical query plans for UPDATE/DELETE queries
- Issue #2138 Fix display of zeros in read-only grid editors
- Issue #2139 Fixed issue causing Message (Connection to the server has been lost.) displayed with Materialized view and view under sql tab
- Issue #2152 Fix handling of "char" columns
- Issue #2156 Added compatibility fixes for newer versions of Jinja2 (e.g. 2.9.5+)

# 13.90 Version 1.1

Release date: 2016-10-27

This release contains a number of features and fixes reported since the release of pgAdmin4 1.0;

#### **13.90.1 Features**

- Issue #1328 Add Python 3.5 Support
- Issue #1859 Include wait information on the activity tab of the dashboards

#### 13.90.2 Bug fixes

- Issue #1155 Display the start value when the user creates sequence
- Issue #1531 Fix to update privileges for Views and Materials Views where "string indices must be integers error" displayed
- Issue #1574 Display SQL in SQL pane for security label in PG and EPAS server
- Issue #1576 Make security label option available in procedure properties
- Issue #1577 Make debug option available for package function and procedure
- Issue #1596 Correct spelling error from evnt\_turncate to evnt\_truncate
- Issue #1599 Ensure the grant wizard works with objects with special characters in the name
- Issue #1622 Fix issue using special characters when creating synonym
- Issue #1728 Properties refreshing after objects are edited
- Issue #1739 Prevent the user from trying to.....

13.90. Version 1.1 609

- Issue #1785 Correctly identify server type upon first connection
- Issue #1786 Ensure errorModel unset property is set correctly when adding a new server
- Issue #1808 Set seconds to valid value in pgAgent job schedule
- Issue #1817 Display message "server does not support ssl" if server with ca-cert or ca-full added
- Issue #1821 Optionally sign both the Mac app bundle and the disk image
- Issue #1822 Handle non-ascii responses from the server when connecting
- Issue #1823 Attempt to sign the Windows installer, failing with a warning if there's no cert available
- Issue #1824 Add documenation for pgAgent
- Issue #1835 Allow users to choose SELECT permissions for tables and sequences in the grant wizard
- Issue #1837 Fix refreshing of FTS dictionaries which was causing error "Connection to the server has been lost"
- Issue #1838 Don't append new objects with the wrong parent in tree browser if the correct one isn't loaded
- Issue #1843 Function definition matches value returned from pg\_get\_functiondef()
- Issue #1845 Allow refreshing synonym node. Does not display message "Unimplemented method (node) for this url (/browser/synonym/nodes/1/7/14301/2200/test)"
- Issue #1847 Identify the collation correctly when reverse engineering table SQL. ERROR: schema "default" does not exist no longer displayed
- Issue #1849 Remove security keys from config.py/config\_local.py
- Issue #1857 Fix error while renaming FTS dictionary and FTS template nodes
- Issue #1858 Ensure the File Manager honours the file type while traversing the directories.
- Issue #1861 Properly generate exclusion constraint SQL.
- Issue #1863 Correctly quote type names in reverse engineered SQL for tables
- Issue #1864 Fix layout of DateTimePicker control help message.
- Issue #1867 Allow package bodies to be dropped.
- Issue #1868 Resolved issue where Integer type of preferences are not updated
- Issue #1872 Fix the file manager when used under Python 3.
- Issue #1877 Ensure preferences values are stored properly.
- Issue #1878 Ensure steps and schedules can be created in empty jobs. ProgrammingError: can't adapt type
- 'Undefined' was displayed
- Issue #1880 Add new indexes to the correct parent on the treeview.

# 13.91 Version 1.0

Release date: 2016-09-29

The first major release of pgAdmin 4. With a more modern look and feel, this release includes the following features;

- Multiplatform
- · Designed for multiple PostgreSQL versions and derivatives
- Extensive documentation
- Multiple deployment models
- Tools
- · Routine maintenance
- Create, view and edit all common PostgreSQL objects
- · Multibyte support

# CHAPTER 14

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

#### pgAdmin 4

Copyright (C) 2013 - 2024, The pgAdmin Development Team

Permission to use, copy, modify, and distribute this software and its documentation for any purpose, without fee, and without a written agreement is hereby granted, provided that the above copyright notice and this paragraph and the following two paragraphs appear in all copies.

IN NO EVENT SHALL THE PGADMIN DEVELOPMENT TEAM BE LIABLE TO ANY PARTY FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING LOST PROFITS, ARISING OUT OF THE USE OF THIS SOFTWARE AND ITS DOCUMENTATION, EVEN IF THE PGADMIN DEVELOPMENT TEAM HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

THE PGADMIN DEVELOPMENT TEAM SPECIFICALLY DISCLAIMS ANY WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE SOFTWARE PROVIDED HEREUNDER IS ON AN "AS IS" BASIS, AND THE PGADMIN DEVELOPMENT TEAM HAS NO OBLIGATIONS TO PROVIDE MAINTENANCE, SUPPORT, UPDATES, ENHANCEMENTS, OR MODIFICATIONS.

**Note:** PostgreS, PostgreSQL and the Slonik Logo are trademarks or registered trademarks of the PostgreSQL Community Association of Canada, and used with their permission.

612 Chapter 14. Licence

A	E
Add Named Restore Point Dialog, 377	EDB BigAnimal Cloud Deployment, 119
Amazon RDS Cloud Deployment, 114	Enabling Kerberos Authentication, 52
Azure Database Cloud Deployment, 123	Enabling LDAP Authentication, 50
<b>D</b>	Enabling OAUTH2 Authentication, 54
В	Enabling two-factor authentication (2FA), 41
Backup and Restore, 395	Enabling Webserver Authentication, 56
Backup Dialog, 395	ERD Tool, 457
Backup Globals Dialog, 402	Event Trigger Dialog, 186
Backup Server Dialog, 404	Exclusion Constraint Dialog, 320
	Extension Dialog, 190
C	External database for pgAdmin user settings,
Cast Dialog, 171	105
Change Ownership Dialog, 47	F
Change Password Dialog, 378	Г
Change User Password Dialog, 48	Foreign Data Wrapper Dialog, 192
Check Dialog, 303	Foreign key Dialog, 323
Clear Saved Passwords, 113	Foreign Server Dialog, 198
Code Overview, 488	Foreign Table Dialog, 203
Code Review Notes, 506	FTS Configuration Dialog, 210
Coding Standards, 492	FTS Dictionary Dialog, 213
Collation Dialog, 174	FTS Parser Dialog, 218
Column Dialog, 306	FTS Template Dialog, 221
Compound Trigger Dialog, 315	Function Dialog, 224
Connect to Server, 138	G
Connecting To A Server, 107	-
Connection Error, 139	Getting Started, 3
Container Deployment, 32	Google Cloud SQL Deployment, 130
Creating a pgAgent Job, 476	Grant Wizard, 379
Creating or Modifying a Table, 303	1
D	
D	Import/Export Data Dialog, 381
Database Dialog, 147	Import/Export Servers, 141
Debugger, 415	Index Dialog, 330
Deployment, 4	Installing pgAgent,474
Desktop Deployment, 22	V
Developer Tools, 415	K
Domain Constraints Dialog, 182	Keyboard Shortcuts, 100
Domain Dialog, 178	

Tabbed Browser, 65

L Language Dialog, 231 Licence, 611 Lock/Restore Account, 49 Login Page, 40 Login/Group Role Dialog, 155	Table Dialog, 347 Tablespace Dialog, 162 The config.py File, 4 Toolbar, 64 Translations, 507 Tree Control, 73 Trigger Dialog, 368 Trigger Function Dialog, 276
Maintenance Dialog, 386 Management Basics, 377 Managing Cluster Objects, 147 Managing Database Objects, 171 Master Password, 136 Materialized View Dialog, 235 Menu Bar, 59	Type Dialog, 285  U Unique Constraint Dialog, 374 User Interface, 58 User Management Dialog, 42 User Mapping Dialog, 294
P	Using pgAgent, 473
Package Dialog, 239 pgAdmin Project Contributions, 487 pgAgent, 473 PostgreSQL Cloud Deployment, 113 Preferences Dialog, 76 Primary key Dialog, 334 Procedure Dialog, 244 Processes, 469 PSQL Tool, 466 Publication Dialog, 252	V View Dialog, 297 View/Edit Data, 446 View/Edit Data Filter, 451
Query Tool, 421 Query Tool Toolbar, 422	
R	
Resource Group Dialog, 153 Restore Dialog, 409 RLS Policy Dialog, 337 Role Reassign/Drop Own Dialog, 167 Rule Dialog, 341	
S	
Schema Dialog, 256 Schema Diff, 452 Search objects, 103 Sequence Dialog, 261 Server Deployment, 27 Server Dialog, 108 Server Group Dialog, 107 Storage Manager, 389 Submitting Pull Requests, 487 Subscription Dialog, 266 Synonym Dialog, 273	

614 Index