

DBeaver User Guide v.23.2

User Guide

Table of contents

DBeaver Desktop Documentation

General User Guide

- Installation
- Application Window Overview
- Views
  - Database Navigator
    - Filter Database Objects
    - Configure Filters
    - Simple and Advanced View
  - Projects View
  - Project Explorer
  - Query Manager
  - Background Tasks
- Database Object Editor
  - Data Editor
    - Navigation
    - Data View and Format
    - Data Filters
    - Data Refresh
    - Data Viewing and Editing

Panels

Managing Charts

Data Search

Data transfer

SQL Generation

Working with spatial/GIS data

Working with XML and JSON

Managing Data Formats

Virtual column expressions

Properties Editor

ER Diagrams

Database Structure Diagrams

Custom Diagrams

Edit mode

SQL Editor

Toolbar Customization

SQL Templates

SQL Assist and Auto-Complete

AI SQL Assistance (ChatGPT)

SQL Formatting

SQL Execution

SQL Terminal

Variables panel

Query Execution Plan

Visual Query Builder

Script Management

Client Side Commands

Export Command

Debug

PostgreSQL Debugger

Search

File Search

DB Full-Text Search

DB Metadata Search

Schema compare

Using Liquibase in DBeaver

Data compare

MockData generation

Spelling

Dashboards, DB monitoring

Projects

Project security

Team work (Git)

Security in PRO products

Bookmarks

Shortcuts

Accessibility

Database Management

Sample Database

Database Connections

Create Connection

Edit Connection

Connect to Database

Invalidate/Reconnect to Database

Disconnect from Database

Change current user password

Advanced settings

SSH Configuration

Proxy configuration

Kubernetes configuration

Kerberos authentication

Connection Types

Tableau integration

Transactions

Auto and Manual Commit Modes

Transaction Log

Pending transactions

Drivers

Database drivers

JDBC-ODBC bridge

Tasks

Data export/import

Data migration

Data Import and Replace

Database backup/restore

Task management

Task scheduler

Composite tasks

Sending results by e-mail

Uploading result to external storage

Cloud Explorer

AWS

AWS Credentials

AWS SSO

AWS Permissions

GCP Credentials

GCP SSO

Azure

DBeaver PRO

Enterprise Edition

Databases support

Classic

Apache Hive/Spark/Impala

Cassandra

Couchbase

InfluxDB

MongoDB

MySQL

Oracle

PostgreSQL

Redis

Salesforce

Cloud

Bigtable

DocumentDB

DynamoDB

Keyspaces

Snowflake

Customizing DBeaver

Changing interface language

Installing extensions - Themes, version control, etc

User Interface Themes

Troubleshooting

Command Line

Reset UI settings

Reset workspace

Troubleshooting system issues

Posting issues

Log files

JDBC trace

Thread dump

Admin Guide

- Managing connections
- Managing variables
- Managing drivers
- Managing preferences
- Windows Silent Install

License management

- License Administration
- How to Import License
- How to Reassign License

Tutorials

- Connecting to Oracle Database using JDBC OCI driver
- Importing CA certificates from your local Java into DBeaver
- SSL configuration
- New Table Creation
  - Creating columns
  - Implementing Constraints
  - Utilizing Foreign-Keys
  - Creating Indexes
  - Incorporating Triggers

Installation

The installation process depends on the distribution type and your Operational System.

Windows / MacOS Installer

The installer distribution is the recommended way to install DBeaver in Windows and MacOS X. It contains all required dependencies. In addition, the installer automatically upgrades DBeaver to the new version if a previous version has already been installed. To install DBeaver, run the installer executable and follow the instructions on its screens.

NOTE:

- The installer does not change any system settings or the Java installation.
- The included JDK will be accessible only to DBeaver.

ZIP Archive

When installing DBeaver manually, without using an installer:

1. Extract the contents of the archive.

   NOTE: Do not unzip the archive over a previous DBeaver version. If you already have any version of DBeaver extracted in the same location - remove it before unzipping the new version.

   NOTE: All configurations, scripts and other necessary data are stored in a separate location (usually in the user`s home directory) so the program deinstallation does not affect them.

2. Run the **dbeaver** executable.

Debian Package

To install DBeaver using a Debian package:

1. Run `sudo dpkg -i dbeaver-<version>.deb`.

2. Execute `dbeaver &`.

RPM Package

To install DBeaver using RPM package:

1. Run `sudo rpm -ivh dbeaver-<version>.rpm`.
2. Execute `dbeaver &`.

NOTE: To upgrade DBeaver to the next version, use `sudo rpm -Uvh dbeaver-<version>.rpm` parameter.

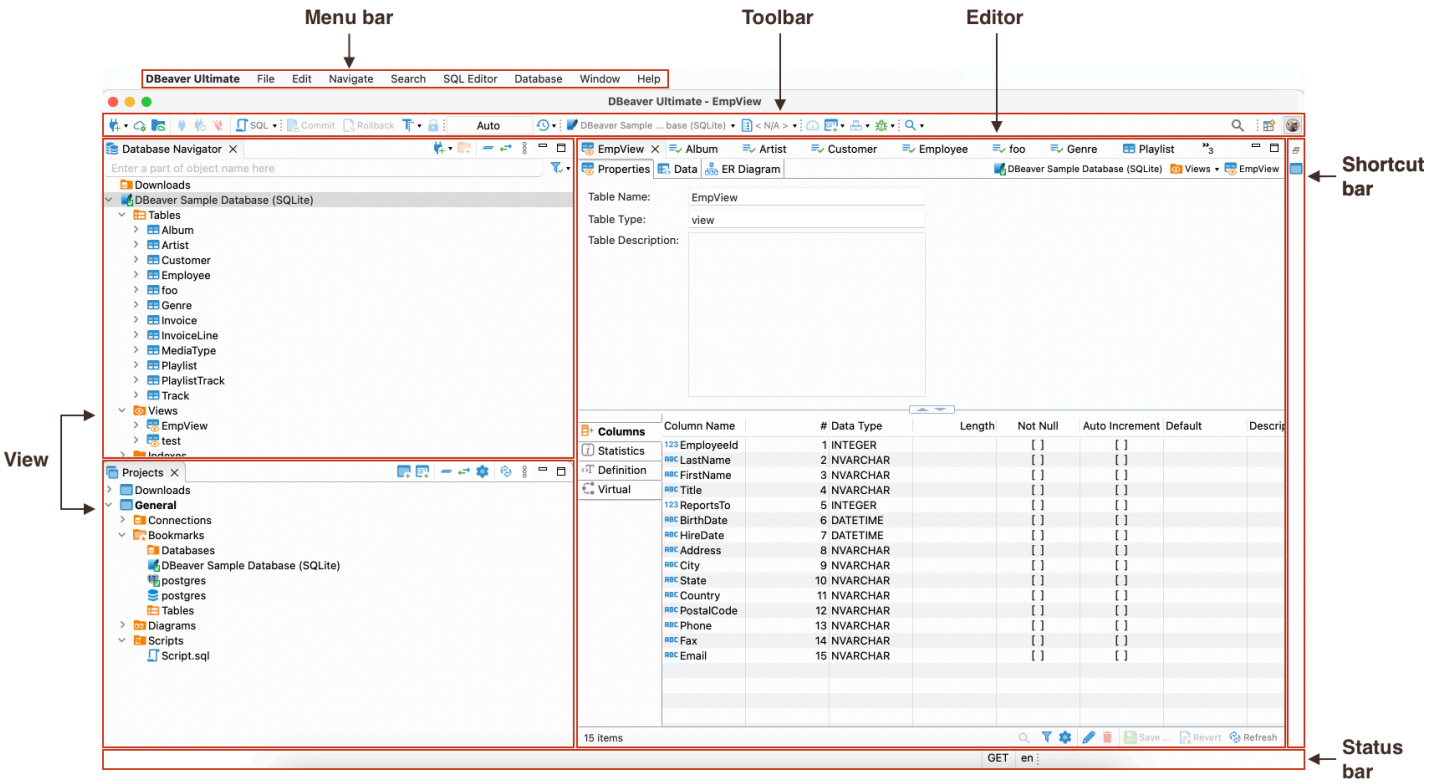
Automatic updates

This feature is available only in Windows and macOS.

From time to time, DBeaver automatically checks for new versions. If the check is not disabled and a new version is available, it will prompt you to decide whether you want to stay or upgrade. If the second option is chosen, the installer will be downloaded and launched upon completion. Note that DBeaver itself will be gracefully closed to avoid data loss.

Application Window Overview

The DBeaver window contains a menu bar, a toolbar, a workspace with one or more editors and views, and a status bar:



Menu Bar

By default, the menu bar contains the following menus:

- The **File** menu contains menu items for creating files, folders, projects, database connections, database projects, ER diagrams, and Import and Export items.
- The **Edit** menu contains global commands like Cut, Copy, Paste, Delete and Add Bookmark targeted at the active element.
- The **Navigate** menu allows navigation through scripts and database objects.
- The **Search** menu provides options to search among files database objects and across data.
- The **SQL Editor** menu is for opening SQL Editor and managing its appearance.

- The **Database** menu allows the management of database drivers, connections, and transactions and reconnecting to and disconnecting from a database.
- The **Window** menu includes items to manage the look of DBeaver window: show/hide and minimize/maximize views and editors, display bars, split editors, and manage other preferences.
- The **Help** menu contains links to information and help resources, and menu items to check the version number and availability of updates.

You can customize the menu bar and the list of menu items to display; for this, go to **Window** -> **Customize Perspective** -> **Menu Visibility** tab.

Toolbar

The toolbar contains buttons for the most basic and frequently used commands:



Some buttons are enabled (colored) and others are disabled (grey). The sets of enabled and disabled buttons change depending on which editor is active in the workspace. Only enabled buttons apply to the active view or editor.

You can customize the toolbar; go to **Window -> Customize Perspective -> Tool Bar Visibility** tab.

You can hide or show the toolbar in the application window. To do it, go to the Window menu, click **Appearance -> Hide Toolbar / Show Toolbar**.

Shortcut Bar

There are two shortcut bars - one on the left and one on the right side of the workspace zone. Shortcut bars host shortcuts of views and editors and appear if at least one view or editor is minimized, otherwise, they are hidden.

Workspace: Views and Editors

Views are windows within the workspace that provide presentations and ways to navigate the information. For more information about particular views, see the Views article.

Editors are windows in which you can interact with the content of files and databases. For more information about particular editors, see the Editors article.

Both views and editors can appear as separate windows or tabs stacked with other views/editors in a tabbed window. The following image shows the title bar of a tabbed window. They become hidden if tabs do not fit in the title bar of a tabbed window. To see the list of hidden tabs, click the Show List icon that also indicates their number:



Several views and editors can be opened simultaneously in the workspace, but only one can be active at a time.

You can change the layout of the workspace by opening and closing views, docking them in different positions in the workspace, collapsing them to the shortcut bar, or expanding them to occupy the whole workspace and restoring them to the latest docked position.

Changing Workspace Layout

You can move views and editors around the workspace and dock them in different positions:

- As a tab in a tabbed window
- As a separate window with a vertical or horizontal layout in any zone of the workspace

You can also swap locations of two views or editors.

To dock a view to a position in the workspace, press and hold the view's title bar, then drag and drop it onto the desired position.

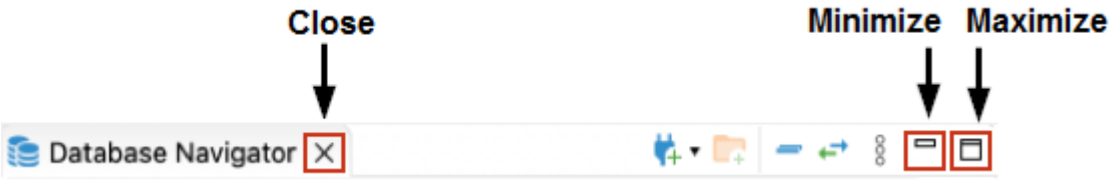
You can resize the view and editor windows. To resize, place the cursor on the window's border until it changes to a double-ended arrow, then click and drag the edge to the needed size.

To close a view or editor, click the Close button, or right-click the title bar of the view / editor, then click one of the options on the context menu (they change depending on the configuration of windows):

- **Close** - to close the active window or tab in a tabbed window.
- **Close Others** (for editors and views that appear as tabs in tabbed windows) - to close all tabs of the current tabbed window except the active tab.
- **Close Tabs to the Right / Left** (also for tabbed windows) - to close all tabs of the current tabbed window that are located to the right / left of the active tab.
- **Close All** - to close all tabs of a tabbed window (close the window).

Maximizing, Minimizing and Restoring View and Editors

All views and editors have the **Close**, **Minimize** and **Maximize** buttons:



The Maximize button changes to the Restore button when a view or editor is maximized.

To maximize a view or editor to the size of the whole workspace, do one of the following:

- Click the Maximize button in the upper-right corner of the view.
- Double-click the title bar of the view or editor.
- On the **Window** menu, click **Appearance** -> **Maximize Active View or Editor**.

When one view is maximized, other views and editors appear as shortcuts on the shortcut bar.

To restore a maximized view or editor to its latest docked position, double-click its title bar or the **Restore** button in its upper-right corner.

When you minimize a view, it wraps into a shortcut on the shortcut bar:



The shortcuts of views and editors may appear on the left or the right shortcut bar, depending on the latest docked position of the view or editor.

To minimize a view, do one of the following:

- Click the **Minimize** button in the upper-right corner of the view.
- On the Window menu, click **Appearance** -> **Minimize Active View or Editor**.

To restore a minimized view or editor to its previous position, click the Restore button on its shortcut in the shortcut bar. To restore a minimized view or editor to a new position, click the view / editor name button under the restore button.

Views

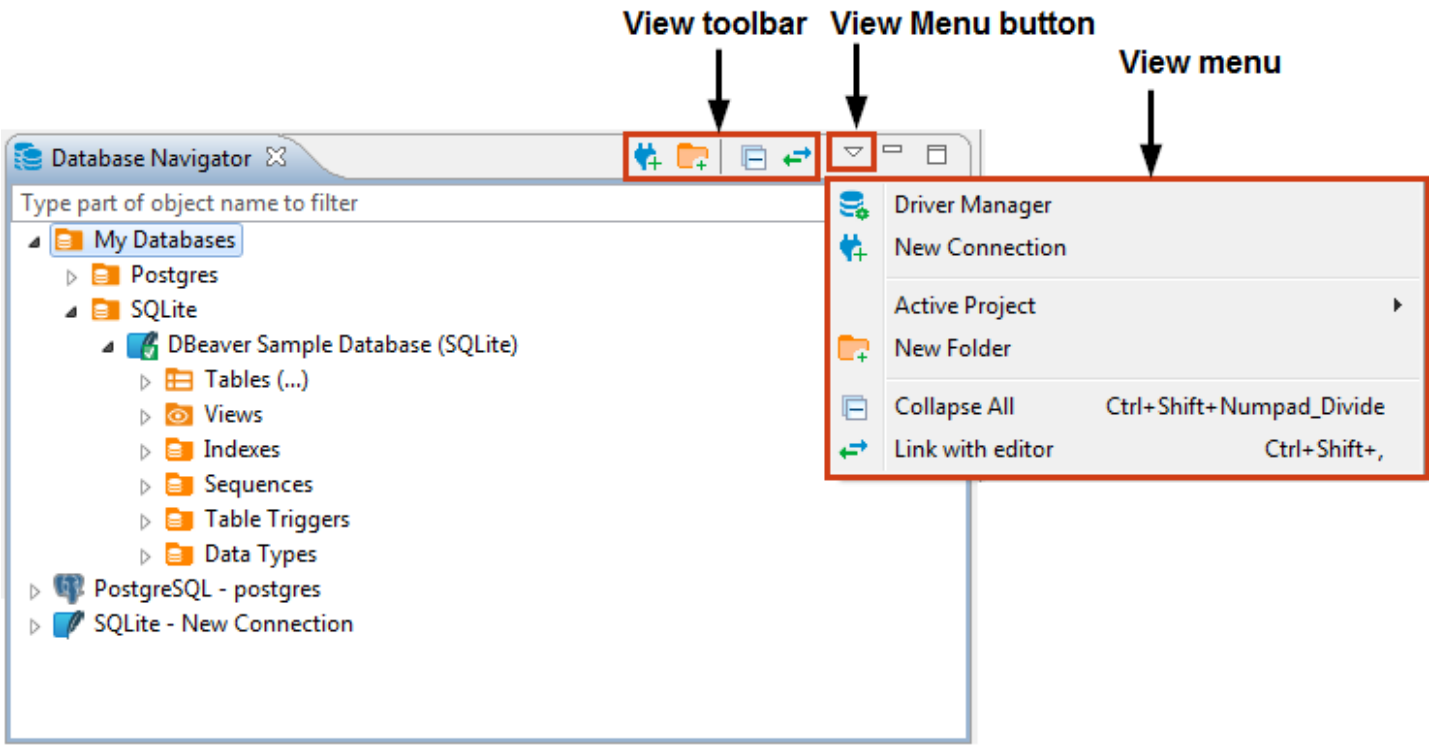
Views are windows within the workspace that provide presentations and ways to navigate the information. The main views in DBeaver are: Database Navigator, Projects and Project Explorer.

To open a view:

- On the Window menu, click **Show View** and then, on the submenu, click the name of the view. Click **Other** if the view is not visible on the submenu.
- For Database Navigator, Projects, and Project Explorer views, on the Window menu, just click the name of the view.

Some views open on demand, for example the Search view opens to show search results.

Views provide their own toolbar and menu:



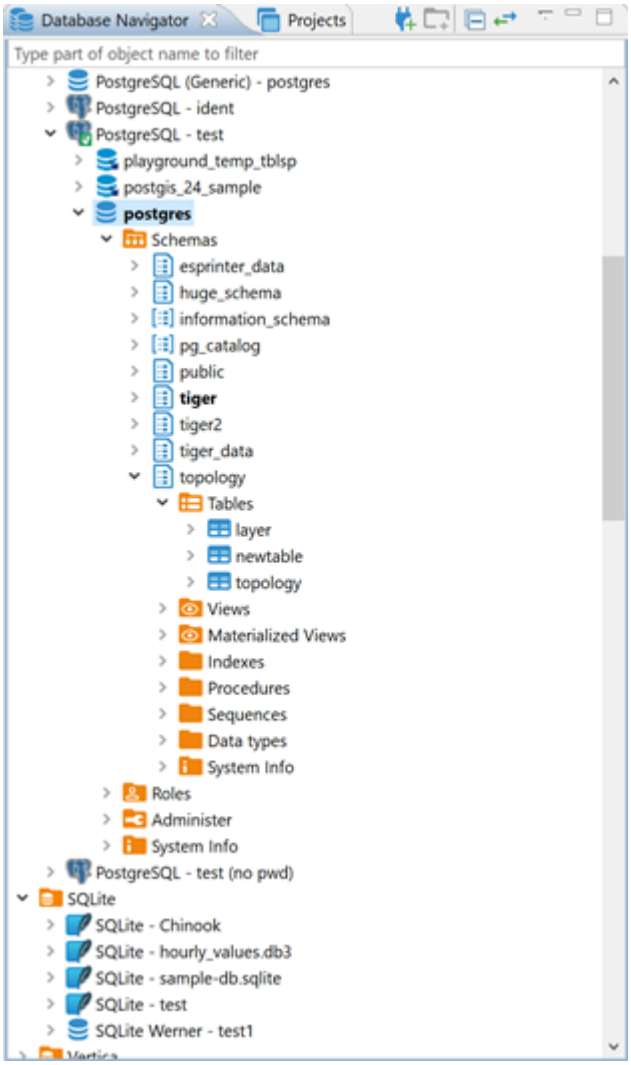
To open the view menu, click the View Menu button in the upper-right corner of the view's title bar, next to the Minimize button.

The toolbar contains buttons applicable to the objects displayed in the view. The set of enabled and disabled buttons depends on the object in focus.

Views also provide context menus for objects they display. To open a context menu for an object, right-click the object.

Database Navigator

Database Navigator is the main view to work with the structure and content of databases. To open Database Navigator, on the Windows menu, click **Database Navigator**. For information on how to change the view layout, please see the Application Window Overview article.



Database Navigator contains a tree of objects, a toolbar and View menu which contain generic items. Each object in the tree has its own context menu. The tree contains the following objects:

- Folders -
- Database connections - and other (icons differ depending on the database type)
- Database objects - various depending on the database type, such as Tables , Views , Columns 123 ABC, Indexes , etc.

To open the view menu of Database Navigator, click the View Menu button ( ) in the upper-right corner of the window. For more information on where to find the view toolbar and menu, please see the Views article.

The menu contains the following items:

Icon	Menu item	Description
	**Driver Manager**	Opens the Driver Manager window that allows you to create, edit and delete drivers for databases. See Database Drivers for information about managing database drivers.
	**New Connection**	Opens the Create new connection wizard. See Create Connection for information about creating connections.
(empty)	**Active Project**	Displays a submenu which allows you to choose a project. See Projects and Projects view for information about projects.
	**New Folder**	Opens a dialog box for creating a new folder
	Collapse All	Collapses the tree to the root level
	Link with editor	Synchronizes the active editor with the element in the database navigator

The toolbar is located in the title bar of the window. Its buttons duplicate the menu items, except for the **Active Project**.

To open the context menu for an object, right-click the object in the tree. The following table summarizes the context menu items for all types of objects that may appear in the tree. Note that the presence or absence of the context menu items for an object depends on the database and object types.

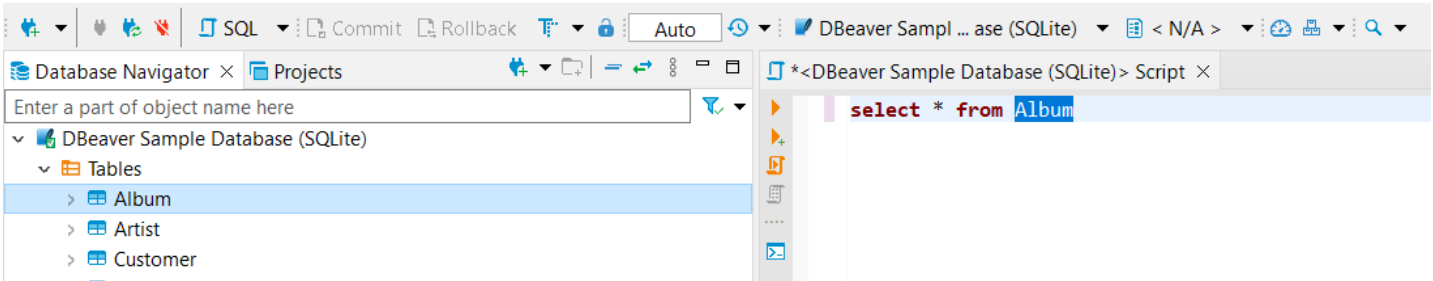
Menu item	Description
Open folder	Opens a folder in a separate view
Create new connections / Create New Connection	Opens the Create new connection wizard
New Folder	Opens a dialog box for creating a new folder
Copy	Copies an object to the clipboard

Paste	Inserts the copied object into a selected folder - most convenient for copy-pasting connections
Delete	Deletes an object **WARNING!** The Delete menu item removes the object not only from the tree but from the database itself or the file system, and this action is irreversible.
Rename	Opens the Rename [object] dialog box
Properties	Opens the Properties for [object] window which allows viewing and modifying the object`s properties
Refresh	Depending on the object, refreshes the object itself, or its parent, or its subnodes – mostly used for refreshing tables and schemes
Connect	Attempts to connect to the database
Invalidate/Reconnect	Checks the status of connection, if it is broken, attempts to reconnect
Disconnect	Disconnects from the database
SQL Editor	Opens a new SQL editor for the connection
Recent SQL Editor	Opens the most recently opened SQL editor
Edit Connection	Opens the Connection Configuration window that allows configuring connection settings
View [objects]	Opens the object in a separate viewer
Edit [object]	Opens the object in a separate editor
Create new [object]	Opens an editor in which you can specify properties and save the new object
Filter	Opens a submenu of one or more filtering options (depending on the object): - Hide [object] - Show only [object] - Configure [objects] filter - Toggle filter - Clear filter See Filters for information.
Copy Advanced Info	Copies the full name of an object
Read Data in SQL Console	Opens an SQL console displaying the object`s data
	- Appears only if you select several objects of the same level

Compare	- Opens the Compare objects wizard which guides you through the steps to generate a comparison report for the selected objects
Generate SQL	Opens a submenu on which you can select the type of SQL query to generate: - SELECT - INSERT - UPDATE - DELETE - MERGE - DDL Clicking one of the items (for example INSERT) generates a relevant query in a separate window.
Export Table Data	Opens the Data Transfer wizard that helps you select a format and export table data
Import Table Data	Opens a window with existing database connections in which you can select a table to import data from
Tools	Opens a submenu that provides tools for database backup and restore, vacuum, etc.

For information on how to filter database objects in Database Navigator, please see the Filter Database Objects article.

While working in the SQL Editor, you can drag the desired object from the Database Navigator to the part of your query. Please pay attention to the selected object name. Running the query immediately with the selected object will cause an error because only the selected part of the script is being executed and fails as it's not a complete query. An extra step to deselect it by clicking somewhere in the SQL Editor is required to remove the selection. Nevertheless, we cannot deselect the object name automatically after dragging. This action is needed when you drag and drop an object into a complex query or long script, and it would be difficult to track what and where exactly you've just dropped, and you can get an unexpected error.



Filter Database Objects

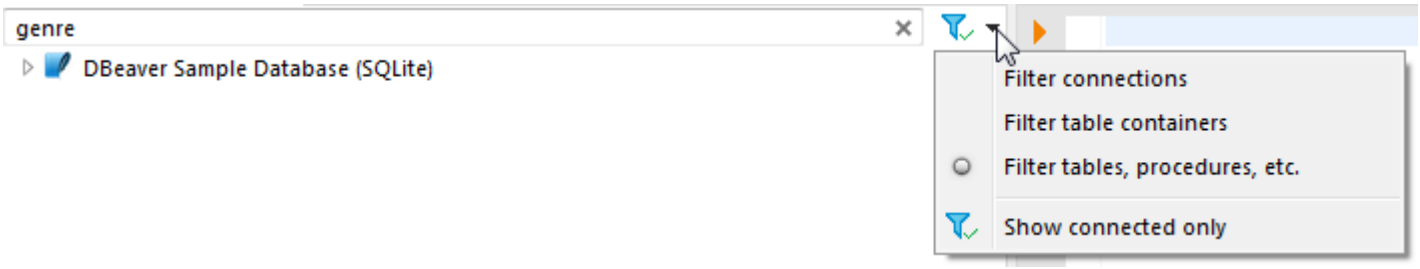
In the Database Navigator and Database Object Editor you can filter database objects to include or exclude some of them from the view. You can filter connections, schemas, tables, views, and procedures. A dots sign `(...)` next to the node`s name indicates that a filter is applied to its sub-nodes: Tables (...)

There are several ways in which you can filter objects. One of the ways is to filter objects by the names of tables and views using the filter field above the tree of objects:

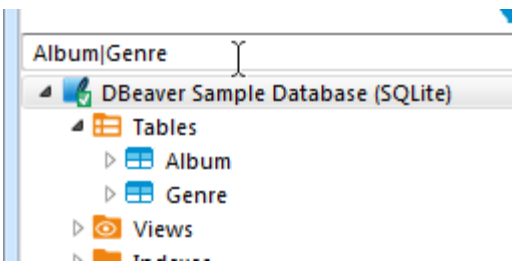


To filter objects by name, type the object name in the field. The tree dynamically updates to show connections /containers/tables/views with that name. To reset the filter, click the Clear icon (✕) on the right end of the field.

You can select the types of filtering objects in the drop-down list on the right. And also use the filter only for active connections.



For multiple filtering use special symbols in the filter field - pipe ("|"), comma (","), or space between object names. You can also use an asterisk symbol to replace the part of the name.



Another way to filter objects is to use the **Filter** item on the context menu of a single object. To filter objects using the **Filter** menu, right-click the object, then click **Filter** on the context menu, and then click one of the items on the submenu:

Filter submenu item	Description
Hide '[object name]'	Hides the current object while displaying the other ones
Show only '[object name]'	Shows the current object while hiding the other ones
Toggle filter	Inverts the filtering – shows hidden objects and vice versa
Clear filter	Removes the filtering to display all objects
Configure [objects] filter	Appears only to the folder or parent nodes of database objects - like 'Tables', 'Indexes', etc. Allows the creation of a complex filter with multiple filtering criteria, see Configure Filters.

The third way of filtering is to use the **Filter** item on the context menu on several objects:

1. Select several objects of the same type using Ctrl or Shift keys.
2. Right-click the selection, then click **Filter**, and then choose one of the options on the submenu:

Filter submenu item	Description
Hide N objects	Hides the selected objects while displaying the rest
Show only selected objects	Shows the selected objects while hiding the rest

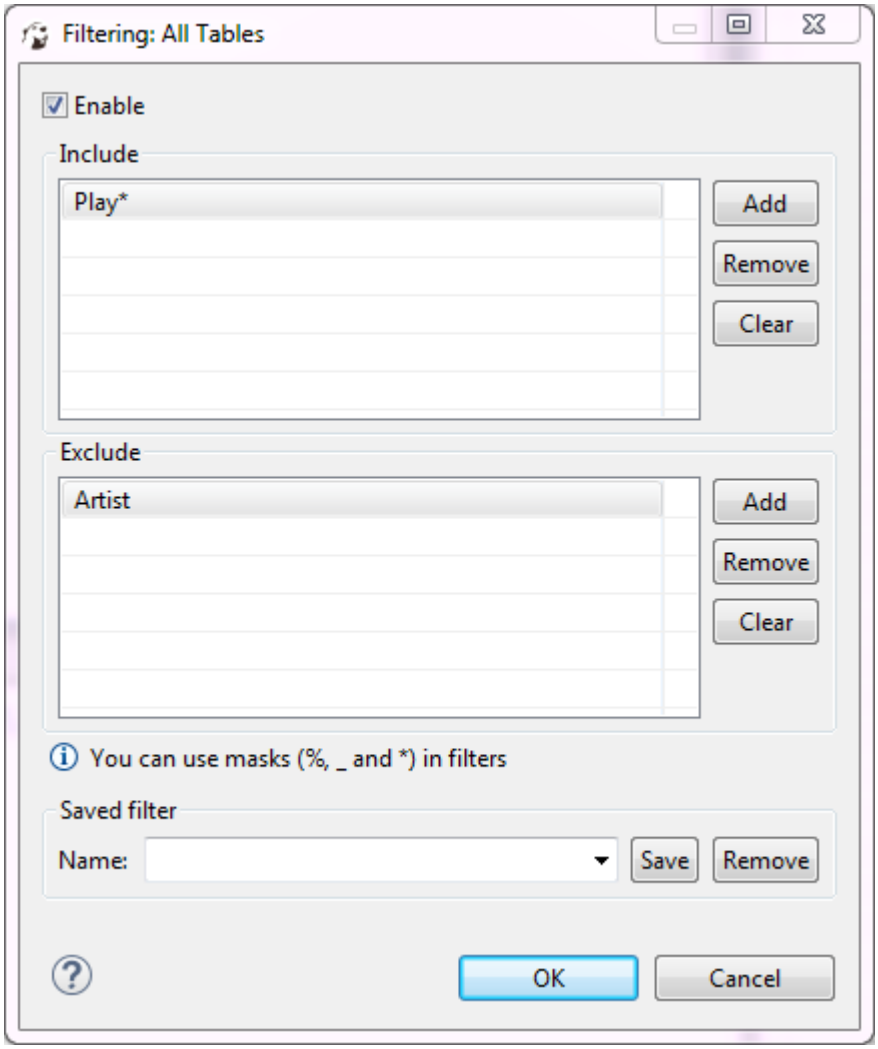
To reset such filters, right-click the parent (folder) node displaying the dots sign `(...)`, and then click **Filter -> Clear filter**.

Configure Filters

You can configure custom filters to filter database objects in the Database Navigator and Database Object Editor.

To configure a custom filter:

1. In the **Database Navigator**, right-click the object and on the context menu click **Filter** -> **Configure [objects] filter**. In the **Database Object editor**, in the toolbar of the **Properties** tab, click the **Filter settings** button ( ). The Filtering window opens.



2. Select the **Enable** checkbox to activate the fields of the window.

3. If you want the filter to apply to all objects of a certain type, for example to all schemes, click **Show global filter**. Otherwise, the filter will apply only to the current object. NOTE: Once you apply the global filter, you cannot revert back to the local filer in the same window. To create a local filter, reopen the Filtering window, see Step 1.

4. For objects that you want to show, click **Add** next to the **Include** field and then, in the field itself, enter the name or combination of symbols to search. For objects that you want to hide, click **Add** next to the **Exclude** field and then, in the field itself, enter the name or combination of symbols to search. **NOTE**: In your search pattern, you can use wildcard characters. The `%` symbol is used to replace any number of characters, while `_` is used to replace a single character. In some database and search systems, the `*` character is also used as a wildcard to replace any number of characters. Always check the system's documentation to ensure you're using the correct symbols.

5. To remove one filtering combination, click the combination in the field and then click **Remove**. To remove all combinations from either of the fields, click **Clear** next to the field.

6. Once you set all filtering criteria, you can save a filter to use for other objects. To save the filter, in the Saved filter area, in the **Name** field, enter the filter`s name and click **Save**.

7. You can also remove any of the saved filters. To remove a filter, in the **Name** drop-down list, click the filter name and then click **Remove**.

8. Click **OK** to apply the filtering criteria. Otherwise, click **Cancel**.

Simple and Advanced View

By default database navagator show all database objects including system objects, all schemas, tables, indexes, administrative and other utility objects.

If you want to change database navigator contentsm simolify or customize it then there are several options. There are 3 modes:

- Simple - minimalistic view for people who mostly wotk with tables
- Advanced - shows all database objects supported by DBeaver
- Custom - manually customized view

When you change view mode you must recovvect to a database (because DBeaver will read different database metadata)

Simple view

- Shows only schemas and tables
- Hides all system objects
- Hides all utility objects
- Hides all intermediate object folders

Advanced view

- Shows all system objects
- Hides all utility objects
- Show all folders and all supported database objects

Custom view

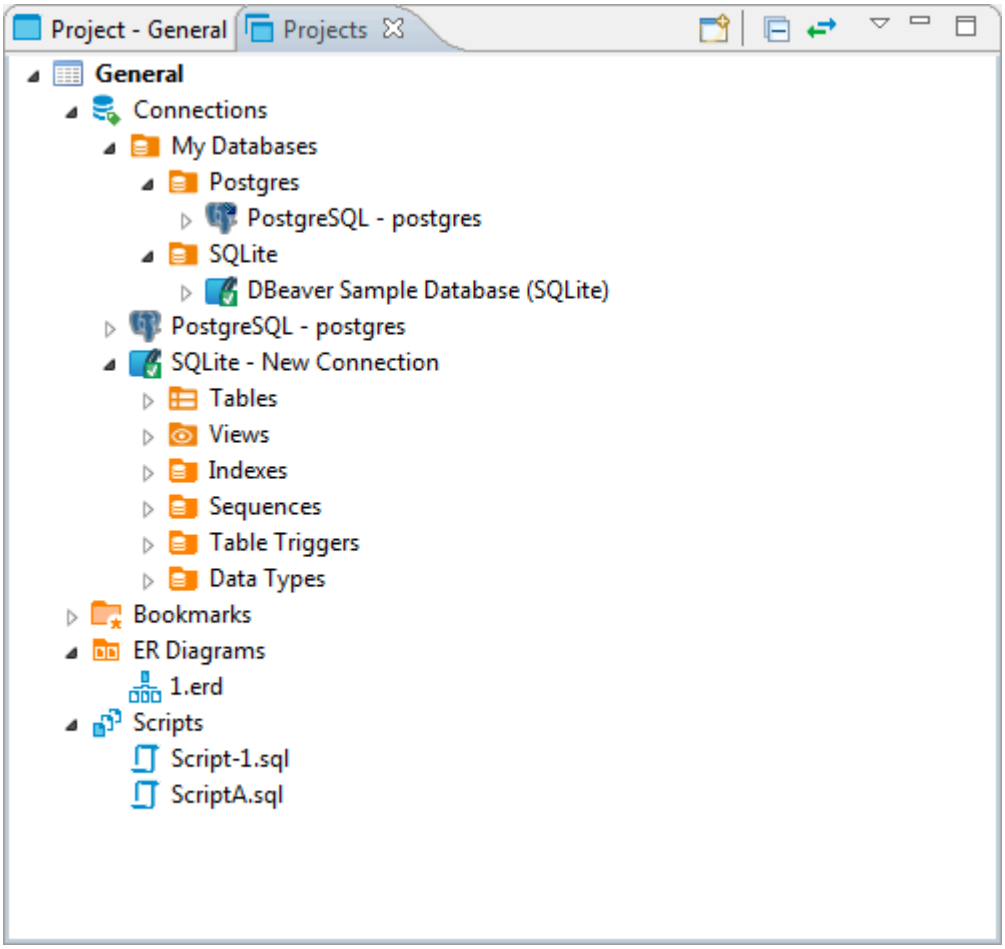
You can configure custom view by setting the following options:

- Show system objects (e.g. system schema pg_catalog or SYSTEM)
- Show utility objects (e.g. internal template databases needed for new database cvreation)
- Show only schemas and tables. It hides all database objects except databases, schemas and tables in these schemas.
- Hide folders. It hides all intermediate logical folders, e.g. "Tables", "Indexes", "Foreign keys". In shows folders contents instead. If there are multiole subfolders then it merges them into a single list.

Projects View

You might need to classify and group database connections into projects. Projects store objects related not to a particular database but to all database connections. These are usually files stored on the file system.

The Projects view displays all projects created in the system and provides tools to manage them. To open the Projects view, on the **Window** menu, click **Projects** (or use `ALT+W+P` shortcut).



For information on how to change the view layout, please see the Application Window Overview article.

The projects are organized into a tree and all have the same high-level structure:

- **Connections** – repeat the content of the Database Navigator view for this project. You can perform the same actions over the objects of the databases as in the Database Navigator.
- **Bookmarks** – contains bookmarks – shortcuts to database objects, see ...
- **ER Diagrams** - contains ER diagrams that you can drag-and-drop here from other folders

- **Scripts** – contains scripts that you can drag-and-drop here from other folders

The Projects view provides a toolbar and View menu which contain generic items. Each object in the tree has its own context menu.

To open the view menu of the Projects view, click the View Menu button ( ) in the upper-right corner of the window. The view menu contains the following items:

Icon	Item	Description
	Create Project	Opens the Create Project wizard
	Refresh Projects	Refreshes the projects tree to display changes caused by creating modifying or deleting projects
	Collapse All	Collapses the tree to the root level
	Link with editor	- Enabled when at least one editor is open, otherwise disabled - Highlights the object in the tree that has its editor open

The toolbar is located in the title bar of the window, its buttons duplicate the view menu items except for the **Refresh Projects** one.

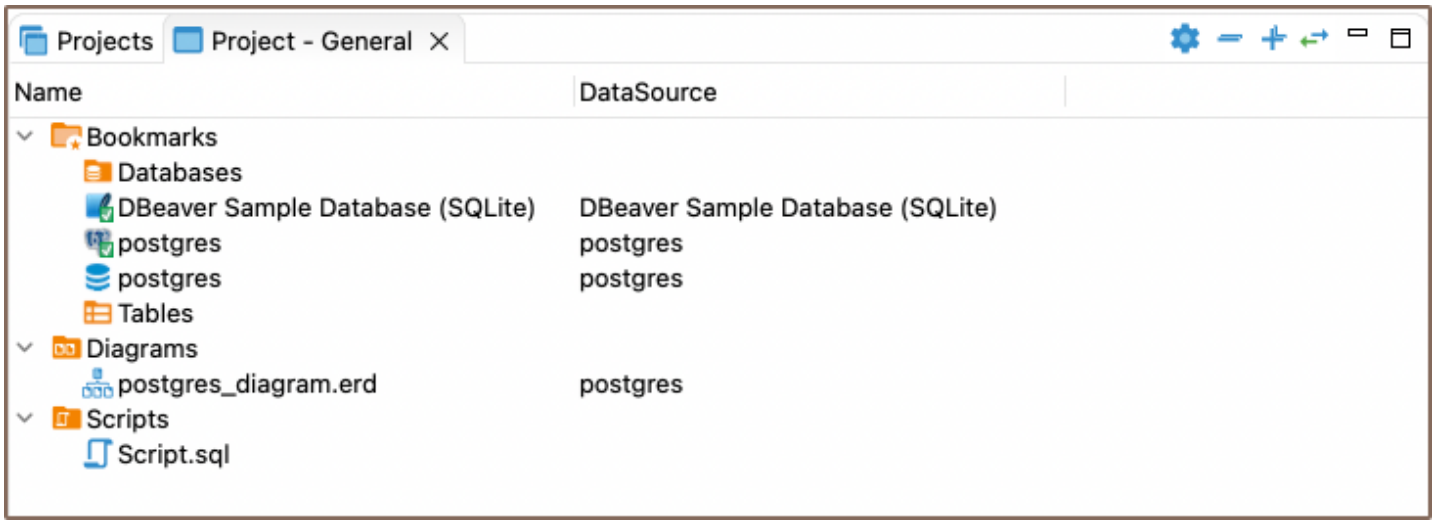
To open the context menu for an object in the tree, right-click the object. For information about context menu items of all objects under the **Connections** node of the tree, please see Database Navigator. The context menus of other nodes in the tree contain some basic items for copy-pasting, renaming, deleting objects, managing their properties, creating folders, etc.

- The **Set Active Project** menu item (for a project root node) makes the project active, that is visible in the Database Navigator.
- The **Link File (SQL Script)** and **Link Folder** menu items allow creating links to files and folders in the file system.

For information about managing projects, please see Projects article.

Project Explorer

The **Project Explorer** view displays detailed contents of the currently active project. To open the **Project Explorer**, click **Window** -> **Project Explorer**.



For information on how to change the view layout, please see the Application Window Overview article.

The title of the **Project Explorer** includes the name of the project: Project – [Project name]. **General** is a project that initially exists in the system by default.

The **Project Explorer** displays the content of a project with metadata. The content includes: Bookmarks, ER Diagrams, and Scripts. The metadata appears in columns which you can hide or show.

The **Project Explorer** view provides a toolbar that contains the following buttons:

Button	Name	Description
	Configure columns visibility	Opens a dialog box in which you can select columns to display in the view
	Collapse All	Collapses the tree to the root level
	Expand All	Expands the tree nodes
	Link with editor	- Enabled when at least one editor is open, otherwise disabled - Highlights the object in the tree that has its editor open

To sort the metadata in the table by a certain column, click the column header.

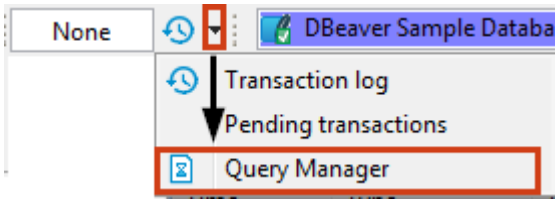
Query Manager

Query Manager is a view that shows the history of all SQL queries that DBeaver has executed during the current session.

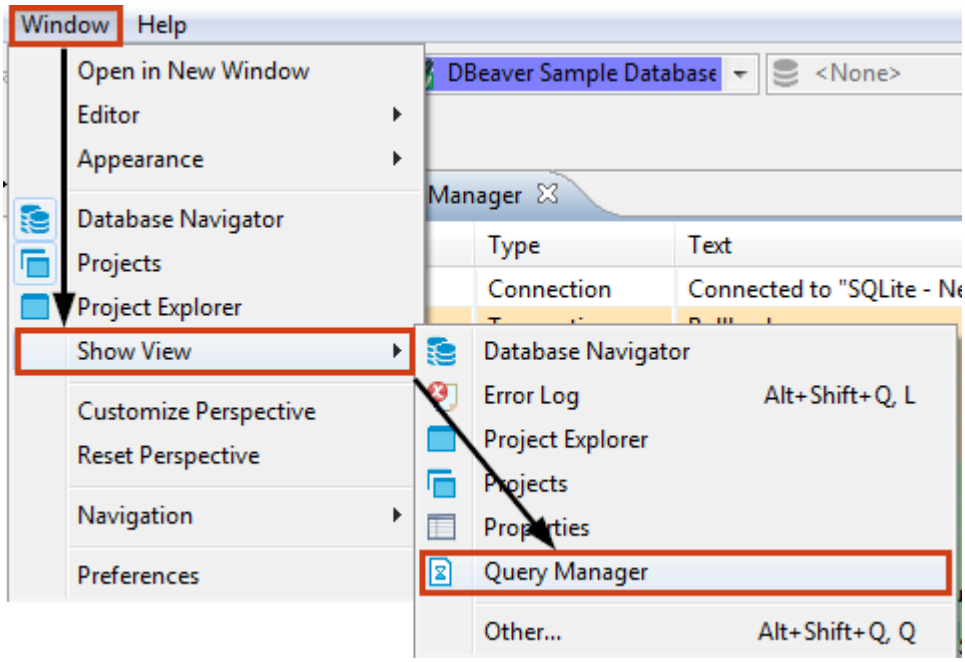
Note DBeaver saves query history after restarting only in Lite, Enterprise, Ultimate and Team editions.

To open the Query Manager, do one of the following:

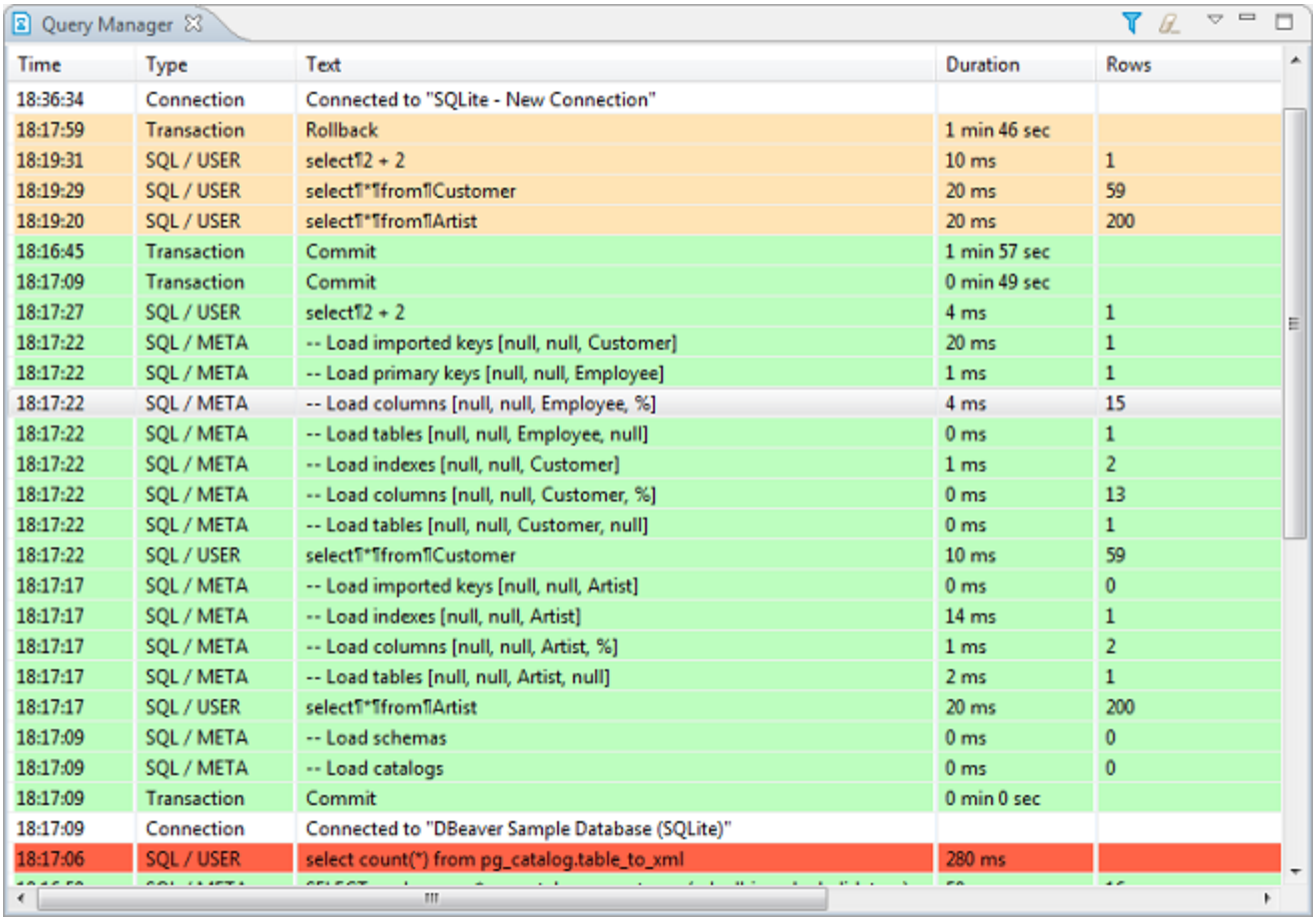
- Click the arrow next to the **Transaction Log** button in the toolbar and then click **Query Manager** on the dropdown menu:



- On the **Window** menu, click **Show View -> Query Manager**:



The Query Manager logs all queries together with their execution statistics (execution time, duration, number of fetched/updated rows, errors, etc.):

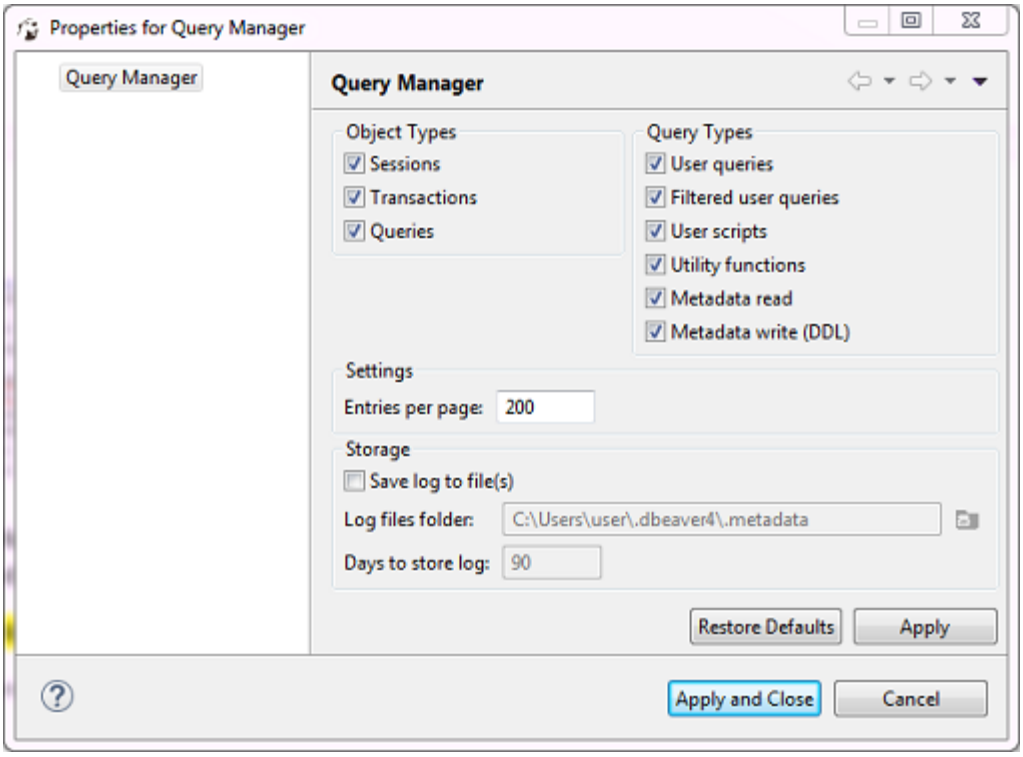


You can modify the look of the Query Manager by filtering queries and setting the number of entries displayed per page, as well as specifying some storage settings. See the 'Query Manager Properties' section below.

To erase all entries from the Query Manager, click the **Clear query manager log** button ( ) in the view`s toolbar. **In DBeaver PRO versions**, you can also open **Help —> Clear History** in the main menu and select **Query log history** option. The query history will disappear after the restart.

Query Manager Properties

To manage the look of the Query Manager, filter the entries, and modify the storage settings, click the **Set query manager filter** button ( ) in the view`s toolbar. The Properties for Query Manager window opens:

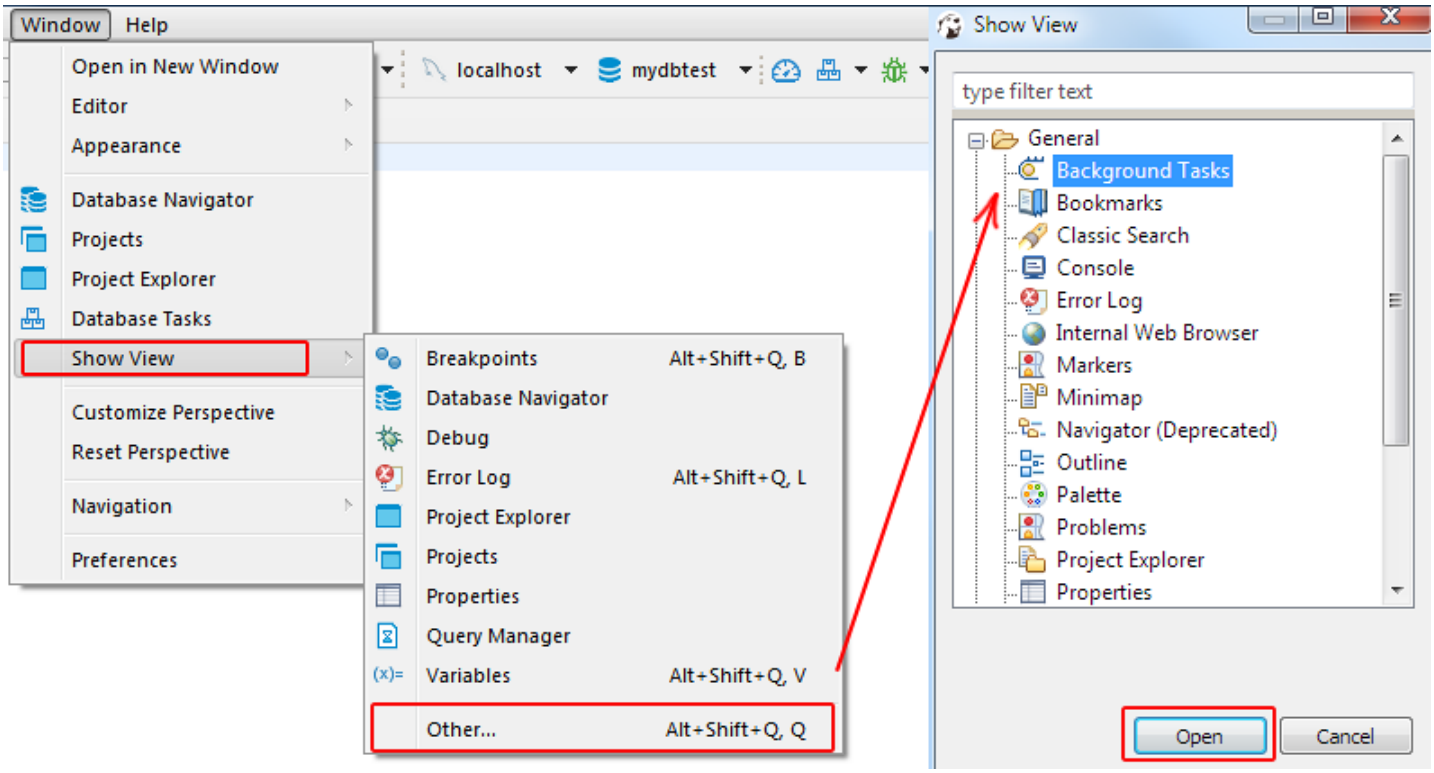


- Select or clear the checkboxes in the **Object Types** section to filter entries by object type. To filter entries by query type, select clear the checkboxes in the **Query Types** section.
- To change the number of entries displayed per page, enter the new number in the **Entries per page** field.
- To set DBeaver to save the query log in a file, select the **Save log to file(s)** checkbox and then specify the file location in the **Log files folder** field.

After you make all necessary changes to the settings, click **Apply** to apply the changes and keep the window open or click **Apply and Close** to apply the changes and close the window. To discard all changes and return to the previous state, click **Restore Defaults**.

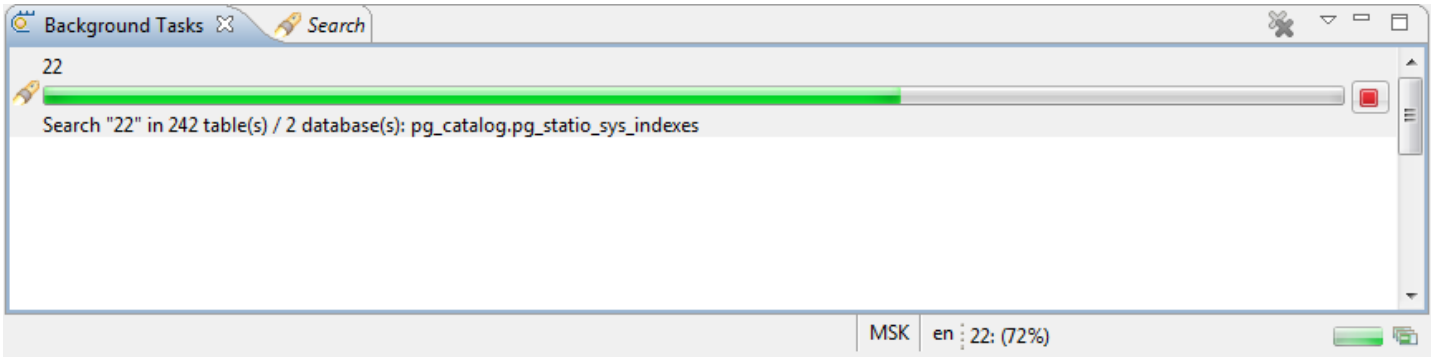
Background Tasks

You can open the Background view from the main menu - click **Window -> Show View -> Other**, then in the Show View window, expand the **General** folder, click **Background Tasks** and then click **Open**:



You can also open the Background Tasks view from some other views or editors by using a special button, for example from the Search view.

The Background Tasks view shows the progress of such background tasks as search, SQL query execution, etc. The view shows data when background tasks take some noticeable time and is useful when you want to track the progress of lengthy operations. If you open this view at a short task, the view will be empty.



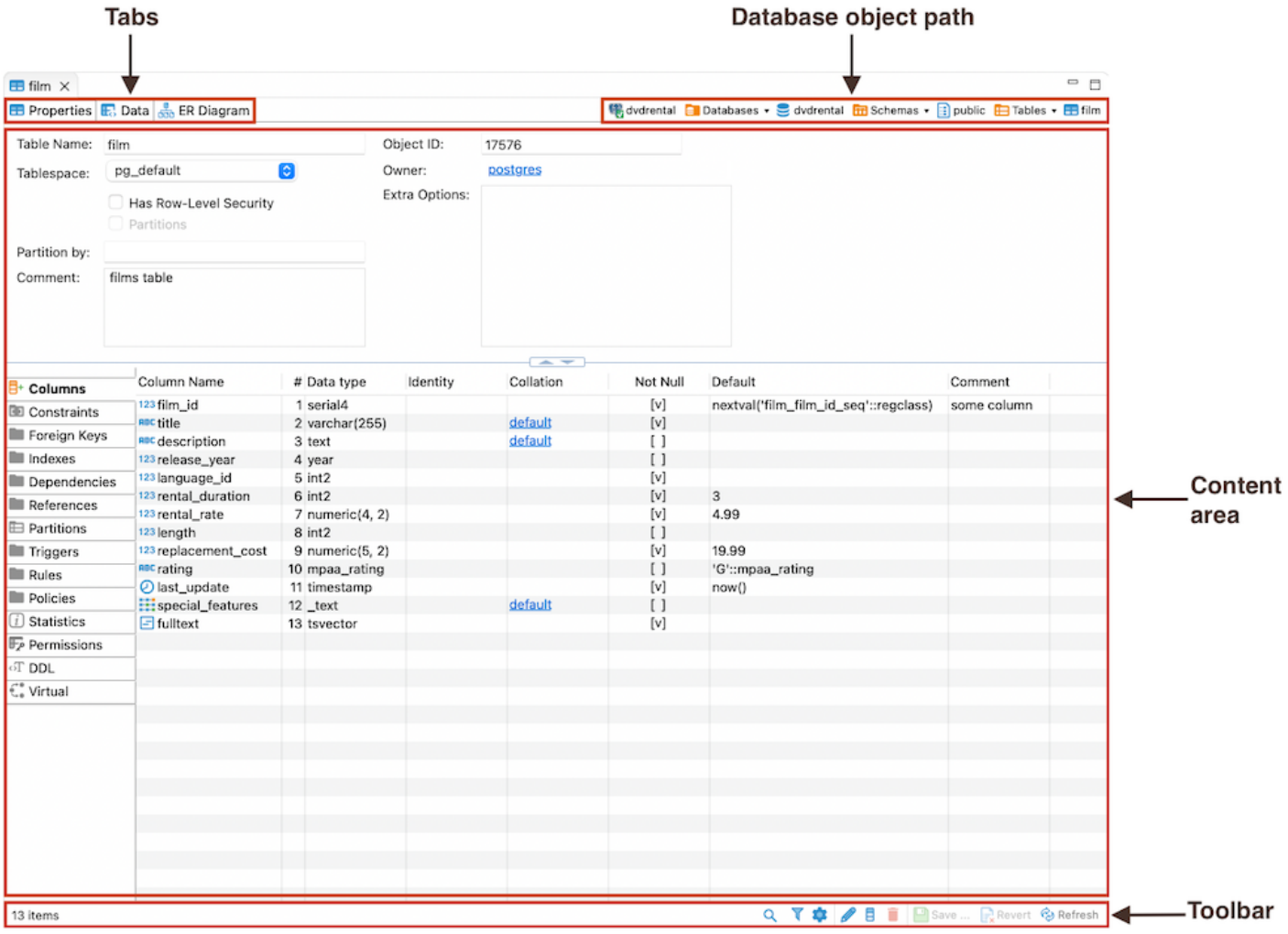
You can cancel the task in progress - click the **Cancel Operation** button ( ).

Database Object Editor

The Database object, or metadata editor is available for multiple database objects such as tables, views and schemas. To open the metadata editor for an object, in the Database Navigator or in the Projects view:

- Double-click the database object
- Click the database object and press `Enter` or `F4`

Upon performing these steps, you will see a window where the tabs contain the following common parts:



Tabs

- The **Properties** tab, available for all objects, encompasses the properties of the database object as well as its sub-entities. For more information, please refer to the Properties Editor article.

- The **Data** tab appears for tables and views and represents the Data Editor.
- The **ER Diagram** tab appears for tables and schemas and displays ERD (Entity Relation Diagrams).

Database object path

The object's path shows the chain of all its parent entities. The entities are clickable: clicking an entity in the path, depending on its nature, either shows its children or opens an editor or a settings window.

Content area and Toolbar

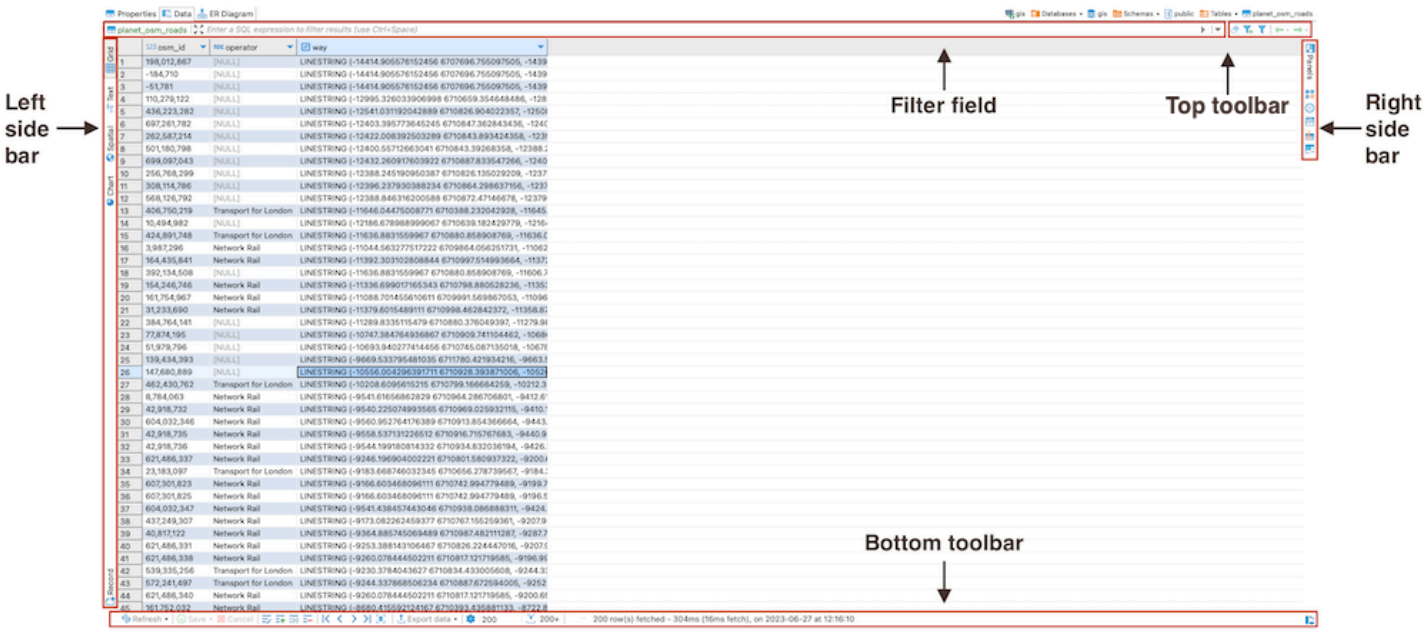
The Content Area and Toolbar are key parts of the Properties tab. The Content Area shows information about a database object and things like its sub-parts or DDL. The Toolbar has different tools on each tab to help with different tasks.

Data Editor

The Data editor appears:

- as the **Data** tab of the Database Object Editor, which is only available for tables and views;
- as the **Results** tab when you run a custom SQL query in SQL Editor.

The Data editor allows to view and edit data of a database table or view. The central part of the Data editor is the data grid. The editor also provides top toolbar, bottom toolbar, left side bar, right side bar and a filter field:



To learn how many rows the table data contains, click the **Calculate total row count** button in the bottom toolbar. The number of rows appears in a status field next to the button: 203

To learn about ways to navigate data in the table data, see Navigation article.

Top toolbar

The top toolbar contains the following buttons:

Button	Name	Description
	Remove all filters/orderings	Removes all filters and orderings applied to the data in the filter field.

	**Save filter settings for current object**	Saves the current filter settings for the database object to apply next time when you reopen it in the editor, see details in the Data Filters article.
	**Custom Filters**	Opens the Result Set Order/Filter Settings window. See Data Filters article for more information.
	Forward and backward - history navigation buttons	Navigate forward and backward in the Data Editor history, see the history section of our article for more information.

Left side bar

The left side bar contains the following tabs:

Button	Name	Description
	Grid	Switches to the grid view of data.
	Text	Switches to the plain text view of data.
	Spatial	Switches to the spatial view. For more details, see the GIS data article.
	Chart	Switches to the chart view. For more details on charts, see the Managing Charts article.
	Record	- Same as pressing `Tab` - Switches the positions of rows and columns so that the columns appear as rows, and the rows hide in one **Value** column. See details in the Table vs. Record Views section of our article.

Right side bar

The right side bar contains the following tabs (see the Panels for more information):

Button	Name	Description
	Panels	Opens panels on the right side of the Data Editor.
	Calc	Opens the result cells calculation panel (SUM, MAX, AVG, etc.).
	Grouping	Opens grouping panel window tools.
	Metadata	Opens Metadata panel.
	References	Opens References panel.

	Value	Opens Value Viewer.

Bottom toolbar

The bottom toolbar provides the following buttons:

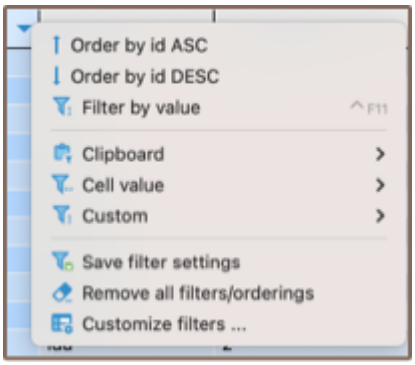
Button	Name	Description
	Refresh	Refreshes the whole results set, including all items that are not visible on the screen, while its dropdown option allows to customize the refresh frequency over a specific period.
	Save	Saves all unsaved changes to the data such as adding, duplicating, deleting rows, inline editing of values. See the Data Viewing and Editing article for information.
	Cancel	Discards all unsaved changes to the data.
	Edit cell value in separate dialog /editor	Opens the cell in focus for editing in a separate editor or dialog box. See details in the Cell Editor section of our article.
	Add new row	Adds a new empty row below the current row, see details in the Adding, Copying and Deleting Rows section of our article.
	Duplicate current row	Copies the current rows and pastes the copy below the current row, see details in Adding, Copying and Deleting Rows section of our article.
	Delete current row	Colors the rows in focus in red to mark them for deletion, see details in the Adding, Copying and Deleting Rows section of our article.
	Move to first row	Moves the focus (highlighting) from the current to the first row of the table.
	Move to previous row	Moves the focus (highlighting) from the current to the previous row of the table.
	Move to next row	Moves the focus (highlighting) from the current to the next row of the table.
	Move to last row	Moves the focus (highlighting) from the current to the last row of the table.
	Fetch all data	Fetches the whole result set making it ready for display. See the Scrolling Results Page section of our article.
	Configure	Opens a menu with settings.

	Result-set fetch size	Displays the initial fetch number of rows in the result set on loading or refreshing.
	Calculate total row count	Calculates the total number of rows in the table.

Note: Some of these buttons may be disabled and may not work if you are using a read-only connection, connecting to a read-only database or if you see the result of a complex query, such as joining two or more tables.

Column context menu

Each column has a context menu, accessed by clicking the downward arrow button , providing different filter options.



Cell context menu

Every cell in the table data also has a context menu – right-click the cell to open the menu. The context menu provides the following items:

Menu Item	Description
Copy	Copies the content of the current cell or column to the clipboard.
Advanced Copy	Opens advanced copy submenu that allows copying data with preset formatting parameters.
Paste	Pastes the copied content to the cells in focus.
Advanced	

Paste	Pastes several values delimited with a tabulation or line break starting from the selected cell.
Delete	Deletes the row that has the cell in focus. **NOTE**: In fact, when users click **Delete**, the system only highlights the red row while the actual deletion happens when users click **Save**.
Edit	Opens a submenu enabling inline editing. See the Data Viewing and Editing article.
Order	Displays a submenu that allows to specify ordering criteria for the data. The submenu contains the most common ordering options that can be applied to the column in focus. See details in Data Filters article. By default, DBeaver orders data by sending a request to the server (the Server-side results ordering checkbox selected). To order data on the client side using DBeaver's internal algorithm, clear the checkbox.
Filter	Displays a submenu that allows to specify filter criteria for the data. The submenu contains the most common filters that can be applied to the cell in focus. See details in Data Filters article. By default, DBeaver filters data by sending a request to the server (the Server-side results ordering checkbox selected). To filter data on the client side using DBeaver's internal algorithm, clear the checkbox.
View /Format	Opens a submenu that provides options for formatting and modifying the view of data. See the Data View and Format article.
Navigate	Opens a submenu that helps users to navigate through the table data. See the Navigation article.
Layout	Changes the layout of data, see the Data View and Format article.
Export data	Opens the Data Transfer wizard that guides you through the steps to export data to a selected format. See the Data Tansfer article. **Note**: The system exports the whole result set including records that are not visible on the screen and preserves all applied data filters and ordering.
Generate SQL	Opens a submenu where you can select the type of SQL query to generate. See the SQL Generation article.
Generate Mock Data	Opens Mock Data Generator. See the Mock Data Generation in DBeaver article.
Logical structure	Opens a submenu allowing you to write virtual column expressions. See the artice about Virtual column expressions.
Open with	Opens the data in external applications like Excel or a web browser.
Toggle result	Opens Value Viewer panel on the right side of the Data Editor.

pannel	
Refresh	Refreshes the whole results set including all items that are not visible on the screen.

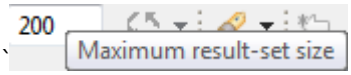
For more information about using the Data Editor, please see the subsections of this article - open them via the contents tree on the right.

Navigation

Scrolling Results Page

If the result set has many rows, you can scroll the results page. To learn how many rows the data table contains, click the **Calculate total row count** button in the bottom toolbar. The number of rows appears in a status field next to the button: 203 . Alternatively, you can right-click a cell in the table and then click **Navigate -> Row Count** on the context menu.

By default, DBeaver limits the number of rows fetched to **200** (you can change this value in the main toolbar or in preferences). The maximum number of rows that DBeaver fetches to display in the Data tab is specified in the Maximum result-set size field in the main toolbar: `



Once you scroll to the last row of the current result portion, DBeaver fetches the next portion (next N rows). You can disable this behavior in preferences. You can also manually fetch the next portion of data equal to the maximum result set size. To do so, click the **Fetch next page of results** button ( ) in the bottom toolbar or right-click the table and click **Navigate -> Fetch next page** on the context menu.

The number of rows fetched is visible in the status field under the data table:



To see the details, click the details button in the status field.

To fetch the whole result set, click the **Fetch all rows** button ( ) in the bottom toolbar or right-click the table and click **Navigate -> Fetch All Data** on the context menu.

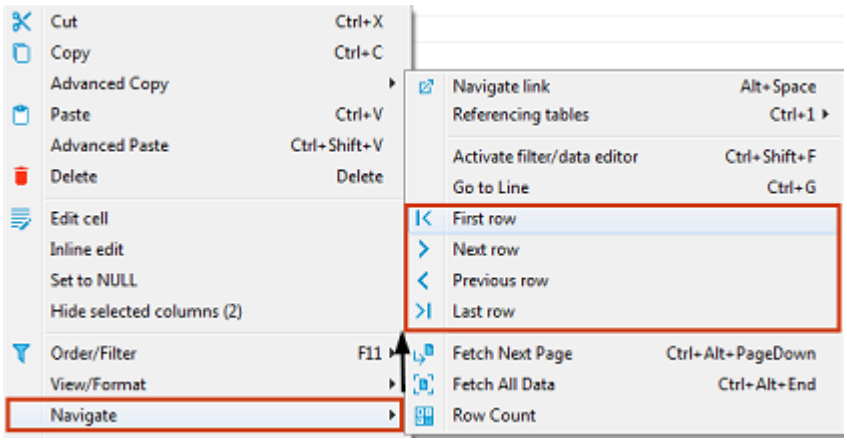
NOTE: Be careful when fetching the whole result set. If it is huge, it might cause program hangup or out-of-memory errors.

You can navigate through the result set using standard shortcuts `Home` , `End` , `PgUp` , `PgDown` , `Ctrl+Home` , (

.

Data Rows

To jump to the first or last row or move one row forward or backward, use the navigation buttons in the bottom toolbar or on the context menu: 



To jump to a specific line, right-click anywhere in the table and click **Navigate** -> **Go to Line** on the context menu. Then in the Go to Row dialog box, enter the row number and click **OK**.

History

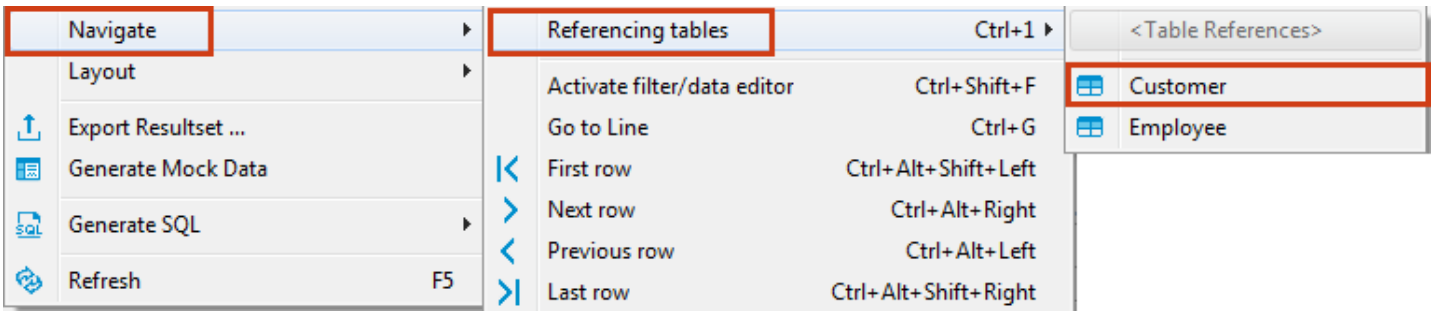
DBeaver remembers the history of actions such as applying filters to data, opening reference tables and other tables via links. You can navigate among such tables and filtered views:

- Use the forward and backward buttons in the top toolbar: 
- Click `Ctrl+Left` or `Ctrl+Right`

Hovering over these buttons displays the names of the tables or filtered views saved in the history.

Navigate Foreign Keys / Referencing Tables

You can navigate with foreign keys or reference tables – those that reference the current table. To open a referencing table, press `Ctrl+1` or right-click the cell and click **Navigate->Referencing tables->[table name]**:



The referencing table opens in the same editor. To navigate back and forth between the initial and referencing tables, use the history navigation buttons (⬅ ➡) in the top toolbar of the editor. In order to open a referencing table in a new window use the `Ctrl+Shift+1` shortcut to the show menu.

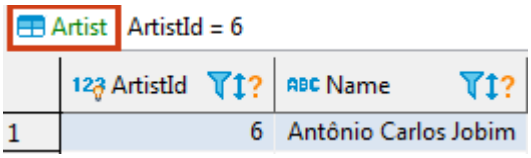
Navigation Links

In the data editor, you can navigate to linked tables – the ones that the current table references. To open a linked table, click the Navigate link icon in a cell that contains it:



	AlbumId	Title	ArtistId
7	7	Facelift	5
8	8	Warner 25 Anos	6
9	9	Plays Metallica By Four Cel	7

Another way is to right-click such a cell and click **Navigate -> Navigate link** on the context menu. The linked table opens in the same editor, filtered by the cell value:



Artist ArtistId = 6

	ArtistId	Name
1	6	Antônio Carlos Jobim

NOTE: The table name in green above the table indicates which table is currently open in the editor.

To navigate back and forth between the initial and linked tables, use the history navigation buttons (⬅ ➡) in the top toolbar of the editor.

You can open a linked table in a separate editor. To do so, simultaneously hold the `Ctrl` key (or `# command` key on macOS) and click the Navigate link icon (↗) in the cell.

Data View and Format

Grid vs. Plain Text Views

You can switch between two data presentations in SE version and four presentations in EE version. Pressing `CTRL+-` switches available presentations in turn.

- To see the data in a grid view, similar to an Excel spreadsheet, click the **Grid** button (Grid) on the bottom toolbar of the editor.
- To switch to the plain text view, click **Text** (Text) on the bottom toolbar.
- To switch to JSON view (available in EE version only for MongoDB documents and JSON tables), click **JSON** on the toolbar.
- To switch to XML view (available in EE version only for XML tables), click **XML** on the toolbar.

Table vs. Record Views

The table view is a standard table (Excel-like) in which columns are vertical and rows are horizontal. This view is the default one. If you click the **Record** button in the bottom toolbar of the editor (Record), or press `Tab` , or right-click a cell and then click **Layout -> Record** on the context menu, the rows and columns switch positions. The columns will appear as rows, and the rows will be hidden in one **Value** column which will show only one row of data. The column headers will shift from the top of the table to its left side:

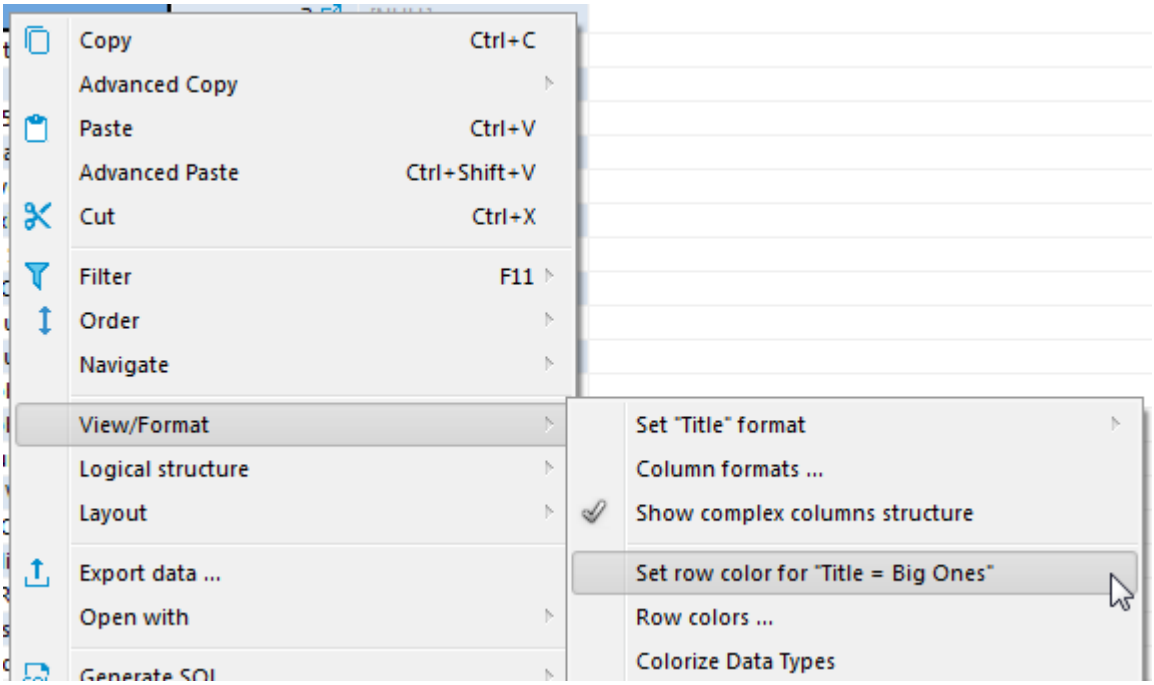
	Value
123 AlbumId	22
ABC Title	Sozinho Remix Ao Vivo
123 ArtistId	16
Column1	[NULL]

The Record view is useful if the table contains a big number of columns. To navigate from row to row of data, use the navigation buttons on the bottom toolbar of the editor:

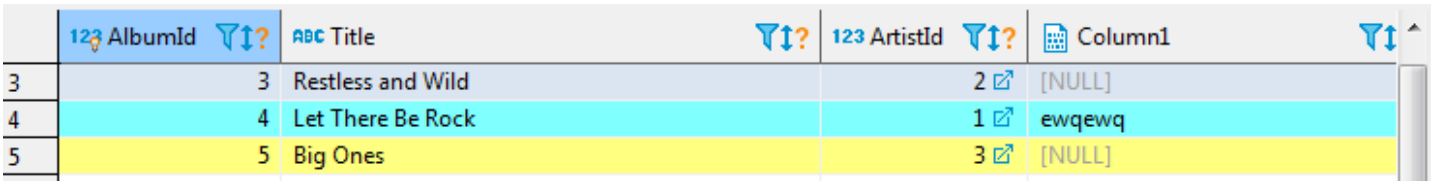
To return back to the standard table view, click the **Record** button again.

Rows Coloring

In the data editor, you can colour all rows that have the same value as a particular cell of a certain column. To do so, right-click the cell and click `View/Format ⇒ Set the row colour for {column name = value}` on the context menu:

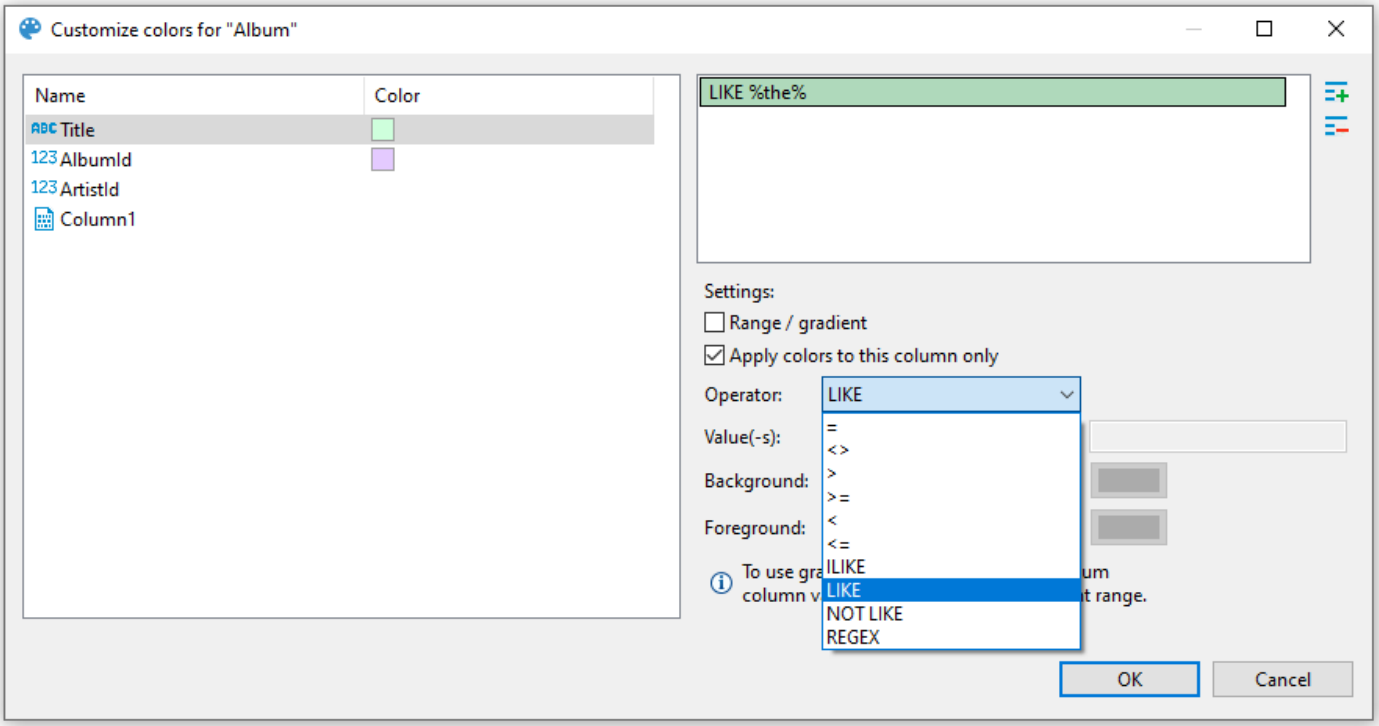


Then choose the colour in the palette window that appears and click **OK**. The current row and all other rows that contain the same value change their colour to the one you have selected:



To remove the coloring by a particular column, right-click the cell again and click `View/Format ⇒ Clear colo` on the context menu.

By choosing `View/Format ⇒ Row colors ...` from the context menu, you can gain more precise control of coloring conditions:



Here, you can define multiple conditions for single column using rich set of predefined operators, change background color and define a range between two values.

Operators work as you may expect. Note that they're executed on the client-side, that means no extra queries are made in order to apply colors.

Value range / Gradient

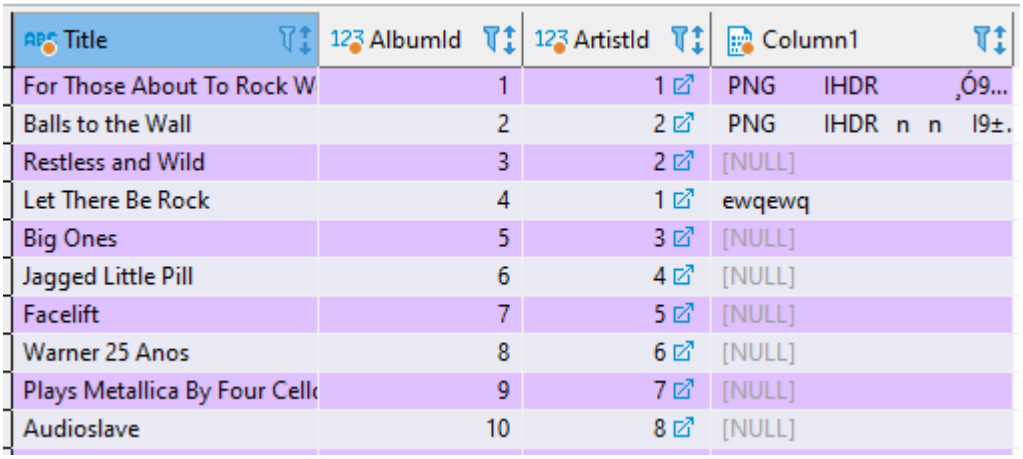
Value range allows you to paint your rows with gradient that fades from first value to second value:

Title	AlbumId	ArtistId	Column1
For Those About To Rock W	1	1	PNG IHDR ¸Ó9...
Balls to the Wall	2	2	PNG IHDR n n I9±.
Restless and Wild	3	2	[NULL]
Let There Be Rock	4	1	ewqewq
Big Ones	5	3	[NULL]
Jagged Little Pill	6	4	[NULL]
Facelift	7	5	[NULL]
Warner 25 Anos	8	6	[NULL]
Plays Metallica By Four Cell	9	7	[NULL]
Audioslave	10	8	[NULL]

In this example we defined a range for column `AlbumId` that fades from `#80c6ff` to `#8000ff` between values `1` and `10`.

Using regex

You can use regular expressions for matching complex values. Otherwise, you can be artistic and, for example, paint rows with odd values in your favorite color:

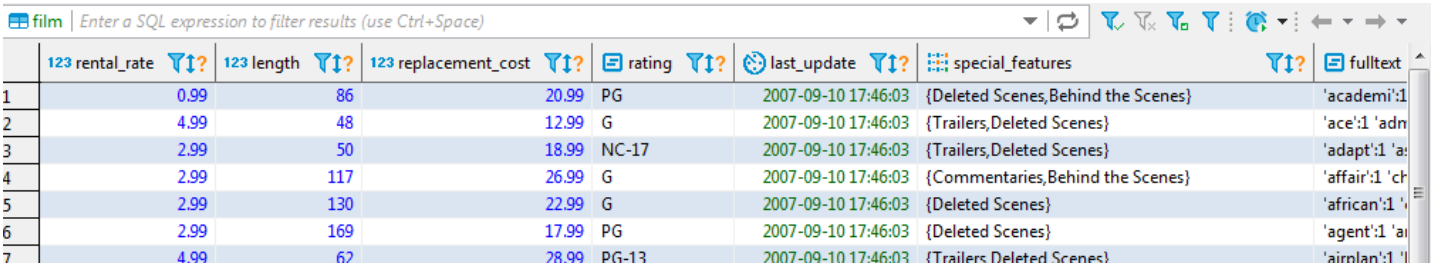


Title	AlbumId	ArtistId	Column1
For Those About To Rock W	1	1	PNG IHDR ,Ó9...
Balls to the Wall	2	2	PNG IHDR n n I9±.
Restless and Wild	3	2	[NULL]
Let There Be Rock	4	1	ewqewq
Big Ones	5	3	[NULL]
Jagged Little Pill	6	4	[NULL]
Facelift	7	5	[NULL]
Warner 25 Anos	8	6	[NULL]
Plays Metallica By Four Cell	9	7	[NULL]
Audioslave	10	8	[NULL]

Snippet for coloring odd rows: `^\d*[13579]$` *; even rows:* `^\d*[02468]$`

Coloring by Data Types

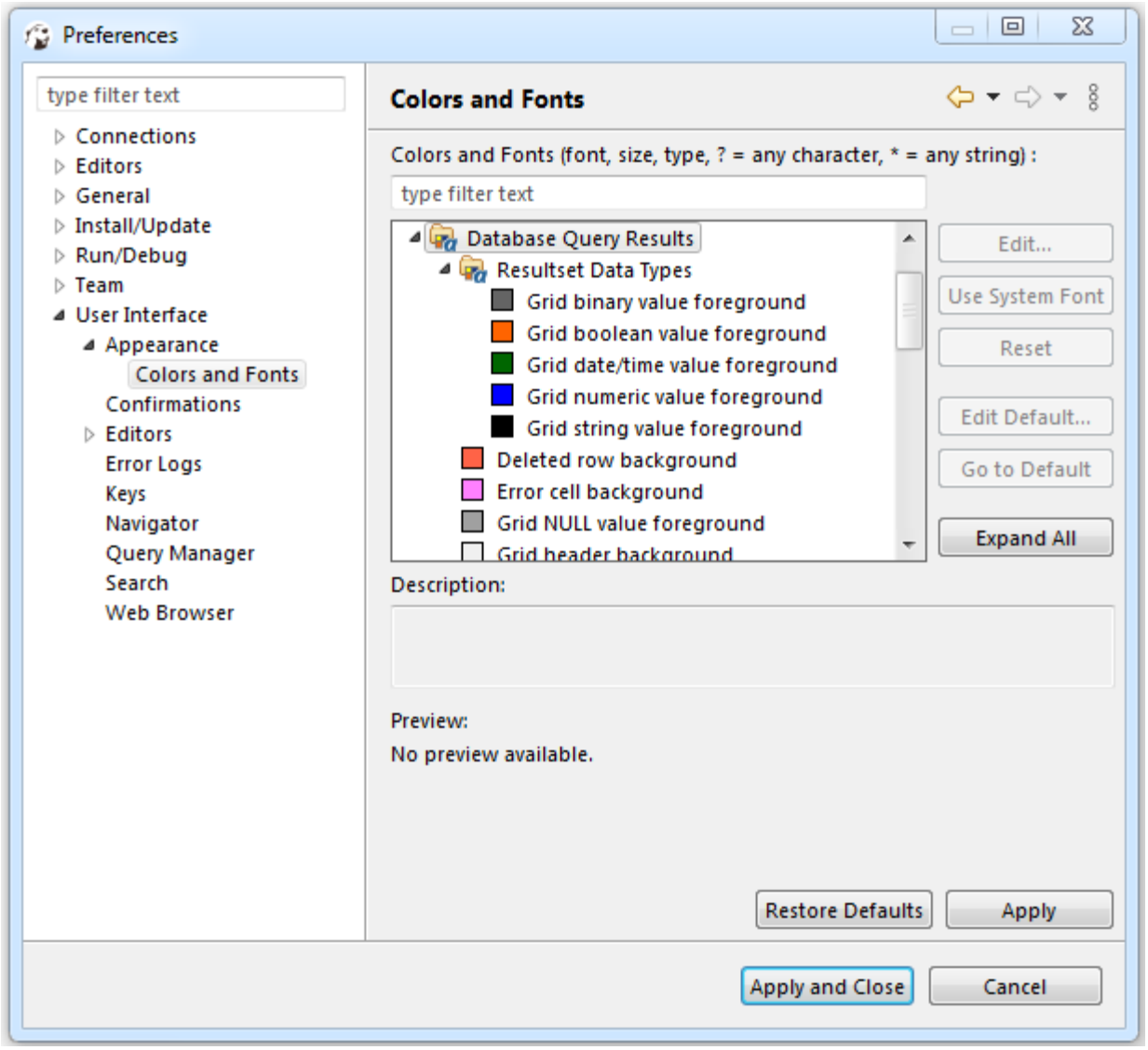
Besides colouring rows by a value, you can colour the values in the columns by data types. To do so, right-click any cell in the table and, on the context menu, click **View/Format -> Colourize Data Types**. The values in the cells will be coloured in different colours according to the current colour preferences:



film | Enter a SQL expression to filter results (use Ctrl+Space)

	rental_rate	length	replacement_cost	rating	last_update	special_features	fulltext
1	0.99	86	20.99	PG	2007-09-10 17:46:03	{Deleted Scenes,Behind the Scenes}	'academi':1
2	4.99	48	12.99	G	2007-09-10 17:46:03	{Trailers,Deleted Scenes}	'ace':1 'adn
3	2.99	50	18.99	NC-17	2007-09-10 17:46:03	{Trailers,Deleted Scenes}	'adapt':1 'a
4	2.99	117	26.99	G	2007-09-10 17:46:03	{Commentaries,Behind the Scenes}	'affair':1 'ch
5	2.99	130	22.99	G	2007-09-10 17:46:03	{Deleted Scenes}	'african':1 '
6	2.99	169	17.99	PG	2007-09-10 17:46:03	{Deleted Scenes}	'agent':1 'a
7	4.99	62	28.99	PG-13	2007-09-10 17:46:03	{Trailers,Deleted Scenes}	'airplan':1 'l

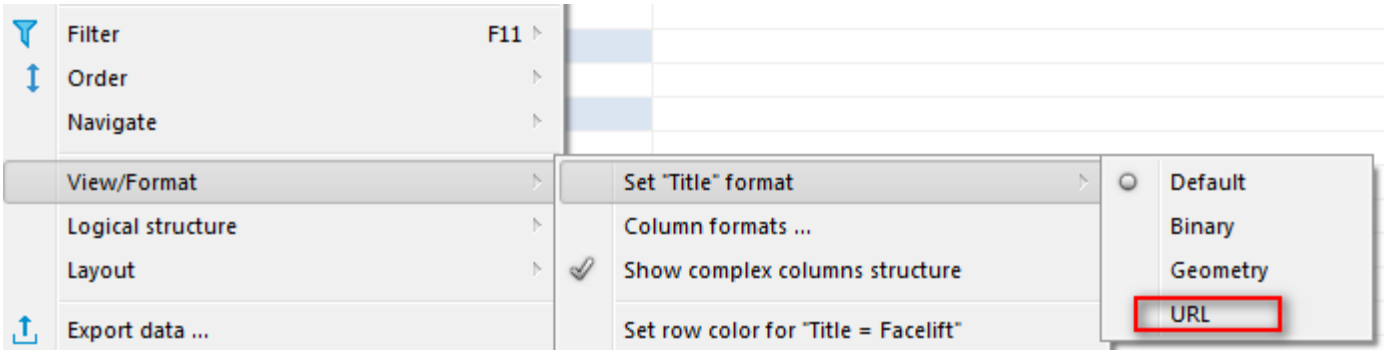
You can change the colour preferences in the Preferences window by: clicking **Window -> Preferences** on the main menu. In the window of the navigation pane on the left, expand **User Interface** and then **Appearance**, and then click **Colours and Fonts**:



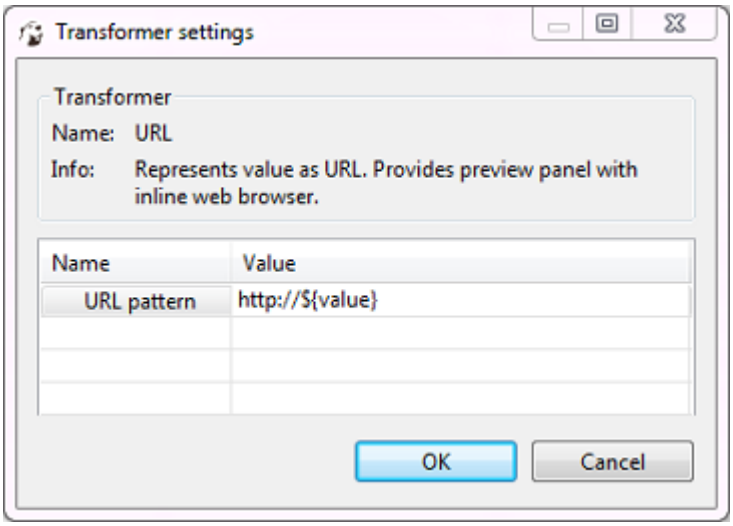
To remove the colouring by data types, on the context menu, click **View/Format -> Colourize Data Types** again.

Transforming Data Presentation

For string and numeric data types, DBeaver provides tools to transform the data presentation into a number of formats, such as URL and Binary for strings and Epoch Time, Number Radix, etc. for numbers. To change the data presentation in a certain column, right-click a cell in the column. Then, on the context menu, click **View/Format -> Set {column name} format** and click the presentation type name:



The Transformer settings window opens showing the value in the chosen format. Click **OK** to apply the change:



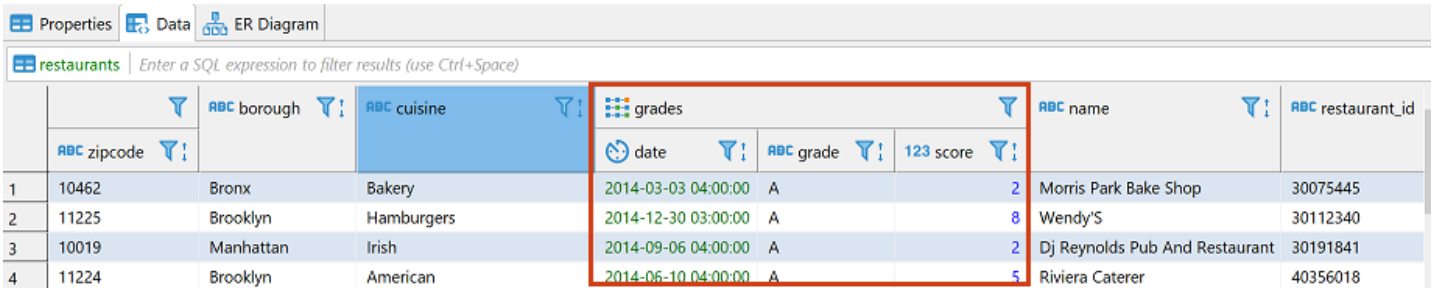
The values in the column appear in the new format.

NOTE: For URL format, the resulting cell provides a link to the URL in a browser window.

To roll back the changes to the default format, right-click any cell in the column, and on the context menu, click **View/Format -> View as -> Default**.

Structurizing Complex Data Types

For complex data types (that themselves represent a structure), such as objects, structures and arrays, DBeaver provides a tool for breaking them into columns:



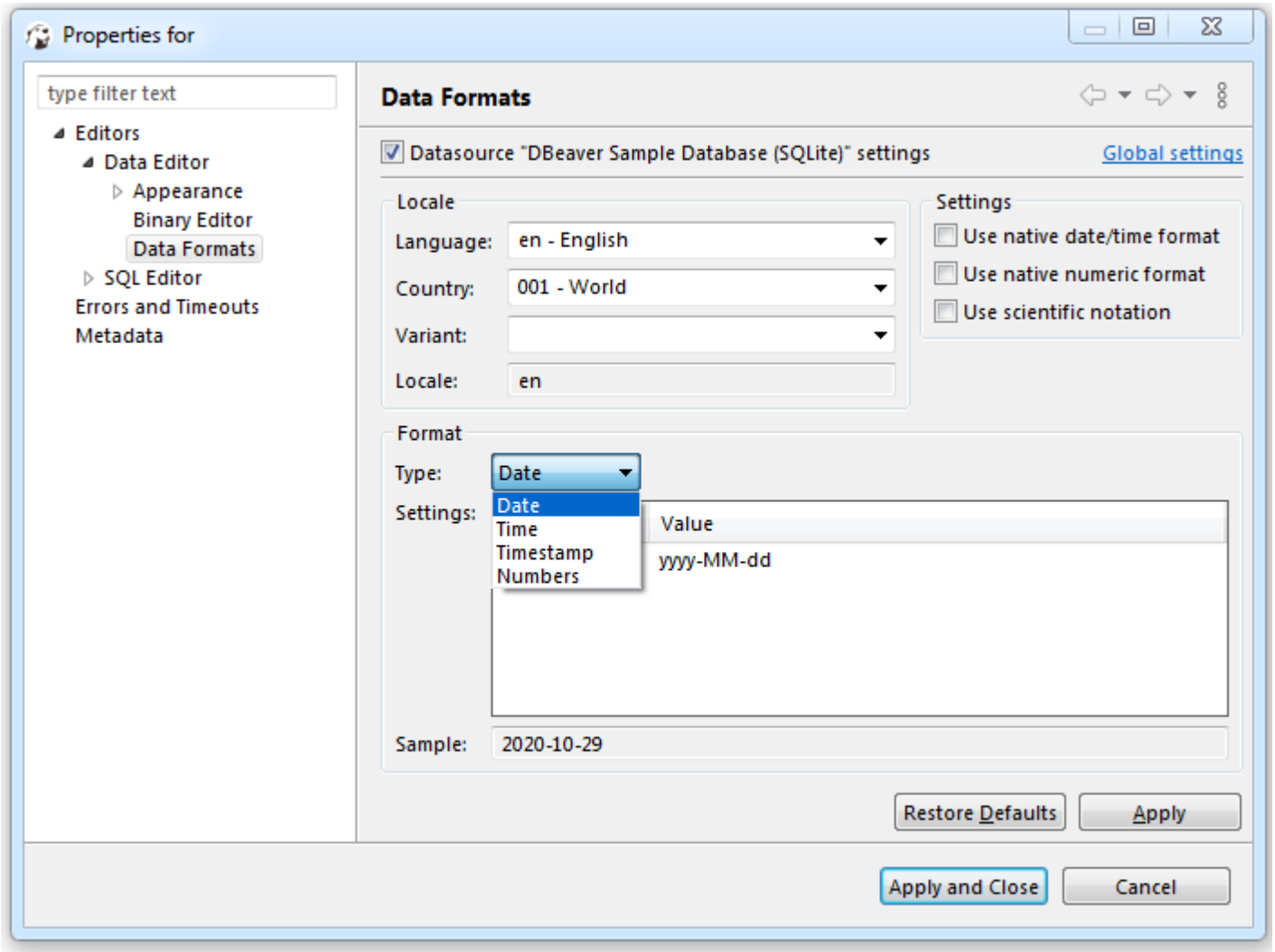
	zipcode	borough	cuisine	grades			name	restaurant_id
				date	grade	score		
1	10462	Bronx	Bakery	2014-03-03 04:00:00	A	2	Morris Park Bake Shop	30075445
2	11225	Brooklyn	Hamburgers	2014-12-30 03:00:00	A	8	Wendy'S	30112340
3	10019	Manhattan	Irish	2014-09-06 04:00:00	A	2	Dj Reynolds Pub And Restaurant	30191841
4	11224	Brooklyn	American	2014-06-10 04:00:00	A	5	Riviera Caterer	40356018

To do so, right-click a cell in the column and, on the context menu, click **View/Format -> Visualize complex columns**

.

Configuring Numeric and Time Data Formats

You can specify the exact format of Time, Timestamp, Date, and Number data used in the currently open database or globally. To specify a format, right-click any cell in the table and, on the context menu, click **View/Format -> Data formats**. The Properties window opens displaying the **Data Formats** page:

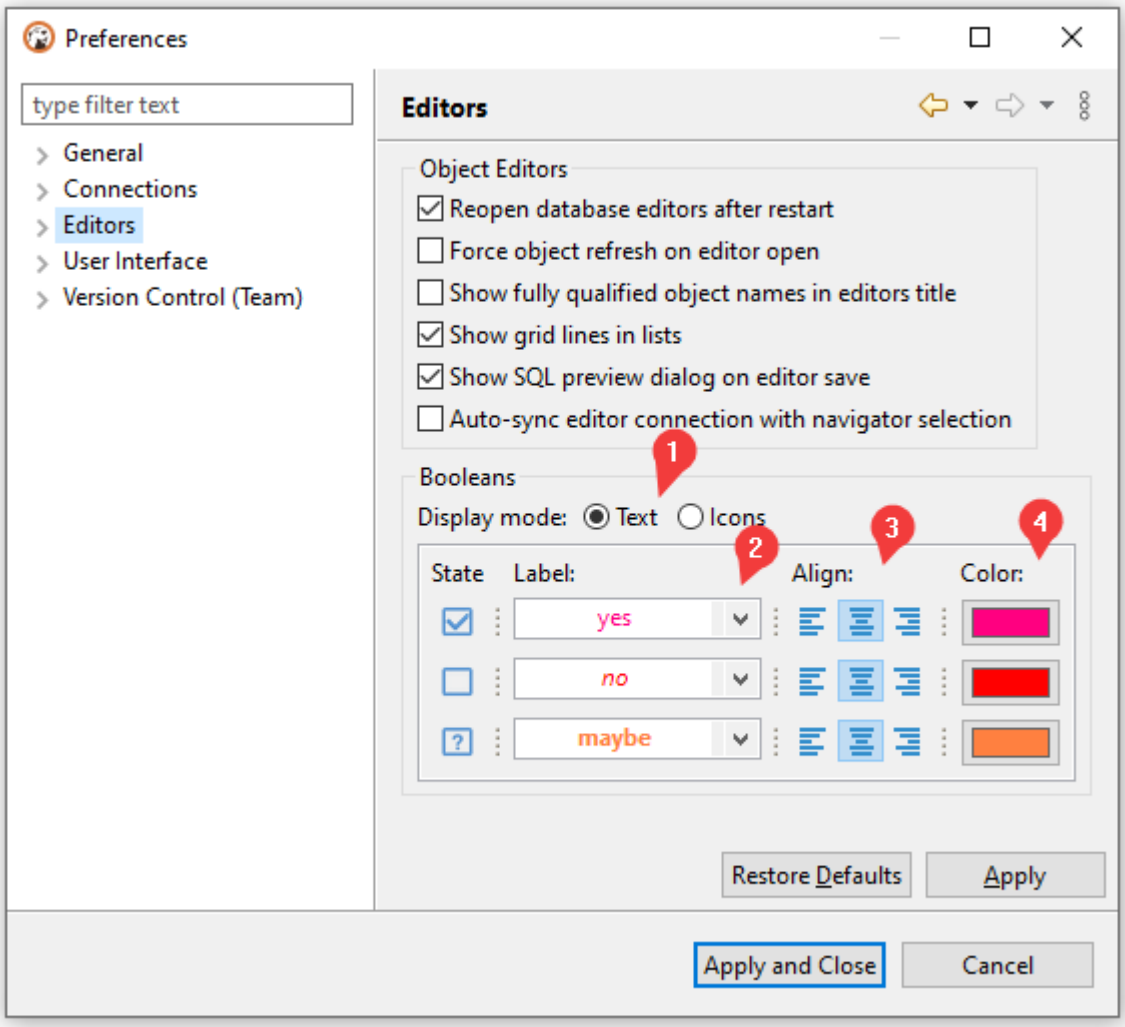


To configure only the format for the current database, select the **Datasource "[Connection name]" settings** checkbox. To configure the settings globally, to all databases that you have in DBeaver, click **Global settings**.

You can specify the locale for the data format in the **Locale** area. In the **Type** dropdown list, click the name of the data type and in the **Settings** table, click the required format.

To apply the changes and make them visible in the table, click **Apply and Close** and then refresh the window (F5).

Configuring Boolean presentation



You can choose between two presentation modes[1]:

- Text-based
- Icon-based

Text-based presentation

This is the most customizable mode. You can:

1. Change **labels** under `Label` column.

   Otherwise, you can use presets available in `Drop-down Menu` [2]

2. Change **alignment** of value inside grid cell.[3]

   Following variants are available: **left**, **center**, and **right**

3. Change **color** of value using color picker under `Color` column[4].

   You can reset color to match current theme's contrast color in `Drop-down Menu` [2] ⇒ `Colors` ⇒ `Use the`

4. Change font **style** in `Drop-down Menu` [2] ⇒ `Styles` .

   Following variants are available: normal, **bold**, and *italic*

Icon-based presentation

This presentation only supports alignment changing.

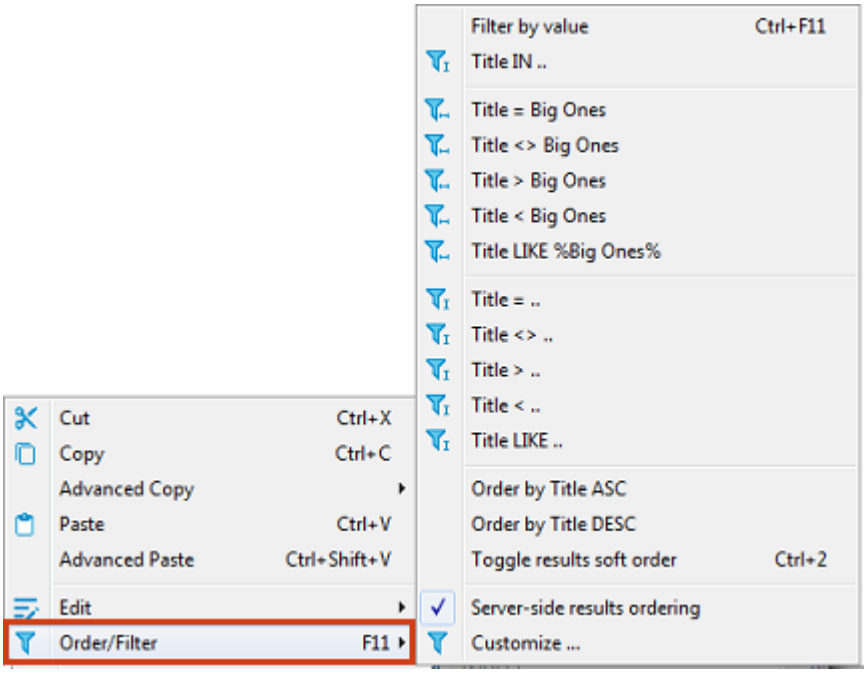
Data Filters

You can apply custom filters to table contents or query results. There are several ways in which you can filter data in the table.

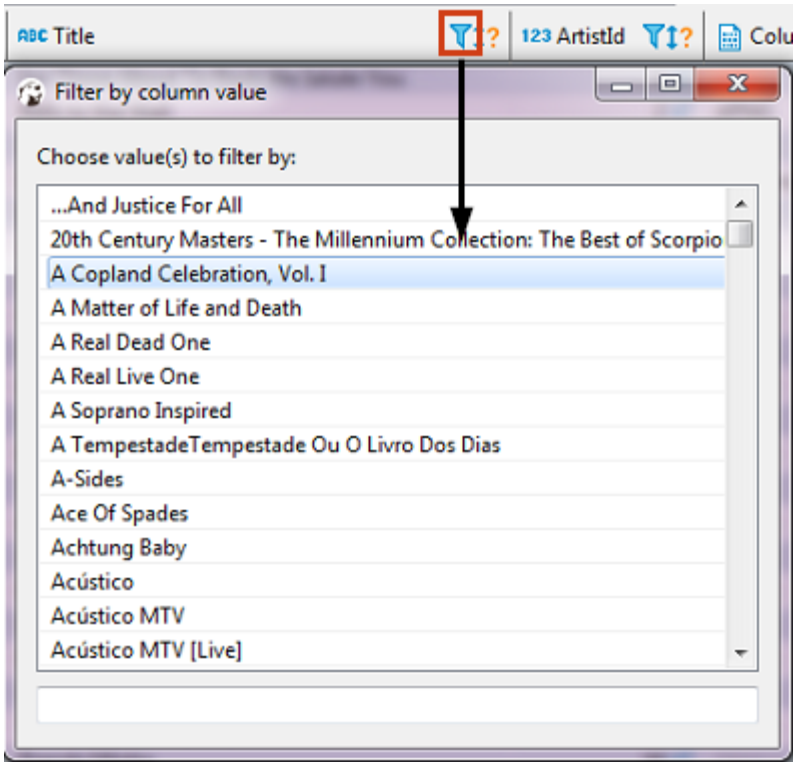
One of the ways is to use the filter field above the table next to the top toolbar. To filter data, enter an SQL expression into the field and click the Apply filter criteria button ( ) next to the field or press `Enter` .



You can apply ready-to-use SQL expressions or SQL expression templates via the context menu. To select a ready SQL expression or a template, press `F11` or right-click the cell, then click **Order/Filter** on the context menu and then click one of the expressions.



The third way is to filter data by a cell value using the filter icon in the column header. To filter data this way, click the filter icon in the column header and then double-click the cell value in the Filter by the column value dialog box:

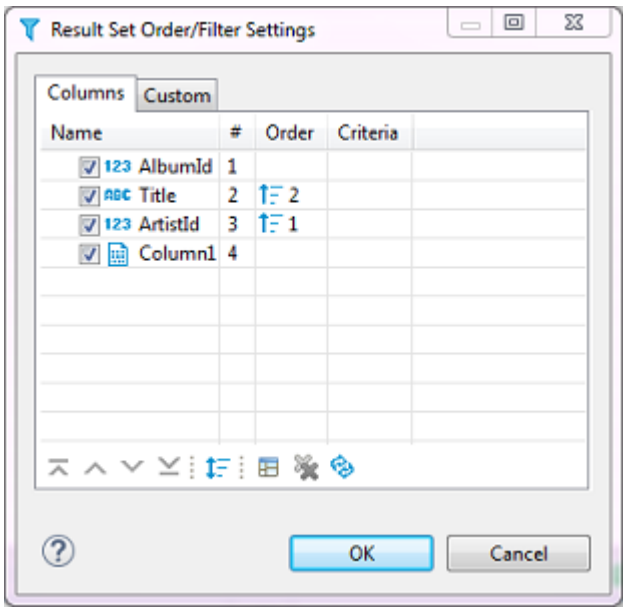


The data updates dynamically. To remove a filter, click the Remove All Filters/Orderings button ( ) in the top toolbar of the editor.

You can save the current filter settings for the database object to apply the next time you reopen it in the editor. To save the current filter settings, click the Save filter settings for current object button ( ) in the top toolbar.

Advanced filters configuration

The main tool for managing the appearance of the data table is the Result Set Order/Filter Settings window.

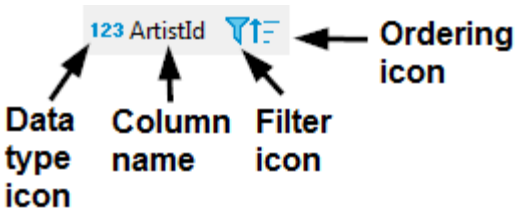


To open this window, click the Custom Filters button ( ) in the top toolbar of the editor or click the Configure button ( ) and then click **Order/Filter** on the dropdown menu.

The Result Set Order/Filter Settings window provides tools to:

- Order data inside columns
- Manage the display of columns in the table
- Manage the order of columns in the table
- Filter data in the table using an SQL expression

Another tool for managing the data appearance is the column headers. In the data table, every column header contains three elements which each has its own function: Data (column) type icon, column name, filter icon, and ordering icon.



- By simply clicking the column name or column type, the icon highlights the whole column.
- You can click the column type icon and then drag and drop the column to a different position in the table.
- You can click the column name and then drag the cursor to the right or left to highlight multiple columns.
- Clicking the ordering icon allows you to order the data in the column in ascending or descending order - see the 'Ordering Data in Columns' section, further in this article
- Clicking the filter icon allows you to filter the data by a cell value, see [TBA]

Ordering Data in Columns

You can order data in columns in one of the following ways:

1. Click the ordering icon ( ) in the header of the column.



The icon has three states:

- Clicking once establishes ascending order ( )
- Clicking a second time changes the order to descending ( )
- Clicking a third time removes the ordering from the column ( )

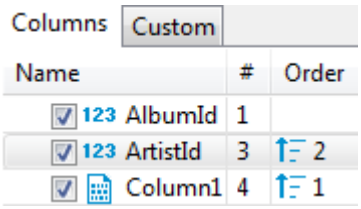
To order data by several columns, go column by column, setting the order with the Ordering icon, starting from the column by which you want to order data first.

2. Click the Custom Filters button ( ) in the top toolbar of the editor to open the Result Set Order/Filter Settings window (see above):

   a) Next to the column by which you want to order data in the first turn, set the ascending or descending order using the same three-state principle as described above.

   b) Set the ordering in other columns by which you want to sort the data in the second, third, etc. turn. The **Order** column indicates the order in which the sorting will happen.

   NOTE: The number (**#**) column indicates the initial order of columns.



   c) To easily move the ordering setting from column to column, you can use the Move up/down/to top/to bottom/ buttons:

To reset the data ordering to its initial state, click the Reset button ( ) in the same window.

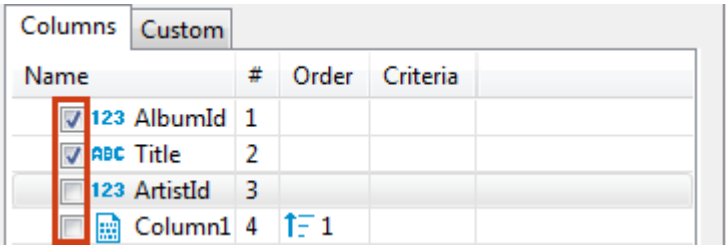
Also, to remove all ordering settings, click the Remove All Filters/Orderings button ( ) in the top toolbar of the Data Editor.

Managing Display of Columns in Data Table

To hide a single column, right-click the column or any cell in it and click **View/Format -> Hide column** on the context menu. To unhide a hidden column, open the Result Set Order/Filter Settings window (see the image at the beginning of this article) and select the checkbox next to the column name, or click the Reset button ( ).

To display or hide columns in the data table, in the Result Set Order/Filter Settings window:

1. Select the checkboxes next to the columns that you want to see in the table and clear the checkboxes next to those that you want to hide.



2. Use the Show All ( ) and Show None ( ) buttons at the bottom of the window.

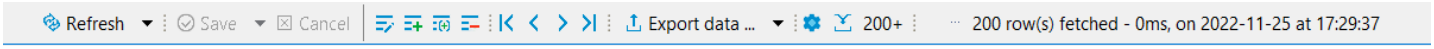
Sorting Columns in Data Table

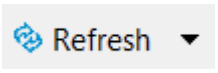
You can modify the order of columns in the data table in two ways:

1. Click the icon in the column header and drag-and-drop the column to a new position.
2. To sort the column alphabetically, in the Result Set Order/Filter Settings window (open by clicking the Custom Filters button ( ) in the top toolbar of the editor), click the Sort button ( )
3. In the Result Set Order/Filter Settings window, click the column to set the focus to it and then move it using the navigation buttons: ( )

Data Refresh

In the bottom left part of the Data editor, you can find the toolbar with actions:

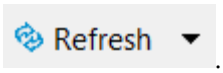


Refreshing is necessary if the database contains changes made by other users working on it simultaneously with you, and you want to see them in your DBeaver window. To refresh data manually, right-click anywhere in the data table and click 

on the context menu or press F5 .

Autorefresh

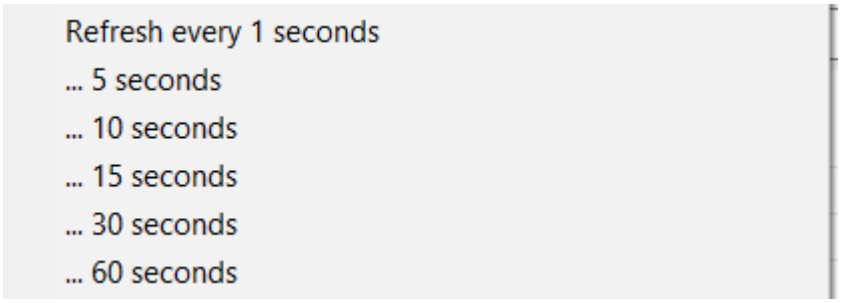
You can also schedule auto-refresh to happen regularly. To auto-refresh the database on schedule:

Click on the arrow in the auto-refresh button 

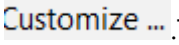
. Now you can configure auto refresh in the two following ways:

1. Use a predetermined duration

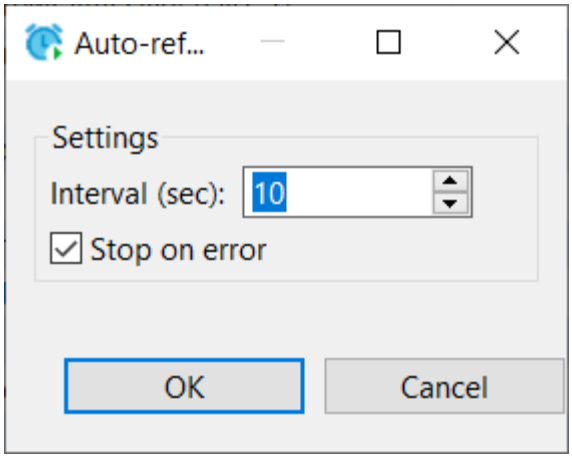
You can select via choosing an available predetermined period in the context menu.



2. Set your duration

You can set your time duration between refreshes. To do that, select 

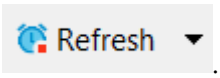
.The Auto-refresh configuration dialog box will open:



a) Set **Interval** in seconds.

b) Select the **Stop on error** checkbox if you want the refresh to stop when it encounters an error or clear it if the refresh should ignore mistakes.

c) Click **OK**.

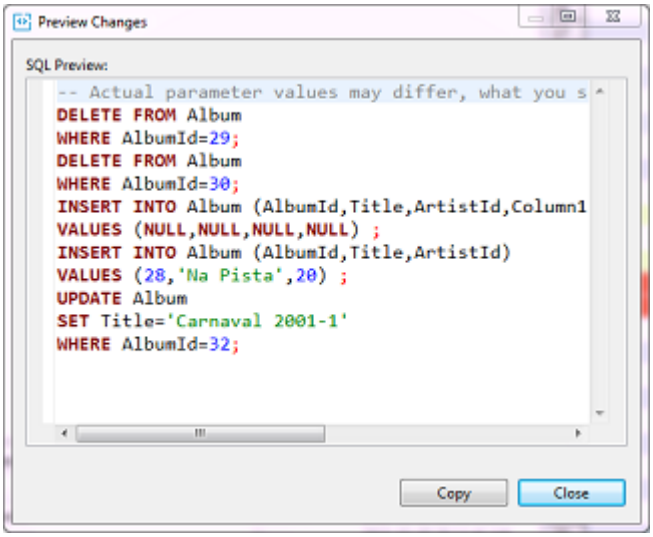
When you perform either of these two alternative options above, the system starts refreshing the data as scheduled, and the Configure auto-refresh button changes to the **Stop auto-refresh** button 

. To stop the auto-refresh, click the **Stop auto-refresh** button or the arrow next to it and click **Stop** on the auto-refresh menu.

Data Viewing and Editing

You can do inline editing (see the *Inline Editing* section below) as well as open the content of a cell in a separate editor (see the *Cell Editor* section below).

When you make any changes to the data and save them using the steps described in this section, the changes will apply to the database itself. Prior to saving the changes, you can preview the SQL script that the system sends to the database to apply the changes there. To see the SQL script, after making changes and before saving or discarding them, click the **Script** button (Script) in the bottom toolbar. The Preview Changes window opens, in which you can only view the SQL script and copy it, if necessary:



Inline Editing

Inline editing is when you modify the content right in the cell. To edit a cell inline, in the table do one of the following:

- Double-click the cell.
- Click the cell to set focus to it and press `Enter`.
- Right-click the cell and click **Inline edit** on the context menu.

The cell is now editable and you can change its value.

To set the cell value to NULL, right-click the cell and click **Set to NULL** on the context menu.

To save the changes, click the **Save** button () in the bottom toolbar. To discard the changes, click the **Cancel** button () in the bottom toolbar.

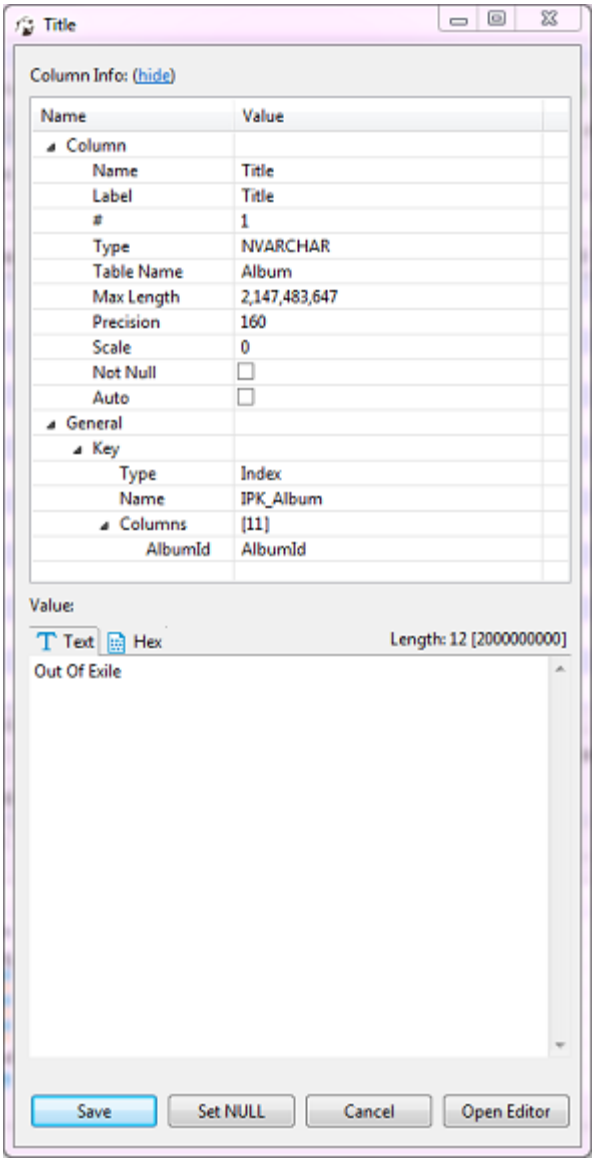
NOTE: Both the **Save** and **Cancel** buttons become editable only when you make changes in a cell and then jump to another cell.

Cell Editor

To edit data in a cell using a separate editor, do one of the following:

- Right-click the cell and click **Edit cell** on the context menu.
- Click the cell to set focus to it and press `Shift+Enter` or click the **Edit cell value in separate dialog/editor** button () in the bottom toolbar.

For cells of CLOB/BLOB data format, this action opens the contents of the cell in a new tab. For all other formats except CLOB/BLOB, this action opens a properties window for the cell:



The window displays properties of the column in the **Column Info** section and provides the **Value** section where you can modify the value of the cell. Edit the value as required and click **Save**. To set the value to NULL, click **Set NULL**. To continue editing the cell in a separate editor (tab), click **Open Editor**.

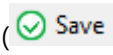
NOTE: DBeaver has full support of CLOB/BLOB data types. You can view values, edit them, and save them back to the database. You can open CLOB/BLOB value in a separate editor (press `Shift+Enter` on a selected cell). You can save/load LOB value to/from regular files. DBeaver can recognize that some BLOB column keeps images (gif, png, jpeg, bmp). In this case DBeaver shows LOB contents as an image. It is convenient to open the value view panel (press F7) and browse images.

Adding, Copying and Deleting Rows

You can add an empty row below the row in focus. To add an empty row, click the **Add new row** button () on the bottom toolbar. Use inline editing or open the cell values in a separate editor to populate them with data (see the sections above).

You can copy any row or several rows currently in focus. To copy rows, highlight one or more rows and click the **Duplicate current row** button () in the bottom toolbar. The duplicate rows appear below the rows in focus.

To delete a row or rows, set the focus to the rows and click the **Delete current row** button () in the bottom toolbar. The rows are coloured red, which means that they are marked for deletion and will be deleted when you save the changes.

To save any such changes, click the **Save** button (

) on the bottom toolbar. To discard the changes, click the **Cancel** button (

) on the bottom toolbar.

NOTE: Some of these buttons may be disabled and may not work if you are using a read-only connection, connecting to a read-only database or if you see the result of a complex query, such as joining two or more tables.

Copying/Pasting Cells

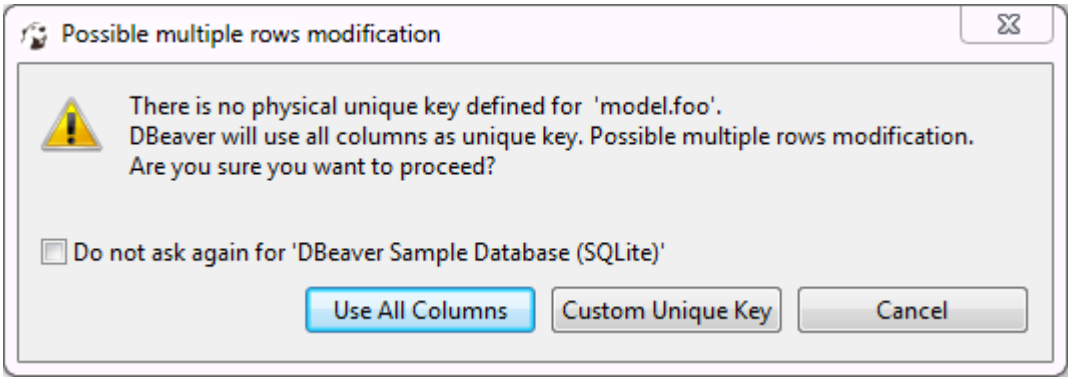
To copy the content of one or several cells to the clipboard in the TAB-delimited format, press `Ctrl+C` or right-click the cell or cell selection and click **Copy** on the context menu. Then you can paste the copied selection into a spreadsheet editor (similar to Excel).

DBeaver provides the advanced copy option that allows configuring additional copy settings (copy with column names/row numbers, configure delimiter and choose value format). To copy cells with additional settings, press `Ct:` or right click the cell(s) and click **Advanced Copy** on the context menu.

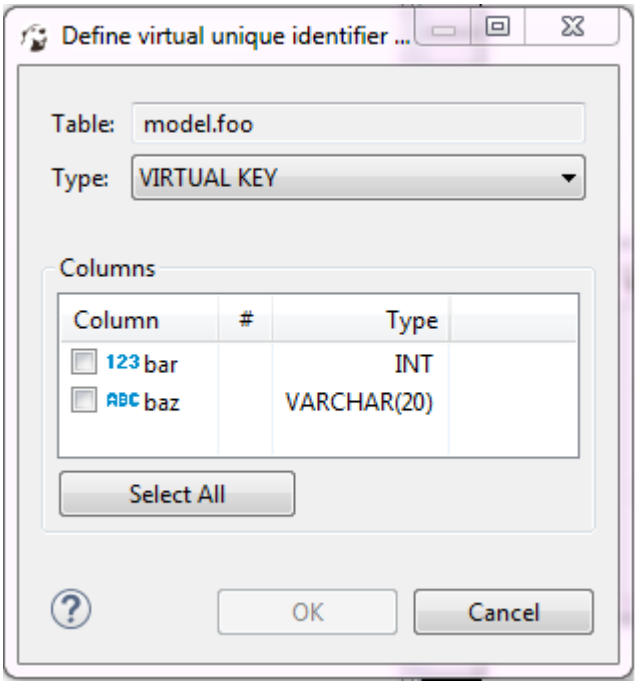
Pressing `Ctrl+V` on a cell pastes the copied content into the cell applying appropriate data type conversion. The **Advanced Paste** option on the context menu or pressing `Ctrl+Shift+V` pastes several cells.

Defining Virtual Keys

To be able to save column value changes, a table must have some unique key (primary key or unique index). Some databases (Oracle, DB2, PostgreSQL) support a special virtual unique column that DBeaver can use to save changes. In other cases, you can define a virtual key – a set of columns that forms a unique combination of values. When you try to save changes in a table without a unique key, DBeaver displays the following error message:



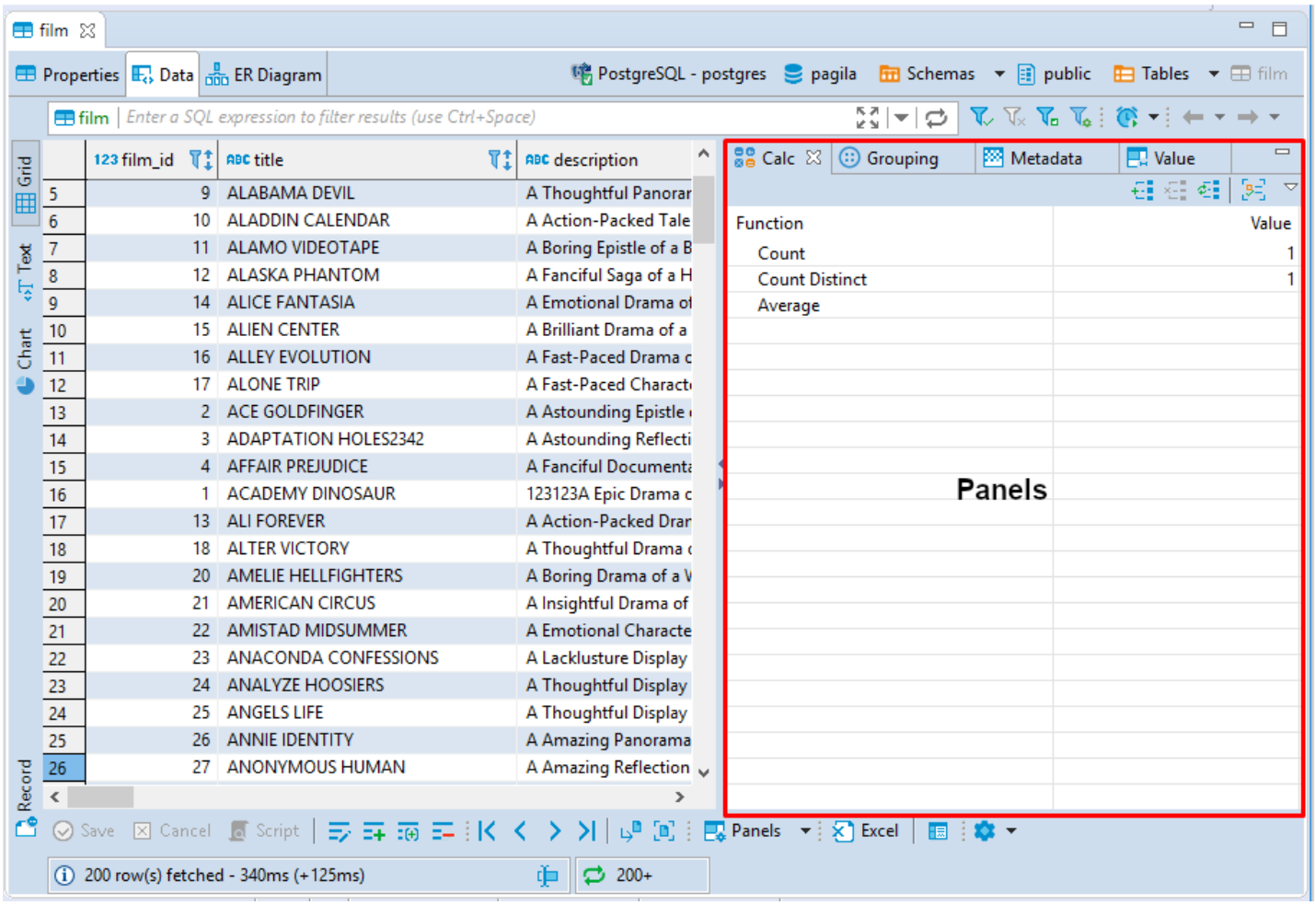
To use all columns as the virtual key, click **Use All Columns**. To create a custom key, click **Custom Unique Key**. Alternatively, to create a unique custom key, you can click the **Configure** ( ) button on the bottom toolbar and then click **Define virtual unique key** on the Configure menu. The Define virtual unique identifier window opens:



To define the key, select some of the columns or click **Select All** and then click **OK**. To remove a unique key from a table, click the **Configure** button in the bottom toolbar and then click **Clear virtual unique key**.

Panels

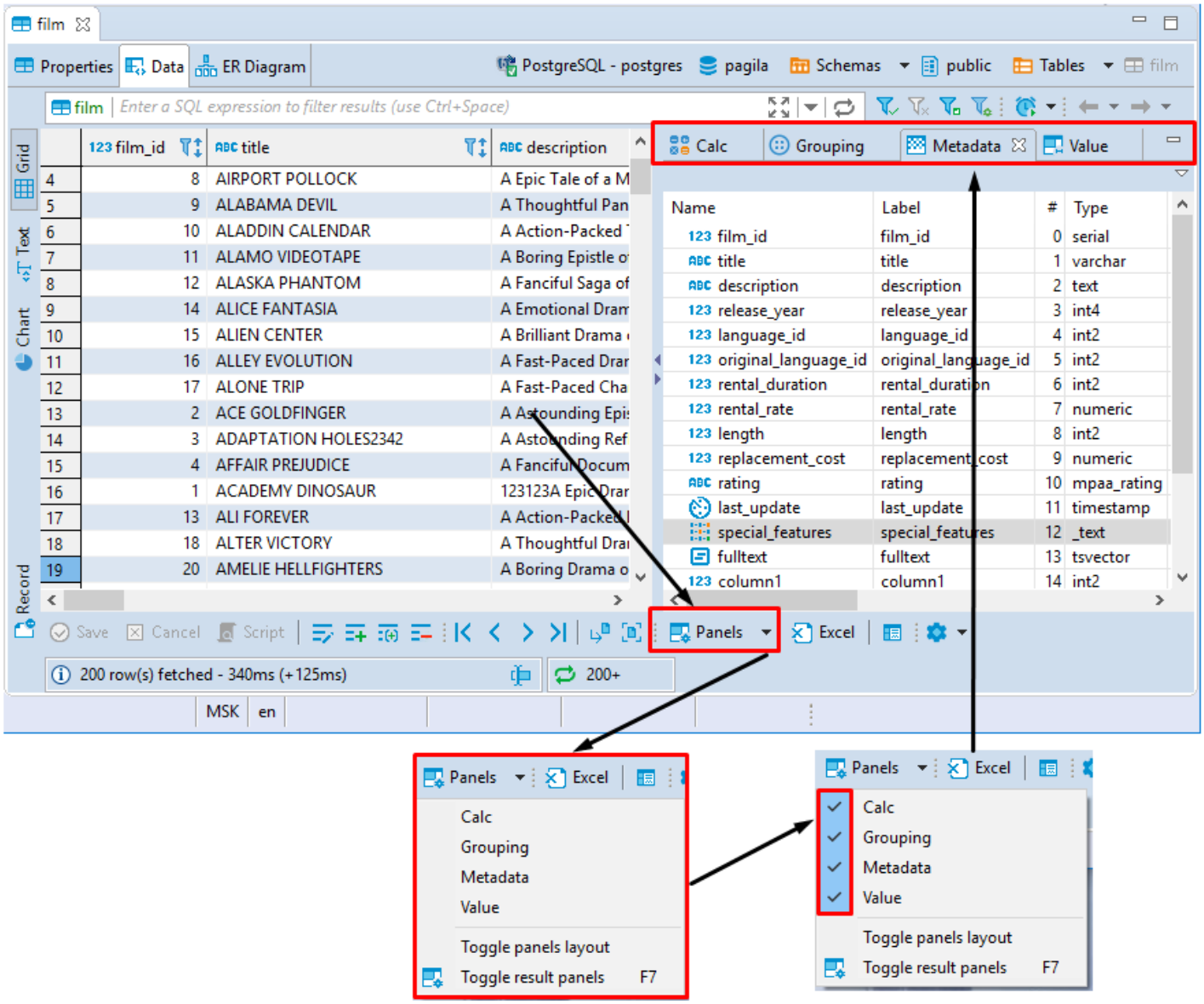
Panels provide additional space in the Data editor in which you can manipulate data. The panels are handy if you work with complex types (structures, arrays), long text data, or BLOBs. Panels appear as tabs in an additional pane in the right hand side of the Data tab:



This additional pane appears only when you open one of the four panels:

- Calc
- Grouping
- Metadata
- Value viewer (default)

To open the panels, click **Panels** on the bottom toolbar. By default, the Value viewer panel opens. Alternatively, you can open the Value panel by pressing F7 on a cell. To open the other panels, click the down arrow next to the **Panels** button and click the name of the panel on the menu:

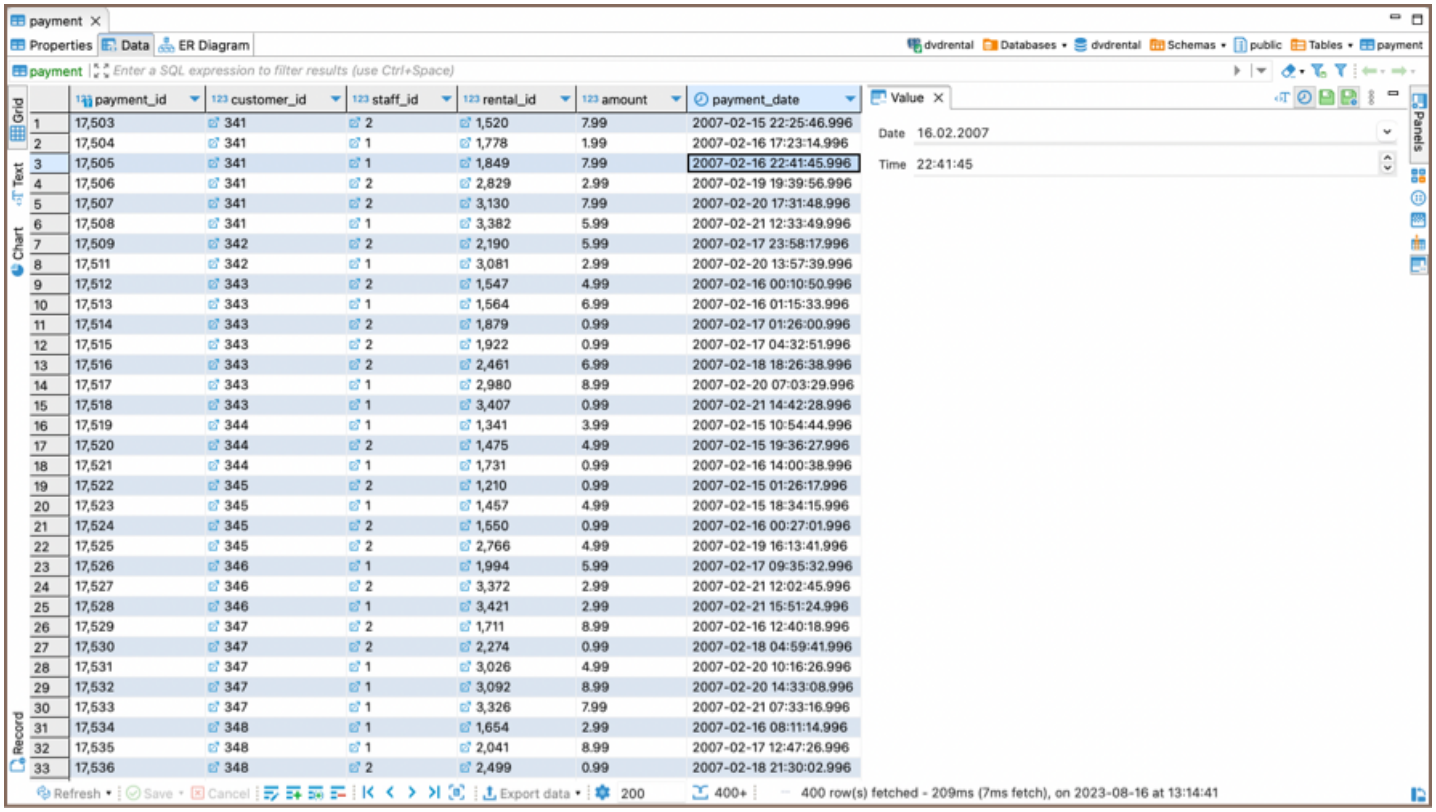


Panels will also open if you try to inline-edit a cell with a complex data type.

To close the panels, click the **Panels** button again or click the standard Close (cross) icon in the upper right corner of each panel.

You can also show and hide panels by clicking the **Configure** button ( ) on the bottom toolbar and then **Toggle result panels** on the Configure dropdown menu.

Value Viewer



The **Value Viewer** panel displays the value currently in focus and allows editing. The toolbar of the **Value Viewer** contains the following common buttons for all data types:

Button	Name	Description
	Apply cell changes	Displays the data table changes in the Value viewer.
	Auto-apply value	Enables the automatic display of changes made in the Value viewer in the data table. When auto-saving is enabled, the changes appear in the data table at the same time when they are made in the Value viewer.

Note: This buttons does not save changes made to the database. To save the changes in the database, you need to use the **Save** button on the bottom toolbar of the Data Editor.

Additional buttons may appear depending on the specific data type being viewed (e.g., `Arrays`, `JSON`, `datetime`, Binary content).

Dictionary handling in Value Viewer

The **Value Viewer** provides specific handling for dictionaries when dealing with foreign keys. If the table contains foreign keys, the **Value Viewer** will display a dictionary with values. For data types that are not `string`, `numeric`, or `datetime`, only one value appear in the dictionary panel.

Working with dictionaries

When working with dictionaries in the **Value Viewer**, the following functionalities are available:

- **Edit Value**: Displays the current value, which can also be edited within the **Value Viewer**.
- **Dictionary** field: Displays the table referenced by the foreign key. Clicking on it will open a new tab with that table.
- **Define Description** field: Clicking on this field will open a window where you can:
  - Choose dictionary description columns or set custom criteria.
  - Define a custom expression for description.
  - Set a column delimiter.
- **Search**: Allows you to type dictionary value or part of description string to search within the dictionary entries.

Preferences Settings for dictionaries

You can further customize the behavior of dictionaries in the Value Viewer by accessing the preferences settings. Navigate to **Window** -> **Preferences** -> **Editors** -> **Data Editor** -> **Dictionaries**.

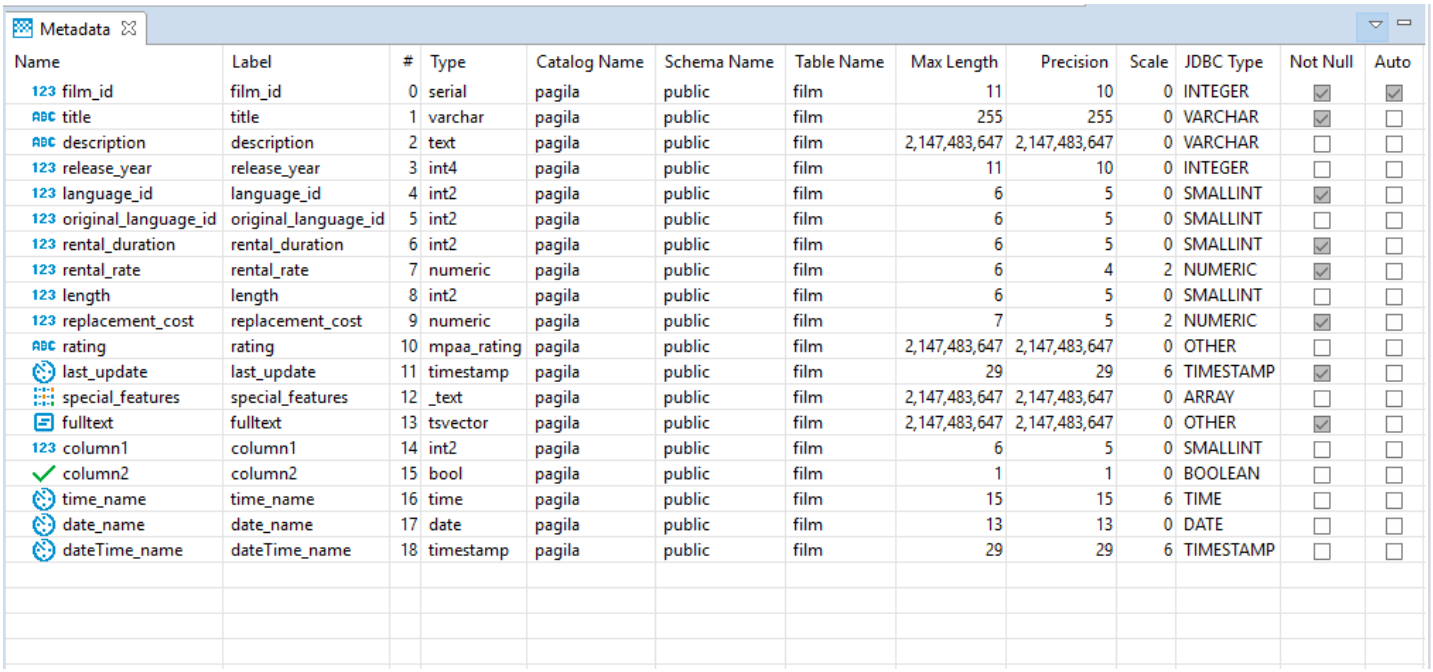
Within these settings, you can configure the following options:

- **Keywords for dictionary description columns**: Allows you to specify keywords that will be used for dictionary description columns.
- **Maximum amount of elements on a dictionary panel page**: This enables you to set a limit for the number of elements displayed on a single page within the dictionary panel.

These preferences provide additional control over how dictionaries are handled and displayed within the **Value Viewer**.

Metadata Panel

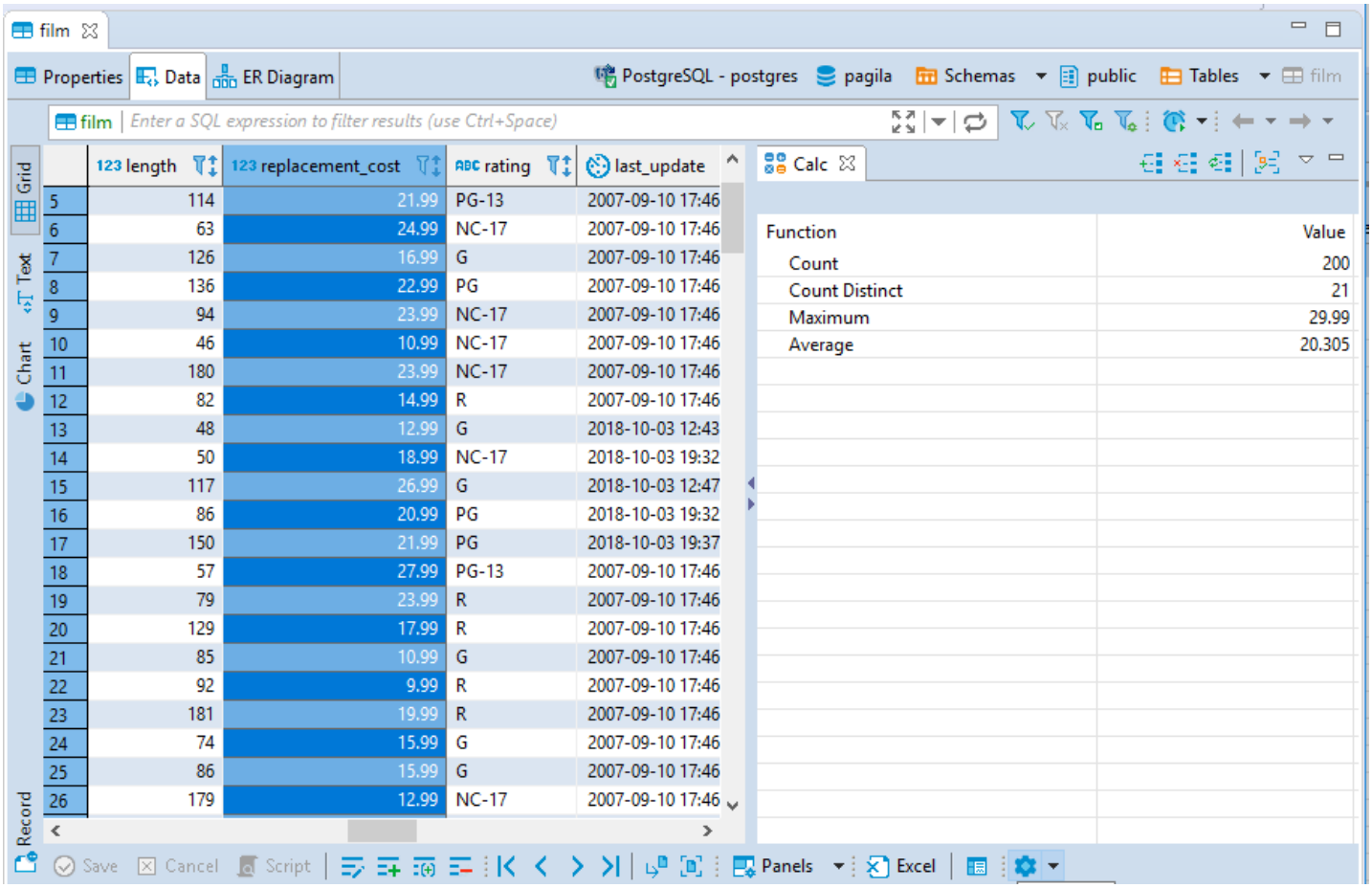
The Metadata panel displays metadata for each cell in the row containing the cell currently in focus. You can just view the metadata.



Name	Label	#	Type	Catalog Name	Schema Name	Table Name	Max Length	Precision	Scale	JDBC Type	Not Null	Auto
film_id	film_id	0	serial	pagila	public	film	11	10	0	INTEGER	☑	☑
title	title	1	varchar	pagila	public	film	255	255	0	VARCHAR	☑	☐
description	description	2	text	pagila	public	film	2,147,483,647	2,147,483,647	0	VARCHAR	☐	☐
release_year	release_year	3	int4	pagila	public	film	11	10	0	INTEGER	☐	☐
language_id	language_id	4	int2	pagila	public	film	6	5	0	SMALLINT	☑	☐
original_language_id	original_language_id	5	int2	pagila	public	film	6	5	0	SMALLINT	☐	☐
rental_duration	rental_duration	6	int2	pagila	public	film	6	5	0	SMALLINT	☑	☐
rental_rate	rental_rate	7	numeric	pagila	public	film	6	4	2	NUMERIC	☑	☐
length	length	8	int2	pagila	public	film	6	5	0	SMALLINT	☐	☐
replacement_cost	replacement_cost	9	numeric	pagila	public	film	7	5	2	NUMERIC	☑	☐
rating	rating	10	mpaa_rating	pagila	public	film	2,147,483,647	2,147,483,647	0	OTHER	☐	☐
last_update	last_update	11	timestamp	pagila	public	film	29	29	6	TIMESTAMP	☑	☐
special_features	special_features	12	_text	pagila	public	film	2,147,483,647	2,147,483,647	0	ARRAY	☐	☐
fulltext	fulltext	13	tsvector	pagila	public	film	2,147,483,647	2,147,483,647	0	OTHER	☑	☐
column1	column1	14	int2	pagila	public	film	6	5	0	SMALLINT	☐	☐
column2	column2	15	bool	pagila	public	film	1	1	0	BOOLEAN	☐	☐
time_name	time_name	16	time	pagila	public	film	15	15	6	TIME	☐	☐
date_name	date_name	17	date	pagila	public	film	13	13	0	DATE	☐	☐
dateTime_name	dateTime_name	18	timestamp	pagila	public	film	29	29	6	TIMESTAMP	☐	☐

Calc Panel

The Calc panel is useful for getting basic statistics across data in several columns and rows:



You can select several columns and rows in standard ways - by pressing and holding the left mouse button or by clicking cells while holding the `Ctrl` or `Shift` keys. The panel updates dynamically to show statistics for the selected data.

To see the data grouped by columns, click the Group by columns button ( ). To remove the grouping by columns and see the summary values for all columns, click the same button again.

By default, the panel applies and displays results for two functions – **Count** and **Count Distinct**. To add other functions, click the **Add function** ( ) button on the toolbar of the panel or right-click one of the rows in the Aggregate panel and click **Add function** on the context menu and then click the name of the function. The following functions are available:

- Sum
- Average
- Minimum
- Maximum
- Median

- Mode

To remove an individual function, click the function and then click **Remove function** ( ) on the toolbar of the panel, or right-click the function and click **Remove function** on the context menu. To remove all functions, click **Reset** ( ) on the toolbar or on the context menu.

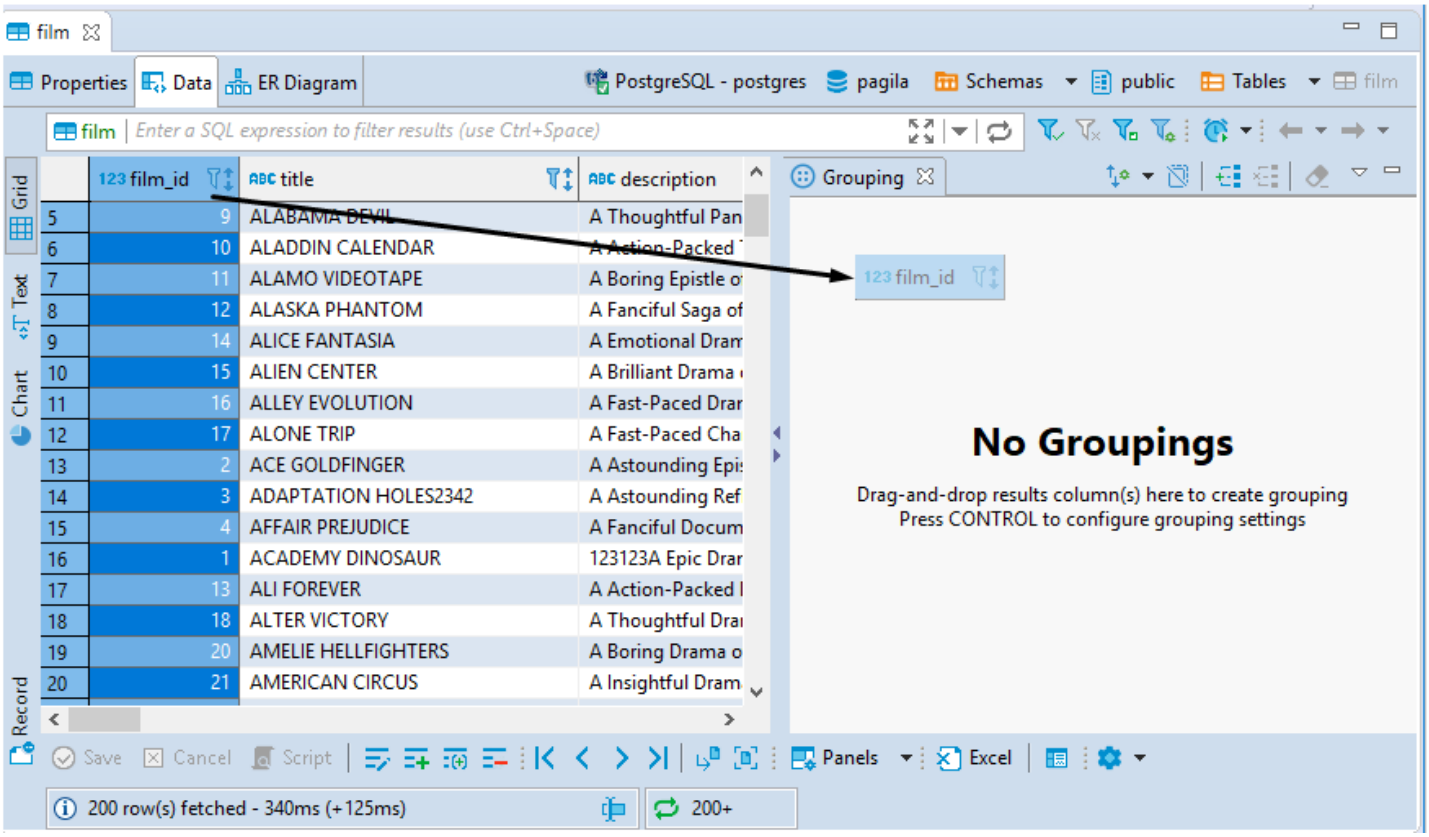
You can copy the value of a particular function to the clipboard - right-click the row and click **Copy Value** on the context menu.

You can also copy all functions with their values - right-click in the table and click **Copy All** on the context menu.

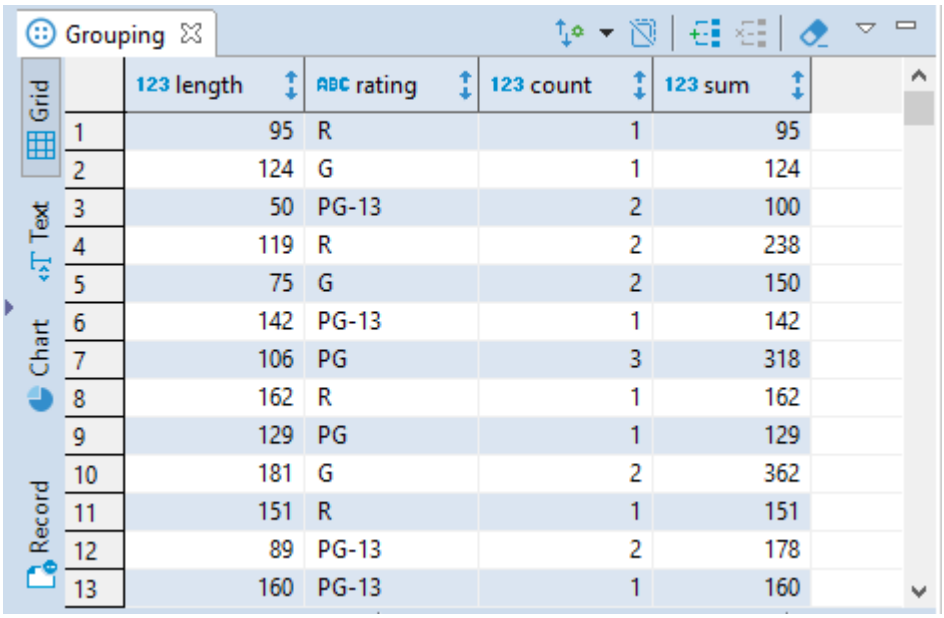
Grouping Panel

The Grouping panel provides tools to calculate statistics based on a table of a custom SQL query. It uses GROUP BY queries to extract unique values for COUNT (default), SUM, AVG, MIN, MAX and other analytics functions displaying the results in dedicated columns.

To obtain the grouping results for one or more columns of a data table, open the Grouping panel, then, in the results table, put the cursor onto the data type icon of the table header so that the cursor turns into a hand pointer ( ), and drag-n-drop the column(s) into the panel:



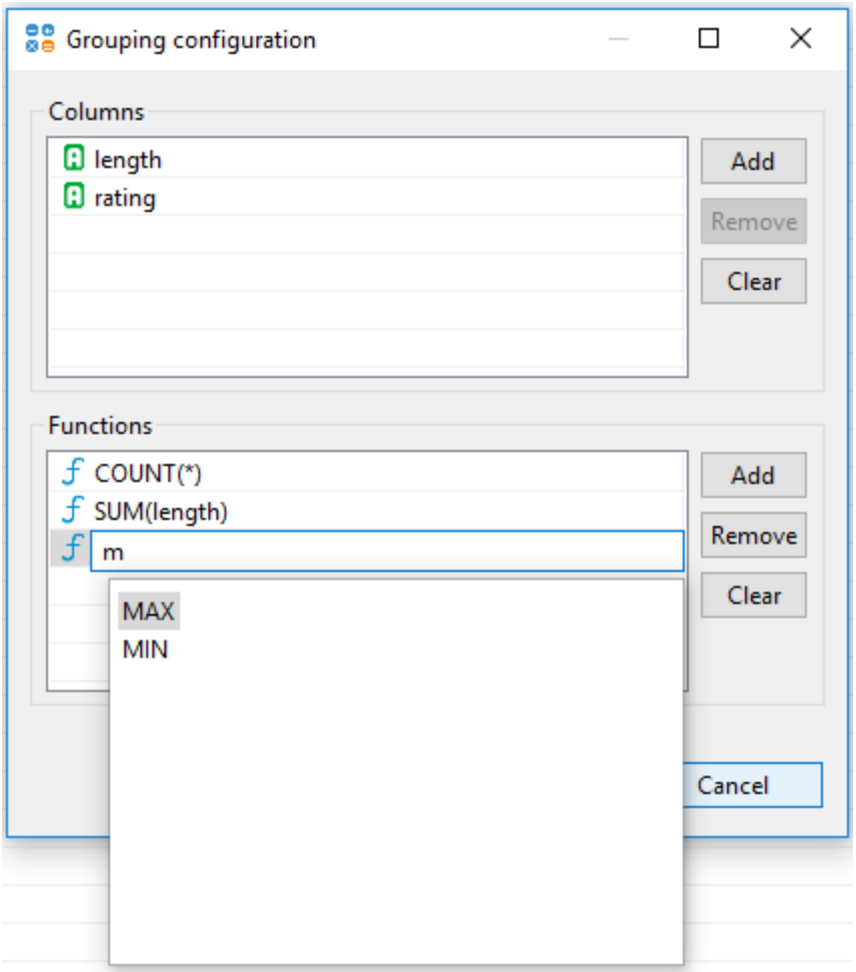
If you add several columns to the panel, DBeaver groups data in the order in which the columns go and calculates statistics based on the grouping.



	123 length	ABC rating	123 count	123 sum
1	95	R	1	95
2	124	G	1	124
3	50	PG-13	2	100
4	119	R	2	238
5	75	G	2	150
6	142	PG-13	1	142
7	106	PG	3	318
8	162	R	1	162
9	129	PG	1	129
10	181	G	2	362
11	151	R	1	151
12	89	PG-13	2	178
13	160	PG-13	1	160

By default, the COUNT function is used. You can add other functions as well. To add a function:

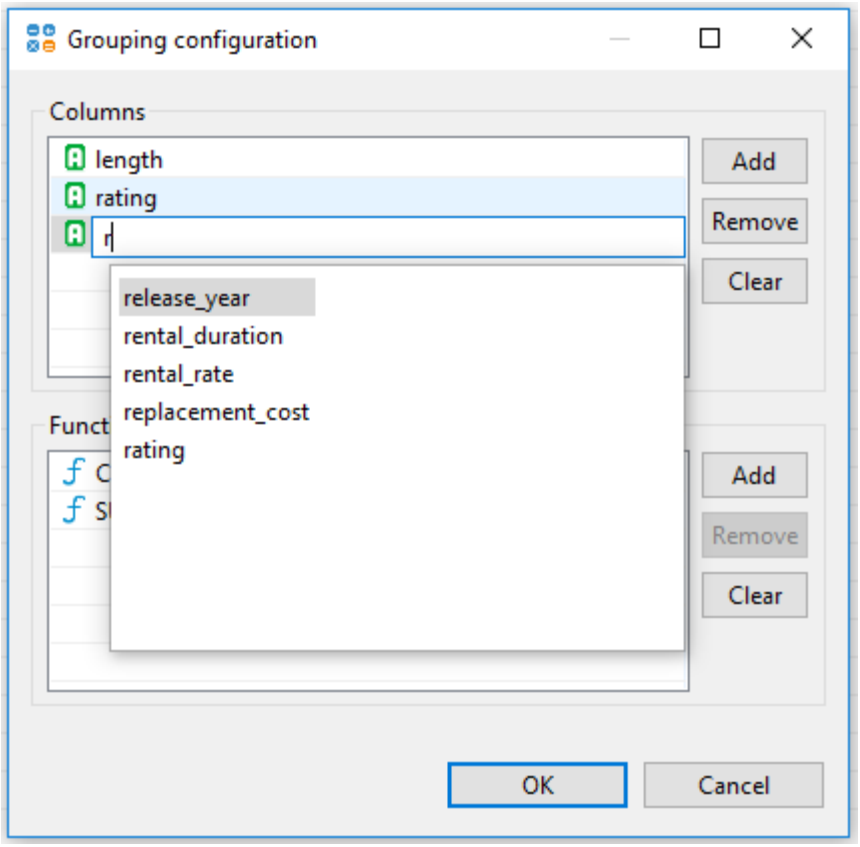
1. Click the **Edit grouping columns** button on the panel`s toolbar.
2. In the Grouping Configuration window, in the **Functions** area, click **Add**, then type the function into the new row:
    - You can use the auto-complete options DBeaver provides.
    - You need to indicate the column name in brackets. COUNT is the only function that supports `*` instead of the column name.
3. Click **OK**:



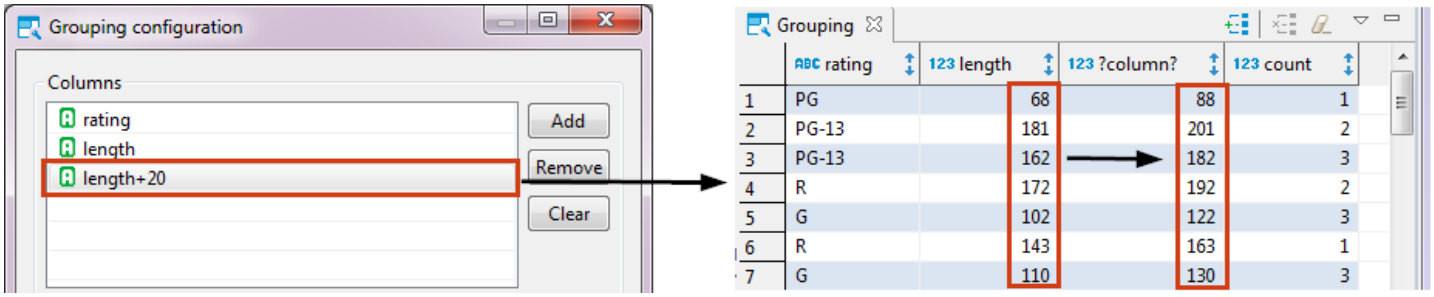
To remove a function, in the same Grouping Configuration window, click the function and click **Remove** and then **OK**. To remove all functions, click **Clear** and then **OK**.

You can also add or remove columns using the same Grouping Configuration window. To add a column:

1. Click the **Edit grouping columns** button on the panel`s toolbar.
2. In the Grouping Configuration window, in the **Columns** area, click **Add**, then type the column name into the new row (you can use auto-complete options DBeaver provides), and then click **OK**:



For MySQL/MariaDB databases you can also add a column with an expression - the expression will be calculated in the resulting column:

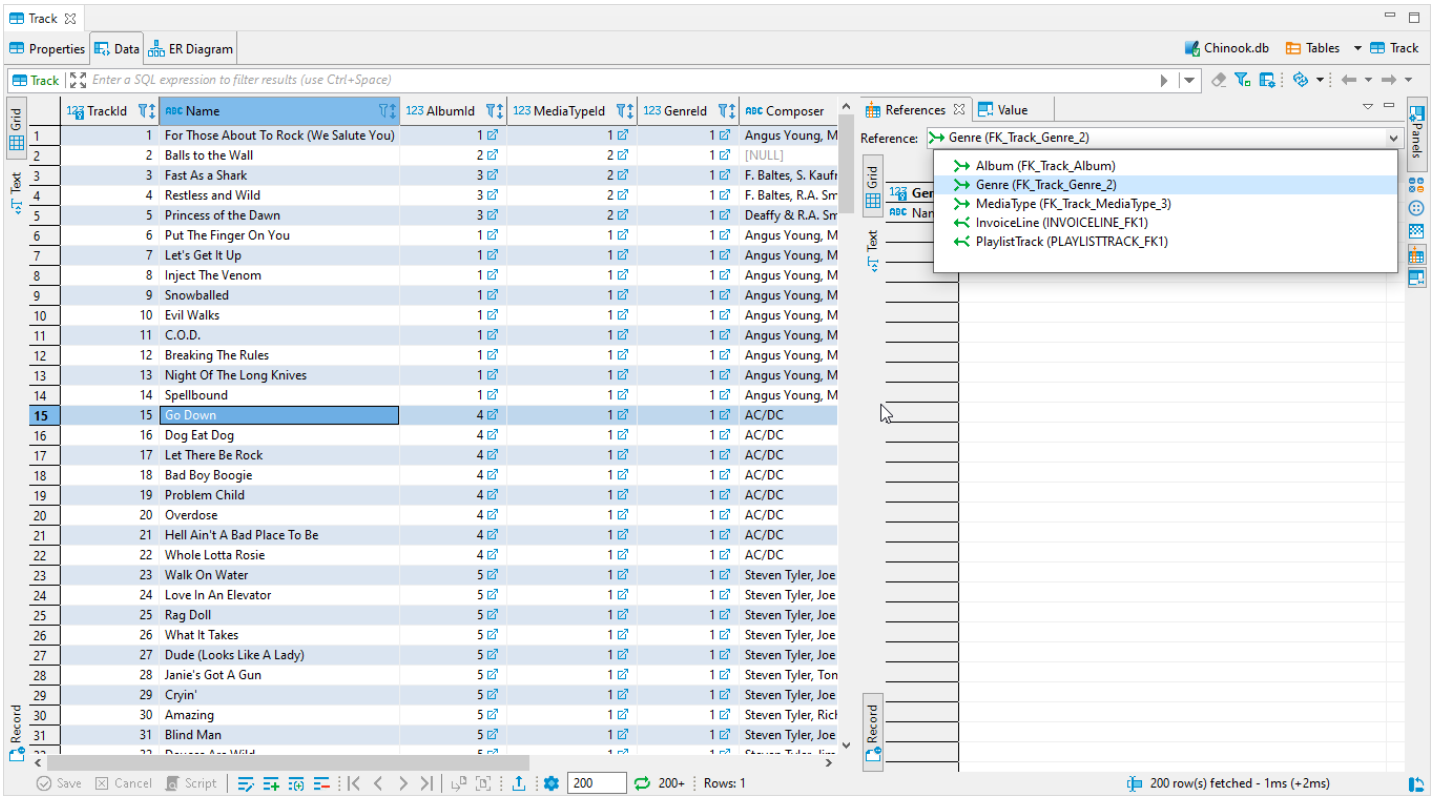


To remove a column, in the Grouping Configuration window, in the **Columns** area, click the column name, then **Remove** and **OK**. To remove all columns, click **Clear** and **OK**.

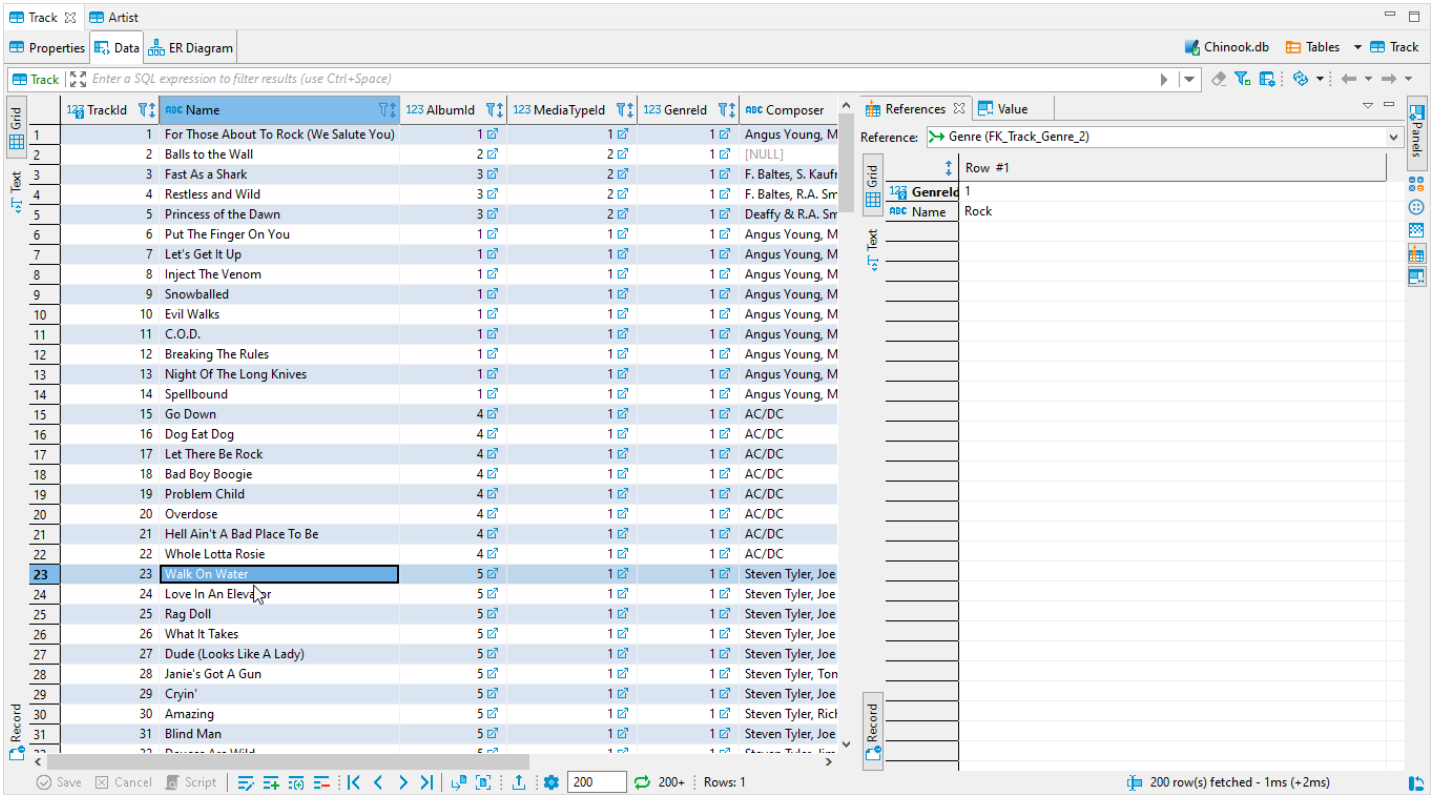
Another way to remove a column is to click the column and then the **Remove grouping column** button ( ) in the panel`s toolbar. Clicking the **Clear grouping** button ( ) removes all results from the Grouping panel.

References panel

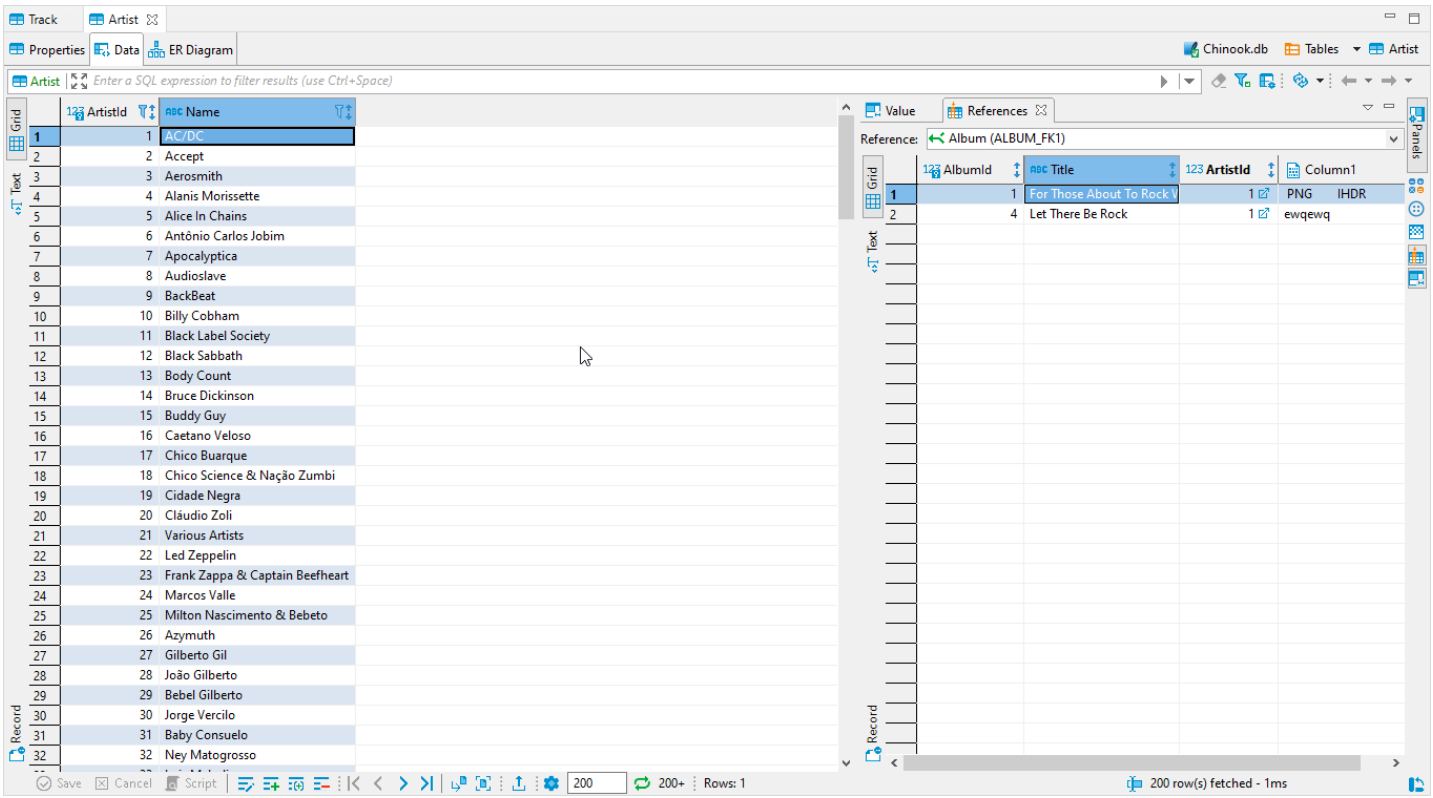
The references panel allows you to see all the related information for the chosen row from other connected tables. The information is presented in an additional data viewer window, filtered to show the information related to the currently selected row. If a table opened in data viewer has a foreign key referencing another table, or it is referenced with a foreign key by another table, all of those connected tables can be picked from a dropdown list.



When a table that is referenced by a foreign key in the current table is chosen, the information from the row corresponding to a referenced key will be shown, in this situation the record mode is enabled by default, but it can be turned off like in a normal data viewer.



When a table that references the current table is chosen, the references panel will show all the rows that refer a selected primary key in the current table.



Managing Charts

Note: This feature is available in Lite, Enterprise, Ultimate and Team editions only.

The default grid view of query results may look unremarkable. Use DBeaver's Charts feature to transform your `SELECT` queries into colorized charts. You can visualize data in **Bar**, **Line**, or **Pie** formats.

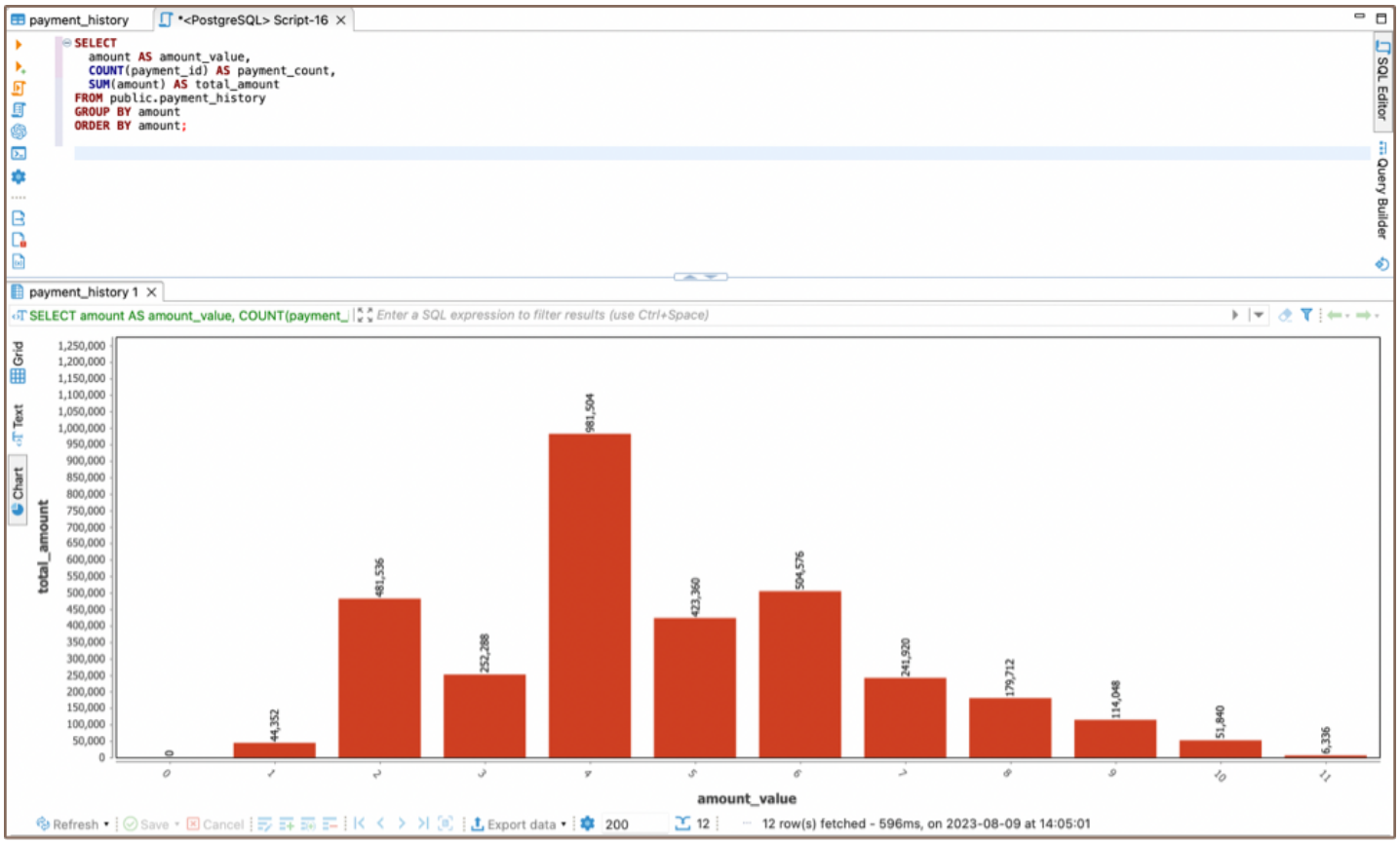
Create a chart directly in the SQL Editor, Data Editor or Grouping Panel. It's a quick and simple way to bring your data to life.

Creating Charts In SQL Editor

Visualizing large data sets makes your analysis faster and more precise. Charts let you grasp important details quickly.

To create a chart in the **SQL Editor**, click the **Charts** button 

on the left vertical toolbar of the query results area. Your data instantly transforms into a visual representation.



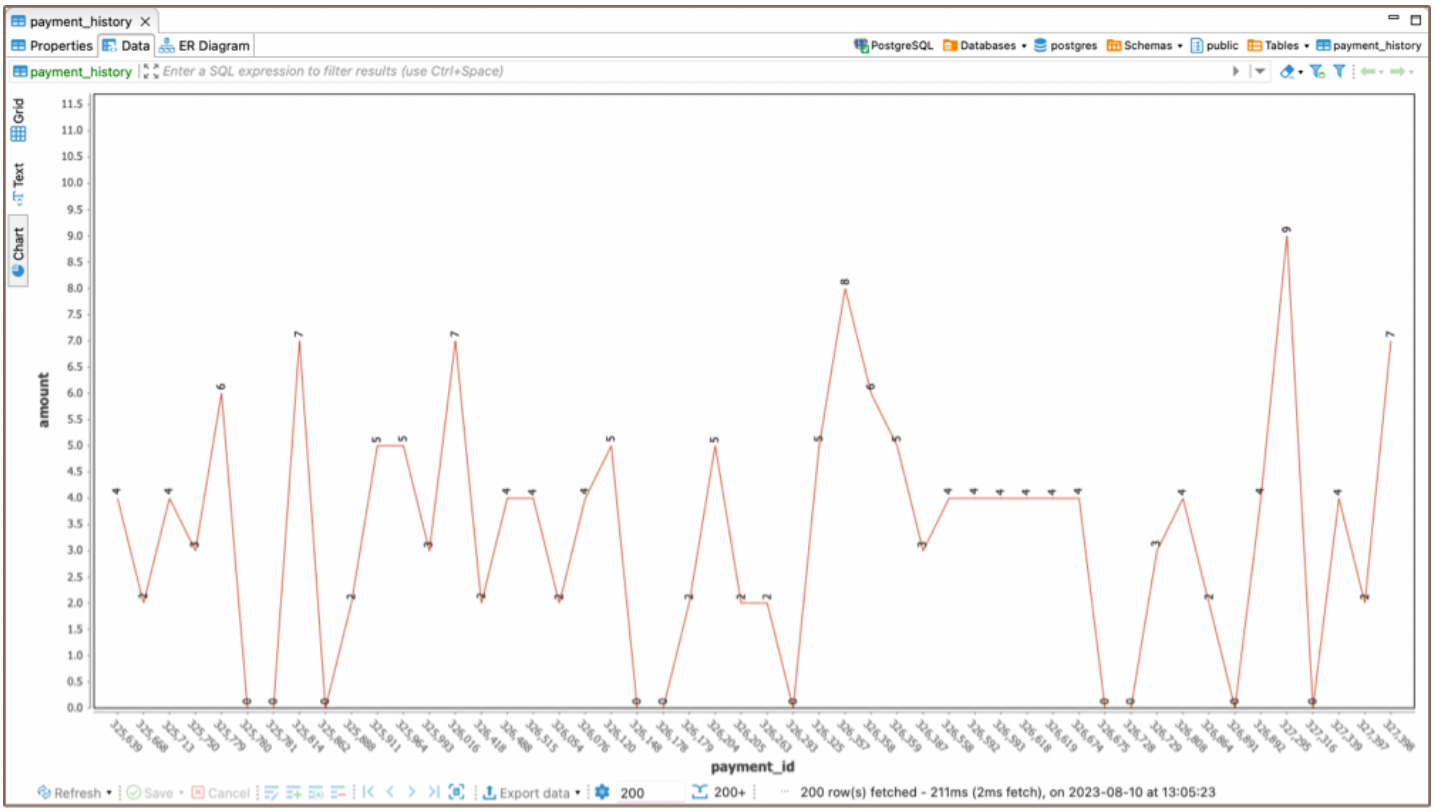
Creating Charts In Data Editor

Charts are a powerful tool for visualizing structured analytical data, such as those stored in Views.

If you want to create a chart for a table, it requires some preparation. First, sort and apply various filters to the table's columns to structure the data. These changes directly affect the appearance of the chart, enabling you to customize the visual representation to fit your needs.

To build a chart using the **Data Editor**, simply press the **Charts** button 

located in the left vertical toolbar. A chart based on your structured data will be created.



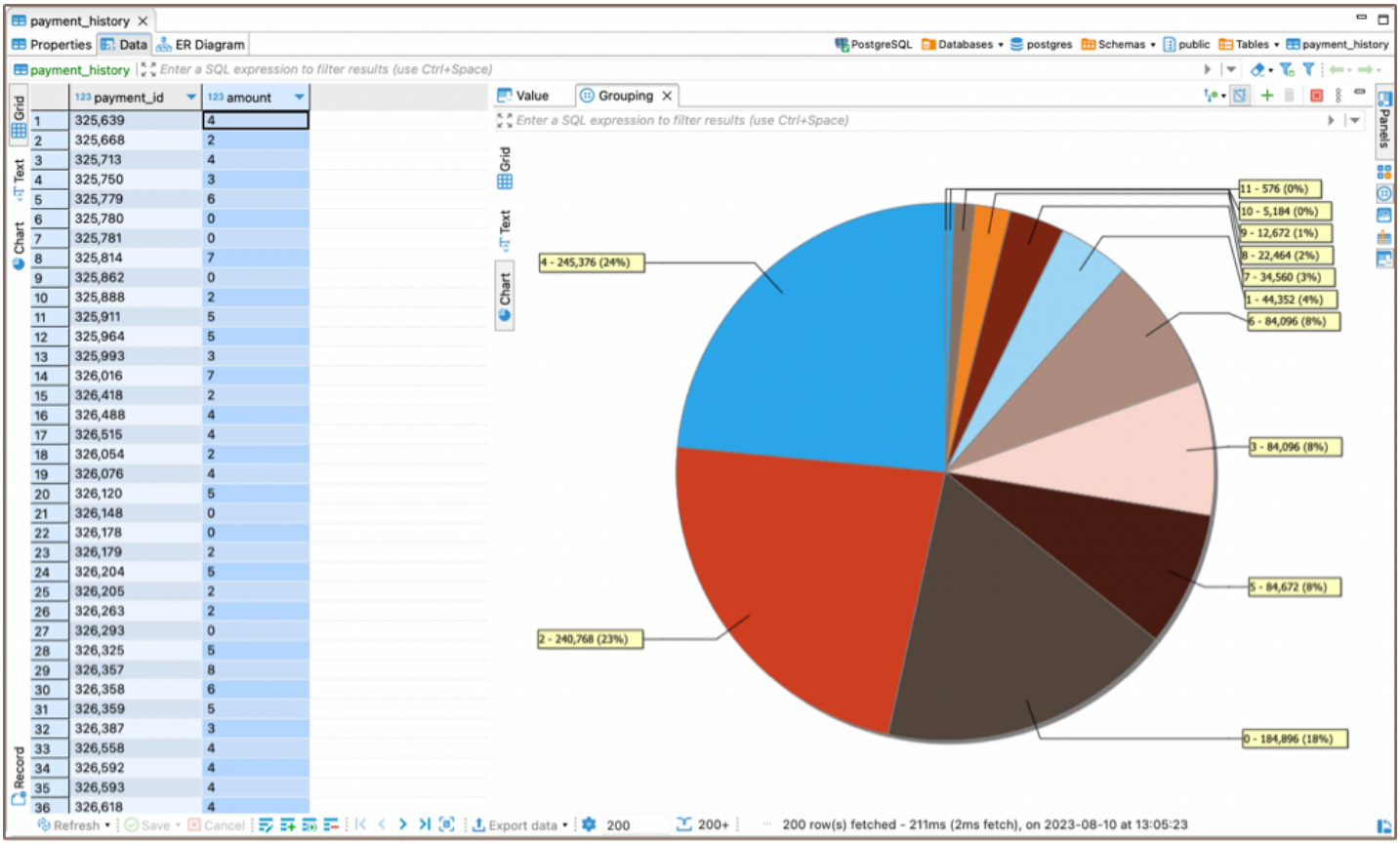
Note: By default, the data for Y-axis is taken from the first column of the table containing numeric values.

Creating Charts In Grouping Panel

The **Charts** feature also works with analytical tools like the **Grouping Panel**. When building a chart for a table that contains grouping results for one or more columns, you can easily switch the source data for the X and Y axes by changing the columns in the Charts Editor.

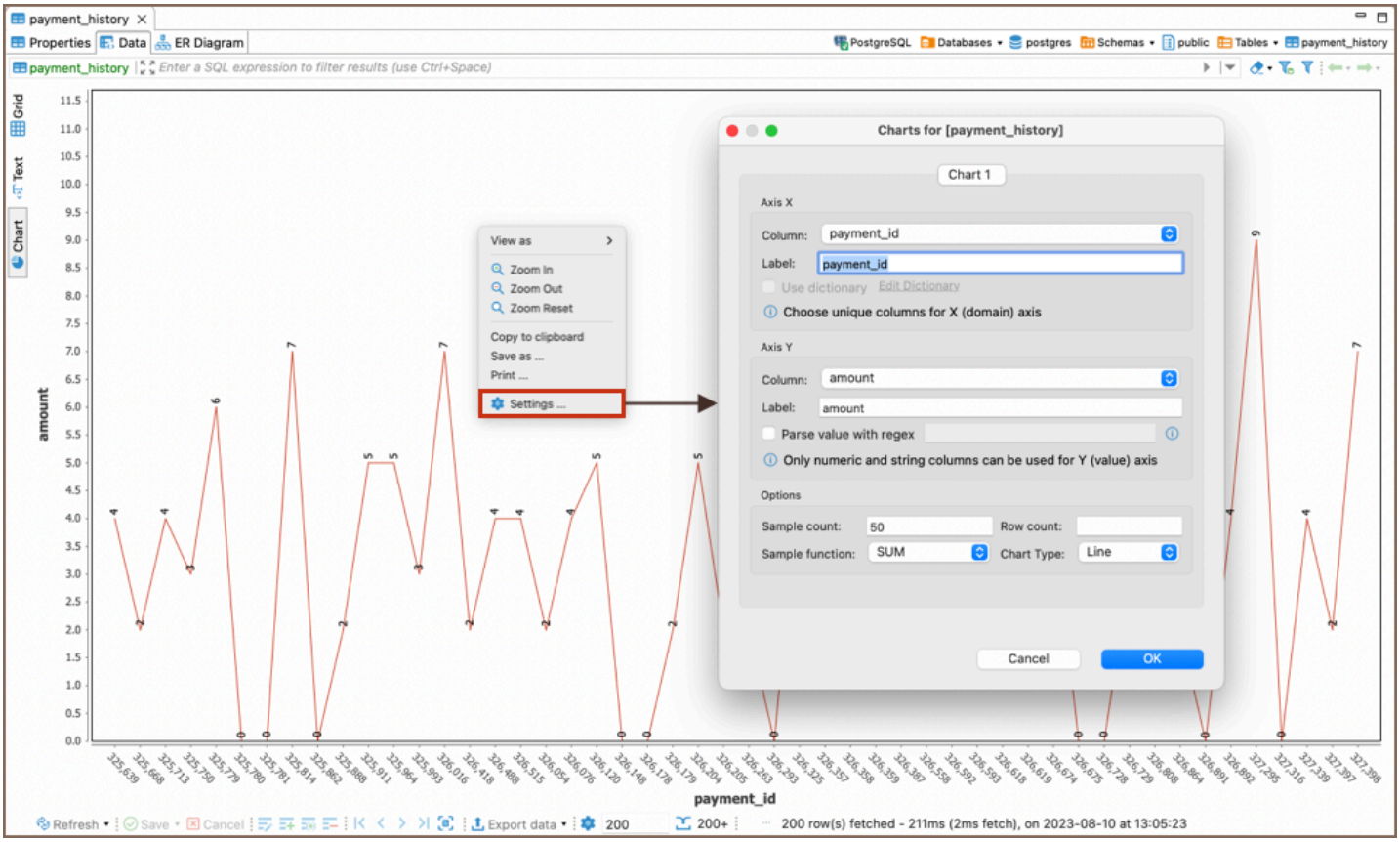
To create a chart in the **Grouping Panel**, simply press the **Charts** button 

on the left vertical toolbar



Editing Chart Settings

To edit the chart settings, right-click inside the chart area to open the chart's context menu, and then select the **Settings** option. The **Chart Editor** will appear, allowing you to make adjustments.



The following chart settings can be adjusted:

Setting Axis X

- In the **Column** drop down list of available columns select a column whose data will be used on X-axis of the bar chart. Make sure you choose unique columns for X-axis.
- Define a user-friendly axis name in the **Label** text field.

Setting Axis Y

- In the **Column** drop down list of available columns, select a column whose data will be used on Y-axis of the bar chart.

  Note, that only numeric and string columns can be used for Y-axis.
- Define a user-friendly axis name in the **Label** text field.
- For advanced customization, you can use the **Parse value with regex** option. Regular expressions (regex) allow you to filter, match, or manipulate the column values according to specific patterns or criteria. This

functionality offers precise control over the data displayed on the Y-axis, enabling you to create more refined visual representations. If you're unfamiliar with regular expressions, the Regular Expressions Tutorial provides comprehensive guidance and examples.

Setting Other Options

You can also set the following chart options:

- **Sample count** - maximum number of columns used for building a chart.
- **Row count** - maximum number of rows used for building a chart.
- **Chart Type** - you can choose how the data will be visualized in **Bar**, **Line**, or **Pie** formats.
- **Sample function** - an aggregate function where the values of multiple rows are grouped together to form a single summary value displayed on Y-axis.

The following sample functions are supported:

Name	Description
AVG	Average value
SUM	The sum of all values
FIRST	The first value
LAST	The last value
COUNT	Total count of all values
MIN	The minimum value
MAX	The maximum value

Copying to clipboard

You can copy a chart to a clipboard by selecting the **Copy to clipboard** option in the chart's context menu.

Exporting Charts

You can export a chart into `PNG` format by selecting the **Save as...** option in the chart's context menu.

Printing Charts

You can print a chart by selecting the **Print...** option in the chart's context menu.

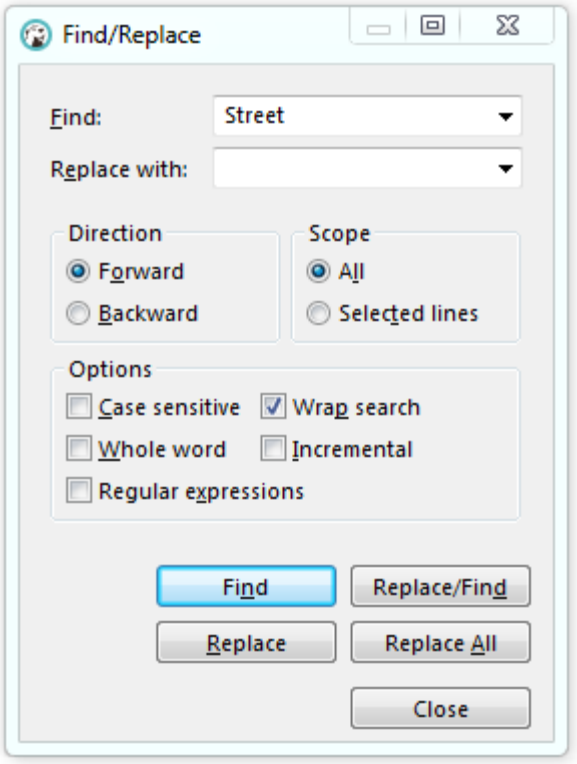
Zooming and Viewing Options

In the chart's context menu, you will find additional options for interacting with the chart:

- **Zoom in**: Enlarge the view of the chart.
- **Zoom out**: Reduce the view of the chart.
- **Zoom Reset**: Return to the default zoom level.
- **View as**: Change the visualization format of the chart. A submenu will appear, allowing you to select between **Bar**, **Line**, or **Pie** formats.

Data Search

To search for data in the result set, press `CTRL+F`. The standard Find/Replace search dialog box will open:



You can also use the Find and Replace feature.

NOTE: The system searches only in already fetched rows.

Setting	Description
Case sensitive	By default, the search is case insensitive.
Whole word	By default, the word specified in the search field can be found in the case when the word is part of another word. Enabling this setting will lead to a particular word search.
Regular expressions	Enable to use regular expressions in the search. In the search field, you can use `Ctrl+Space` for autocomplete regular expressions templates.
Wrap search	Enable this setting to find matches throughout the object from the beginning, and not only from the focus point to the end of the object.
Incremental	Matches are found incrementally as you type, if this setting enabled.

Data export/import

Data transfer is a crucial feature that enables you to export and import data in various formats as well as moving data between tables in the same or to different databases.

- Export data
- Import data
  - Importing from CSV
  - Importing from Excel
  - Importing from XML
  - Importing from database table

Note: The data transfer operation runs in the background, allowing you to continue working with your database during the export or import process. However, try to avoid changing data in the tables that have been selected for export or import until the process is complete.

Export data

You can transfer data from one database to another or export it in different types and formats:

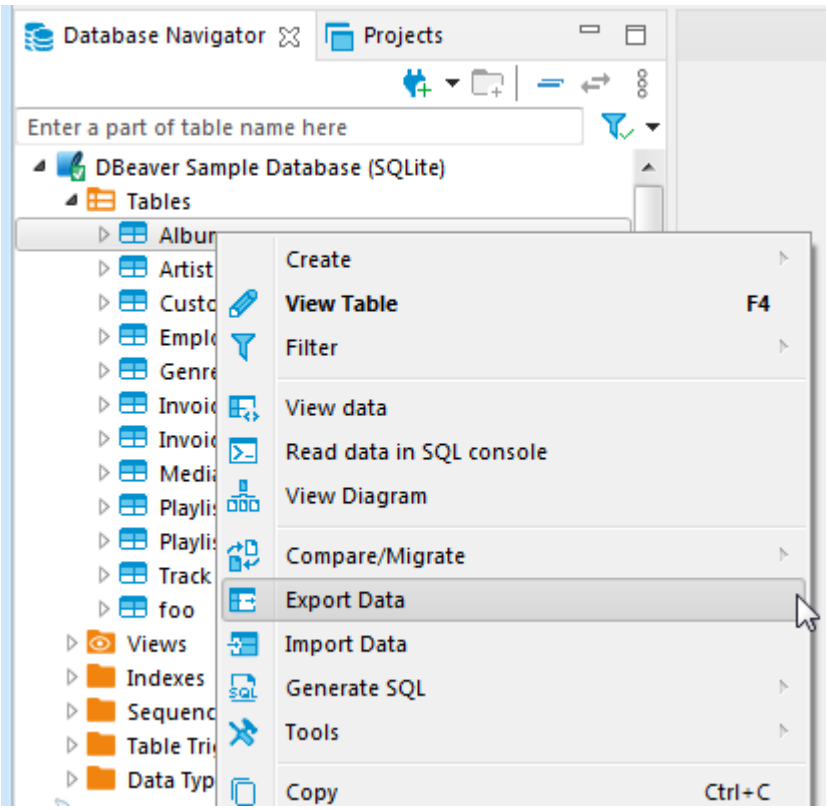
- `CSV`
- `XLSX` (Excel spreadsheet)
- `HTML`
- `XML`
- `TXT`
- `JSON`
- `Markdown` files
- `SQL` statements

- Source code arrays

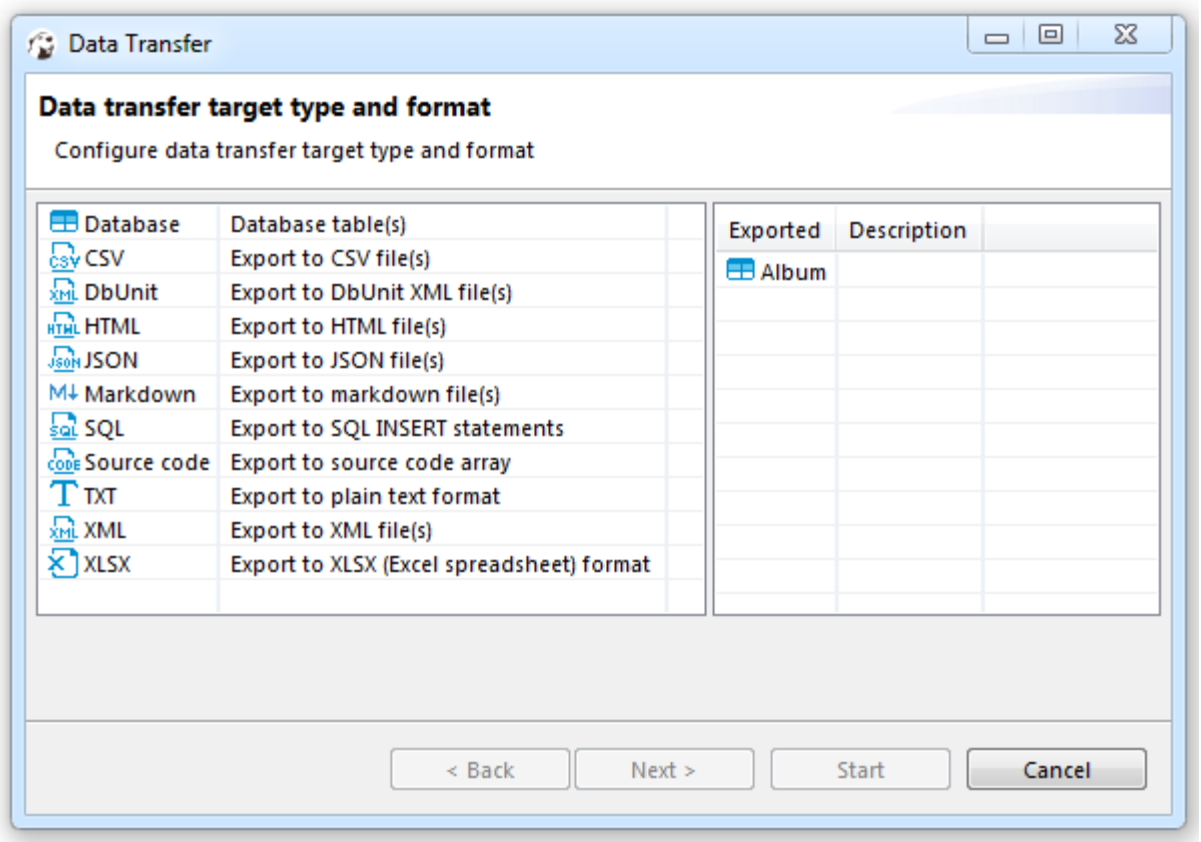
The process of exporting data follows a similar procedure for the supported formats. Therefore, in this guide, we will focus on the data exporting process using the `CSV` format.

1. Select the table or tables you want to export. In the context menu, choose **Export Data**.

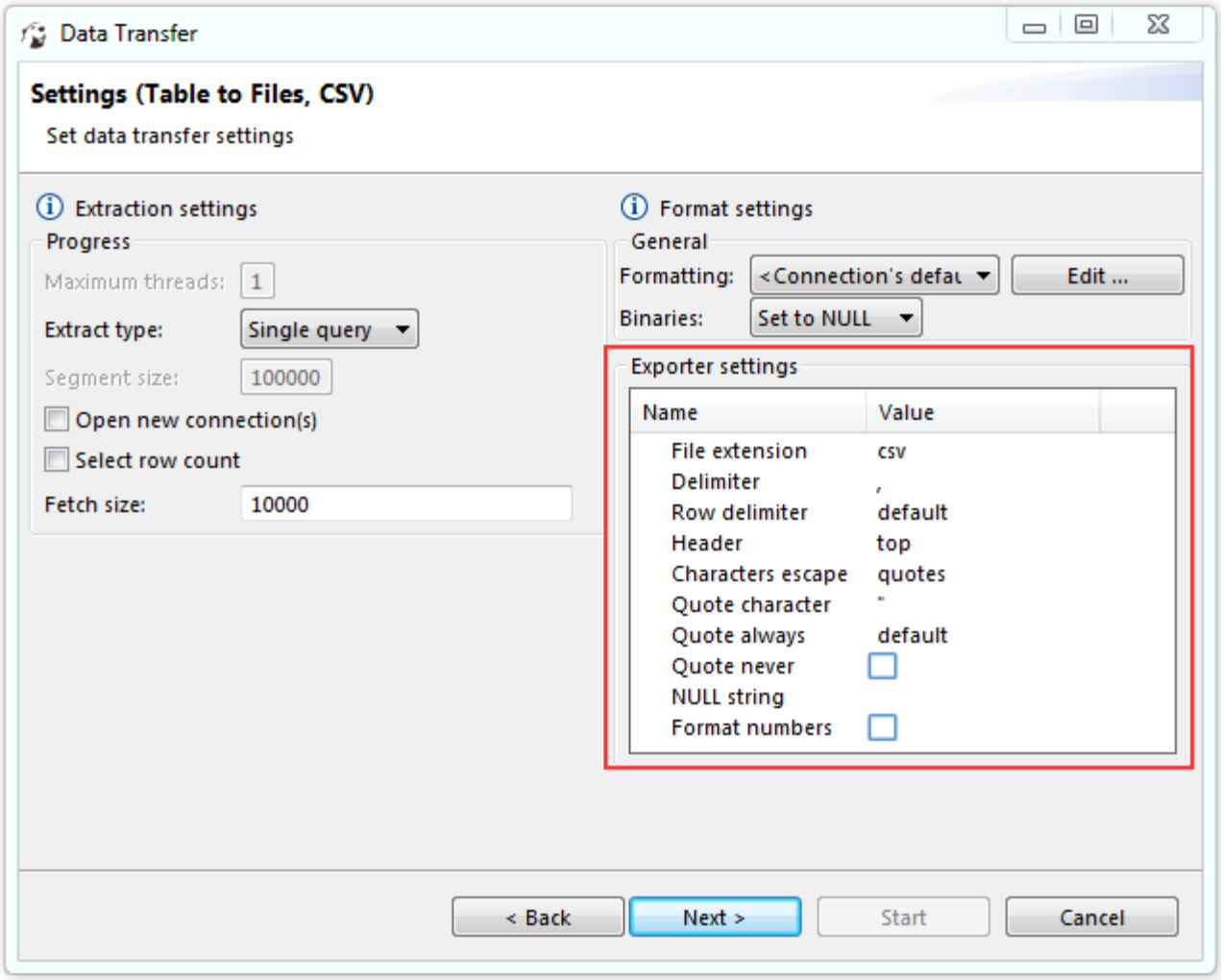
   Note: you can also export data from custom SQL query results. To do that, choose **Export data** in the results context menu.



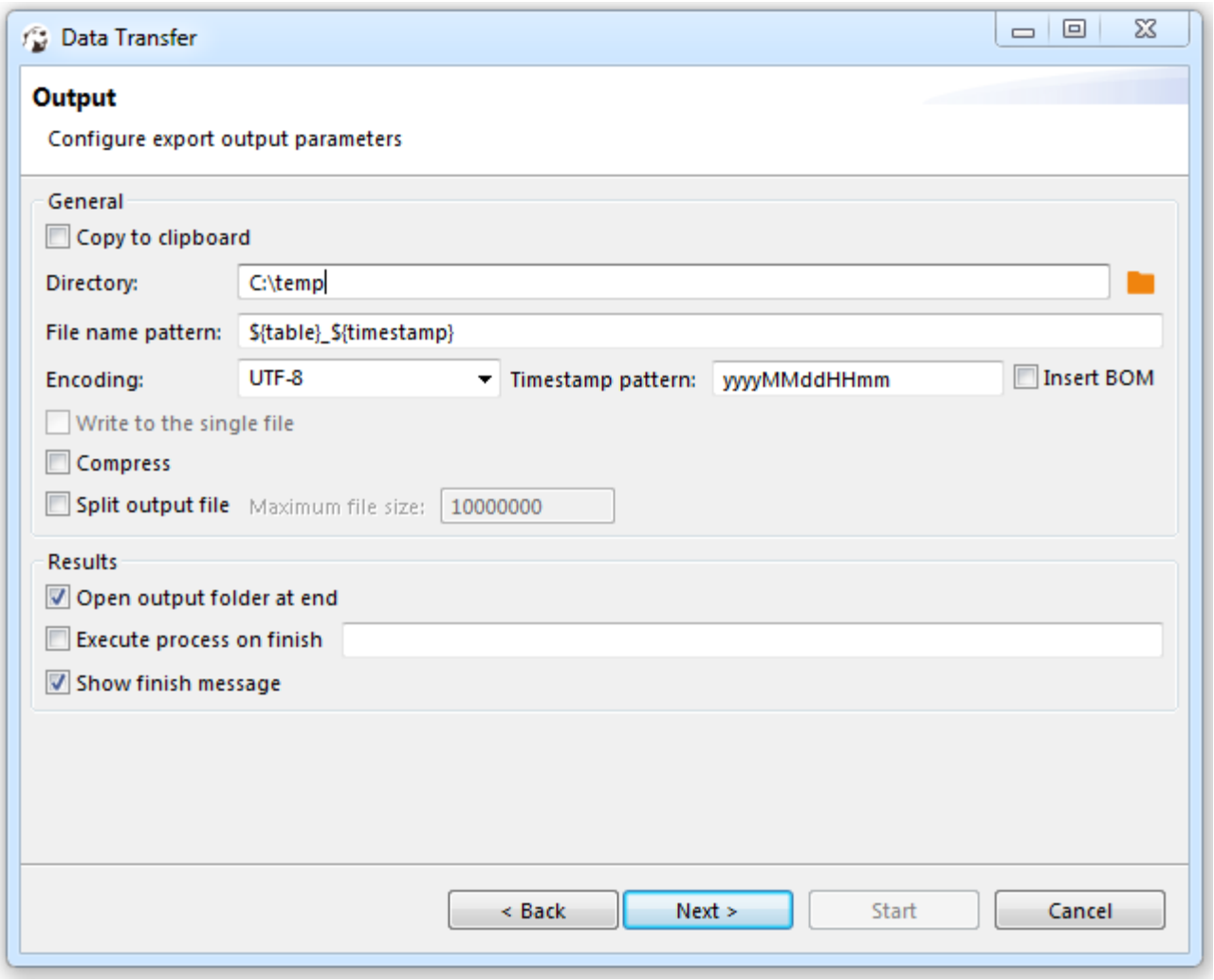
2. In the window that appears, choose `CSV` and click **Next**.



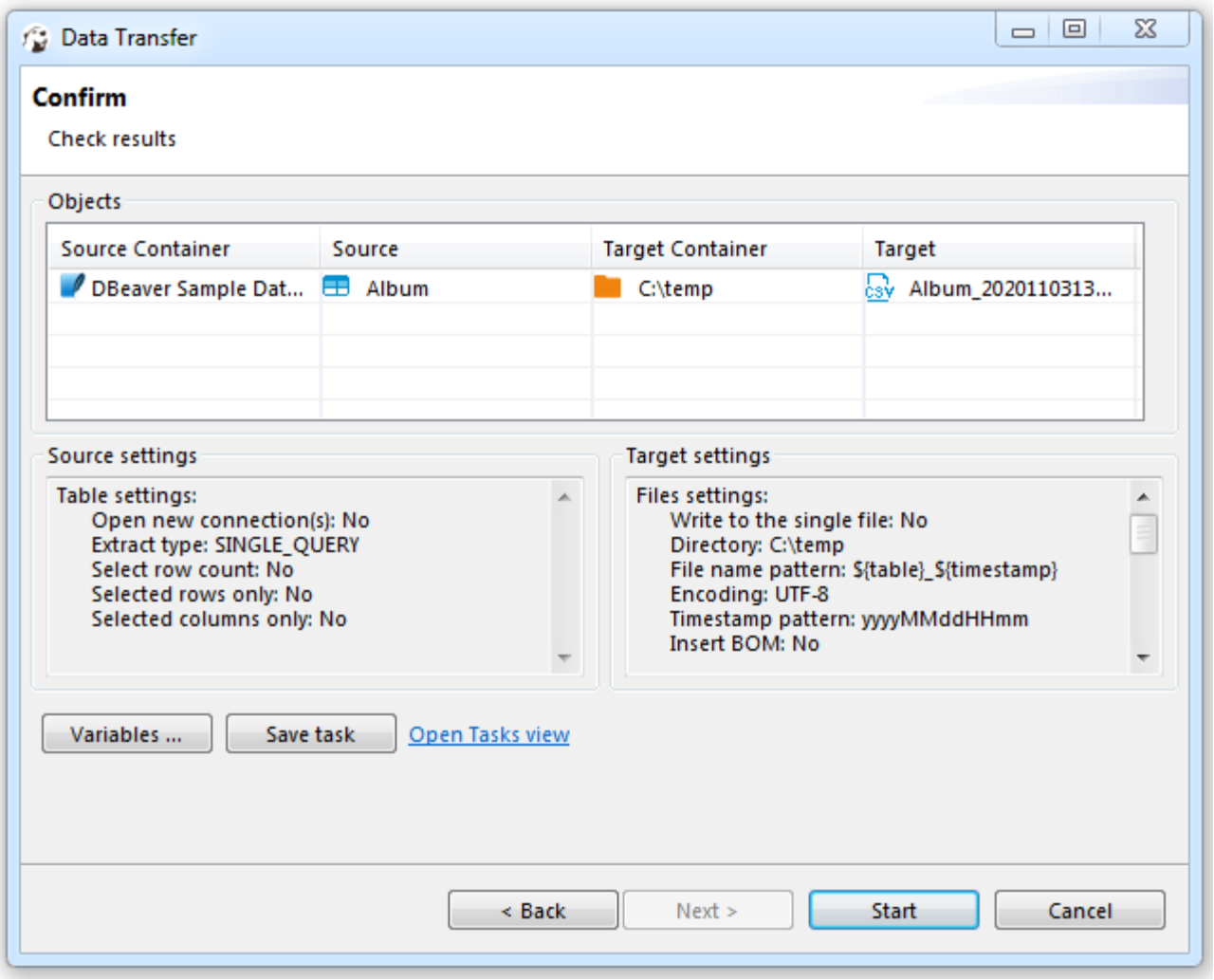
3. Set your data extraction options (how the data will be read from the tables). This may affect the extraction's performance. And set the export format option. They are specific to the data format you chose in step 2:



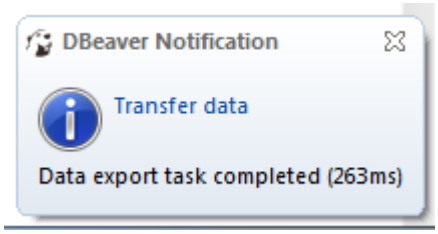
4. Set options for output files or clipboards. **Note**: Timestamp pattern is used here to target the file name pattern:



5. Review what you want to format and which format you will export. You can also save all your settings as a task in this step or change the task variables:



6. Press finish. See extraction progress. You can keep working with your database during the export process as the extraction will be performed in the background.

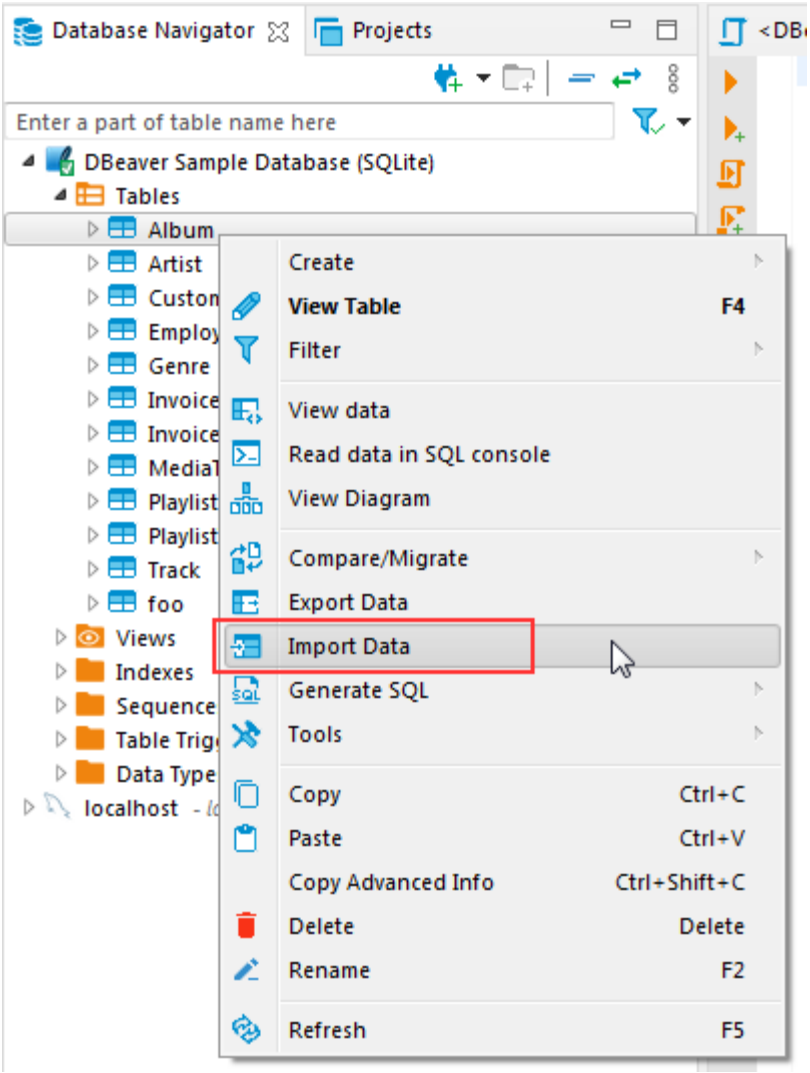


Import data

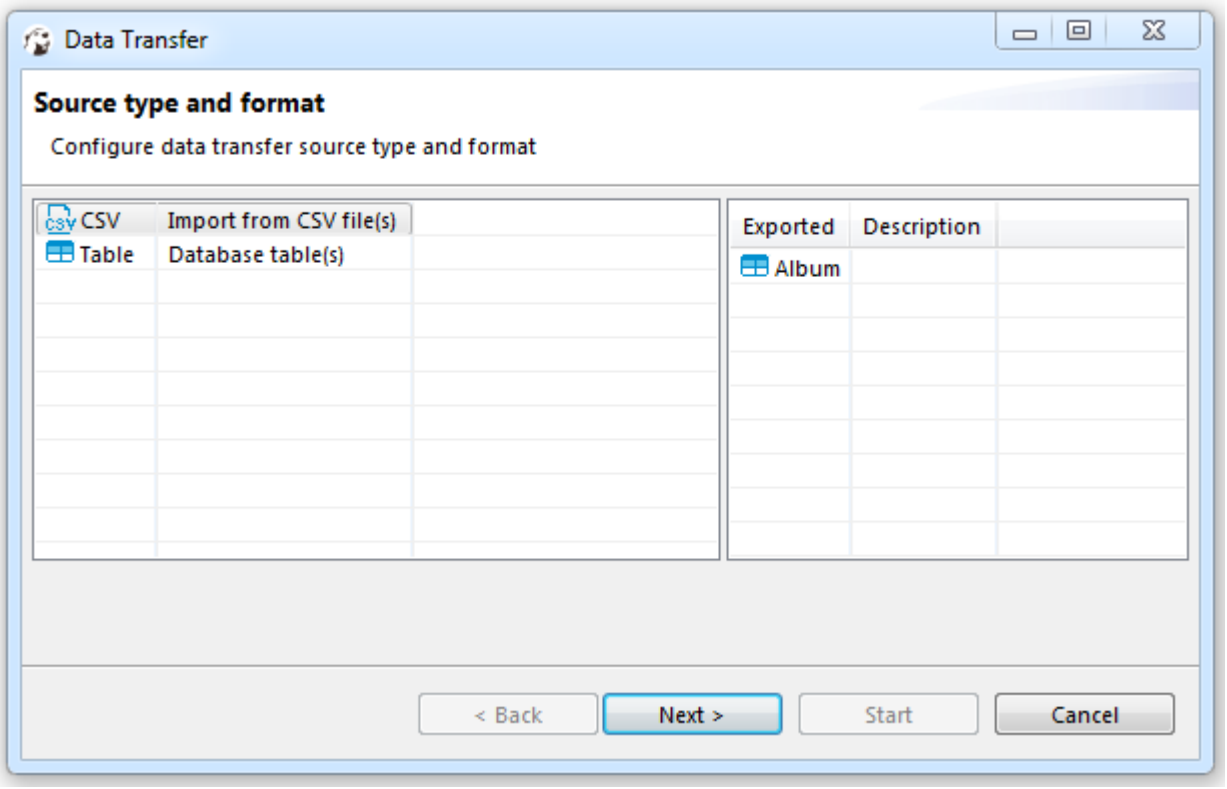
You can import data to your database from `CSV`, `XLSX`, and `XML` files.

Importing data from CSV file

1. Select one or more tables to which you want to import data. In the context menu, choose **Import Data**:

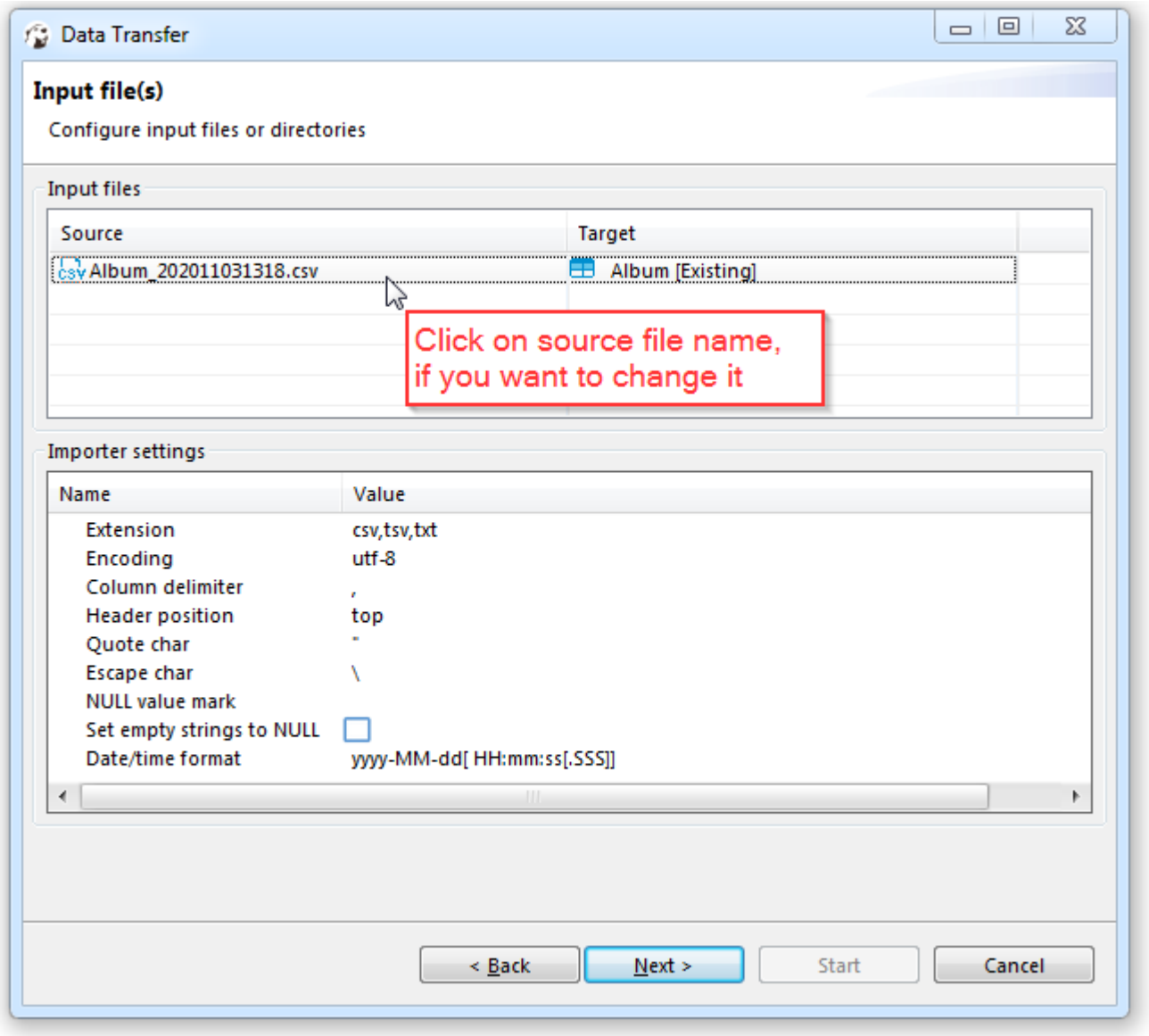


2. In the window that appears, choose `CSV` and click **Next**.

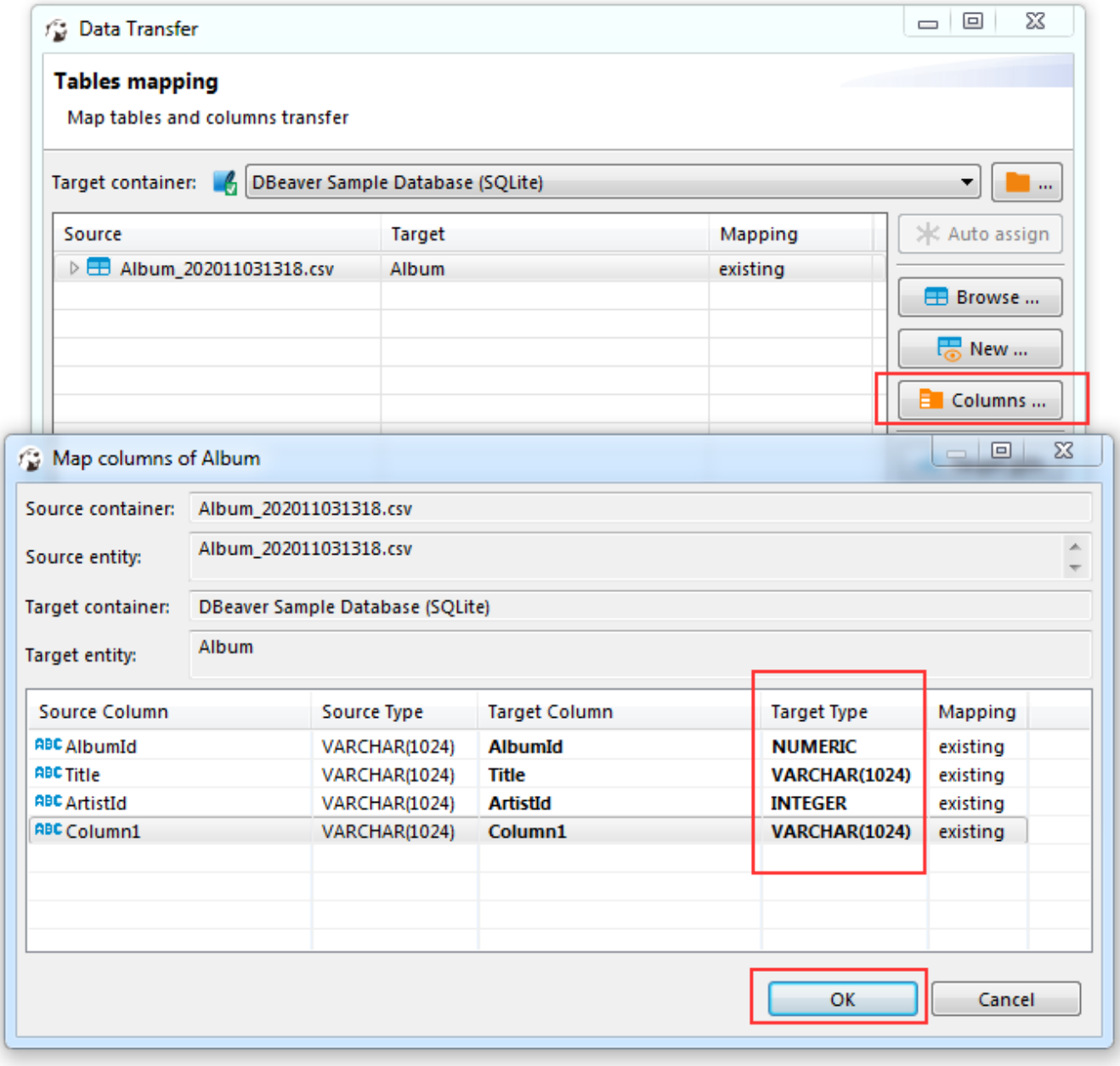


3.

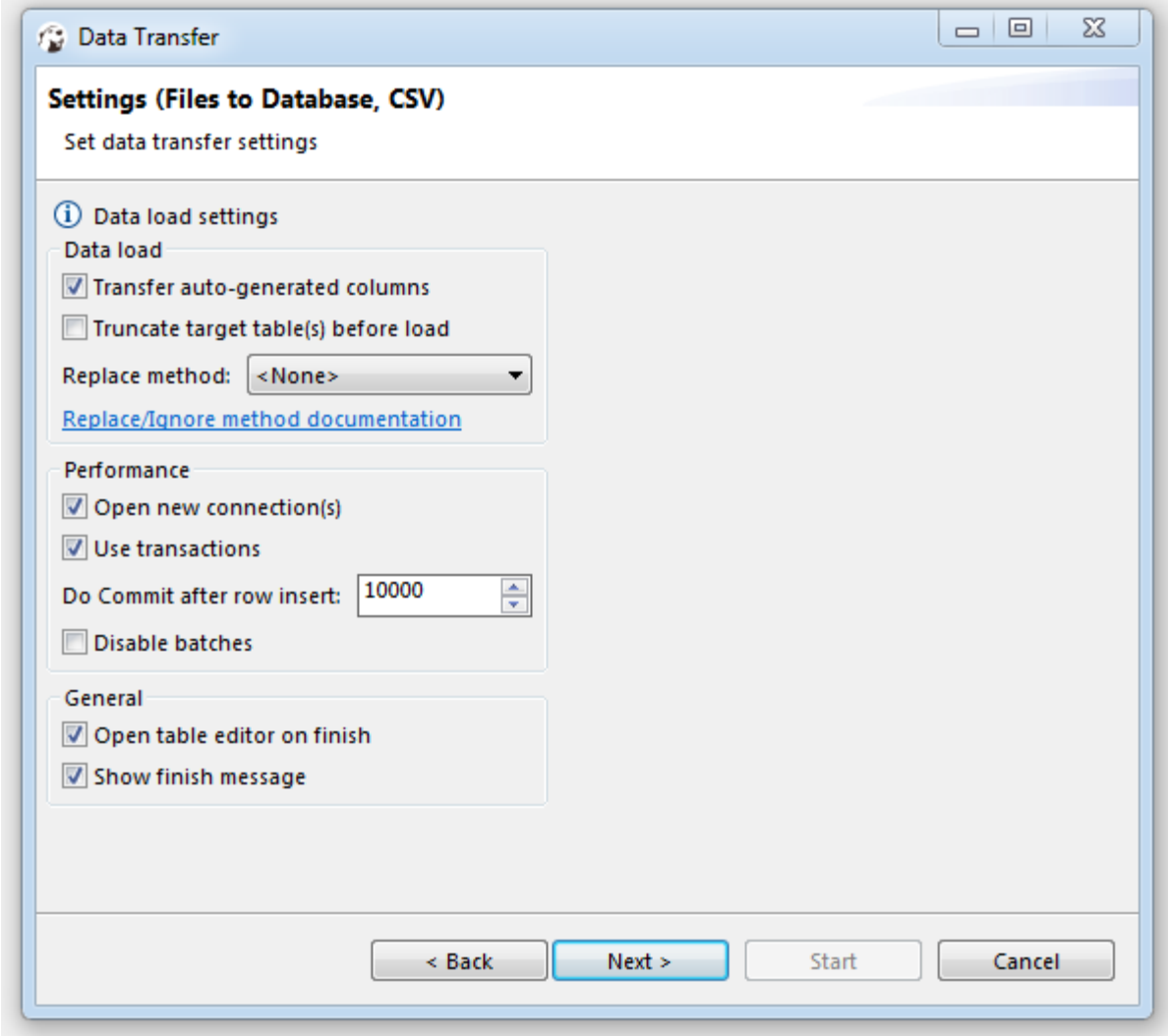
Select the input `CSV` file for each table you want to import. You can change the Importer settings (format specific) during this step:



4. Set CSV-to-table mappings. You must set a column in the `CSV` file for each database table column. You can skip columns (the value will be set to `NULL` in the target table column). You can set constant values for the table column if there is no source column in the `CSV` .

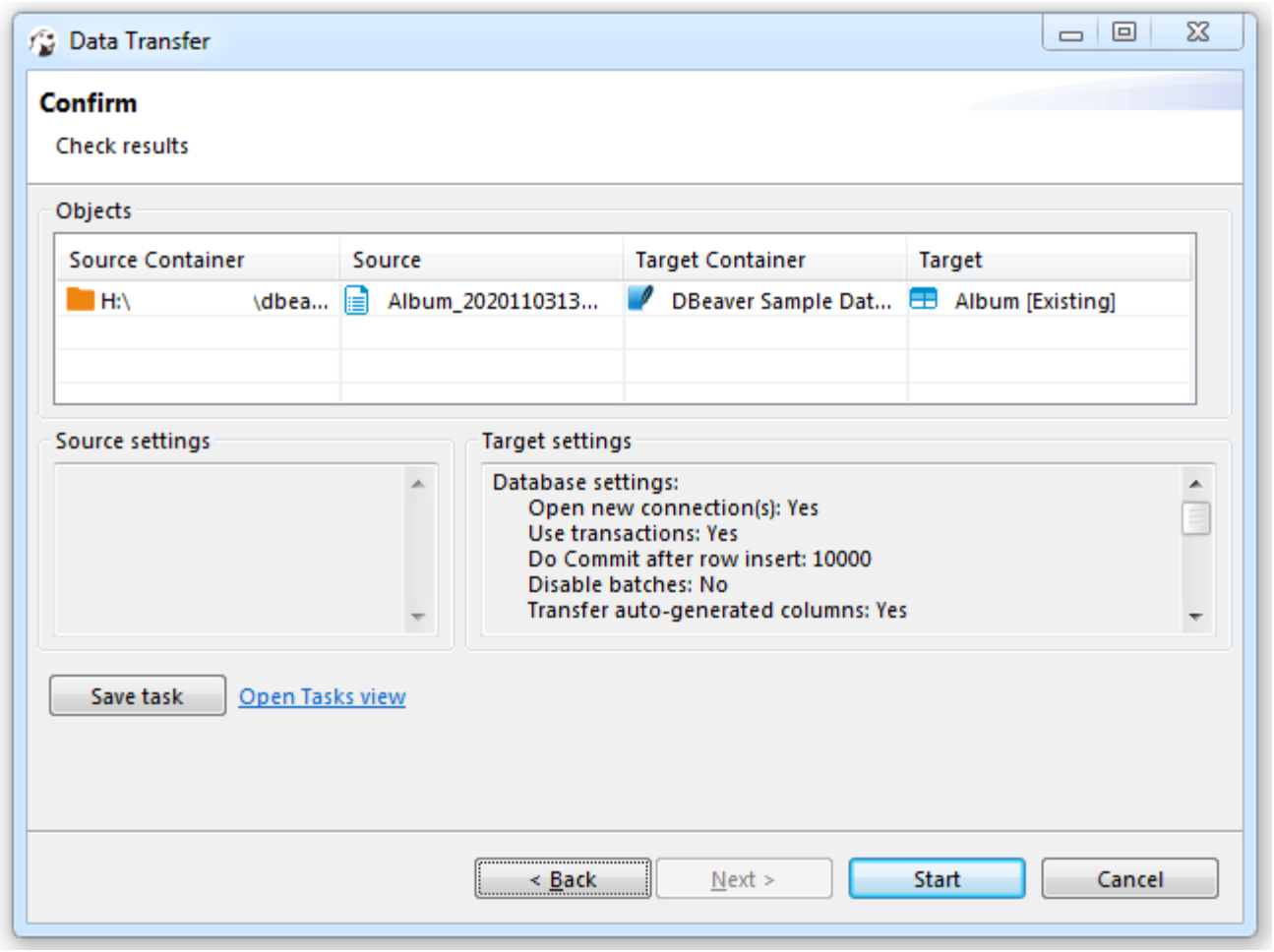


5. Set options for loading data in the database. These options may affect the loading's performance:

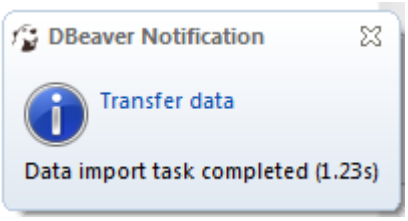


Read our guide on Data Import and Replace to learn more about the replacing method option.

6. Review which file(s) and to which table(s) you will import. You can also save all your settings as a task in this step:

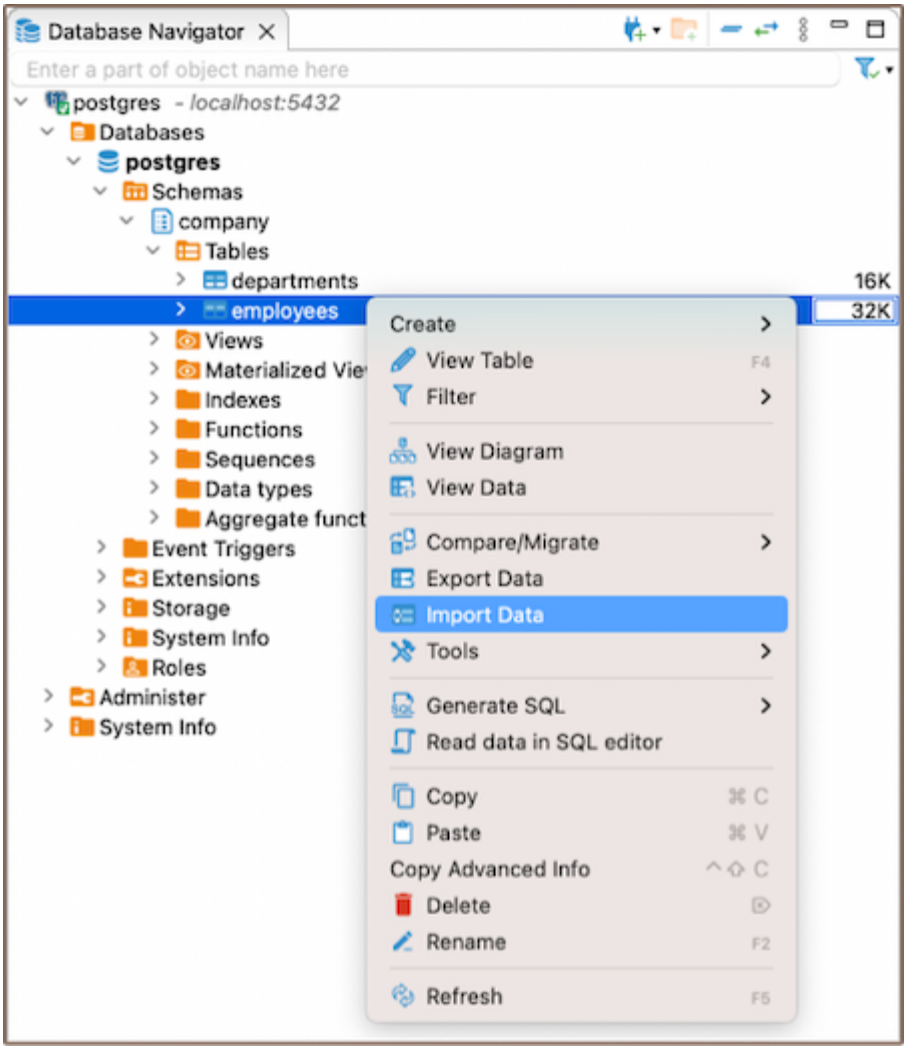


7. Press finish. See extraction progress. You can keep working with your database during the export process as the data loading will be performed in the background. In the end, you will see the following status message:

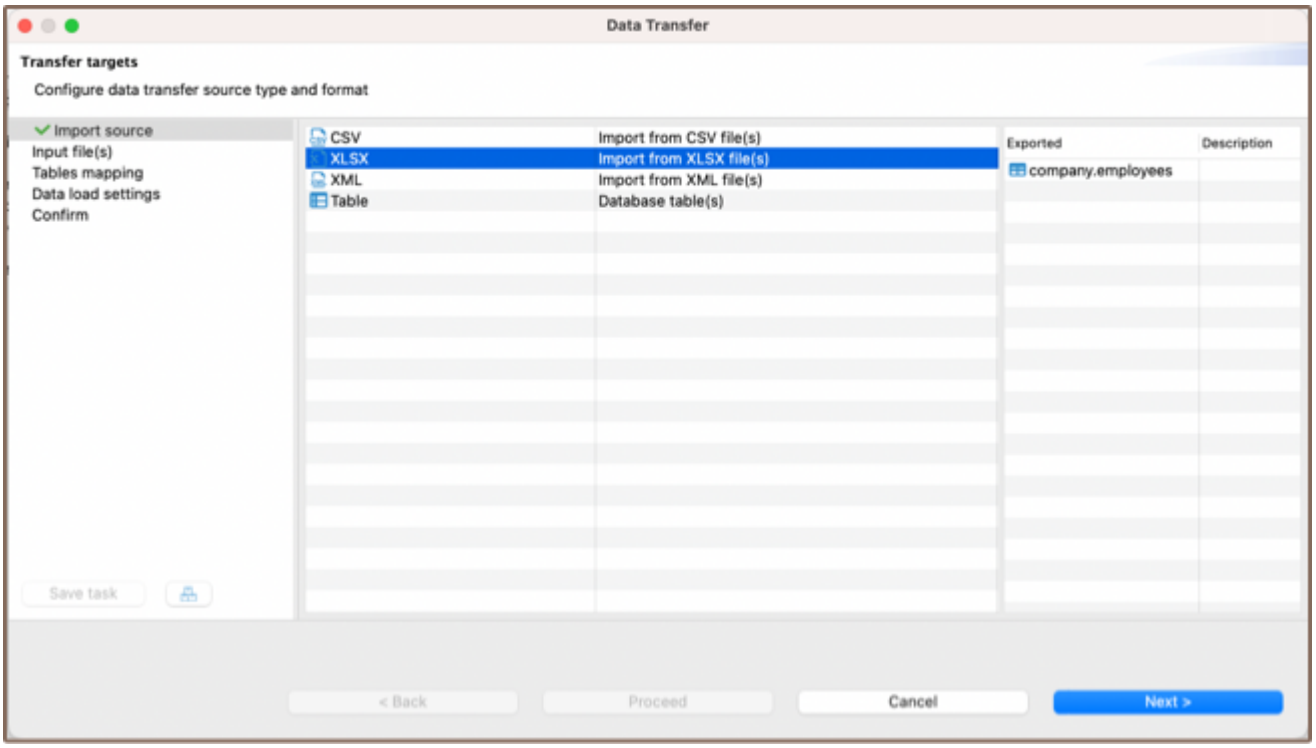


Importing data from Excel file

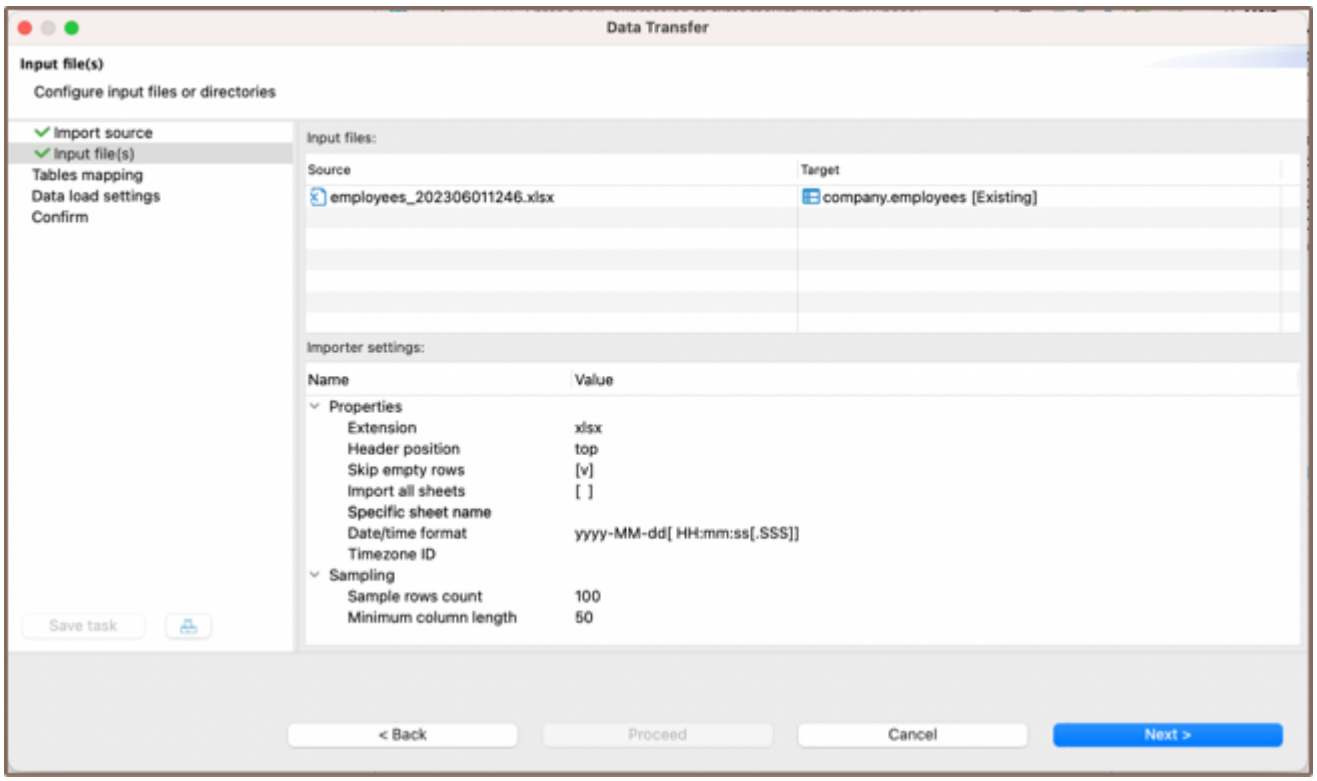
1) Choose the database table (or tables) you want to import data into. Do this by right-clicking on the table name in the **Database Navigator** section and then clicking on **Import Data**.



2) In the window that appears, choose `XLSX` and click **Next**.



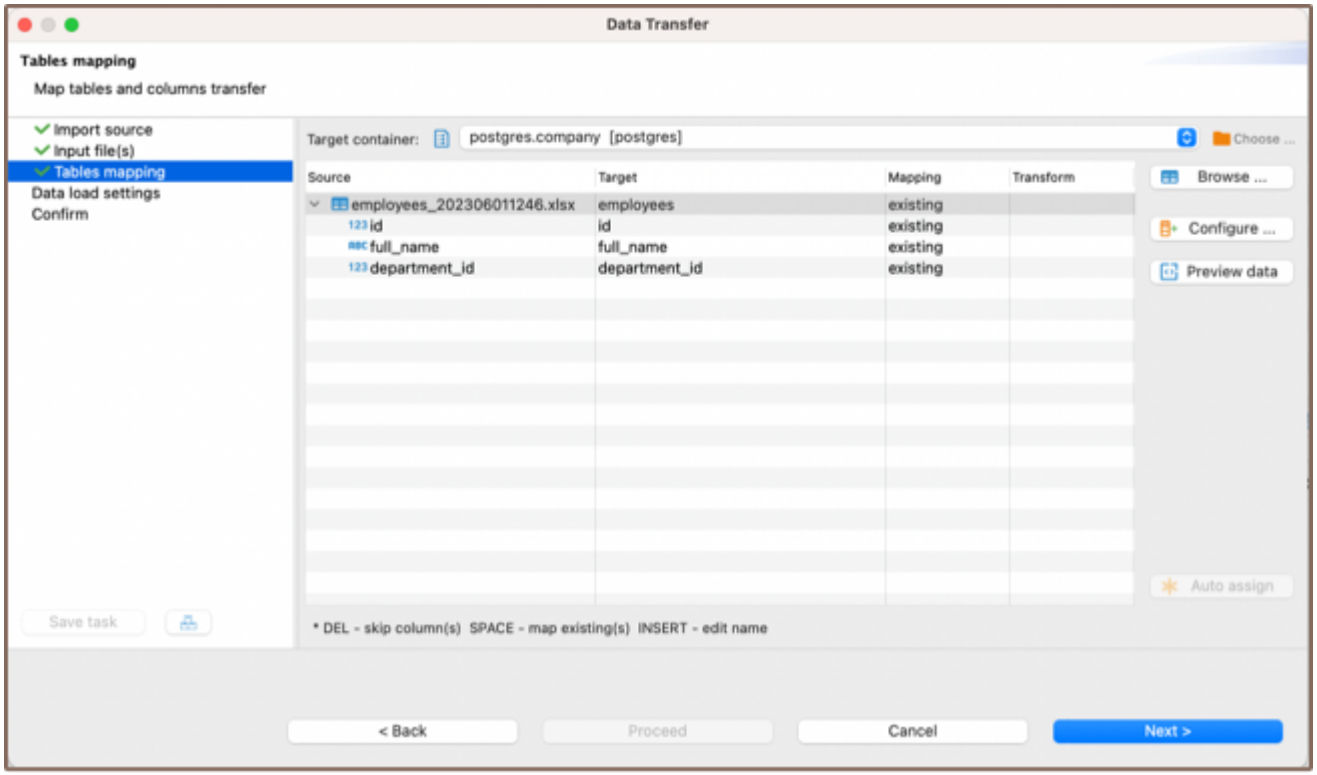
3) In the following window, choose the file that contains the data you wish to import into the table. Select the appropriate settings described below, then click **Next**.



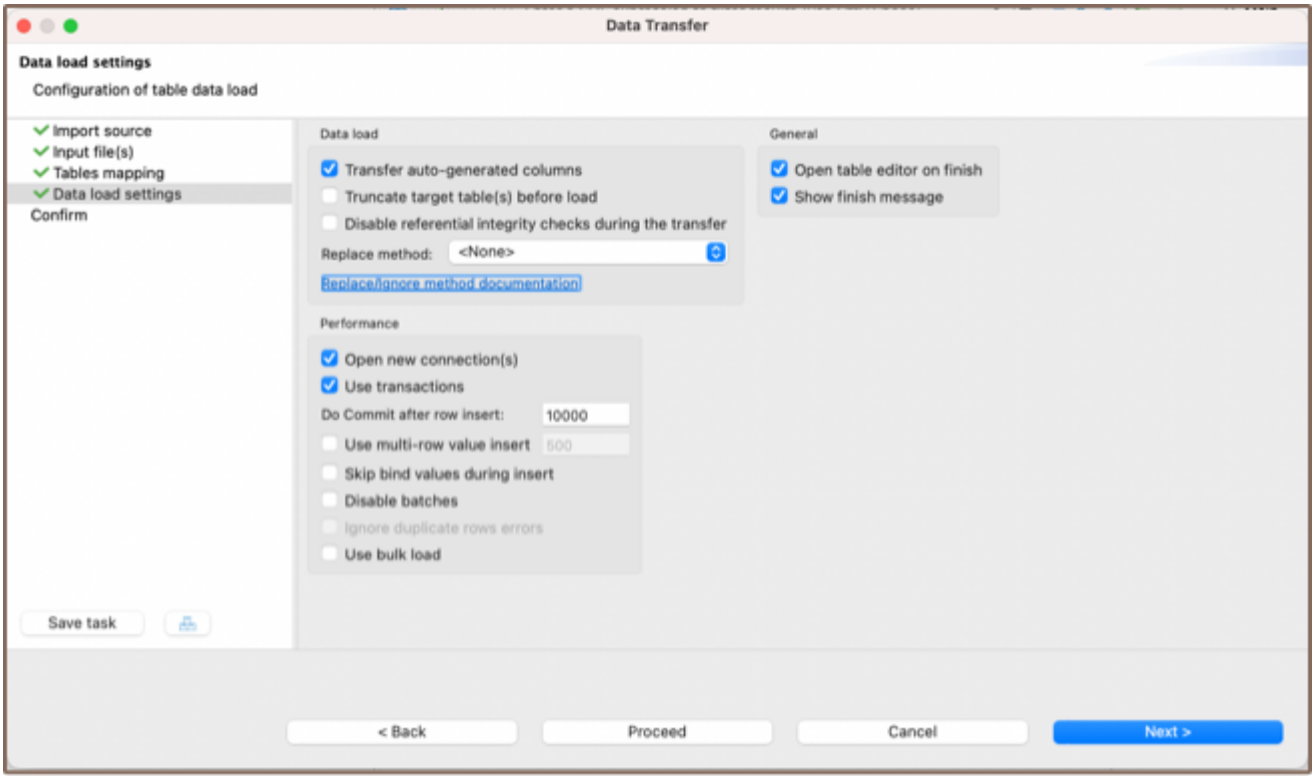
Setting name	Description
Header position	Determines the location of the column names in the Excel table, either at the top or none. This setting specifies whether the column names are located in the first row of the Excel table or if there are no column names present.
Skip empty rows	If this setting is enabled, any open string values encountered during the data processing will be ignored and not inserted into the corresponding cells in the row. If the setting is disabled, all cells in the row will be filled with a `NULL` value if an empty string is encountered.
Import all sheets	Specifies that all sheets in the file should be imported during the data import process.
Specific sheet name	Enables you to choose a particular sheet from the Excel file for importing during the data transfer process.
Date /time format	Use this setting to specify the date format used in the `XLSX` file. This is used to clarify the date format during the import process and does not affect the output data. You can refer to the **java DateTimeFormatter** documentation for details on the format pattern syntax.
	The local machine timezone is used by default. There are three ways to specify the timezone:

Timezone ID	1) Local zone offset: Specify the offset from UTC in the format of either a positive or negative number (e.g., +3, -04:30). 2) Specific zone offset: Specify the offset from GMT or UTC in the format of GMT+/-X or UTC+/-X (e.g., GMT+2, UTC+01:00). 3) Region-based: Specify the timezone using a region-based identifier such as UTC, ECT, PST, etc.
Sample rows count	Determines the number of rows that will be used as a sample to estimate the length and data types of the imported data.
Minimum column length	This value is used when creating a new column and, if necessary, specifying its type. It indicates the minimum number of characters or digits expected in the column. This information helps determine the appropriate data type and size for the column during the creation process.
Save task	Opens the Save Task window to assist in creating a task during the data transfer process. This window provides options and settings for creating and configuring a task related to the data transfer operation.

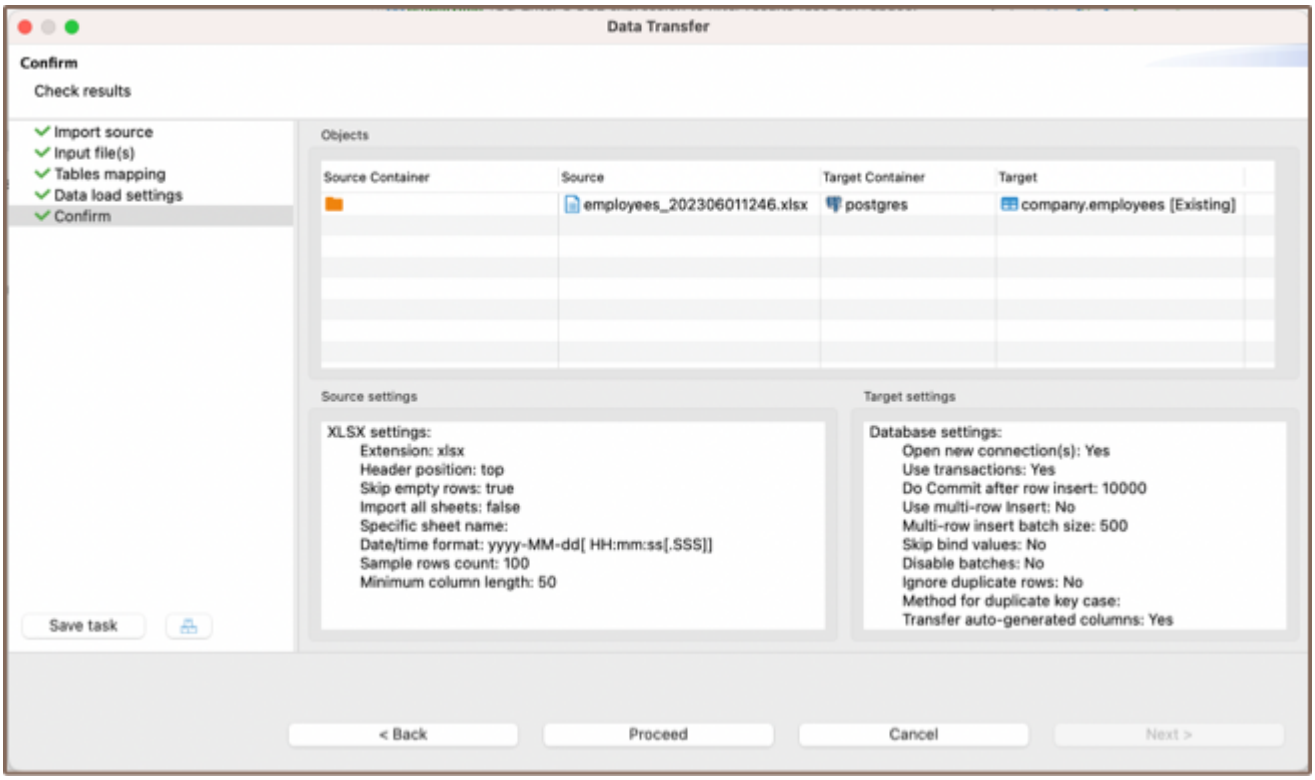
5) In the next window, set XLSX-to-table mappings. Please refer to our mapping process guide for more detailed information.



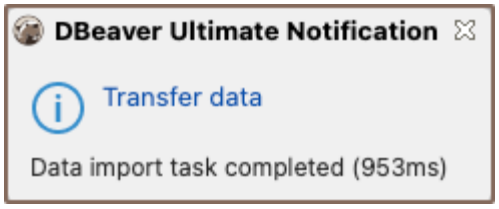
6) Select your data load settings in the subsequent window, and then click **Next**. For more information, please refer to our article's section Data load settings article.



7) In the final window, you can review all the settings you selected earlier. If you missed something, you could go **Back** and fix it. When you are ready, finish the import by clicking **Proceed**.



8) If the `XLSX` file is valid and there are no errors, you will see a notification window with information about the completion of the task. You can keep working with your database during export, as the data loading will be performed in the background.



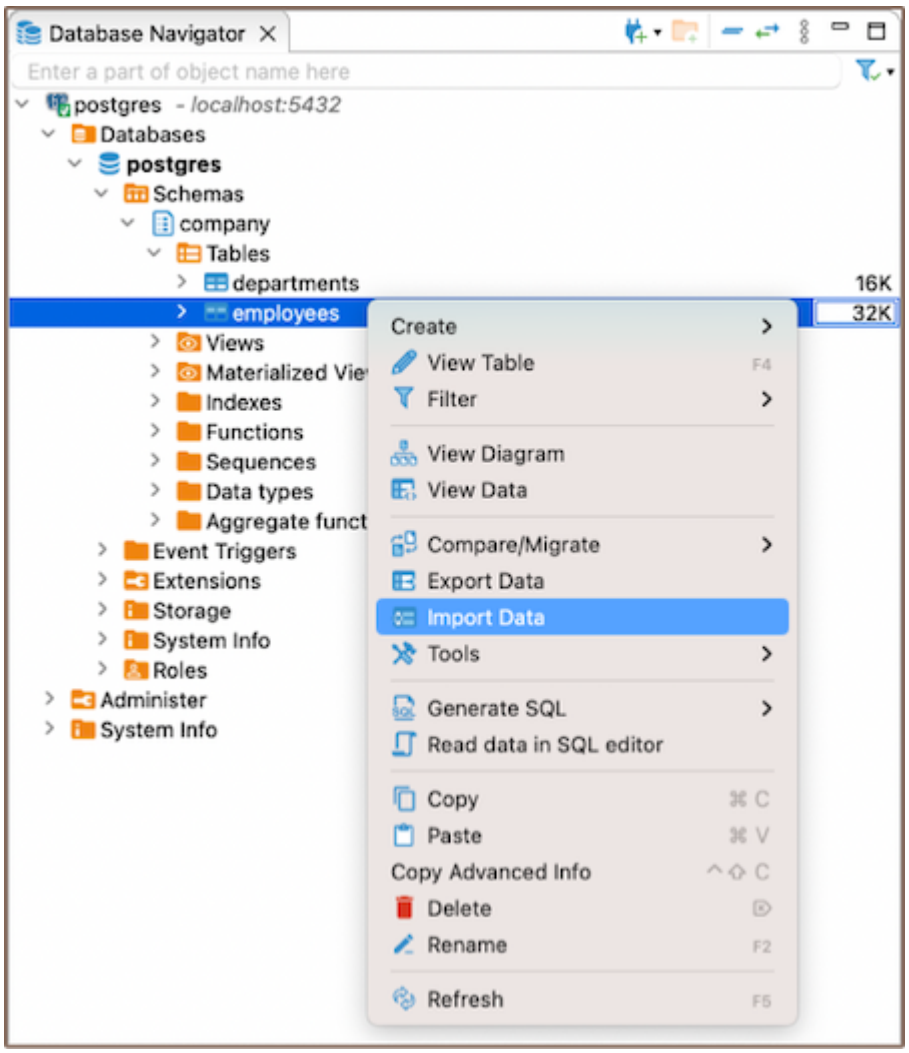
Additional features

When importing from Excel files, DBeaver provides additional features to enhance your data transfer experience:

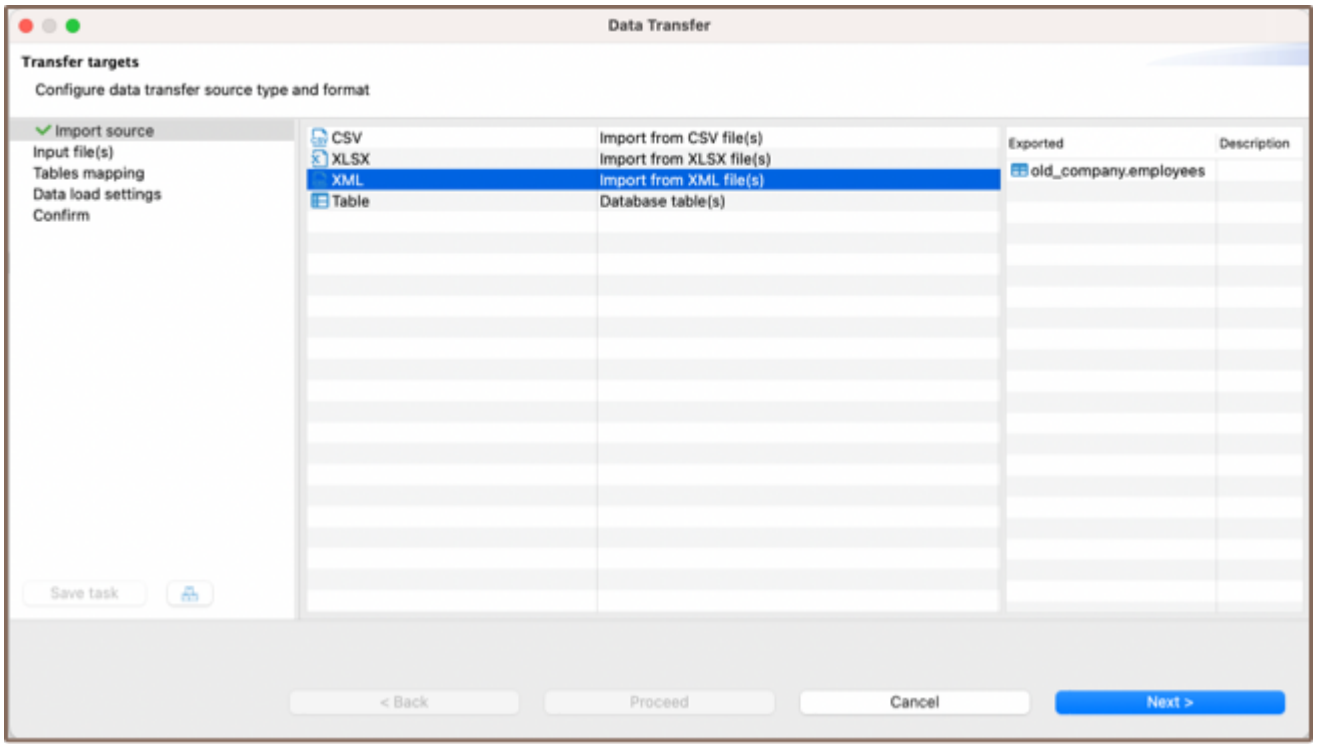
- **Allow importing multiple sheets into different tables**: DBeaver allows you to import multiple sheets from a single `XLSX` file into different tables in your database.

Importing data from XML file

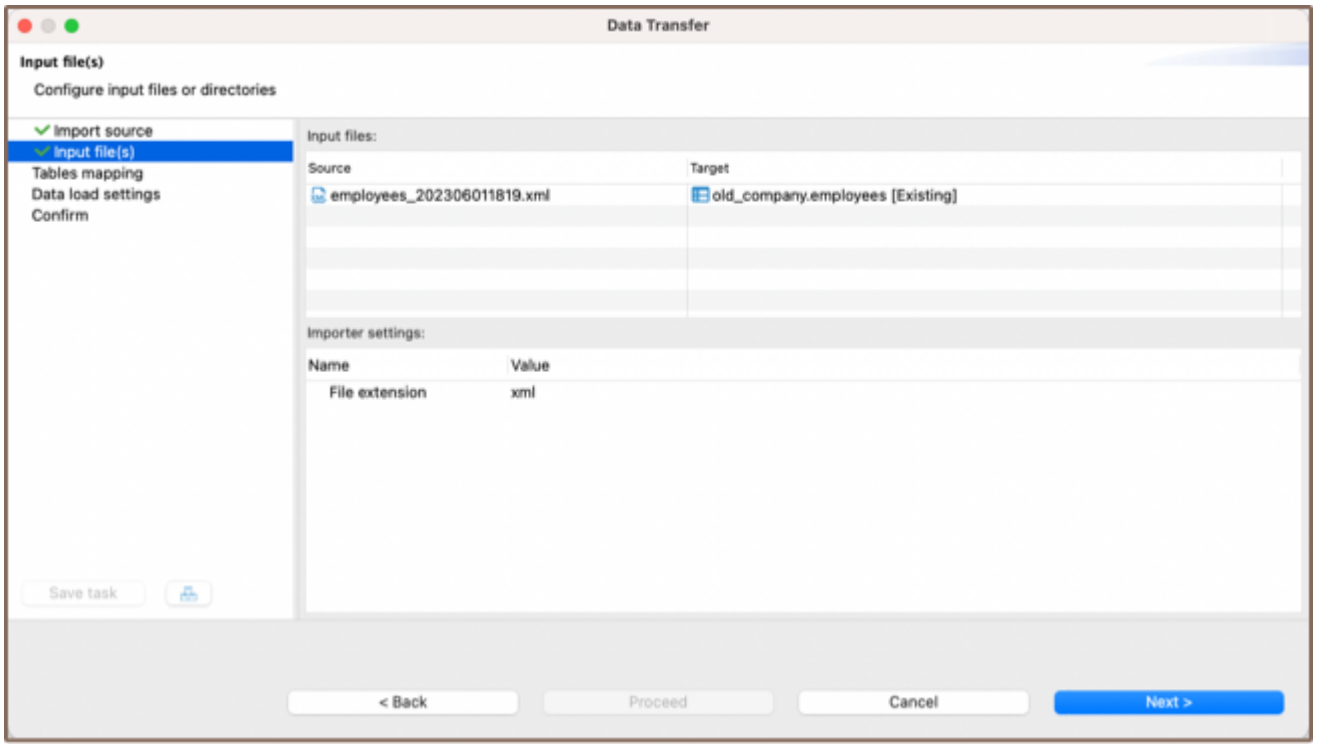
1) Select the database table (or tables) where you want to import data. Do this by right-clicking on the table name in the **Database Navigator** section, then clicking on **Import Data**.



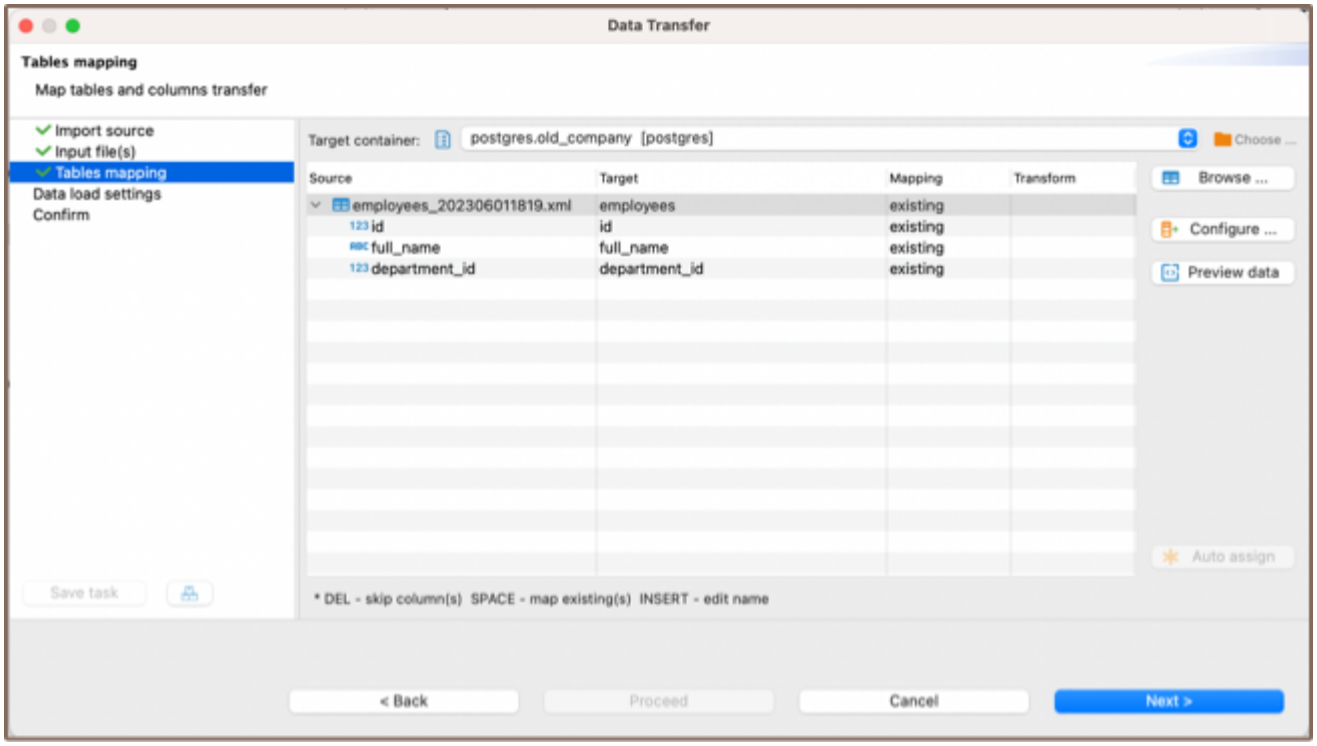
2) In the window that appears, select `XML` and then click **Next**.



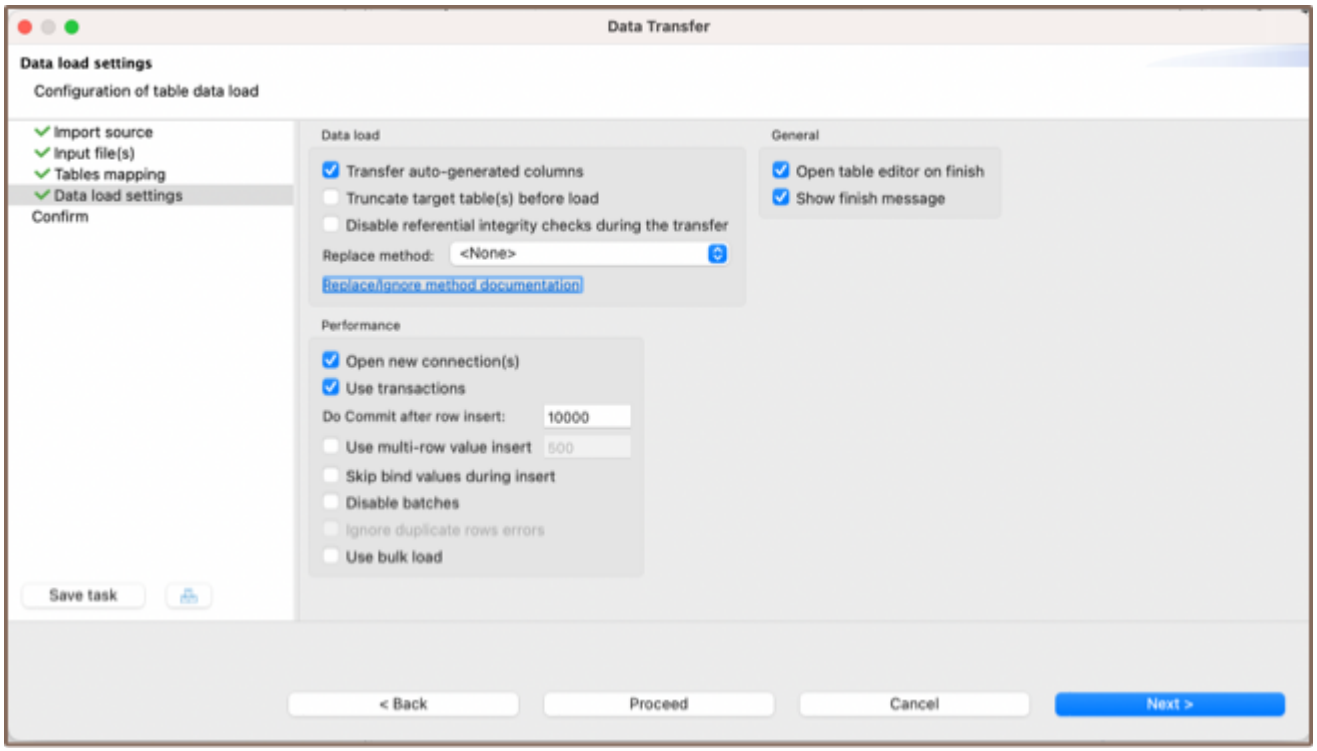
3) In the following window, select the `XML` file that contains the data you want to import, then click **Next**.



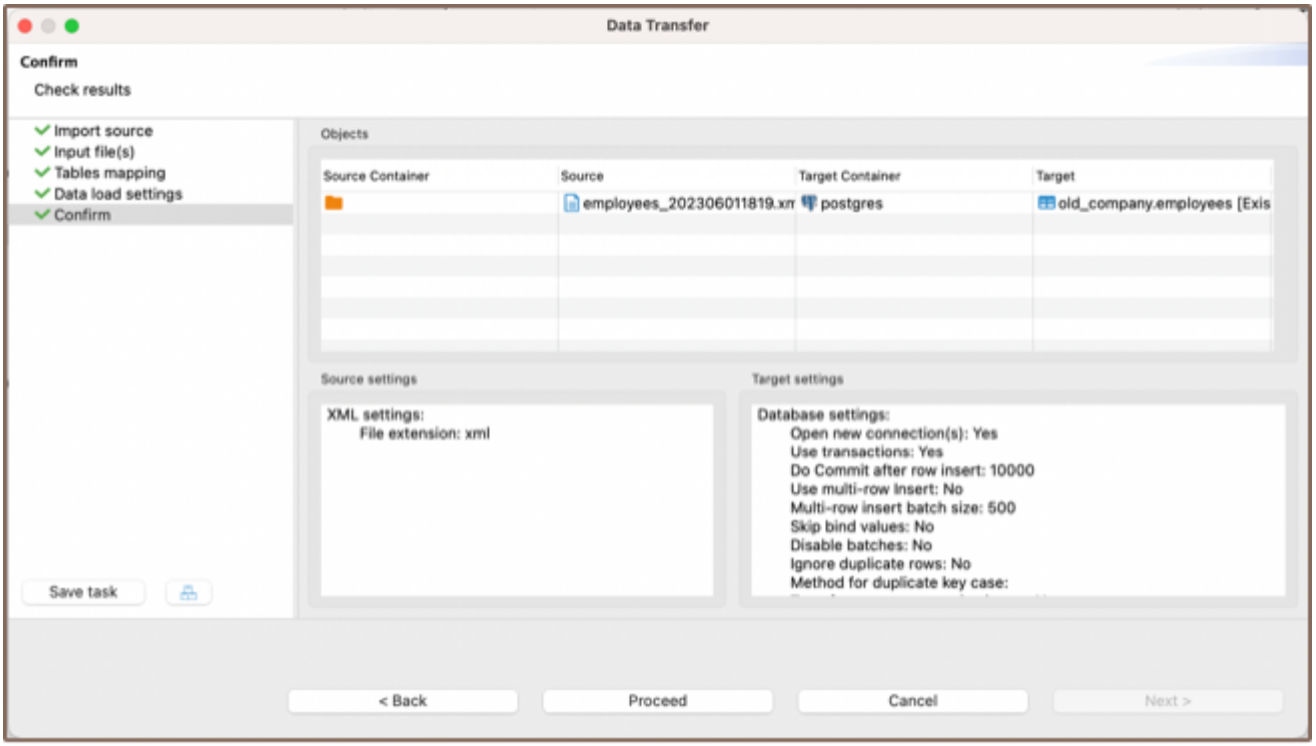
4) In the next window,w set XLSX-to-table mappings. Please refer to our guide for detailed information on the mapping process.



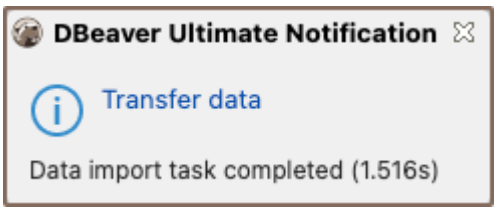
5) select your data load settings in the subsequent window, and then click **Next**. For more information, please refer to our article's section Data load settings.



6) In the final window, you can review all the settings you selected earlier. If you missed something, you can go back and adjust it. Once you are happy with everything, finish the import by clicking **Proceed**.



7) If the file is valid and there are no errors, you will see a notification window with information about the completion of the task. You can keep working with your database during export, as the data loading will be performed in the background.

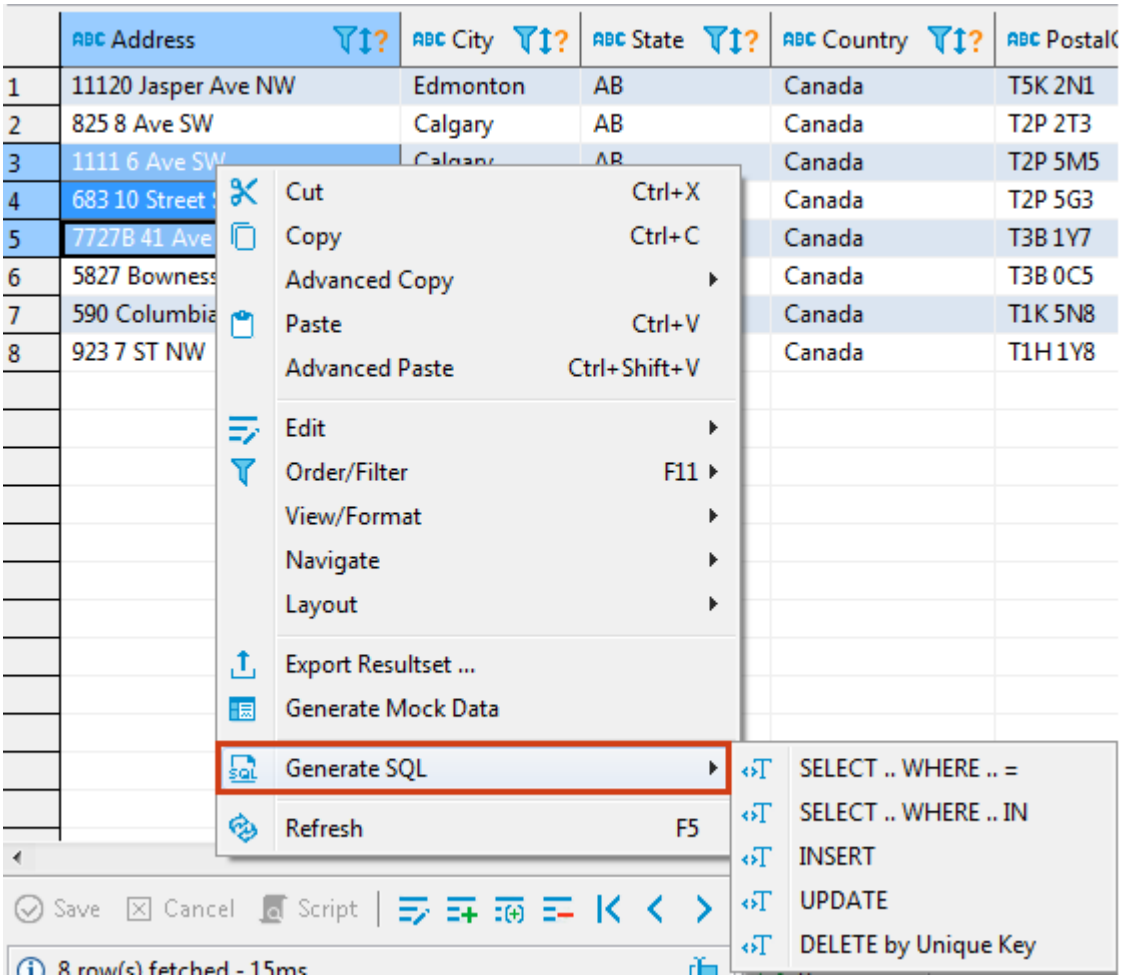


Importing data from the database table

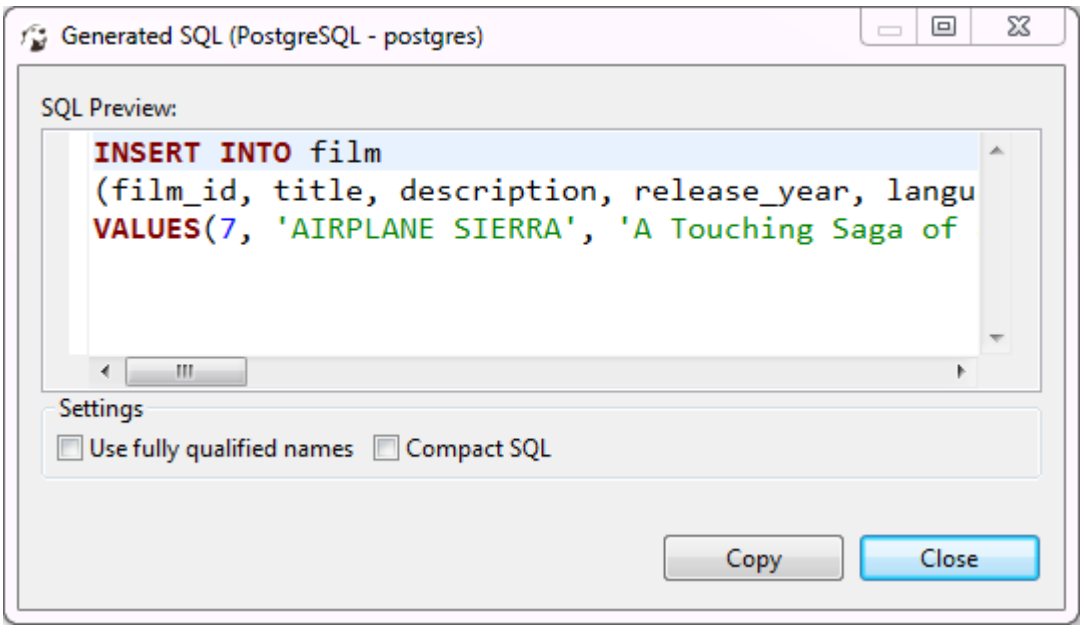
DBeaver offers seamless data migration capabilities, allowing you to transfer data from one database table to another. For more detailed instructions and insights on data migration, you can refer to our Data Migration guide.

SQL Generation

You can generate SQL statements (SELECT/INSERT/UPDATE/DELETE) based on selected rows. To generate SQL, right-click the selected rows and click **Generate SQL** and then one of the SQL commands on the context menu:

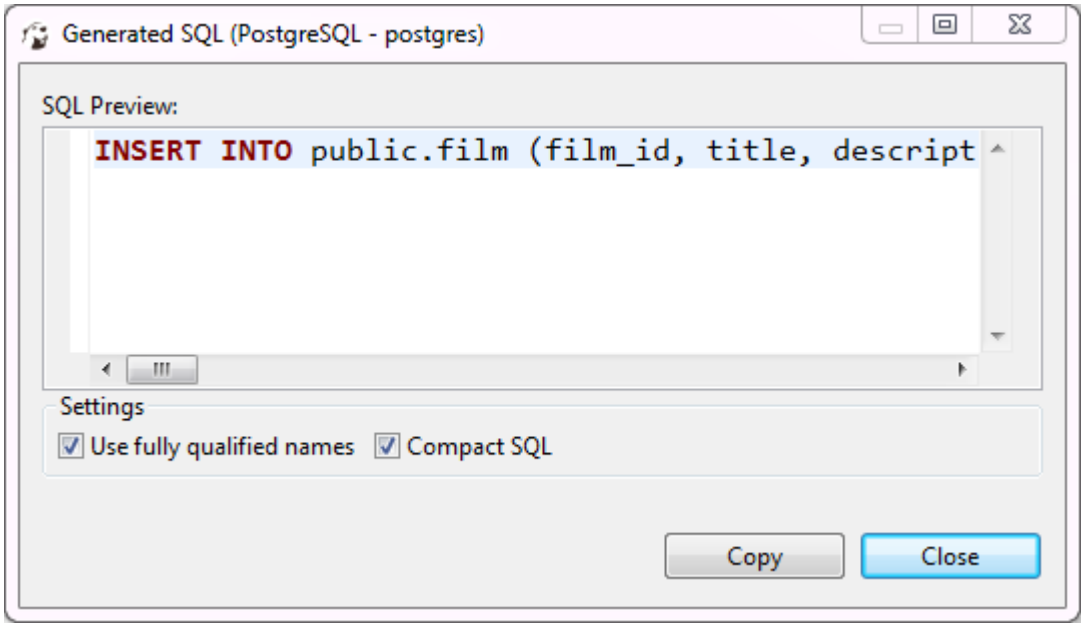


The SQL result opens in a separate window where you can view and copy it:



To use table names in the format '[schema name].[table name]', select the **Use the fully qualified names** checkbox.

To wrap the SQL query into one line, select the **Compact SQL** checkbox:



Working with spatial/GIS data

Spatial data, often represented as either geometric or geographical values, can be visualized on a map or graph. A geometric object is typically composed of a sequence of points that define its shape. For a more comprehensive understanding of spatial data, you can refer to this detailed explanation.

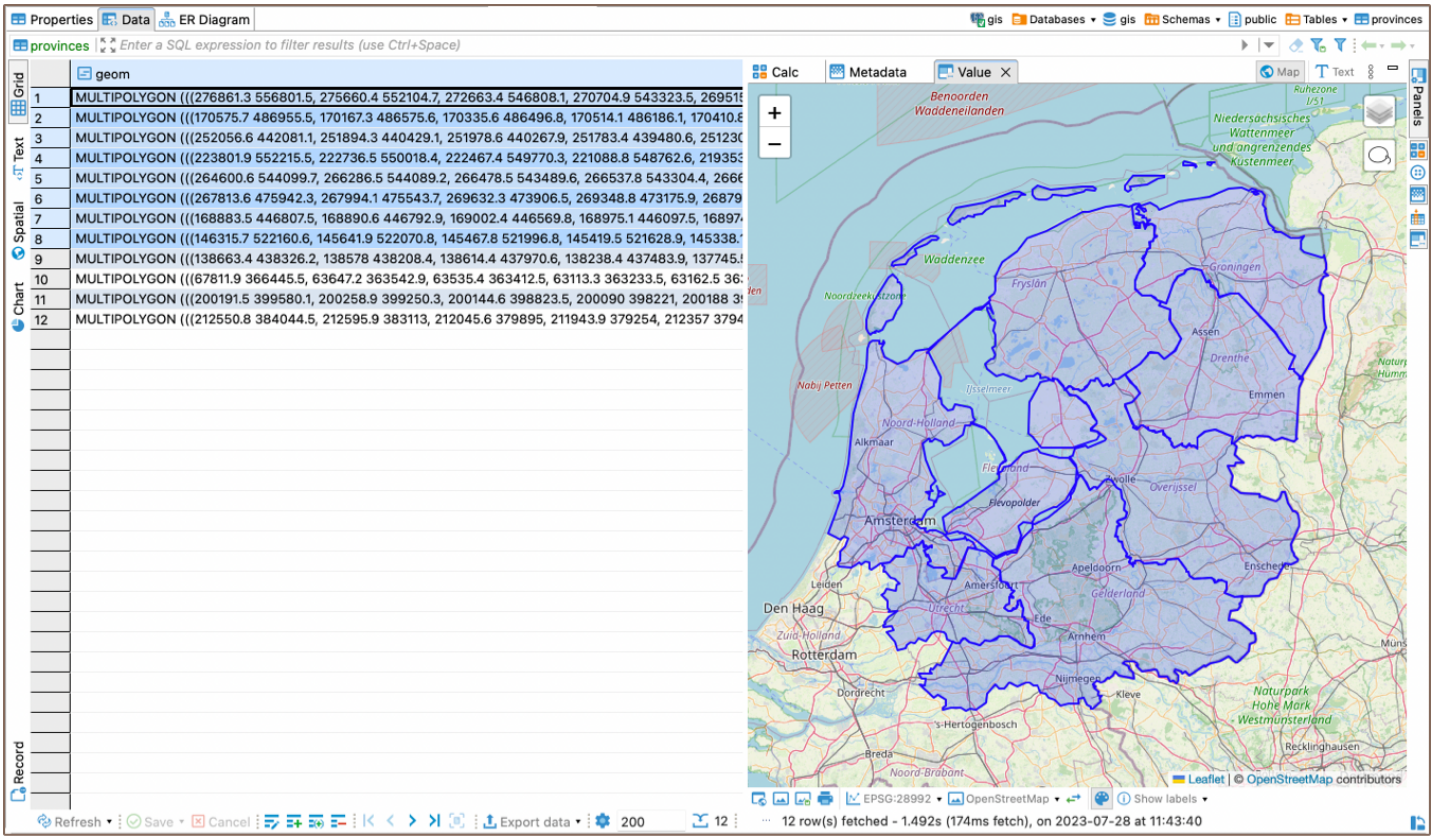
DBeaver's support of spatial data covers the following databases:

- PostgreSQL (PostGIS)
- MySQL
- SQLite (GeoPackage)
- H2GIS
- SAP HANA
- Oracle ★
- SQL Server ★

Each of these databases has its own set of features and capabilities for handling spatial data, and DBeaver is equipped to work effectively with each of them.

Spatial data viewer

The Spatial data viewer in DBeaver is a tool designed to visualise and interact with spatial data. It provides a graphical representation of your spatial data, enhancing your ability to understand and analyze it.



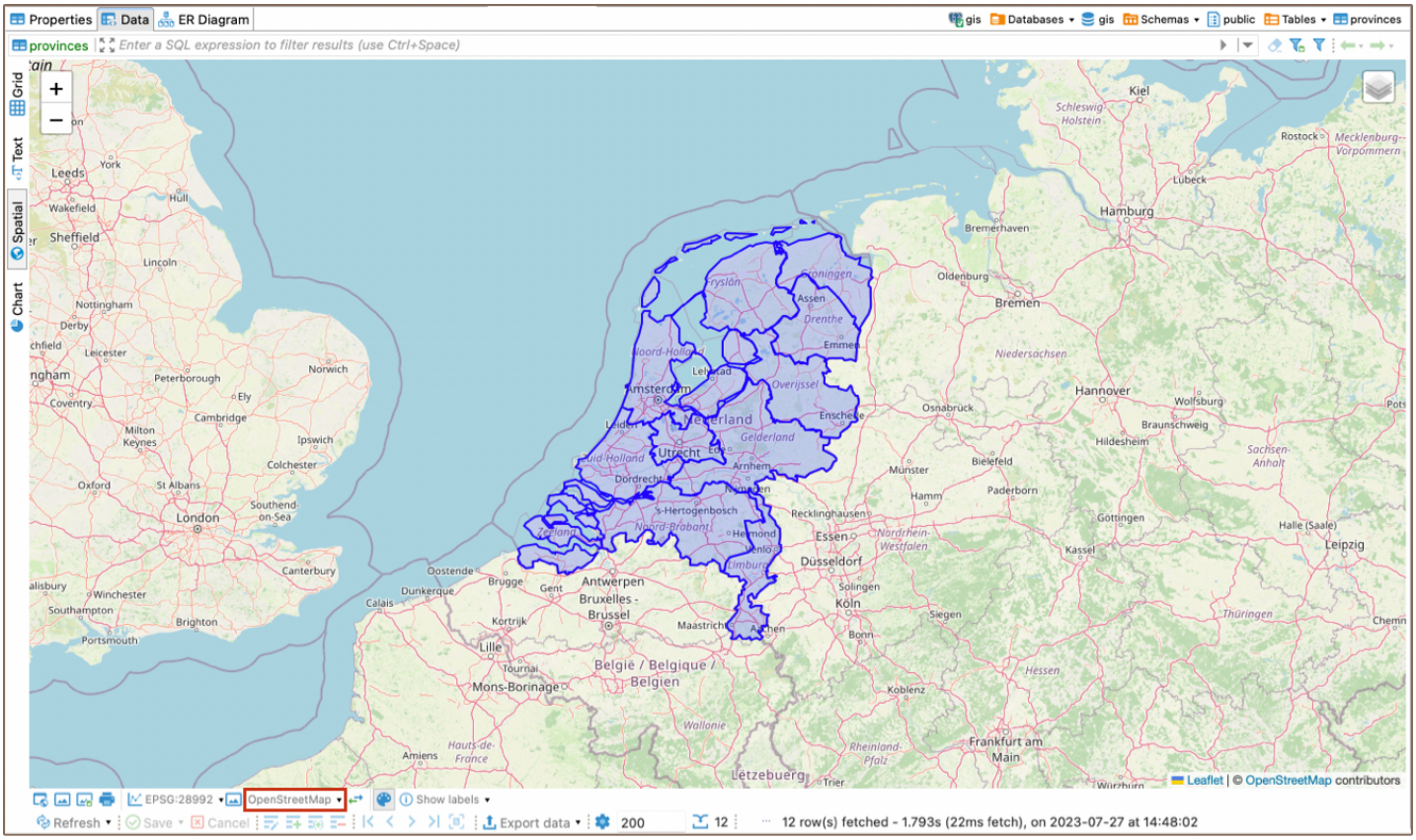
Differentiating data on the map

Every table column has its own color on the map. This helps you find the information you need on the map if you know which column it belongs to. If you click on an object on the map the following information will be displayed:

1. Name of the column in the header
2. Displayable data (strings, numbers, dates etc.) from every other column in the corresponding row

Tile layer management

DBeaver has several predefined map tiles. The tiles can be chosen with the combo below the viewer:



You can select which tile layers you want to see using the layers button. This button includes a drop-down menu where you can either select a layer or click on the **Manage** option.

The **Manage** option provides the ability to upload your own layers and hide pre-installed layers from the menu.

Defining custom tile layer

At this point, you may be wondering what to put in the Layers definition box. Here is a brief explanation.

DBeaver's spatial data viewer uses Leaflet under the hood. To define a custom tile layer, you wll need to provide arguments for the `L.tileLayer()` function in the Layers definition box. This function is responsible for loading and displaying the tile layers on the map.

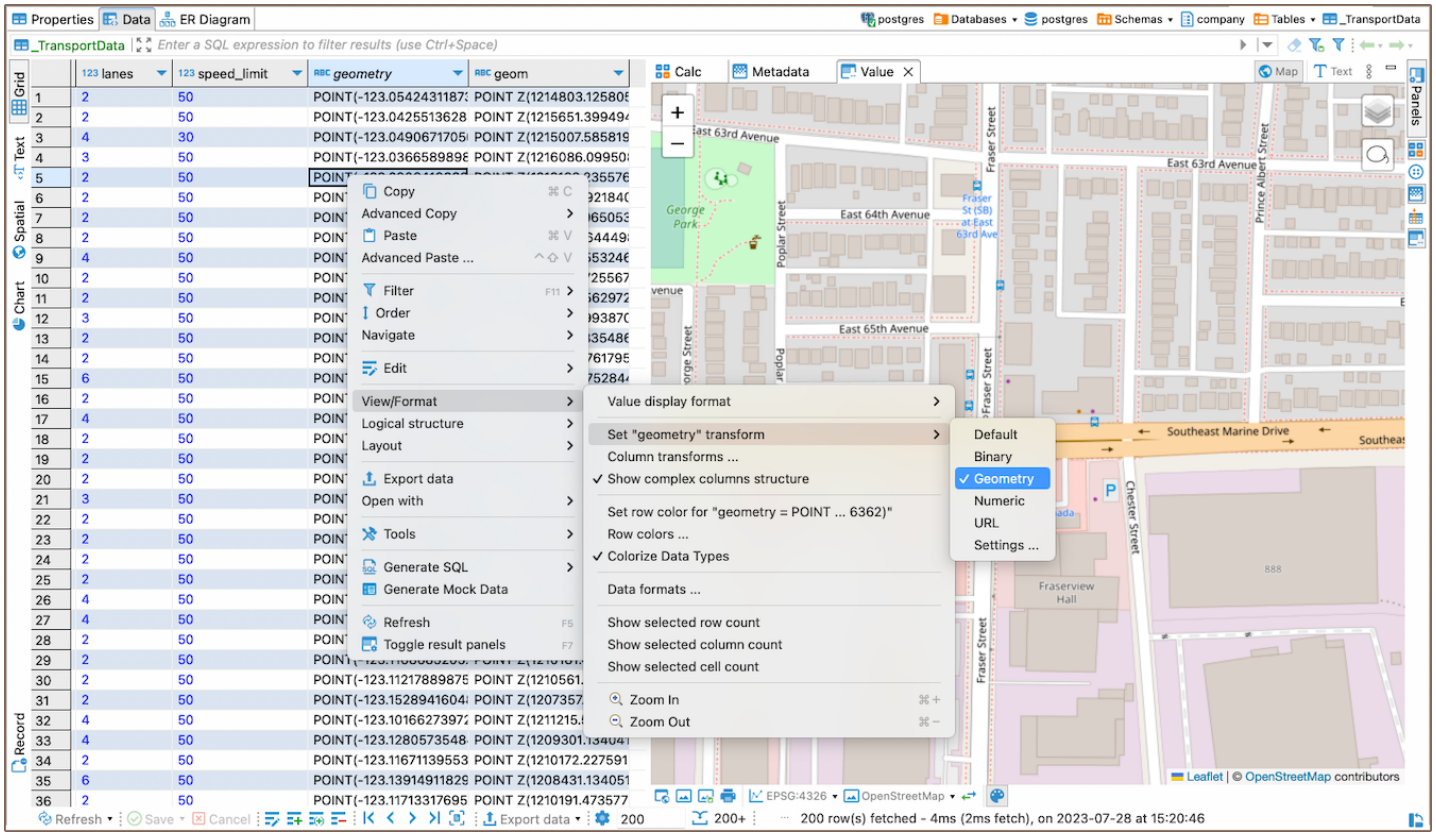
For more detailed information on how to use the `L.tileLayer()` function, refer to the official Leaflet documentation. To get started, you can also look at the definition of predefined tiles in DBeaver.

Viewing string or binary data from any Database on a map

In DBeaver, you can visualize your geodata on the map, regardless of whether it is stored as string or binary data. To do this, select the data cell and navigate to the **View/Format** setting. From there, choose the **Set columnName format** option and select **Geometry** from the available formats.

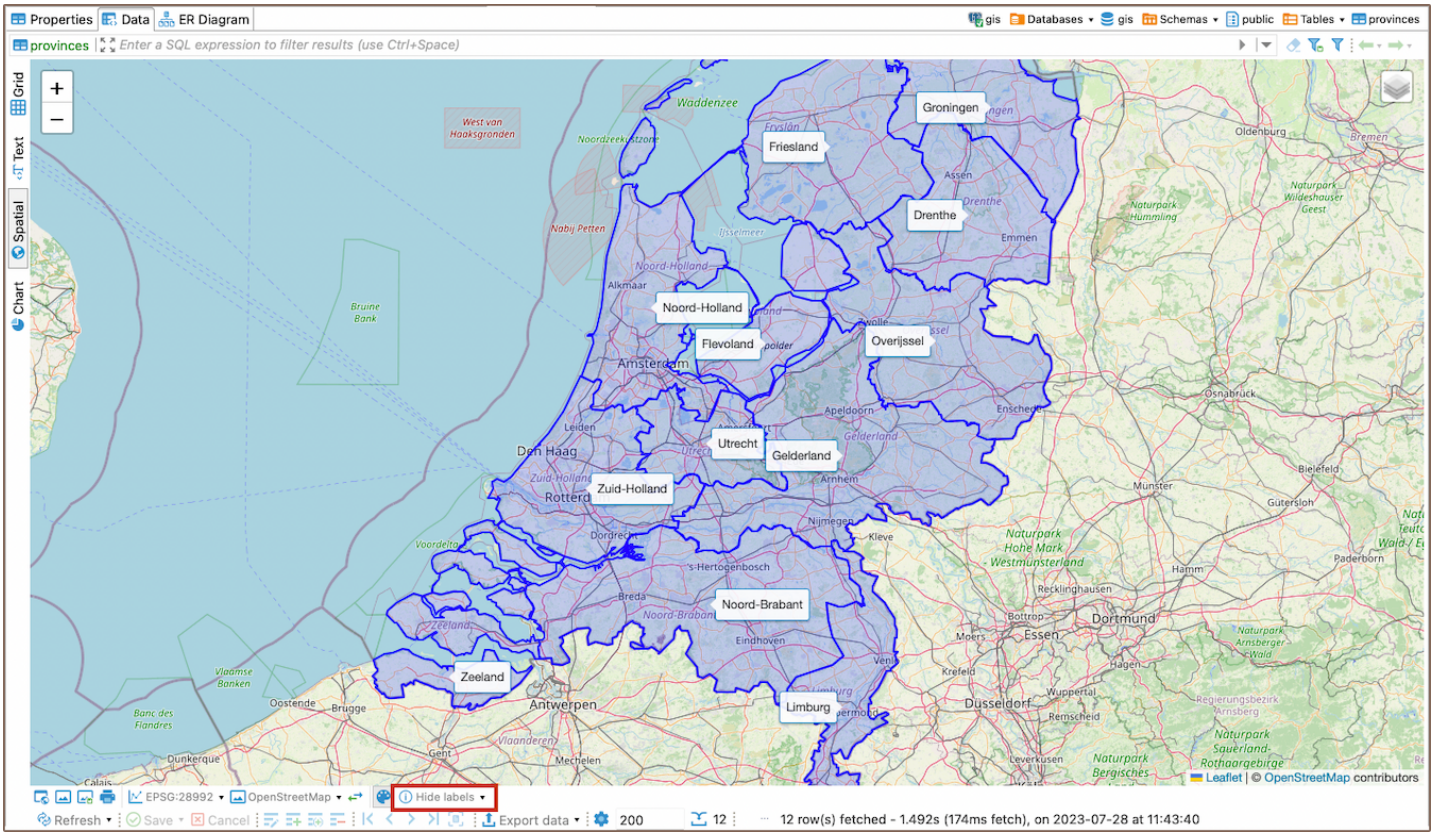
This feature allows you to transform and visualize your data in a spatial context, enhancing your understanding of the data.

Tip: It is particularly useful when working with geospatial datasets, as it enables you to see the geographical distribution and relationships of your data.



Labels on a map

You can also enable labels on the map using the **Show labels** button. This feature allows you to visualize specific data on the map, providing a more detailed view of your spatial data.



The **Show labels** button includes a drop-down menu with an option to **Configure labels**. Selecting this option opens a window where you can customize what data will be visualized in the labels, the order of the data, and the delimiter between the data.

Tip: Once the labels are enabled, the button's name changes to **Hide labels**, allowing you to easily toggle the display of labels on the map.

Copying the viewport/selection bounds

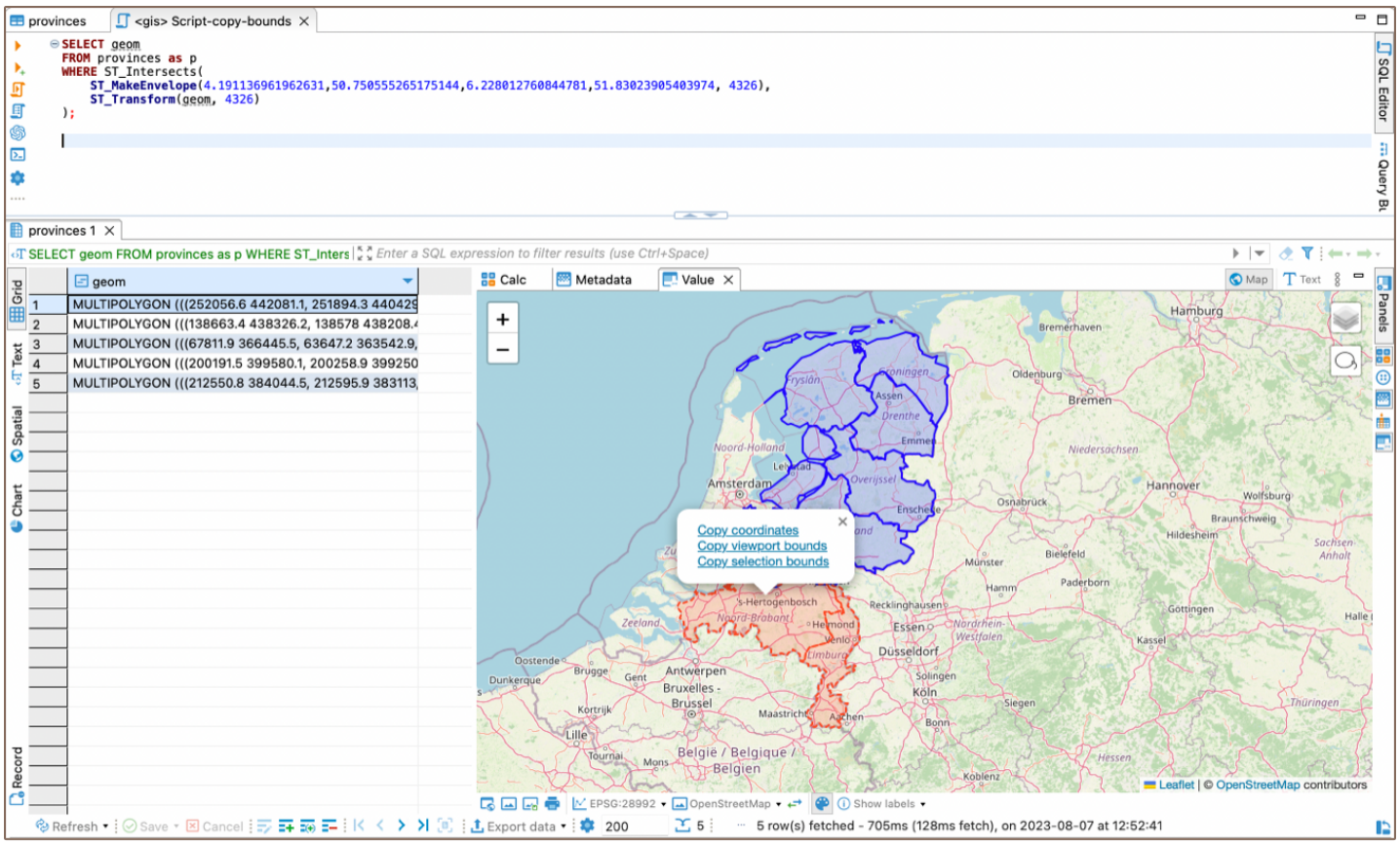
DBeaver's spatial data viewer also includes a feature that allows you to copy the extent of the current viewport area.

To use this feature, adjust the viewport to the area of interest using the zoom and drag functions. Then, select the **Copy viewport bounds** option. This action will copy the coordinates that represent the extent of the current view area. You can then paste these coordinates into your SQL query or any other place where you need to use them.

Tip: This feature can be useful when you need to examine all spatial data within a specific area or when you want to check their attributes in relation to a particular spatial object.

Here's an example of how you can use these coordinates in an SQL query:

```
SELECT geom
FROM your_table
WHERE ST_Intersects(
    ST_MakeEnvelope(<paste bounds here>, 4326),
    ST_Transform(geom, 4326)
);
```



In this query, you replace `<paste bounds here>` with the coordinates you copied from the viewport.

- The `ST_MakeEnvelope` function creates a rectangular polygon using these coordinates.
- The `ST_Intersects` function then finds any spatial objects in the provinces table that intersect with this rectangle.
- The `ST_Transform` function is used to ensure that the geometries are in the same coordinate system (in this case, `EPSG:4326`).

Tip: If you want to copy the extent of a specific selection, you can use the **Toggle lasso** tool to **Copy selection bounds**, and copy the coordinates.

Additional features

Copying coordinates

You can copy coordinates to the clipboard from any point on the map: just right-click anywhere and select **Copy coordinates**. It copies the coordinates formatted as `latitude`, `longitude` to the clipboard.\ **NOTE:** The coordinates are copied according to `EPSG:4326 CRS` and are just raw numbers. You may need to remove a comma and switch the `latitude` and `longitude` places to correctly insert it into a database.

Miscellaneous buttons

You can use buttons at the bottom of the view for additional features:

- Open the temporary generated `HTML` file in your default browser.
- Copy the current map to the clipboard as a picture.
- Save the current map as a picture into the selected folder.
- Print the current map.
- Flip the `latitude` and `longitude` coordinates in the source data. This can be useful if the data in your table is saved in (`latitude` / `longitude`) format while the Leaflet reads it as (`longitude` , `latitude`). This button does not change anything in the source data, it just changes how this data is read to show the accurate information on the map.

Viewing Curve Geometries

You can now visualize the curve geometries in DBeaver's spatial data viewer. This enhancement allows you to work with complex spatial data types such as `CIRCULARSTRING`, `COMPOUNDCURVE`, `CURVEPOLYGON`, `MULTICURVE`, and `MULTISURFACE`.

You can use SQL functions like `st_geomfromtext` and `st_curvetoline` to create and view curve geometries.

Tip: This feature can be specifically useful when dealing with complex spatial datasets in PostGIS that include curved geometries. It allows for more precise representation and analysis of geographical features that are not adequately represented by simple polygons or lines, such as circular areas of interest, roads that follow curved paths, or areas defined by a radius from a certain point.

Working with XML and JSON

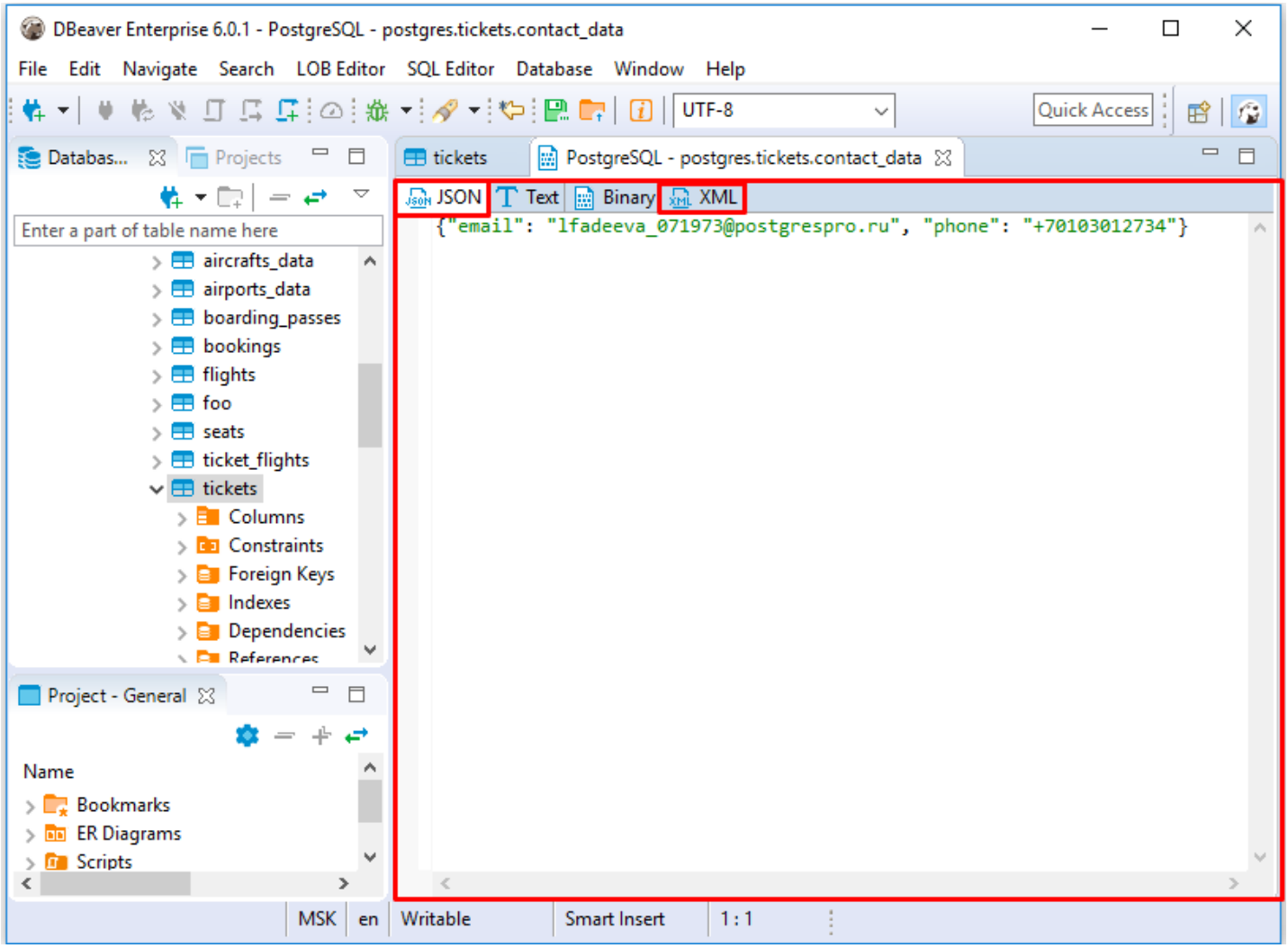
DBeaver supports XML and JSON column types (in relational databases) by using standard JDBC interfaces. This feature was added in JDBC4 so you will need JDBC4 compliant driver for your database.

In the Data Editor, you can edit XML/JSON data right in the table cells/ However, a huge amount of data may require a larger editor so you might want to save XML/JSON scripts to a local file or upload this type of data from a local file.

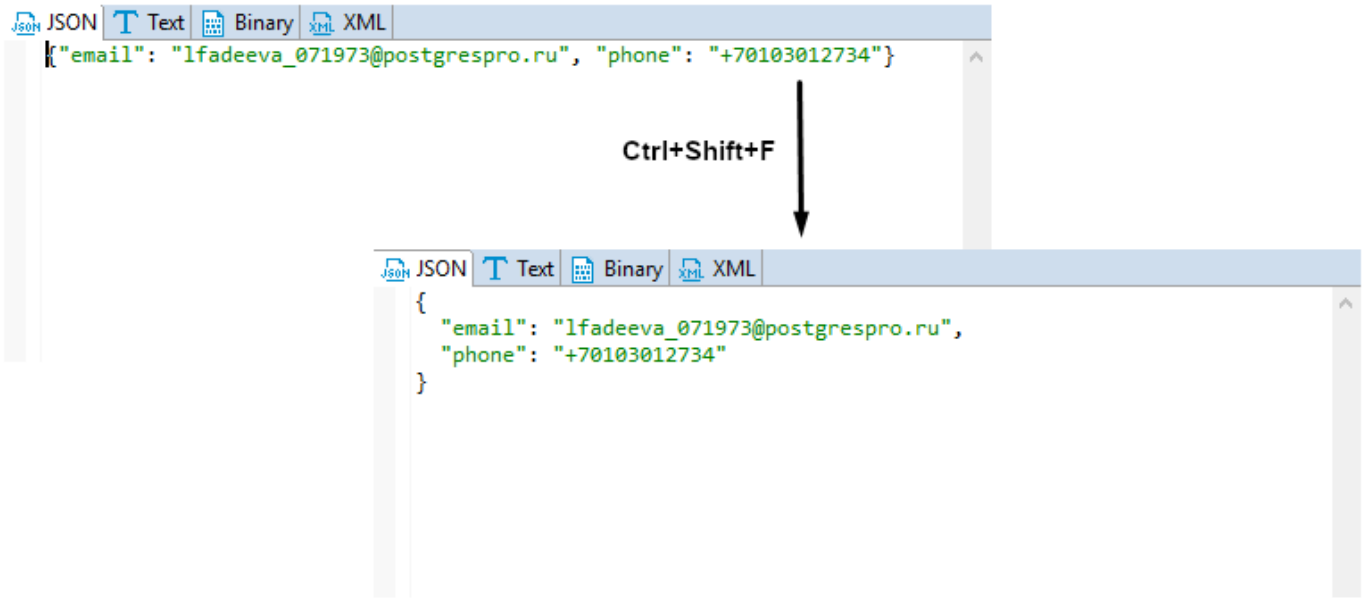
To open the full-size XML/JSON editor click the cell containing data in XML/JSON format and press `Shift + Ente`

.

By default the editor opens on JSON tab, open XML tab to modify XML data.



To auto-format XML/JSON script press `Ctrl + Shift + F` keyboard buttons.



Use `Ctrl + S` keyboard shortcut to save the changes made.

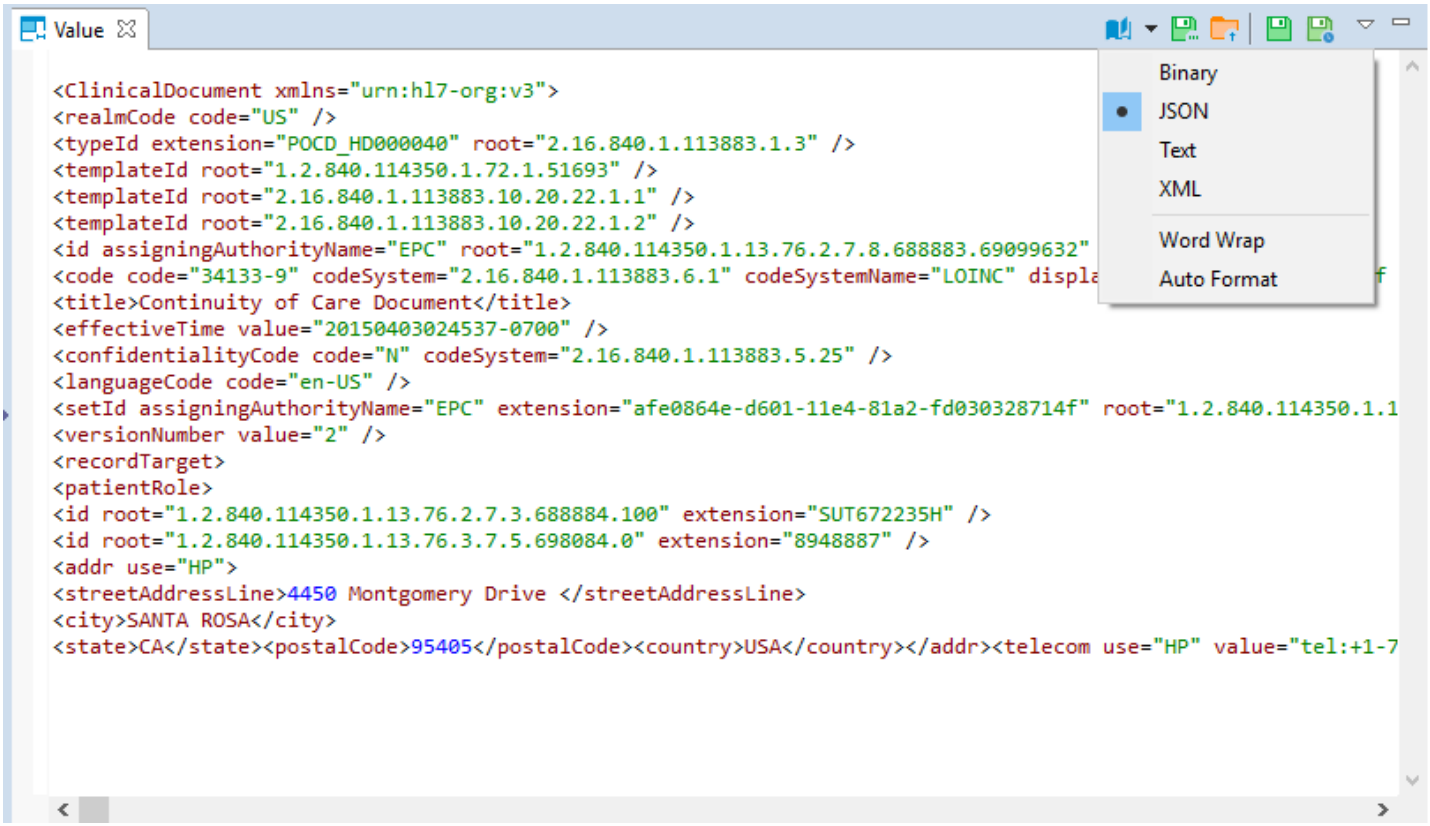
You can also edit XML/JSON content, save it locally and upload it from a local file with the help of **Value** panel toolbar.



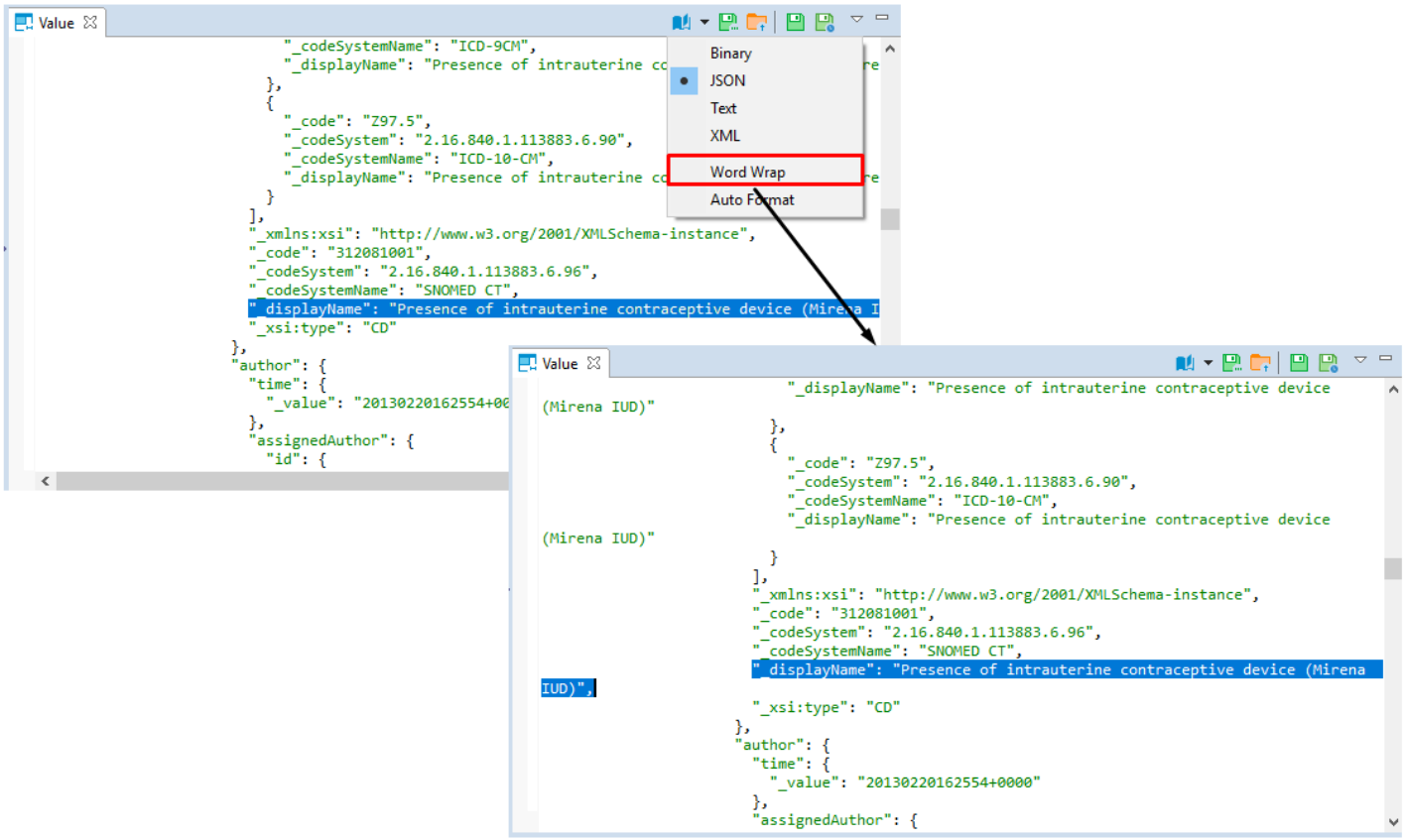
To upload data from a local file, press the **Load from file...** button .

To save the content to a local file, press the **Save to file...** button .

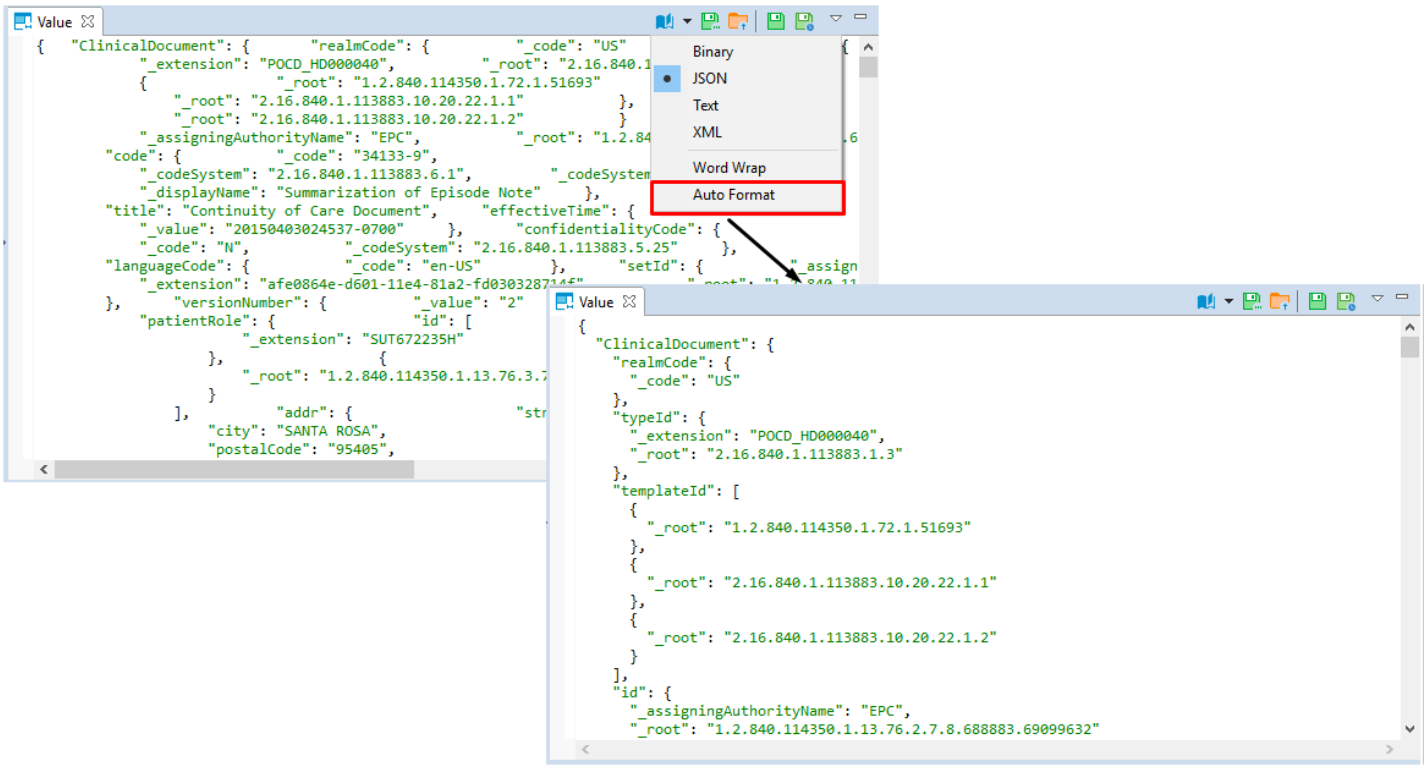
To switch between the formats, press the **Content viewer settings** button and select the format.



Use **Word Wrap** feature that wraps the text within a screen.



Use **Auto Format** feature to automatically change the appearance of XML/JSON script (fix spaces around operators / commas, fix indentation, etc) and make it more readable.

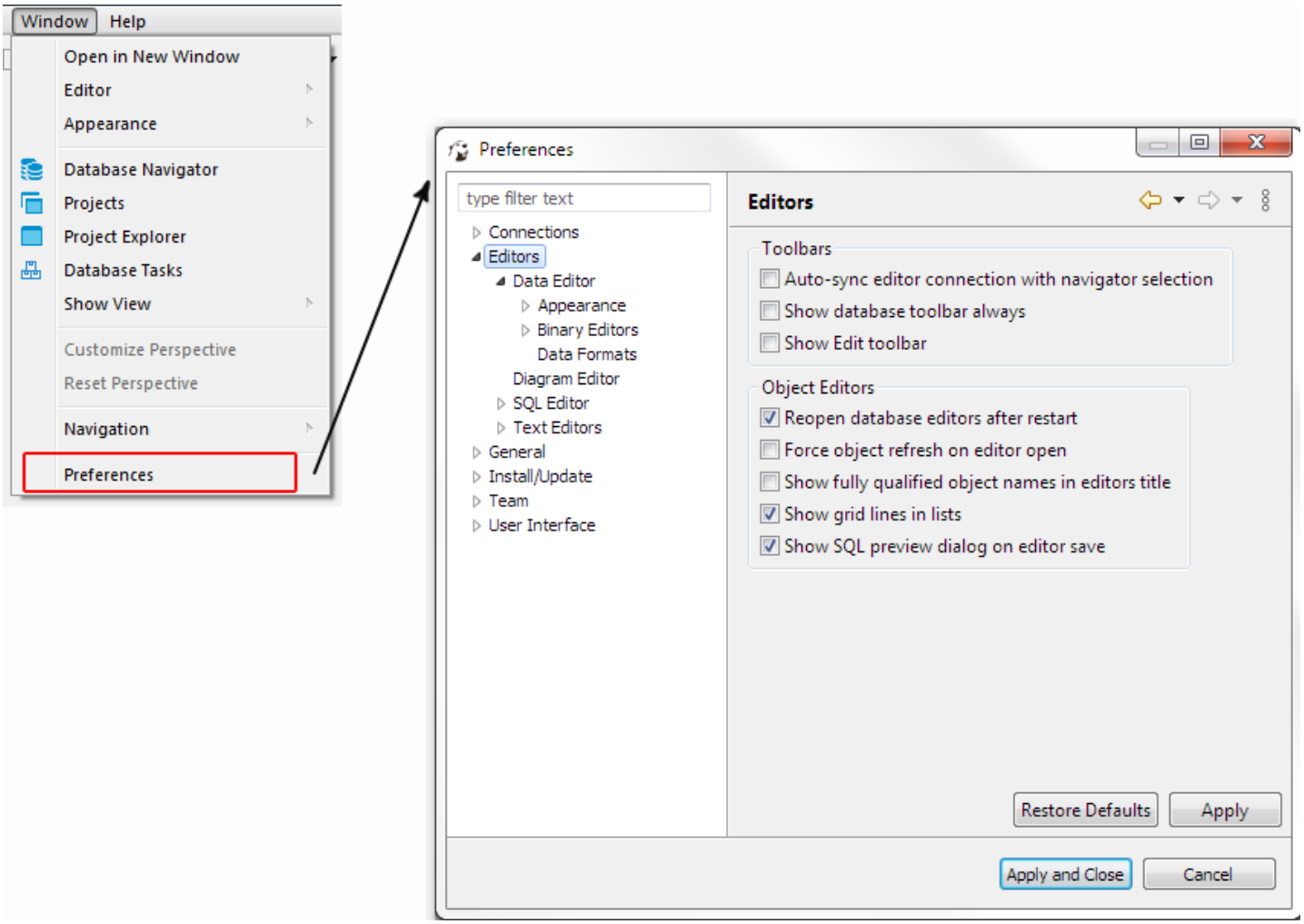


To learn more about **Value** panel, see Panels.

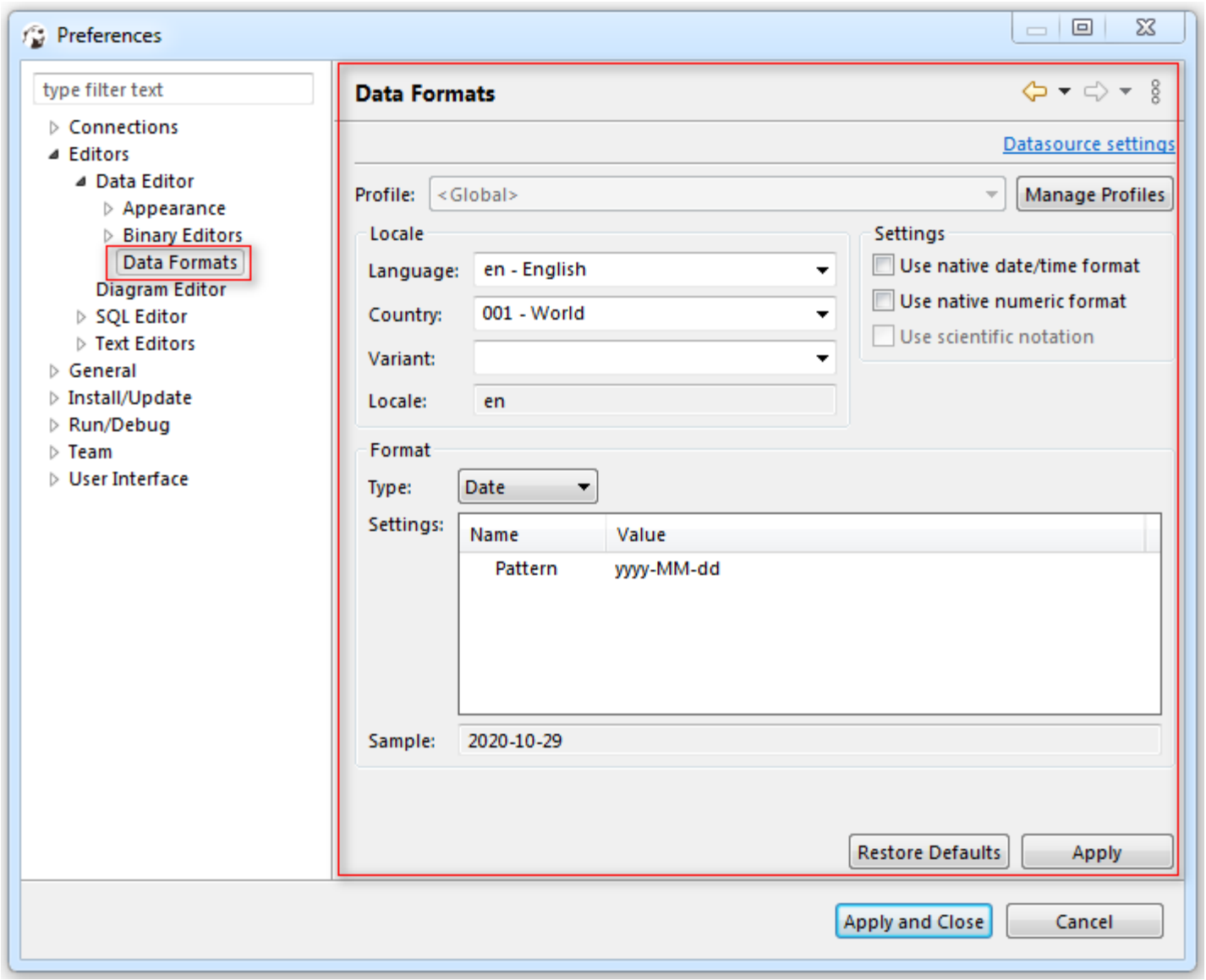
Managing Data Formats

The DBeaver formatting functions allow you to set up database locale and change datasource format settings. This feature can be very useful, for example, for database migration.

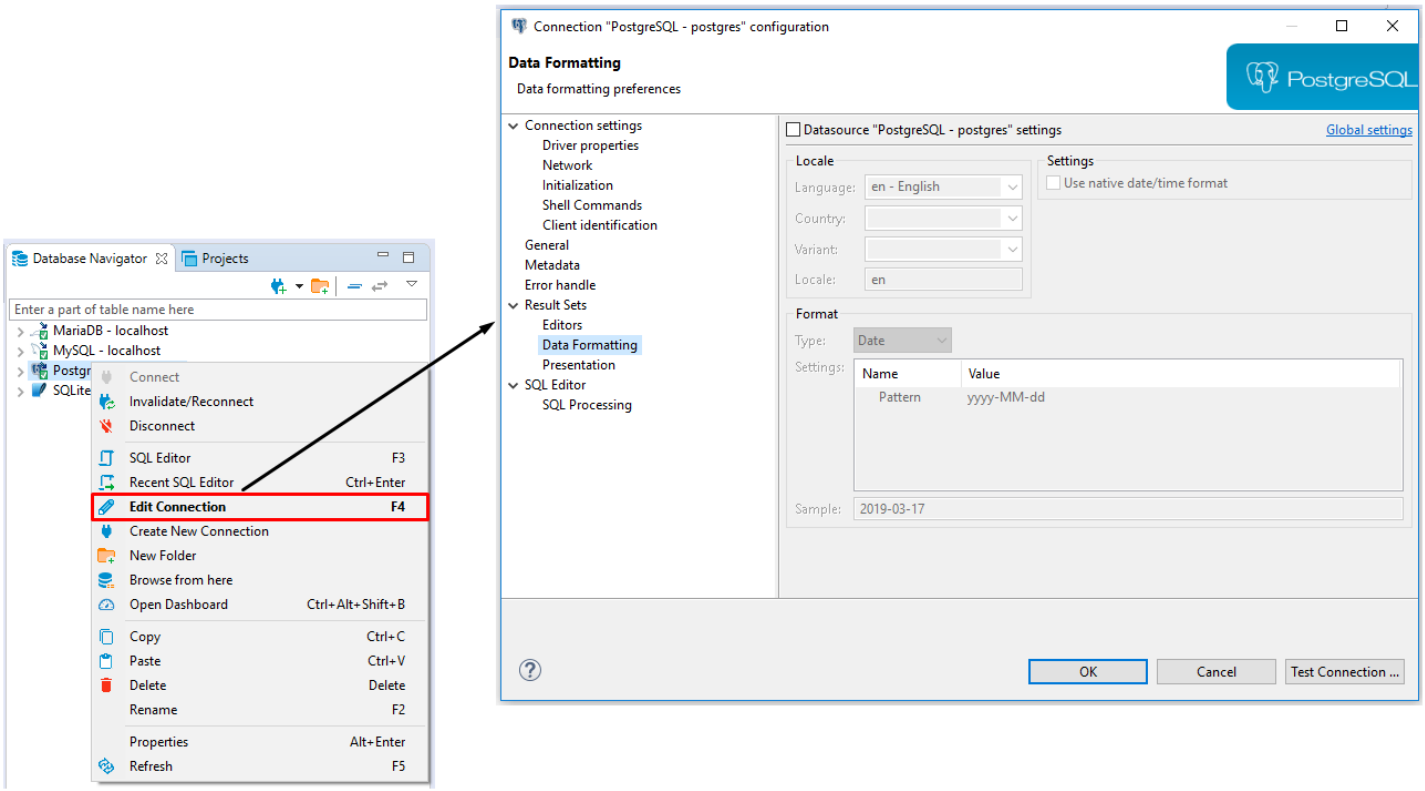
To change the data format settings use the option Window -> Preferences in main menu.



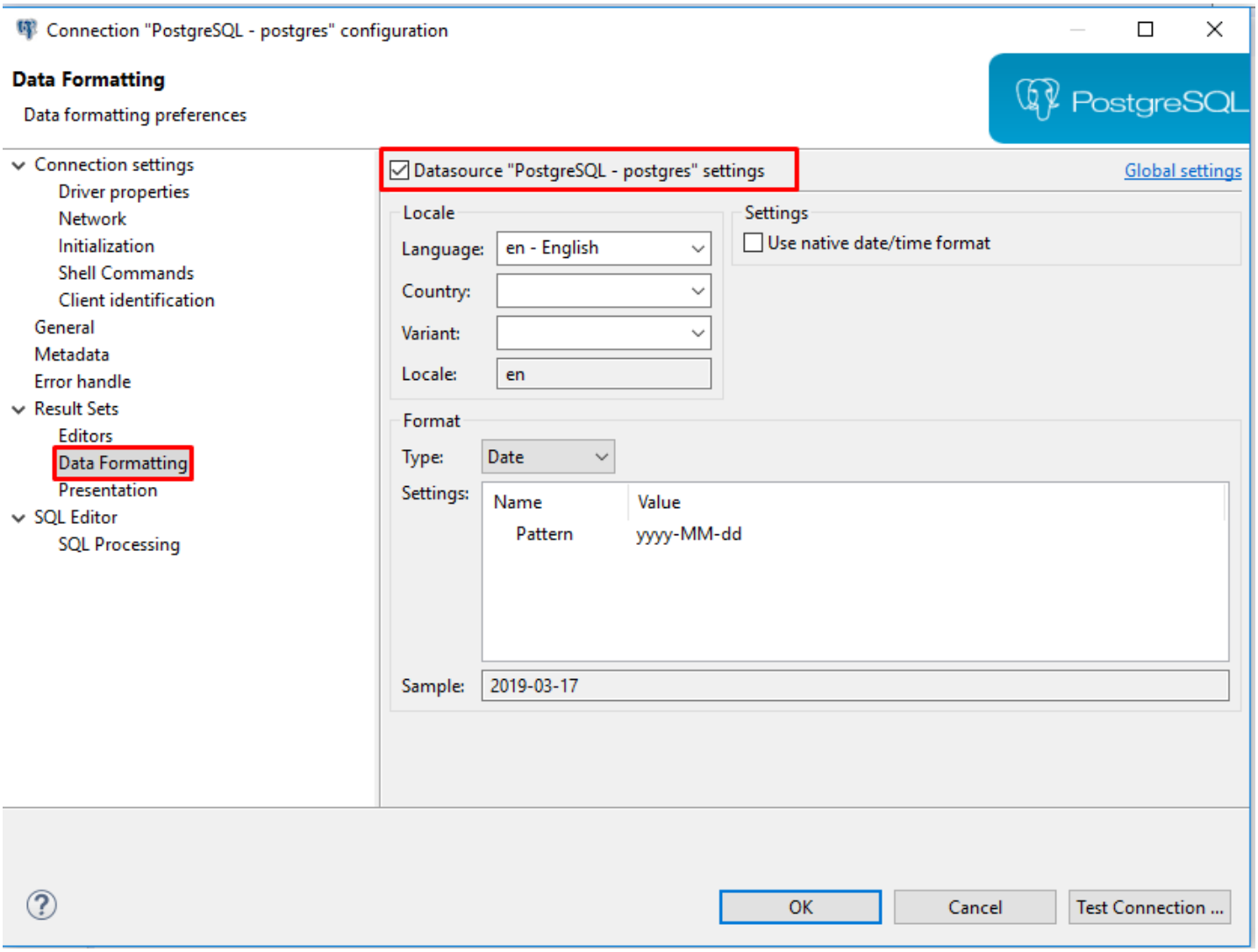
In the Preferences dialog box go to Editors -> Data Editor -> Data Formats.



Or, in the Database Navigator right-click a connection and select **Edit Connection** menu option.



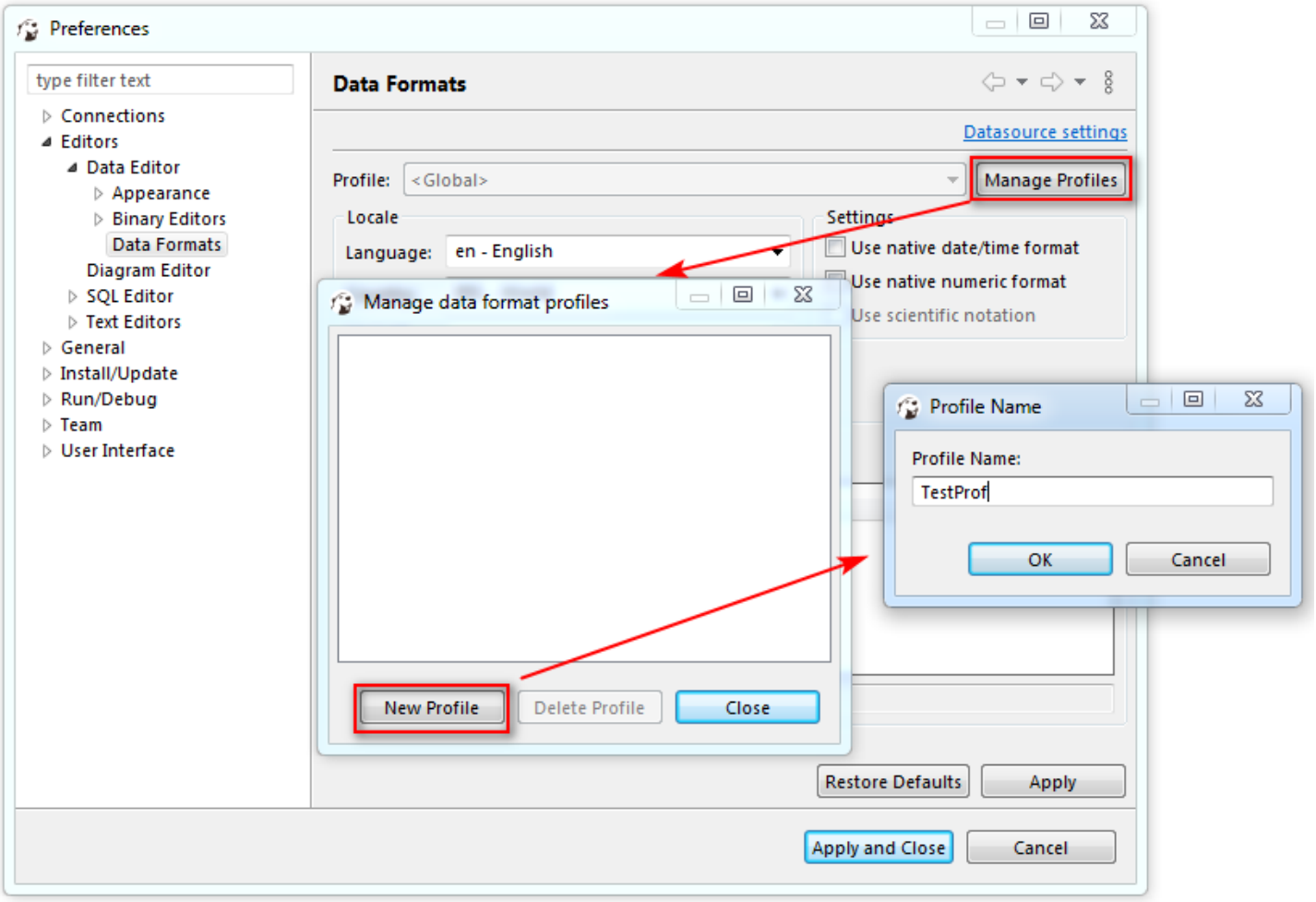
In the right area of the opened **Data formatting preferences** dialog window go to Data editor -> Data Formats and select the **Datasource settings** check box in the left area to customize the data format settings.



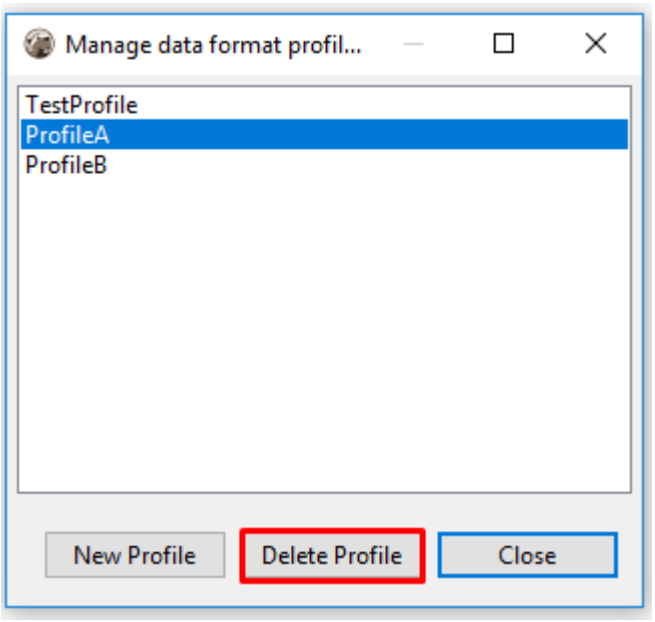
Data Format Profiles

Data format profiles allow you to apply a set of data format preferences to the whole current project by one click.

To create a data format profile press the **Manage Profiles** button. In the opened dialog window press button **New Profile**, define the name and press **Create**.



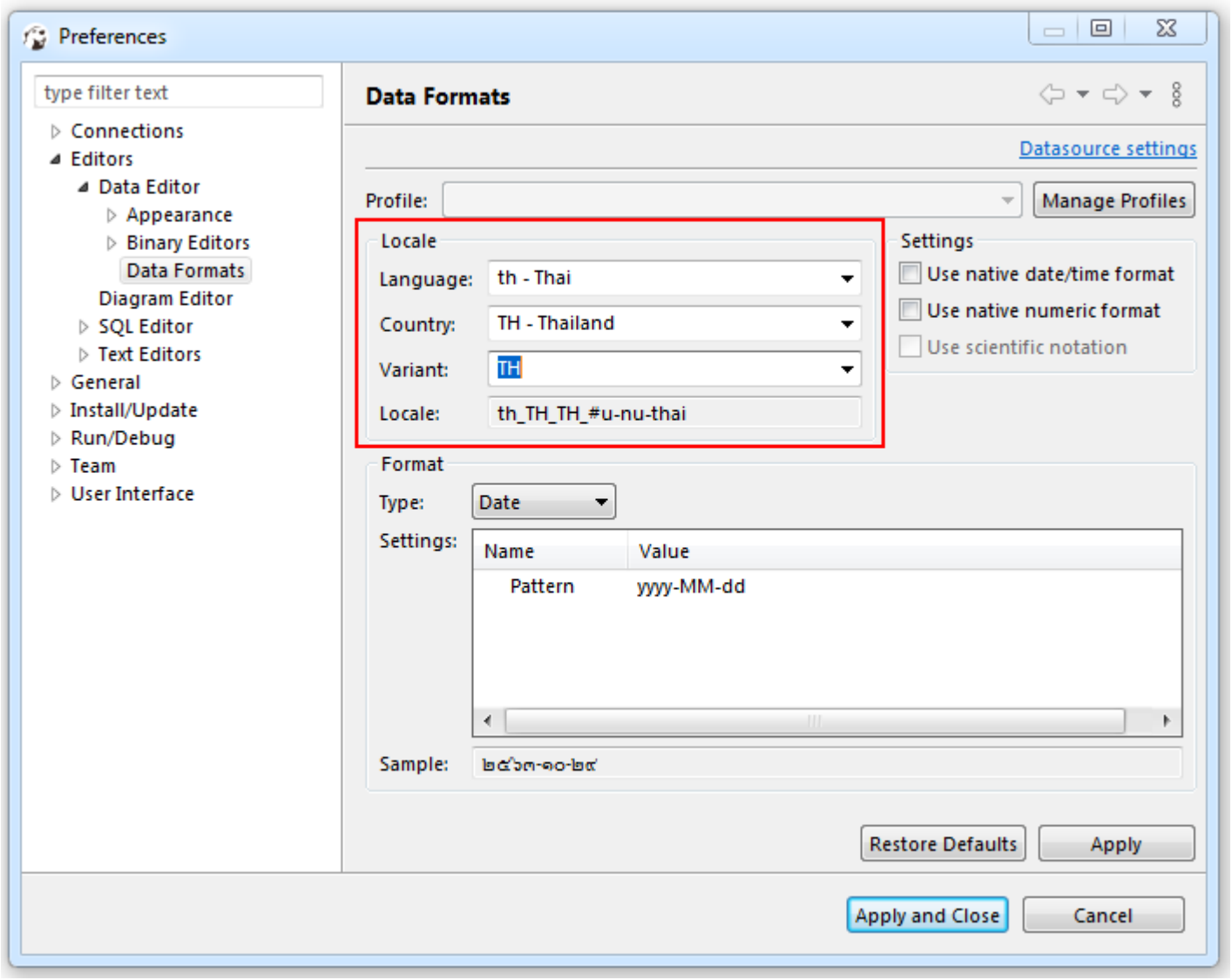
To delete a data format profile press the **Manage Profiles** button, then in the opened dialog window select the profile you want to delete and press the button **Delete Profile**.



Changing Data Formats

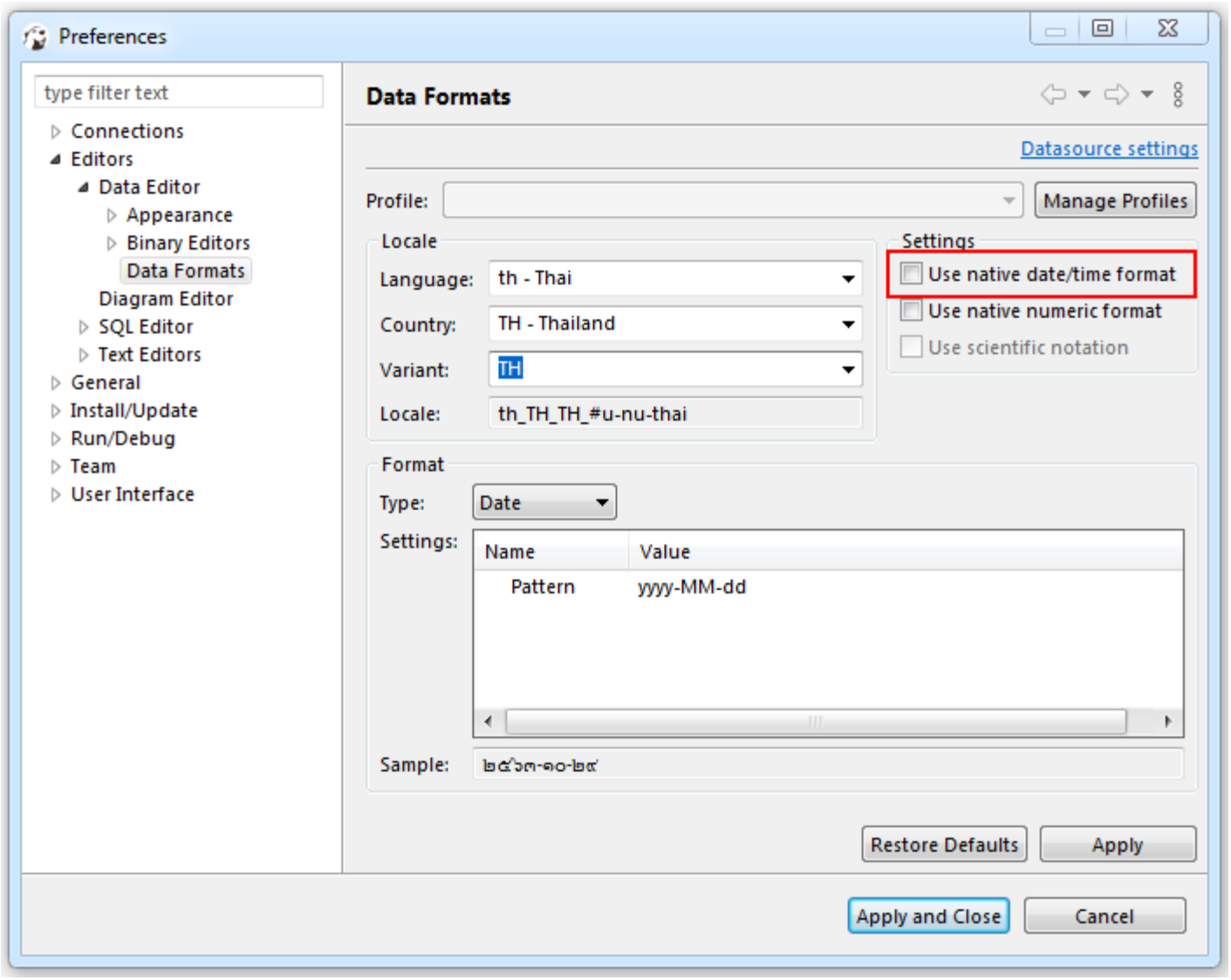
The following groups of data format settings can be adjusted:

Locale



To define this setting select a language, country and variant if available.

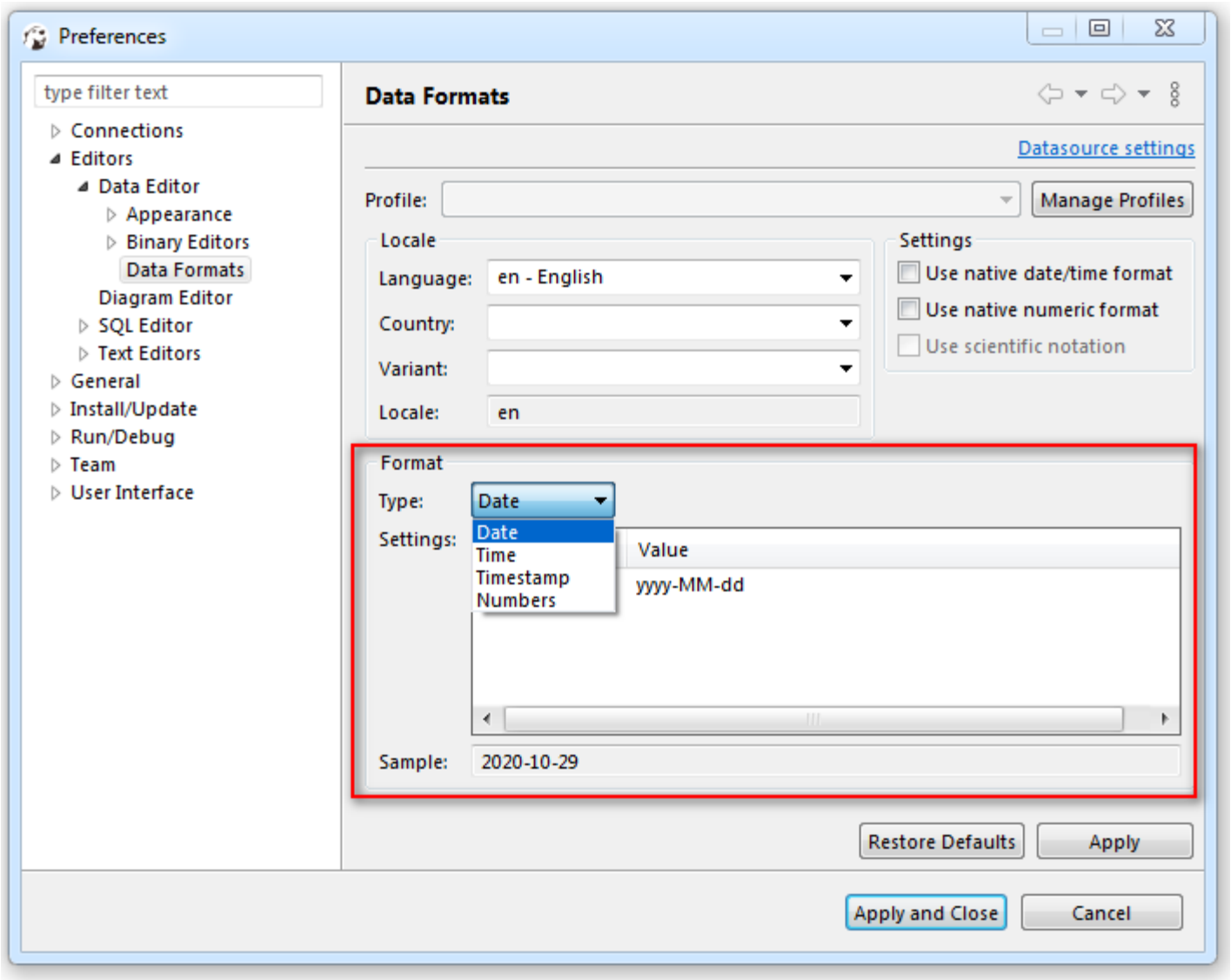
Native Date/Time Mode



Select **Use native date/time format** check-box and the data format originally built-in to the datasource will be used.

You can change the format of the following data types:

Data Type Format



The format of the following data types can be customized:

- Date
- Time
- Timestamp
- Numbers

Date

The default value for this data type is *yyyy-MM-dd*.

Pattern	Description
yyyy or y	Year of era (4 digits)

yy	Year of era (2 last digits)
YYYY	Week year
M	Month in year without leading zeros
MM	Month in year
MMM	Short month name in year
MMMM	Month name in year
D	Day in year
d	Day in month without leading zeros
dd	Day in month
E	Day name in week
G	Era designator
z	General time zone
Z	RFC 822 time zone

Time

The default value for this data type is *HH:mm:ss*

Pattern	Description
H	Hour in day (0-23) without leading zeros
h	Hour in day in am/pm (1-12) without leading zeros
HH	Hour in day (0-23)
hh	Hour in day in am/pm (1-12)
a	Am/pm marker
m	Minute in hour without leading zeros
mm	Minute in hour
s	Second in minute without leading zeros

ss	Second in minute
S	Millisecond
ffffff	Microseconds

Timestamp

The default value for this data type is *yyyy-MM-dd HH:mm:ss*

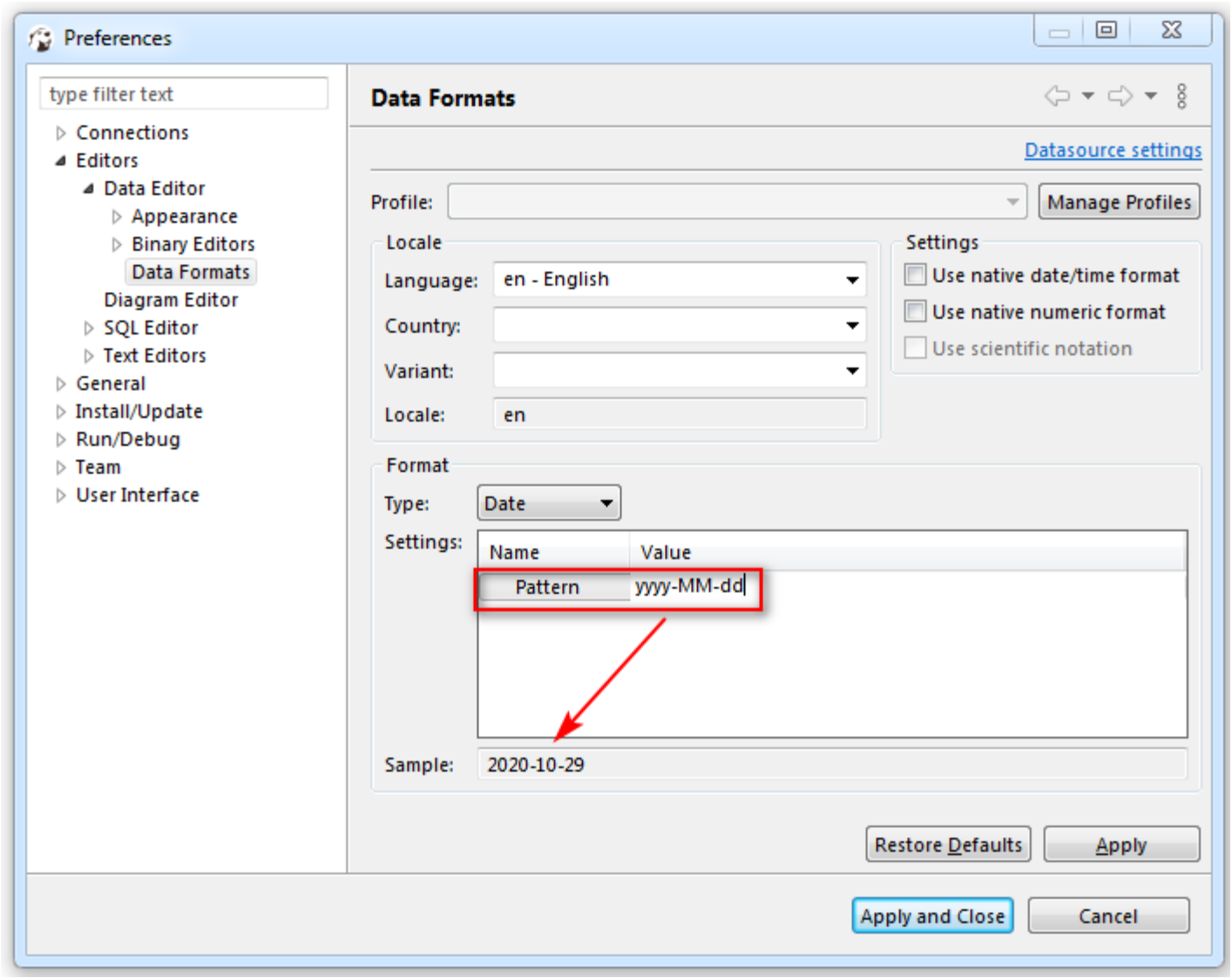
Link on Java date pattern documentation

Numbers

The following parameters can be configured from this type of data:

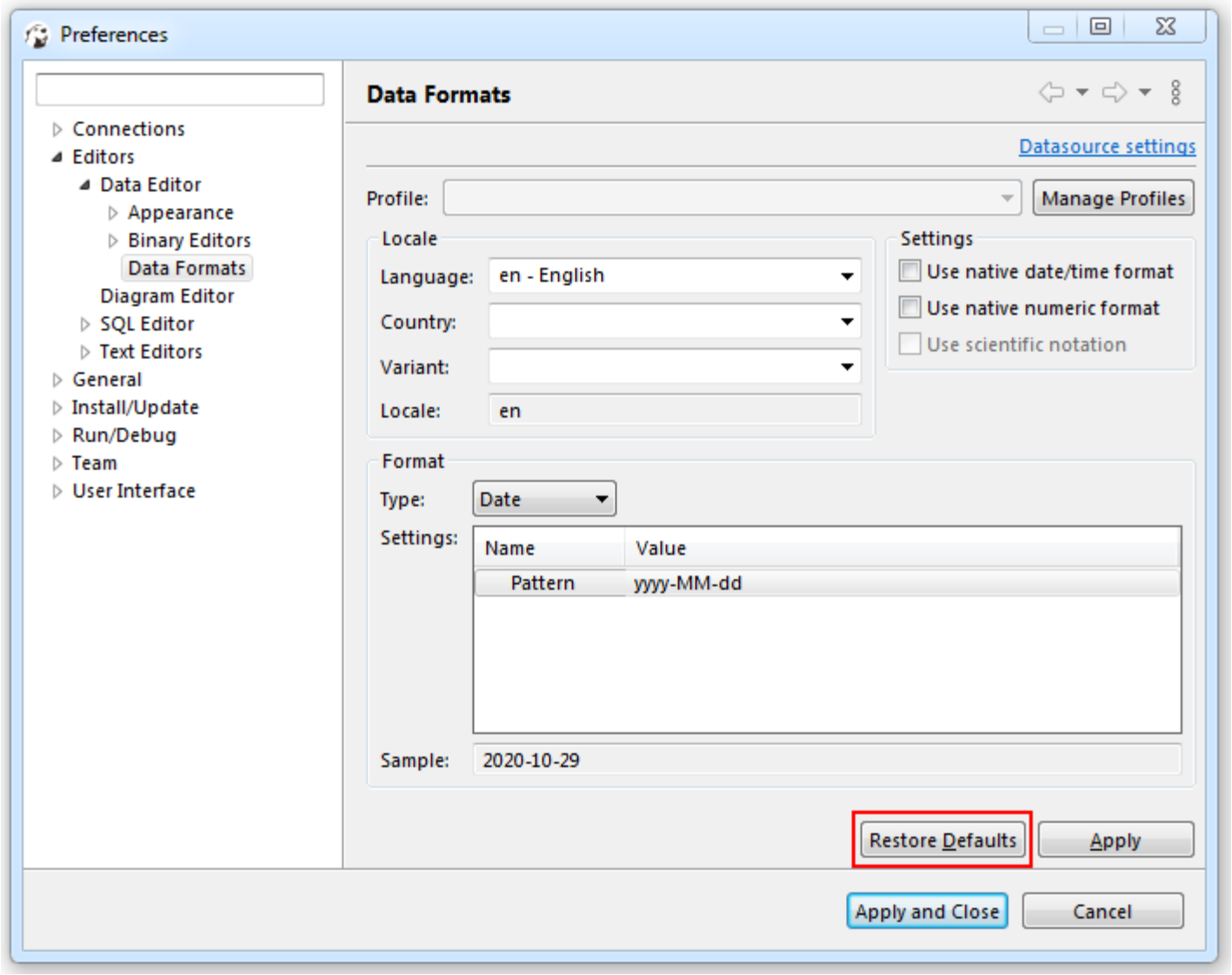
- **Use Grouping** - Long numbers can be hard to read if they have too many digits. For example, the factorial of 30 is 33 digits long! Select this check-box to enable Grouping mode, in which digits are displayed in clumps of 3 or 4 (depending on the current radix) separated by commas.
- **Maximum integer digits** - Defines the maximum number of digits to the left of the decimal point.
- **Minimum integer digits** - Defines the minimum number of digits to the left of the decimal point.
- **Maximum fraction digits** - Defines the maximum number of digits to the right of the decimal point.
- **Minimum fraction digits** - Defines the minimum number of digits to the right of the decimal point.
- **Use data type scale for fraction digits** - Some numeric columns or parameters may have a predefined scale, that is the maximum number of digits to the right of the decimal point. Select this check-box if you want the predefined precision to be used.
- **Rounding mode** - Specifies a rounding behavior for numerical operations capable of discarding precision. Each rounding mode indicates how the least significant returned digit of a rounded result is to be calculated. To learn more, please refer to Oracle documentation.

To change the data type format, change the value displayed in the **Pattern** area,save the changes made by pressing the **Apply** button and observe the expected result in the **Sample** field.



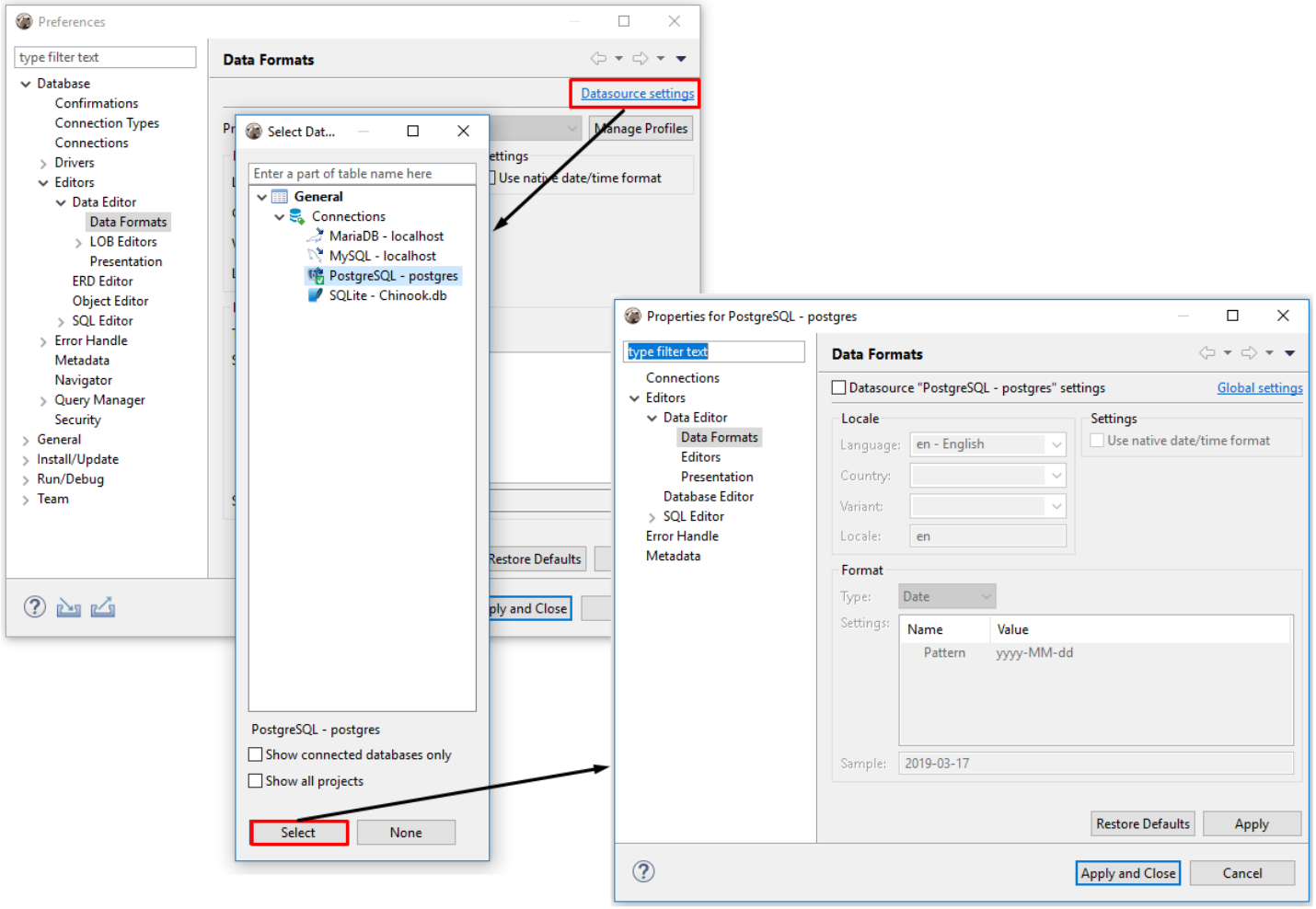
Restoring Default Data Formats

To restore the default data format settings, press the **Restore Defaults** button.



Datasource Settings

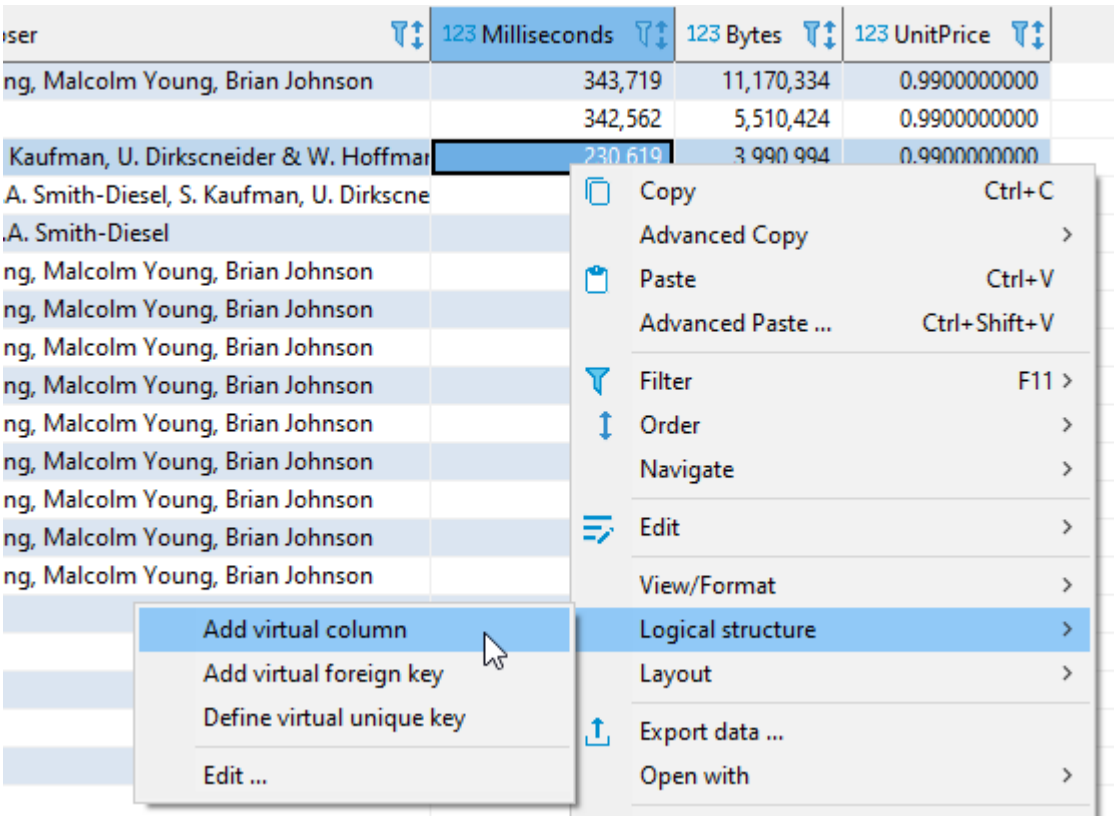
Press **Datasource settings** link to change data format settings for a particular datasource, then adjust the settings in the opened dialog box.



To save changes made press **Apply**.

Virtual column expressions

Expression language



You can use standard JavaScript-like expression language. DBeaver uses the Jexl engine to process expressions.

Language references and examples can be found here: http://commons.apache.org/proper/commons-jexl/reference/syntax.html

Column values

All columns' values in the current result set can be referred to by name. Expression `column1 + column2` will produce the sum of two numeric columns or concatenation of two string columns `column` and `column2`.

Standard functions

Standard functions are declared in namespaces.

You can refer to the functions in the namespaces as variables - `nsName.functionName(parameters)`).

math

You can access all math functions as `math.function(parameters)`.

You can find all supported math functions here: https://docs.oracle.com/cd/E12839_01/apirefs.1111/e12048/functmath.htm

geo

Function	Parameters	Description
wktPoint	(longitude, latitude)	Produces WKT (geometry) point out of two coordinates. Default SRID is 4326.
wktPoint	(longitude, latitude, srid)	Produces WKT (geometry) point out of two coordinates and SRID

content

If you have JSON or XML columns in your table, you can add a virtual column with an expression for these columns.

Use content.json()[parameter1][parameter2] pattern to create expression for JSON column.

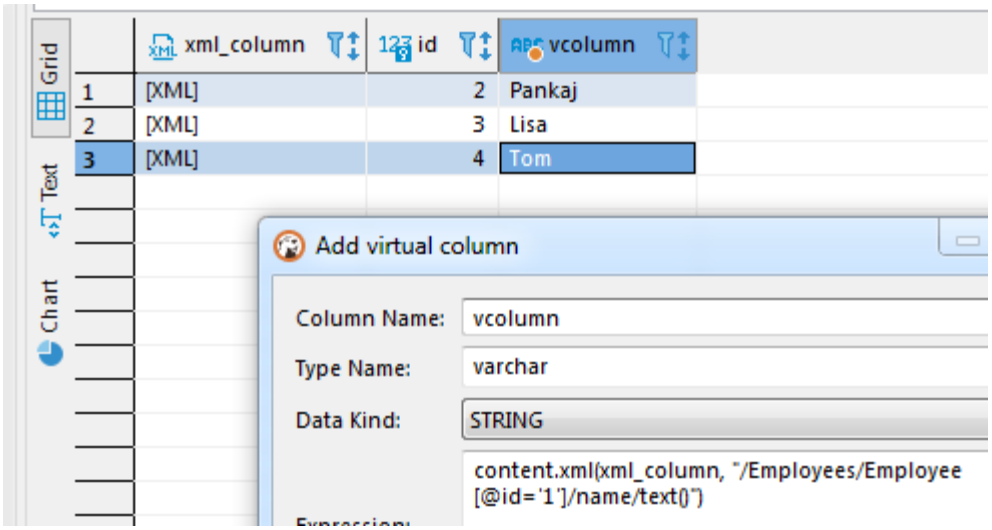
Expression example: `content.json(column1)['glossary']['GlossDiv']['title']`

You can read more about JSON parameters in the link on the top.

Use content.xml(columnName, "expression") or content.xml(columnName, "returnType", "expression") patterns to create expression for XML column. The quotation is important for parsing processes. XML expression can return types: string, number, boolean, node, nodeset - all these types can be used (!in quotes) for return data type clarification. content.xml(columnName, "expression") returns string by default

Expression example: `content.xml(column1, "nodeset", "/Employees/Employee[gender='Female']/name/text()")`

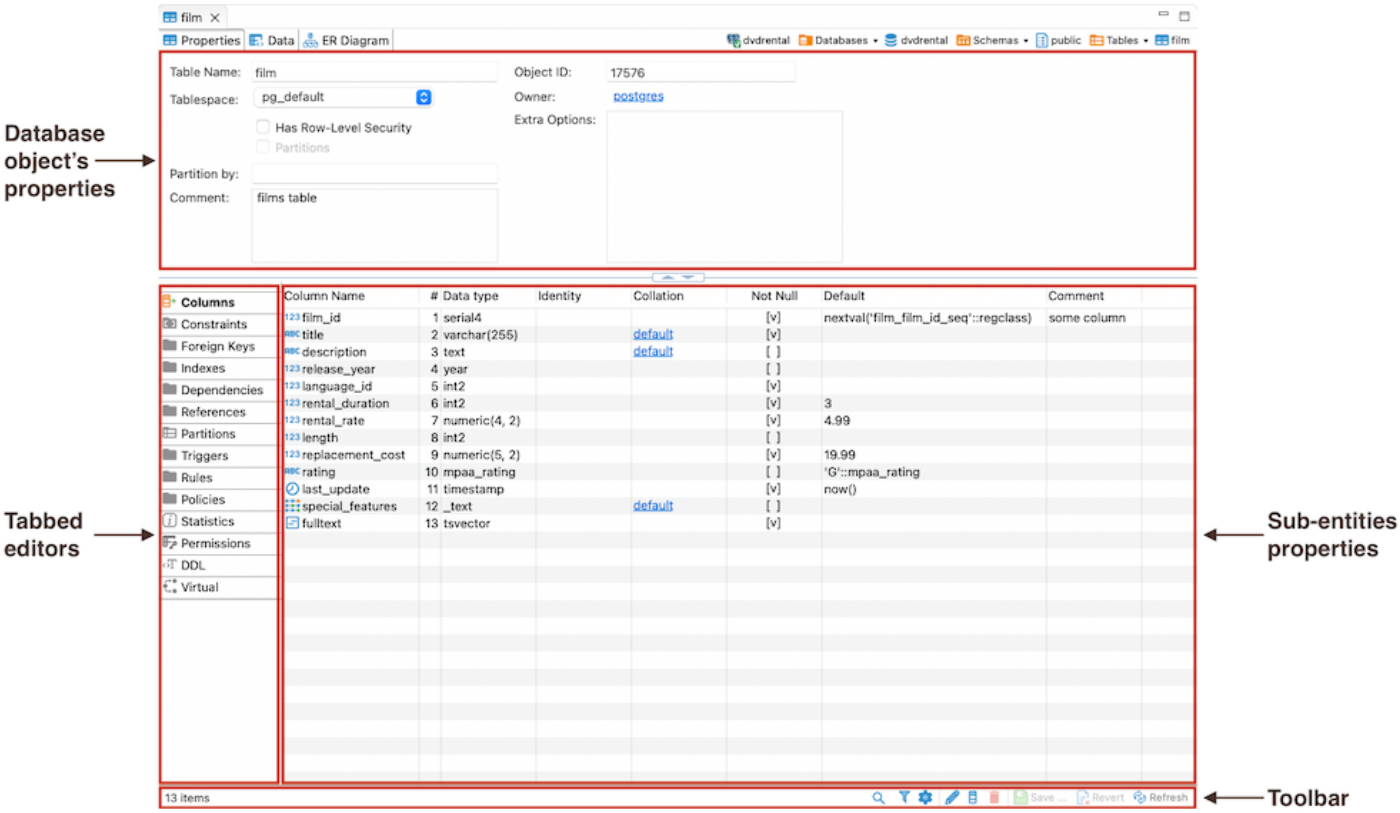
You can read more about XPath here: https://en.wikipedia.org/wiki/XPath


Grid
Text
Chart
xml_column
id
vcolumn
1
[XML]
2
Pankaj
2
[XML]
3
Lisa
3
[XML]
4
Tom
Add virtual column
Column Name:
vcolumn
Type Name:
varchar
Data Kind:
STRING
content.xml(xml_column, "/Employees/Employee
[@id='1']/name/text()")

Properties Editor

Overview

When you open the **Properties** tab of the [Database Object Editor], you'll encounter the following interface elements:



- **Database object's properties**: displays the properties of the current database object.
- **Tabbed editors**: These vary depending on your database. Depending on the database you're working with, you'll see different tabs reflecting that particular database's unique characteristics and requirements.
- **Sub-entities properties**: These contain the properties of the object's sub-entities. To access these properties, click the tabs in the **Tabbed Editors**.
- The **Toolbar** provides tools for managing these sub-entities.

Key Operations

To interact with the **Properties Editor** page, you can use Toolbar controls, execute commands from context menu associated with sub-entities, and utilize related keyboard shortcuts.

Toolbar controls

The toolbar at the bottom of the editor provides the following tools for the majority of sub-entities, except for some specific ones like SQL-based views (**DDL** and **Source**):

Button	Name	Description
	Search items	Displays a search field next to the button: - Type in the search combination - the content updates dynamically - To remove the filter, click the cross icon next to the search field.
	Filter settings	Opens the Filtering window, which allows setting a custom filter, see Configure Filters.
	Configure columns	Opens the Configure columns dialog box to select the columns to display or hide in the current view.
	Refresh the selected items	Depending on the database type, refreshes either the current item or its parent or the whole database object – reloading data from the database.
	View	Opens an editor/viewer for the item currently in focus.
	Create new [items]	Creates a new item of the same type as currently displayed in the open view, for example, a column.
	Delete database object	Deletes the item currently in focus.
	Save the current contents	- Same as the **Save** button on the main application toolbar - Same as `Ctrl+S` (or `#S` for MacOS) Opens the **Persist Changes** window that allows saving changes in the currently open sub-entity.
	Revert to the last saved state	Reverts all changes made to the database object to the previous saved state.

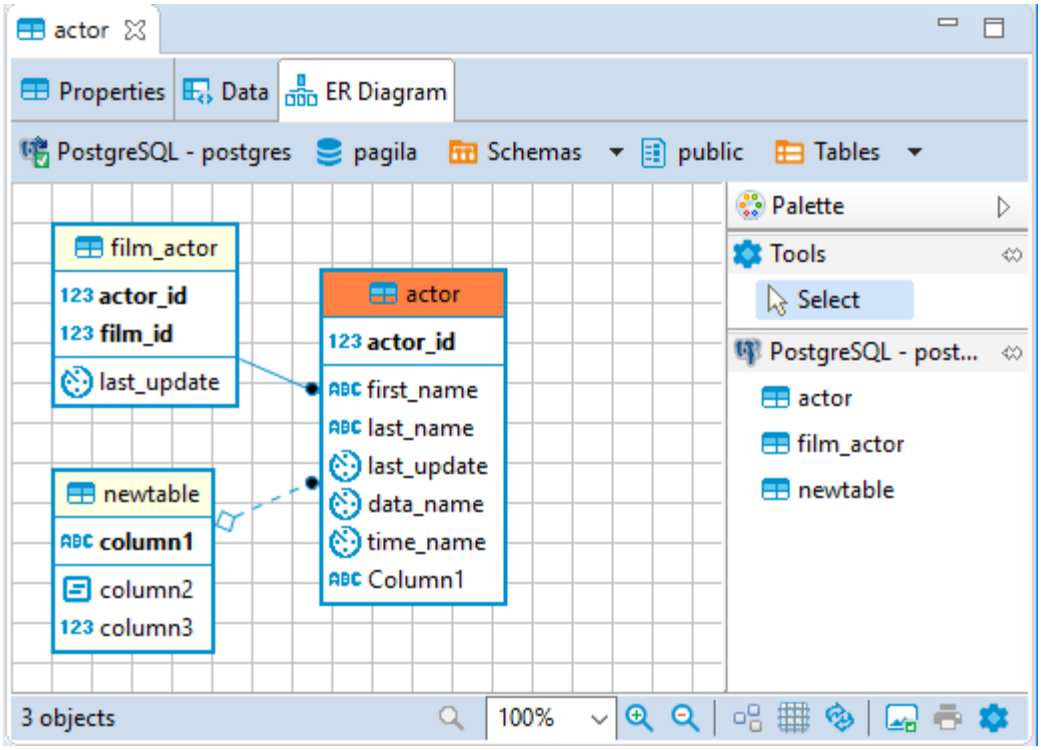
Context menu

The **Properties** of the sub-entities have context menus that offer the same commands as the Database Navigator.

To access these menus, right-click on the item.

ER Diagrams

ER diagrams appear on the rightmost tab of the Database Object Editor:



Entity Relation Diagrams (ERD) are graphic presentations of database entities and the relations between them. DBeaver allows you to view the diagrams of existing tables and whole database schemas, see Database Structure Diagrams. DB also allows the creation of custom diagrams, see Custom Diagrams.

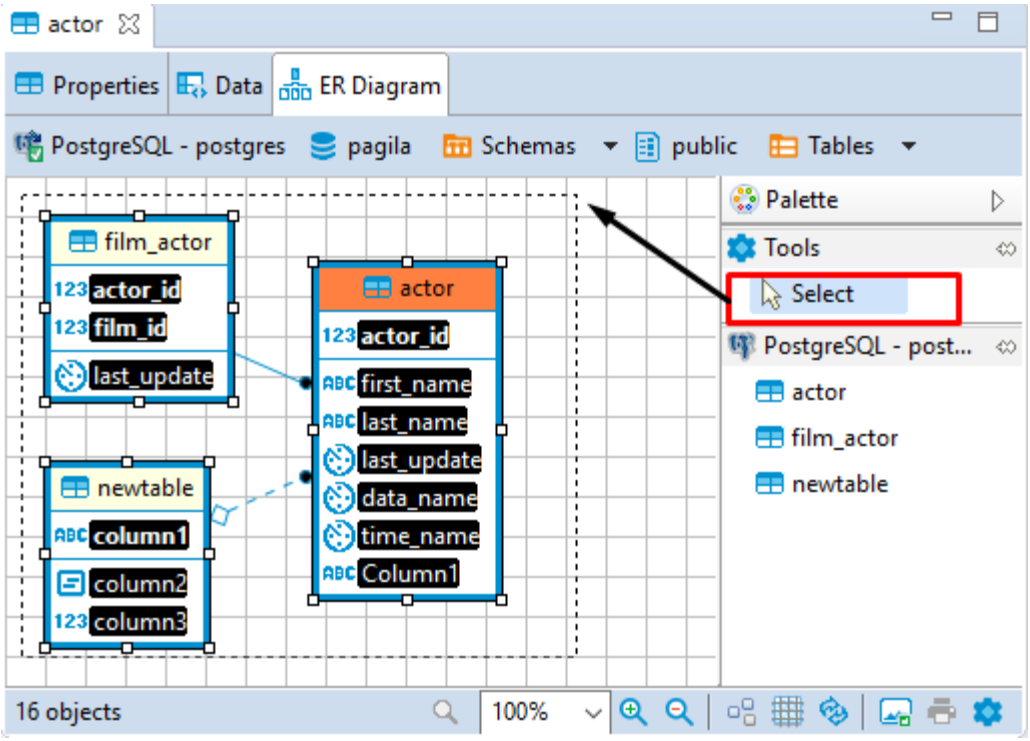
By default DBeaver uses IDEF1X notation.

Both types of diagrams provide the same tools to adjust their view and structure. They can be printed and exported to image file formats.

Selection of Elements in Diagrams

You can use one of the two tools to select elements in diagrams:

- Select – supports both, single and multi-select modes. To select a single element (table, connection, entity inside a table) in a diagram, just click that element. To select multiple elements, similar to using the Marquee tool, click outside the first element and draw until all elements you need are in focus:

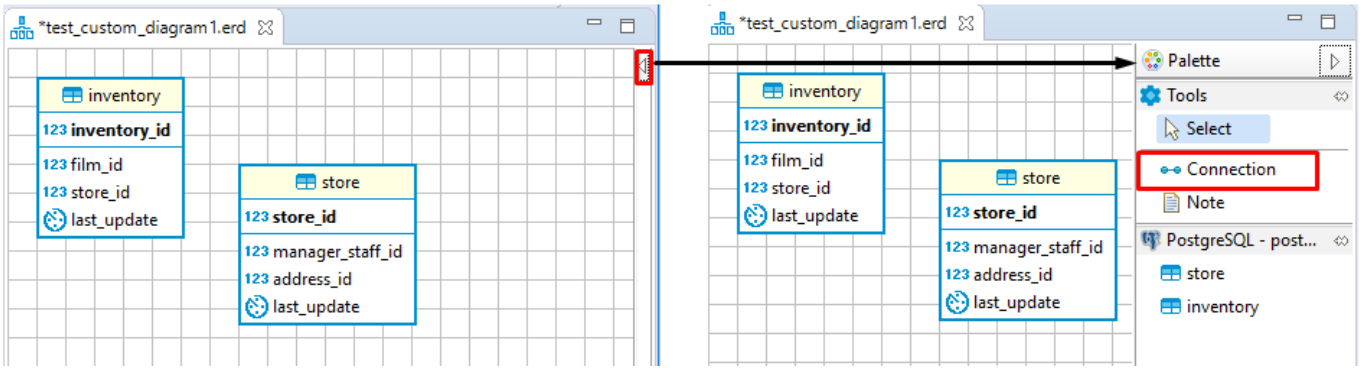


Structure Adjustment

NOTE: All changes to existing database schemas cannot be saved and are intended for exploration purposes only.

You can do the following structural changes in the diagrams.

- Add new tables to a diagram by drag-n-dropping them onto the diagram field from the Database Navigator.
- Rearrange tables in the diagram by dragging them all over the space. You can select several tables and drag them to a new location.
- Auto-arrange tables into a compact view after manual rearrangements: click the **Arrange Diagram** ( ) in the toolbar or on the context menu (right-click anywhere on the diagram tab).
- (Available for Custom Diagrams only) - connect tables with a connector: click the **Show Palette** button ( ) in the upper-left corner of the diagram tab and then, in the Palette panel, click **Connection**:



Now click the tables that you want to connect with each other in turn, one by one. To stop the connection line, double-click the last table

- (Available for Custom Diagrams only) - removes tables and connections: right-click the table or conection and click **Delete** on the context menu or just click the table or connection and press `Delete` .

View Adjustment

You can adjust the view of any diagram in the following ways:

- Enable/disable the diagram grid: Click **Toggle Grid** ( ) in the toolbar.
- Modify attributes visibility: Right-click the diagram and, on the context menu, click **Show Attributes** and then select one of the options:
  - **All** - all attributes
  - **Any keys** - primary and foreign keys
  - **Primary key** - only primary keys
  - **None** - no attributes
- Modify attributes presentation: Right-click the diagram and, on the context menu, click **View Styles** and then select one of the options:
  - **Show Icons**
  - **Show Data Types**
  - **Show Nullability**

- Show Comments
- Show Fully qualified names

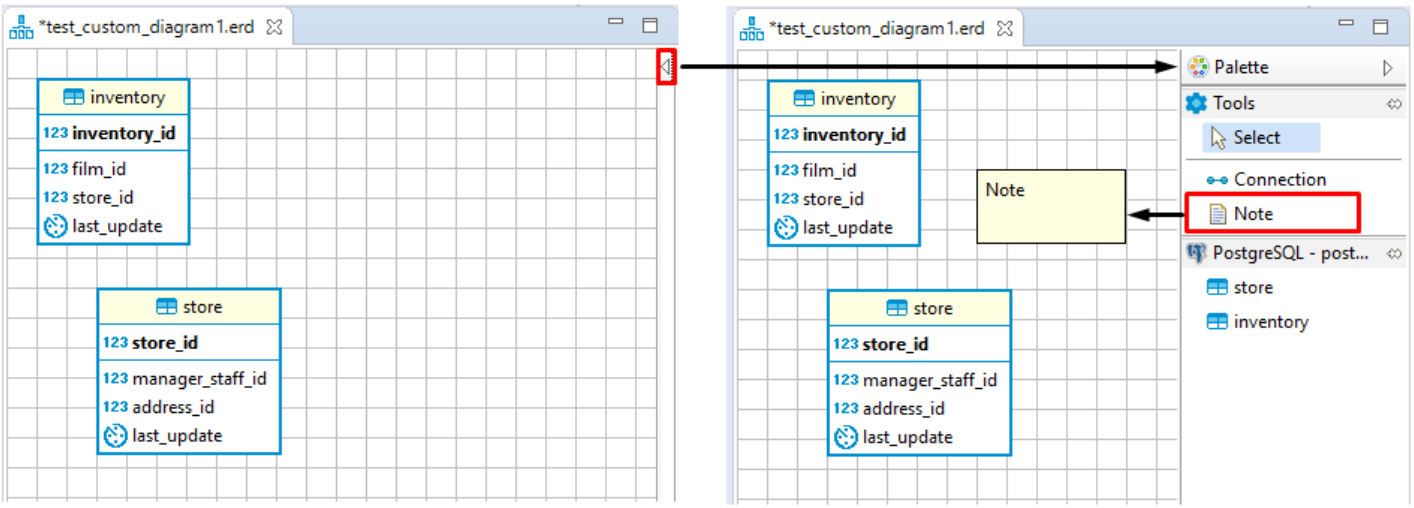
- Change the color of the entities/notes: Right-click the header of the entity or comment and then click **Set color** on the context menu. Then you can select the color and click **OK**.
- For elements located in front of/behind others, bring an element to the front or send it to the back: Right-click the element and then click **Bring to front** / **Send to back** on the context menu.
- Zoom the diagram in/out: Click the **Zoom In/Zoom Out** buttons or choose the scaling value in the dropdown list in the toolbar: 

Refresh

To see changes made by others to the database schema, you might need to refresh the diagram: click **Refresh Diagram** () in the toolbar.

Notes

You can create notes only in Custom Diagrams. To create a note, click the **Show Palette** button () in the upper-left corner of the diagram tab. Then, in the Palette panel, click **Note** and click anywhere in the diagram tab. Now you can double-click the **Note** box to enter the note text:



Search in Diagram Entities

To search among entities of a diagram, click the **Search items** button () in the toolbar, then type in the search combination. The entities that contain the search combination are highlighted in the diagram. To remove the filter, click the cross icon next to the search field.

Bindings

Navigation and selection:

- Use `ARROWS` to navigate between tables.
- Press `SHIFT` | `#` + `ARROWS` to select multiple tables.
- Press `CTRL` | `#` + `ARROWS` to select additional tables using `SPACEBAR`.
- Press `SPACEBAR` to select the current table.

Table Manipulation:

- Press `.` (period) to change the mode to move/resize tables, then use `ARROWS` and `ENTER` | `#` to move/resize tables.

Note: To use this feature on macOS, you may need to enable accessibility permissions for DBeaver. This can be done in `System Preferences -> Security & Privacy -> Accessibility`.

Focus:

- Press `ENTER` | `#` to focus on attributes in the table.
- Press `BACKSPACE` | `#` to leave the focus.
- Use `|`, `?`, `\` to focus on associations.

- Press `ALT/FN + 1` | `#/FN + 1` to focus on the diagram.
- Press `ALT/FN + 2` | `#/FN + 2` to focus on the palette.
- Press `ALT/FN + 3` | `#/FN + 3` to focus/open the outline.
- Press `ALT/FN + 4` | `#/FN + 4` to focus on the parameter view.

Other Functions:

- Press `CTRL + SHIFT + ENTER` | `# + SHIFT + #` to open the selected table diagram.

Diagram Export

You can export (save) a diagram as an image (PNG, GIF, BMP formats) or as a file in GraphML format. To export a diagram, click **Save diagram in external format** () in the toolbar.

Diagram Printing

To print a diagram, press `CTRL+P` or click **Print Diagram** () on the toolbar.

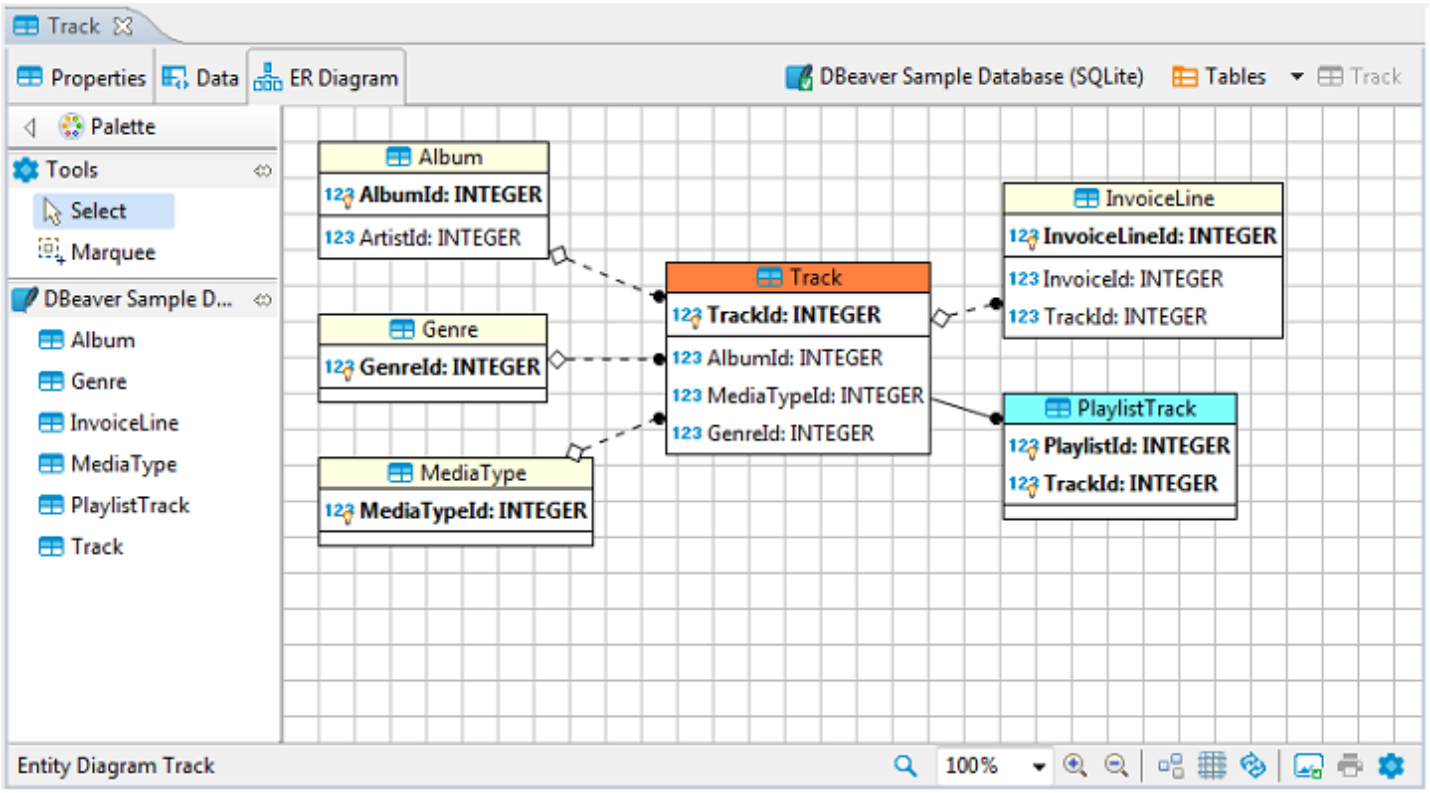
Settings

To modify the diagram settings, click **Configuration** () on the toolbar.

Database Structure Diagrams

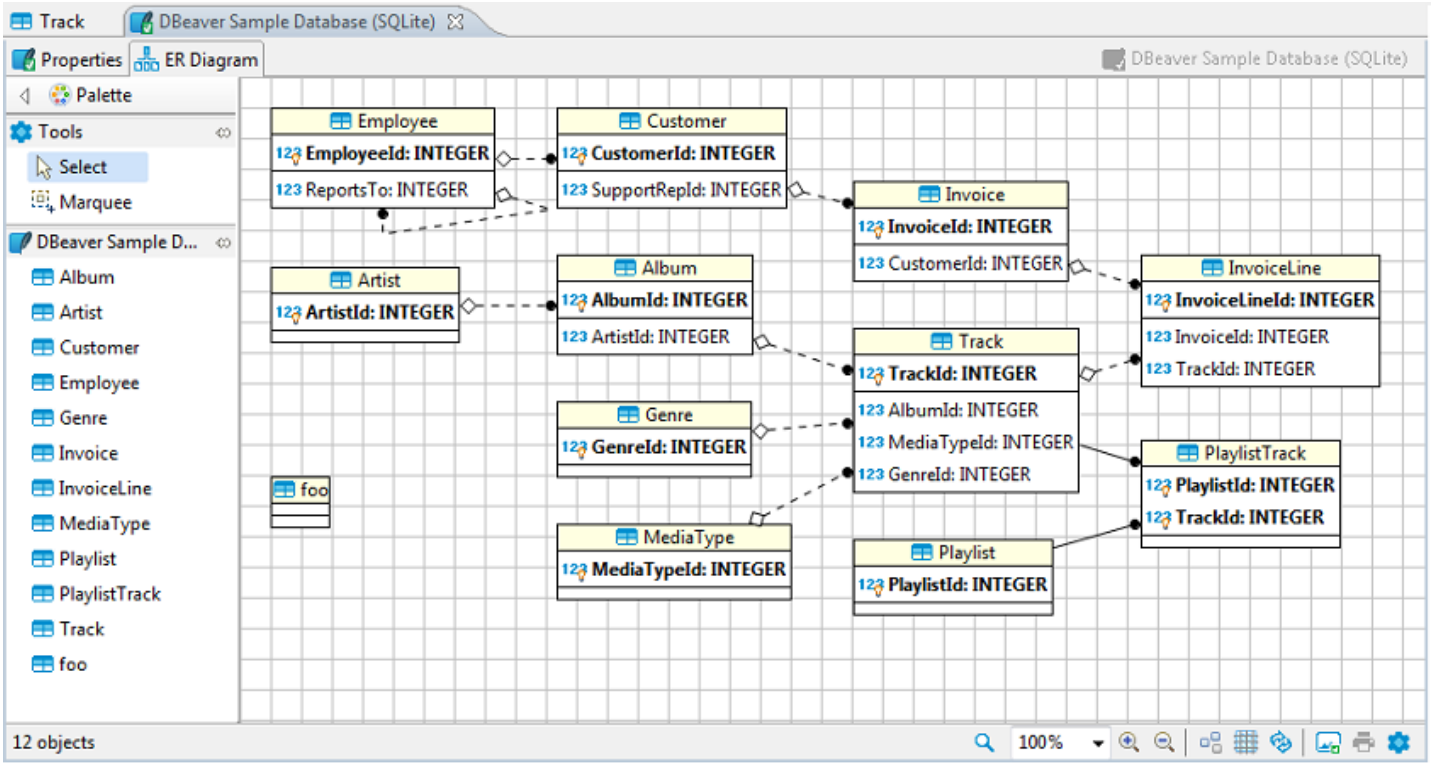
You can view a database structure in the standard ERD (Entity Relation Diagram) form. ER diagrams are available for all tables and schemas (databases).

The ER diagram for a table shows the table itself and its relations with other tables inside the schema. To view the ER diagram for a table or view, double-click the table or view in the Database Navigator and then, in the Database Object Editor, switch to the **ER Diagram** tab:



To view the ER diagram for a full database schema, double-click the schema name in the Database Navigator or the previous node in the path (usually - **Tables**):

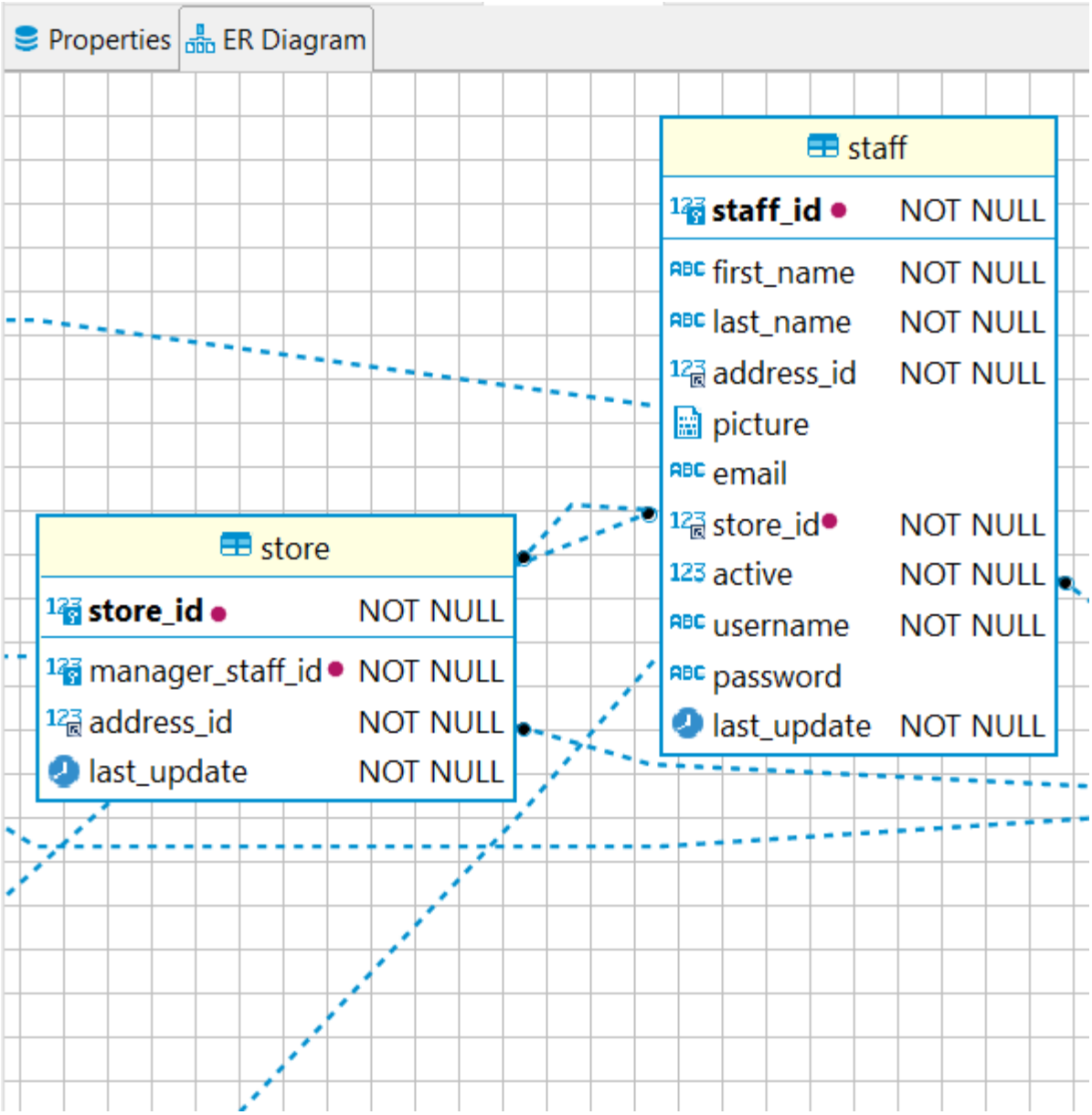




NOTE: Table and schema diagrams are read-only. You can rearrange the layout, drag-n-drop elements inside a diagram but you cannot save the changes state or delete/add anything. This is because the diagrams represent the actual state of databases.

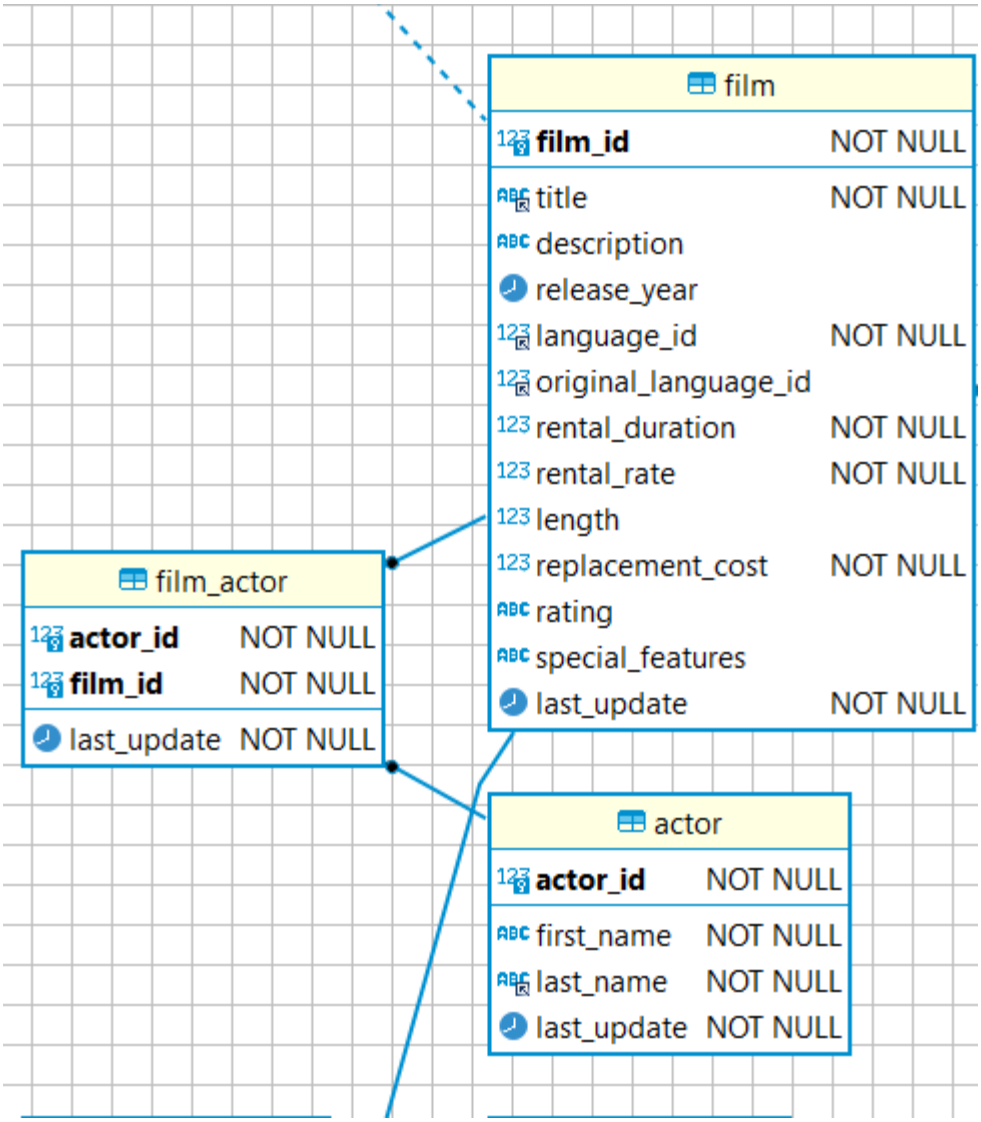
Relationship Notation

Lines representing the relationship between tables can look different depending on the nature of the relationship. Please note that any line can have only one beginning and one end. We may have links between two tables not only for one pair of attributes, which means that several associations will be displayed:

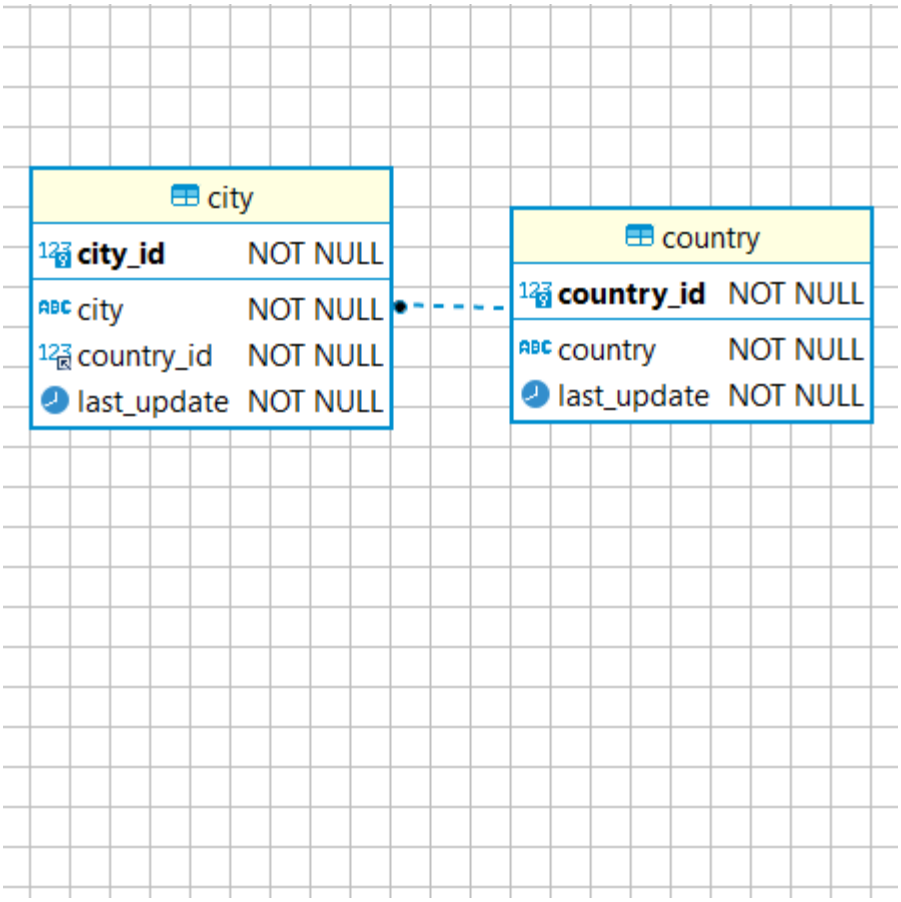


Notation	Description
	A solid line means that the foreign key column in one table is a primary key in another table
	A dashed line means that the foreign key column is not a primary key but a regular column in another table
	A black dot is used as the start of the line
	A diamond is used at the end of the link when the column in the source table is optional (there is no NOT NULL)

A one-to-one relationship is always a solid line due to the unique primary key - foreign key relationship:



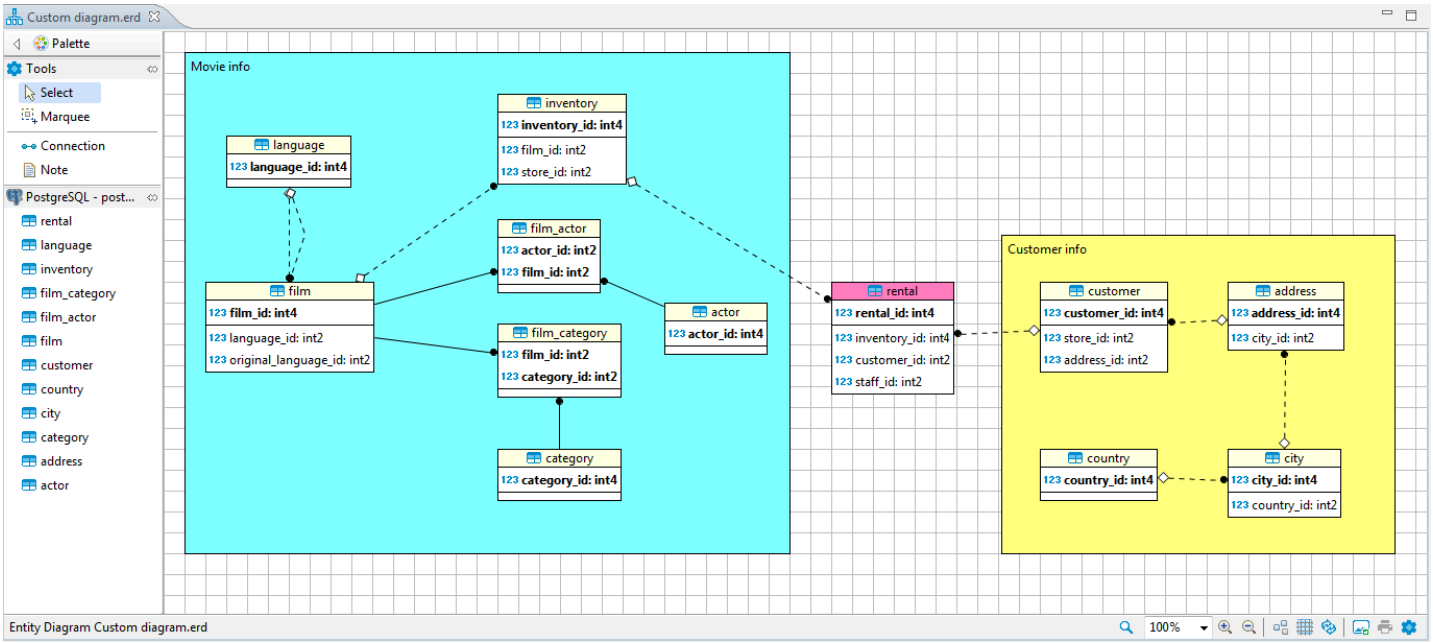
A dashed line is for the one-to-many relationship:


city
city_id NOT NULL
city NOT NULL
country_id NOT NULL
last_update NOT NULL
country
country_id NOT NULL
country NOT NULL
last_update NOT NULL

Custom Diagrams

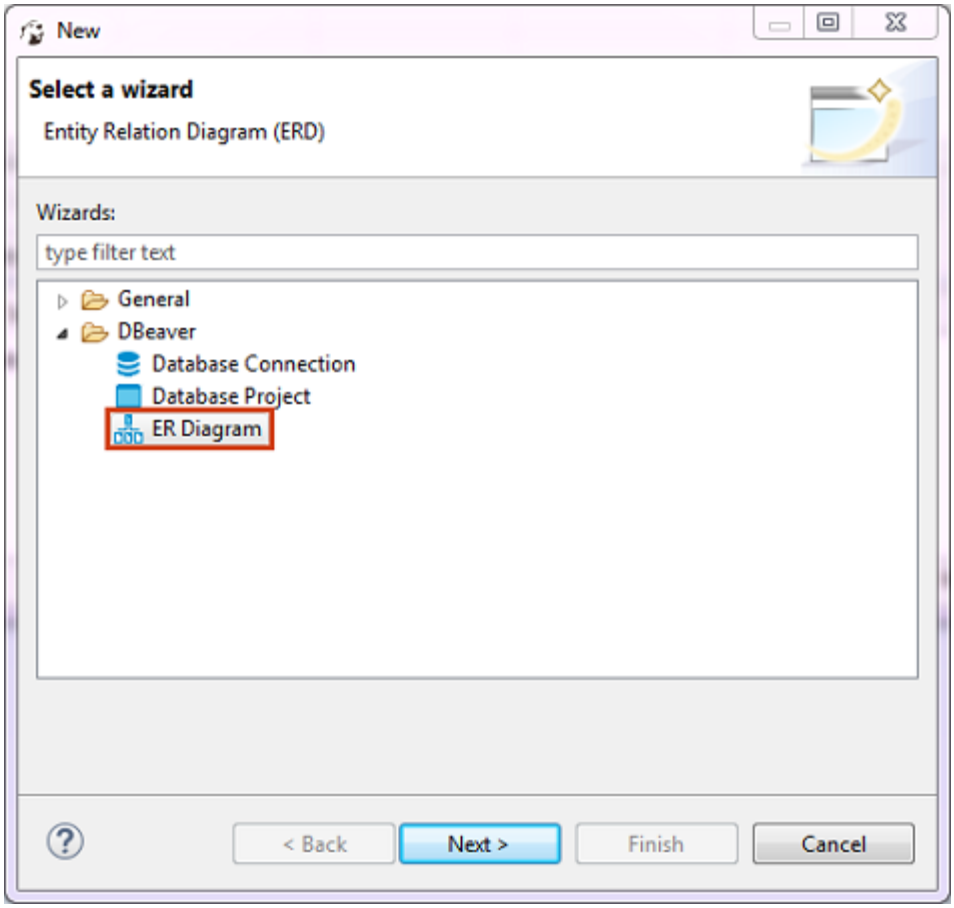
You can create custom ER diagrams that can contain any tables, relations and notes.

However, even custom diagrams may contain only real existing database entities (tables).

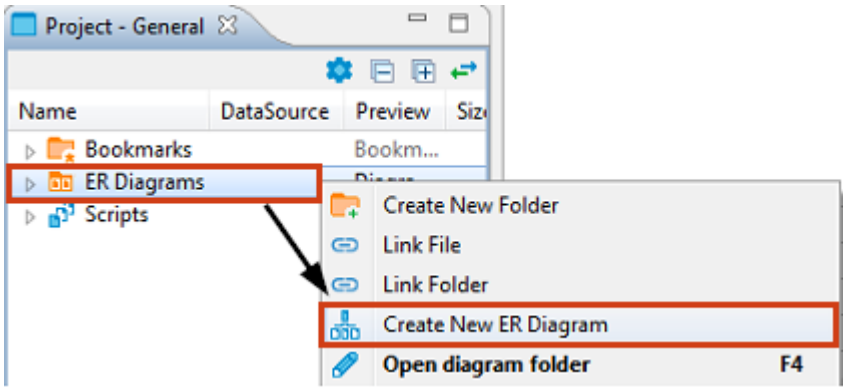


You can create a custom diagram in one of the ways:

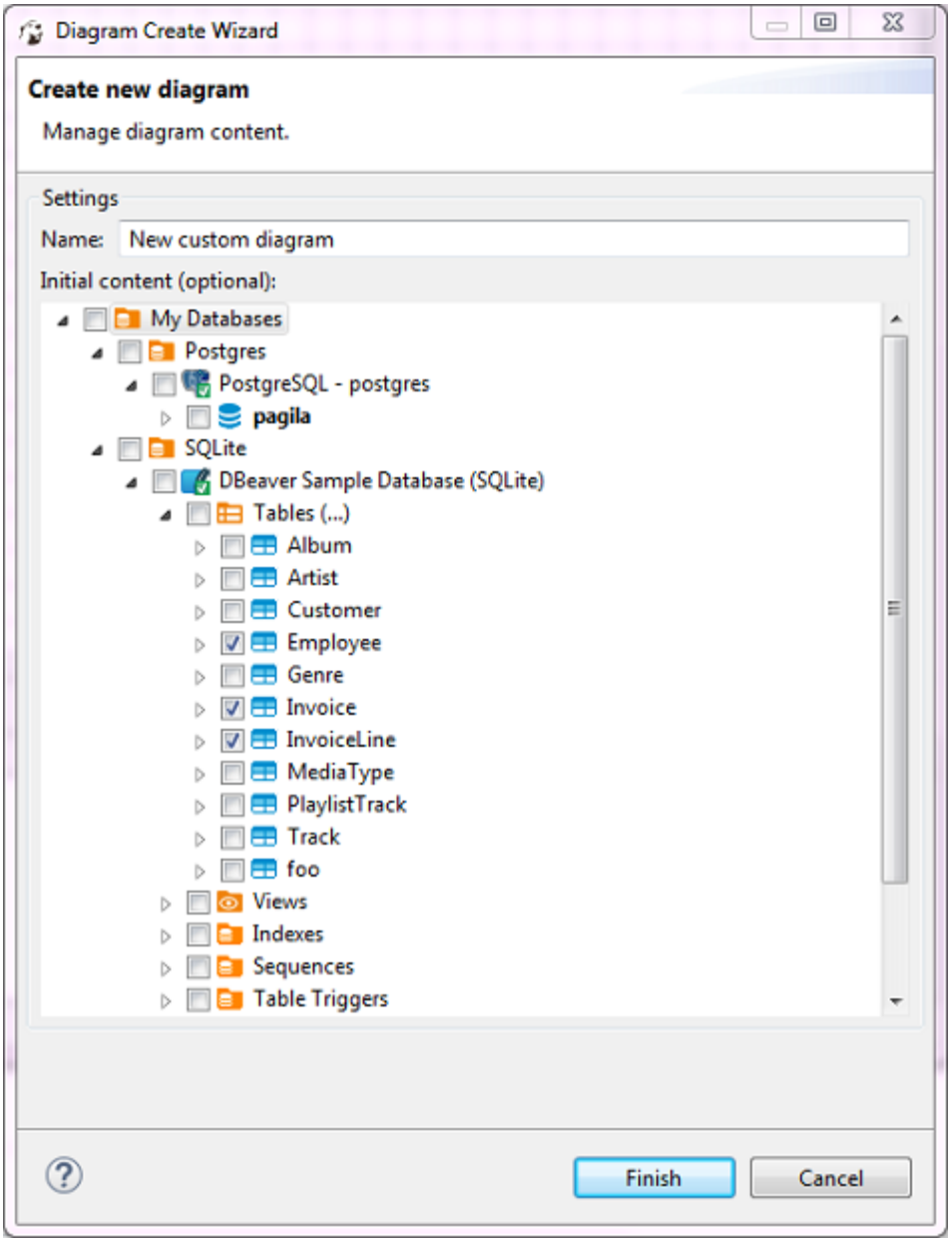
1. On the DBeaver main menu, click **File -> New**. Then in the new diagram wizard, click **DBeaver -> ER Diagram**, and then **Next**:



2. In the Project Explorer view, right-click the **ER Diagrams** node and then click **Create New ER Diagram** on the context menu.



In both cases, in the Diagram Create Wizard, specify the diagram name and (optional) choose the initial diagram contents (set of tables):



The new diagram appears in a separate editor. Now you can drag-n-drop any number of tables into it. You can add tables from different connections as well as from different database types (for example, combine Oracle and MySQL tables in one and the same diagram).

You can also add notes and custom relations (associations) using the ERD palette on the left side of diagram tab - see details in the ER Diagrams article. For example, to create a diagram similar to the one shown at the beginning of this article, you need to:

1. Add required tables and relationships between them and move them around to create a well-shaped structure (see *Structure Adjustment* section of the ER Diagrams article).
2. Add notes (see the *Notes* section of the ER Diagrams article).

3. Stretch the notes to cover the intended tables, then send the notes to the back, and then set a color to the tables and notes (see the *View Adjustment* section of the ER Diagrams article).

Undo/redo functions are fully supported in diagram editing.

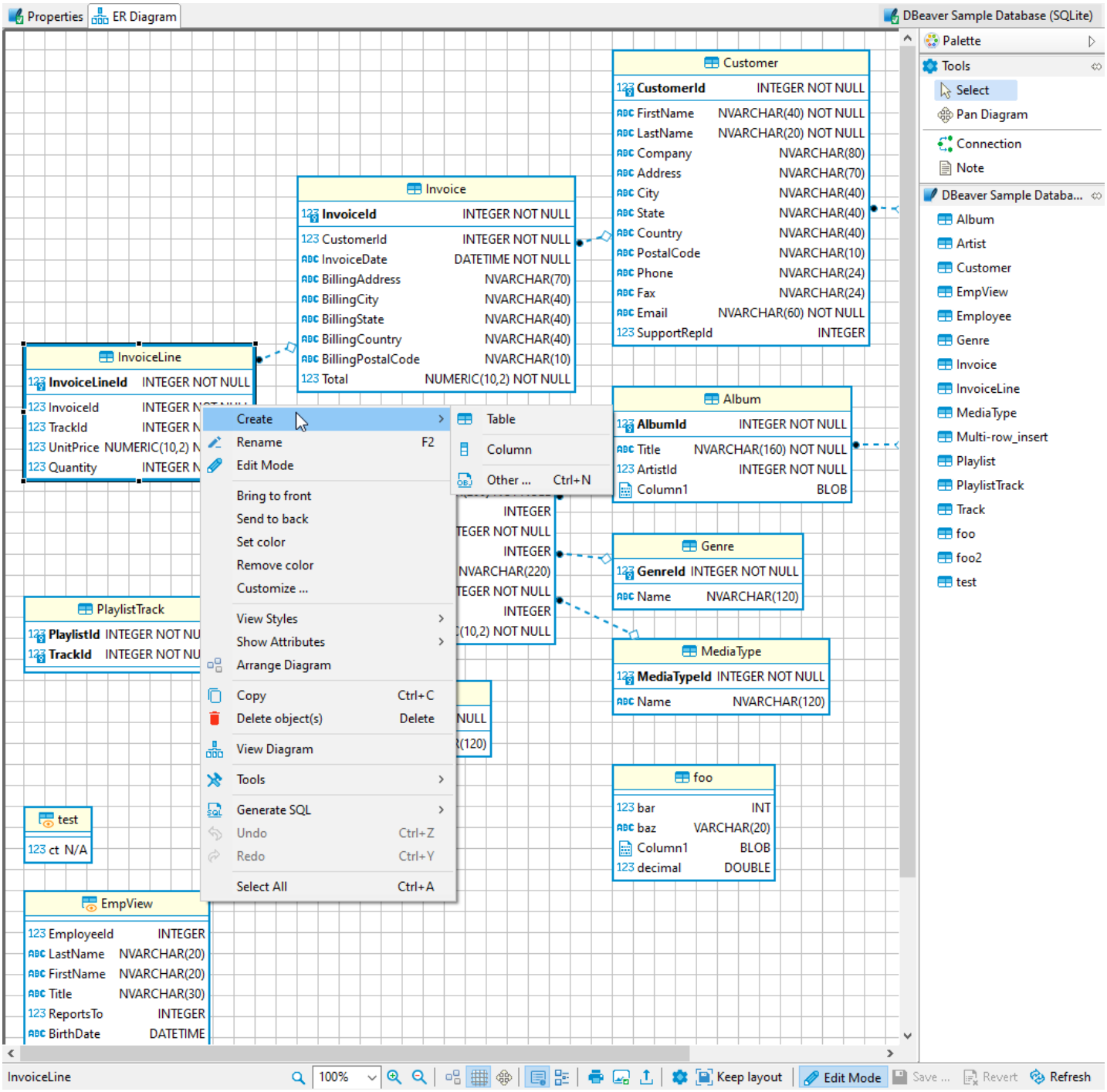
Edit mode

Note: This feature is available in Enterprise, Ultimate and Team editions only.

Edit mode for ER Diagrams is a special feature that lets you create database objects while using the visual presentation of ER Diagrams. It can be accessed on the ERD tab of any database object that supports it. Changes made to a diagram in edit mode will generate an SQL script that can be executed to persist all the changes made. You can enable Edit mode by either choosing it from the context menu or by clicking a button on the bottom toolbar.

Working in Edit mode

To create an object you need to right-click on the diagram and select an object that you want to create. Foreign keys are created similarly to virtual foreign keys in custom diagrams. Just drag a column from a table for which the foreign key is being added and drop it in the referenced table. This will open a window that lets you choose a unique key in the referenced column.



Saving changes to a database is performed by clicking a save button. This will show you a preview of a generated script that you then can execute or cancel. The revert button will cancel all the changes that you did to a diagram after the last save.

SQL Editor

You can create multiple SQL scripts for a single connection. Every script opens in its own SQL editor. To open an SQL editor for some connection:

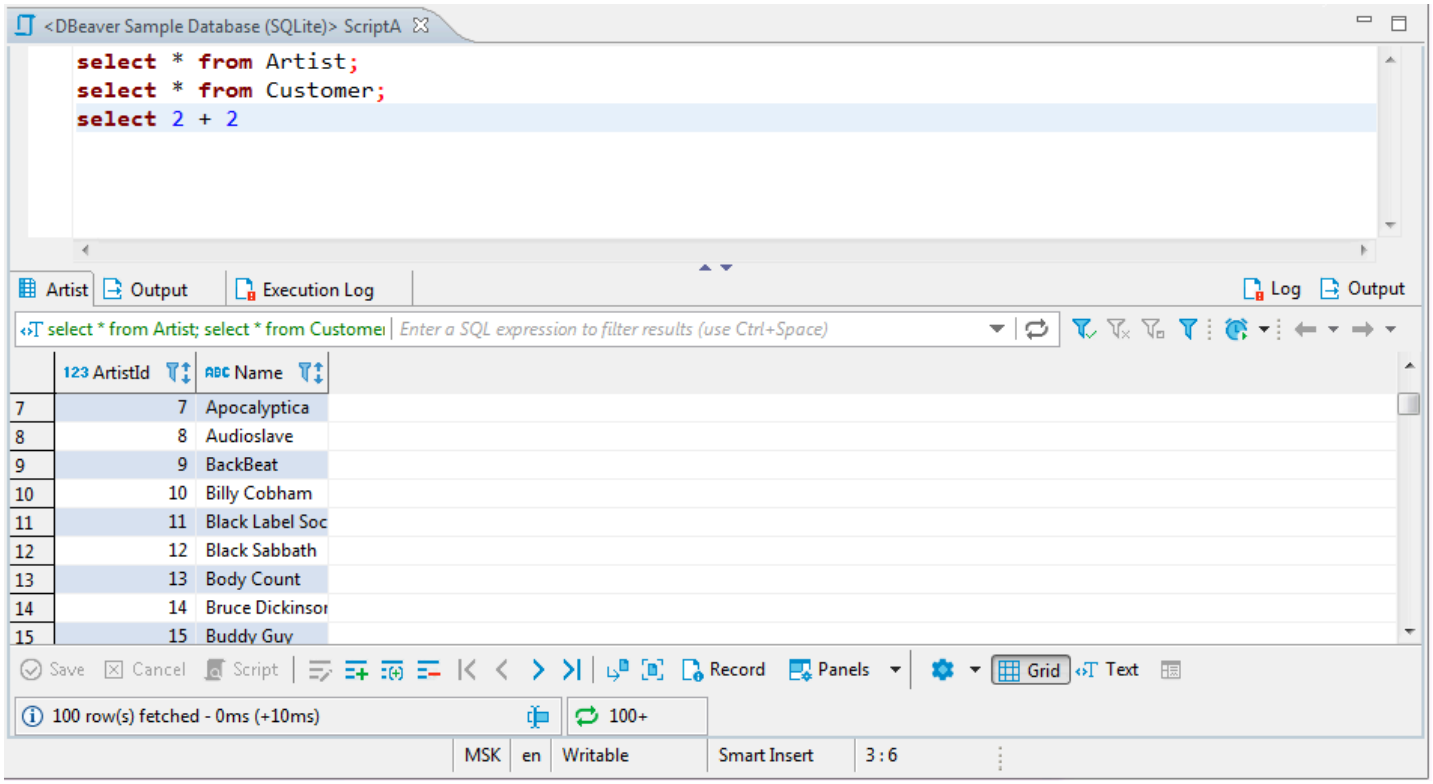
- Click this connection in the Database Navigator view and press `F3` or click **SQL Editor -> SQL Editor** on the main menu. Alternatively, click **SQL Editor** on the context menu of this connection. DBeaver opens the Choose SQL script editor with saved SQL scripts linked to this connection. Click the SQL script to open it in a separate tab.
- Click **Recent SQL Editor** on the context menu for this connection or on the main menu (**SQL Editor -> Recent SQL Editor**). This opens the latest used SQL editor. You can also open the most recent SQL editor using `Ctrl+` shortcut in the Database Navigator view.
- If you need to create a new SQL script, on the main menu, click **SQL Editor -> New SQL Editor** or press `F3` and then click **New Script** in the Choose SQL Script window.

DBeaver uses SQL syntax highlighting which depends on the database associated with the script. Different databases have different sets of reserved keywords and system functions.

NOTE: SQL Editor for a connection is different from SQL console for a table or view (right-click the table and click **Read data in SQL console**). Unlike the console, it can save scripts and changes made to them.

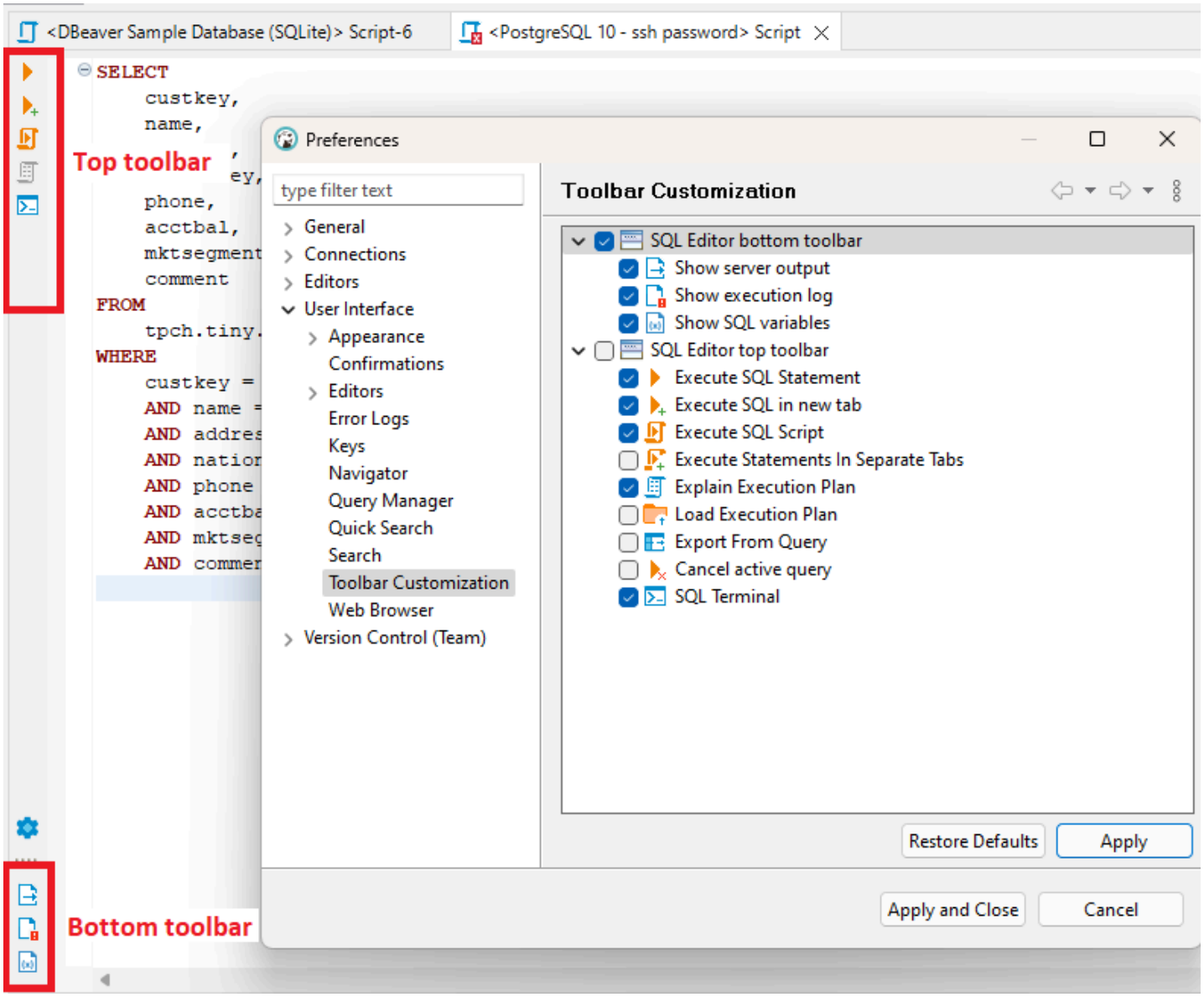
You can see all your saved SQL scripts in the Project Explorer view in the **Scripts** folder.

The SQL editor includes the script panel at the top and the results panel at the bottom:



You can open the SQL editor preferences by pressing `Alt+Enter` .

Also SQL Editor has toolbar which is customizable. See Toolbar Customization.

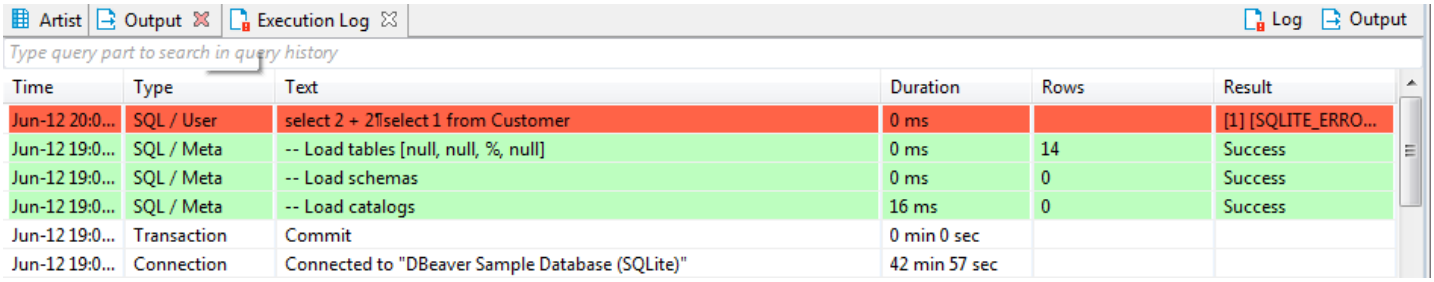


Results Panel

The results panel displays tabs with results in various formats. The tabs resulting from script execution represent instances of the Data Editor. You can create, edit and execute SQL scripts in the script panel and then see the results in the result tabs.

The results panel provides **Output** and **Log** views of results.

The execution **Log** tab contains all queries executed in the current SQL editor:



The **Output** tab contains all server-side database messages/warnings generated by a database when you execute queries. This feature is supported only by a few database engines (Oracle, SQL Server and some other ones).

Layout Adjustment

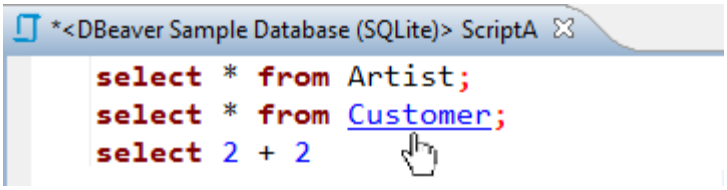
You can modify the layout of the SQL Editor by showing/hiding the results panel and changing the horizontal /vertical position of the panes.

- To toggle (hide/show) the results panel, press `CTRL+6` or right-click anywhere in the script pane and, on the context menu, click **Layout -> Toggle results panel**.
- To maximize the results panel, press `CTRL+Shift+6`, or double-click the results tab name, or right-click anywhere in the script panel and, on the context menu, click **Layout -> Maximize results panel**.
- To switch between the script panel and the results pane, press `Alt+6` or right-click anywhere in the script panel and, on the context menu, click **Layout -> Switch active panel**.

To position both panels horizontally, right-click anywhere in the script panel and, on the context menu, click **Layout -> Horizontal**.
To position both panels vertically, right-click anywhere in the script panel and, on the context menu, click **Layout -> Vertical**.

Hyperlinks

You can press and hold `Ctrl` and at the same time move the mouse over the SQL text. If DBeaver recognizes some identifier as a table/view name, it presents it as a hyperlink. You can click the hyperlink to open this object`s editor:



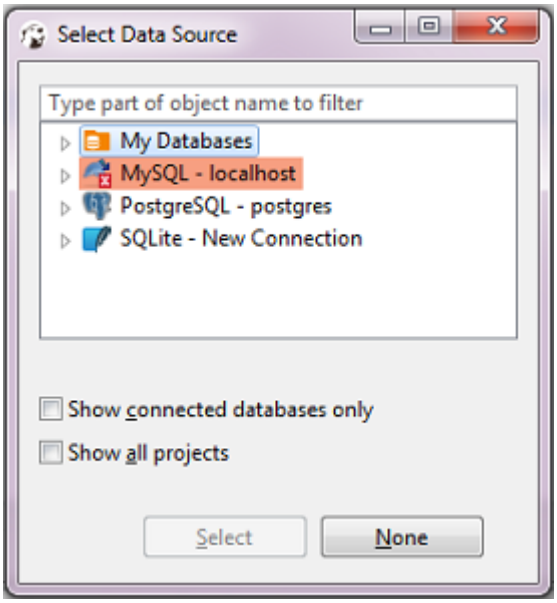
Active Database/Schema Selection

You can change the connection associated with the current SQL editor or change the active database/schema, at the same time retaining the SQL text.

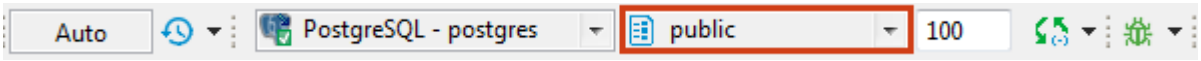
To change the connection, press `Ctrl+9` or click the **Active datasource** box on DBeaver`s main toolbar:



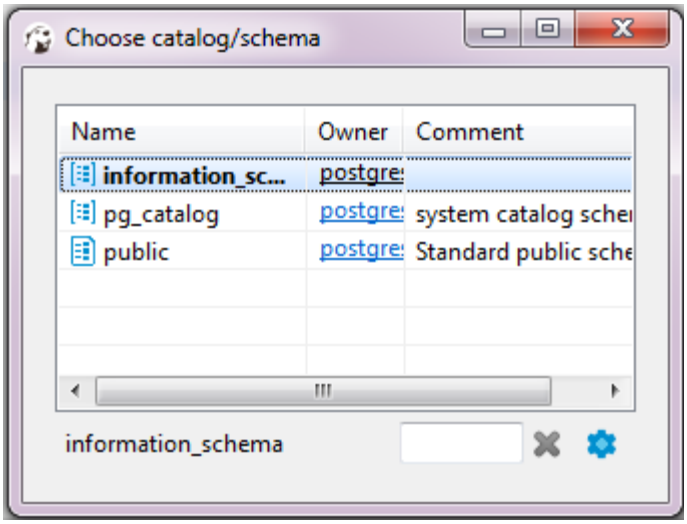
The Select Data Source dialog box opens. In the tree of connections, click the required connection and then click **Select**. To disassociate the SQL Editor with any connection, click **None**:



To change the active schema, press `Ctrl+0` or click the **Active Catalog/Schema** box in DBeaver`s main toolbar:



The Choose catalog/schema dialog box opens. In the list of schemas, double-click the required schema:



If there are many schemas and they do not fit in the dialog box use the search field to find the schema you need:

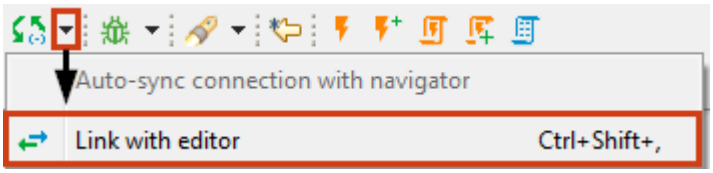


To configure the set of columns to be visible for each schema in the dialog box, click the **Configure columns** button ().

You can easily associate the SQL Editor with the connection that is currently in focus in the Database Navigator (the focus can be on any object of the connection - a table, a folder, etc.) - click the **Set connection from navigator** button on DBeaver`s main toolbar:



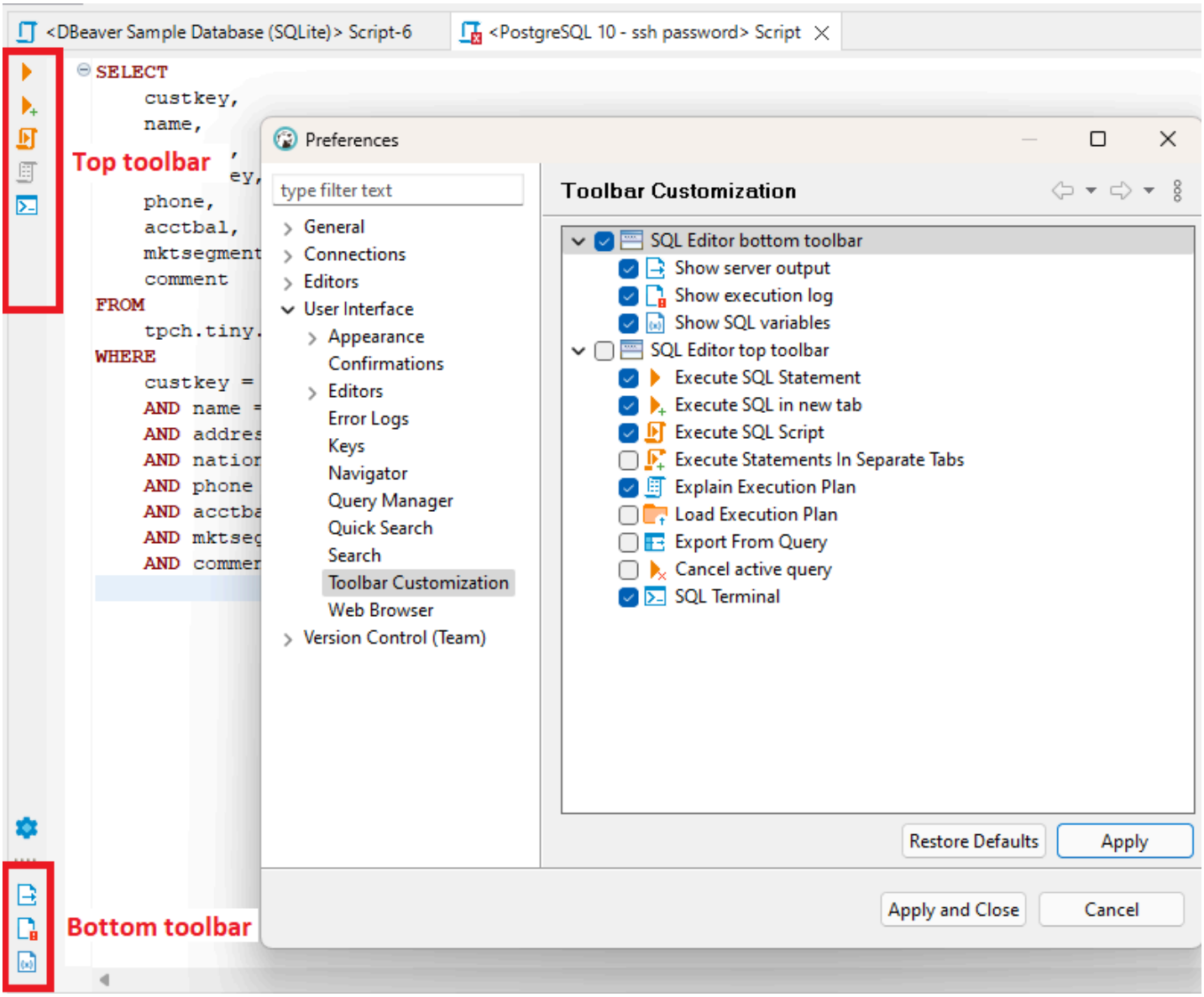
The reverse action is also possible: you can set the focus of the Database Navigator to the active connection of the SQL Editor - press `Ctrl+Shift+,` or click the arrow next to the **Set connection from navigator** button in DBeaver`s main toolbar and then click **Link with editor**:



Toolbar Customization

You can customize your SQL Editor toolbar to bring more commands with

`Preferences->User Interface->Toolbar Customization`.

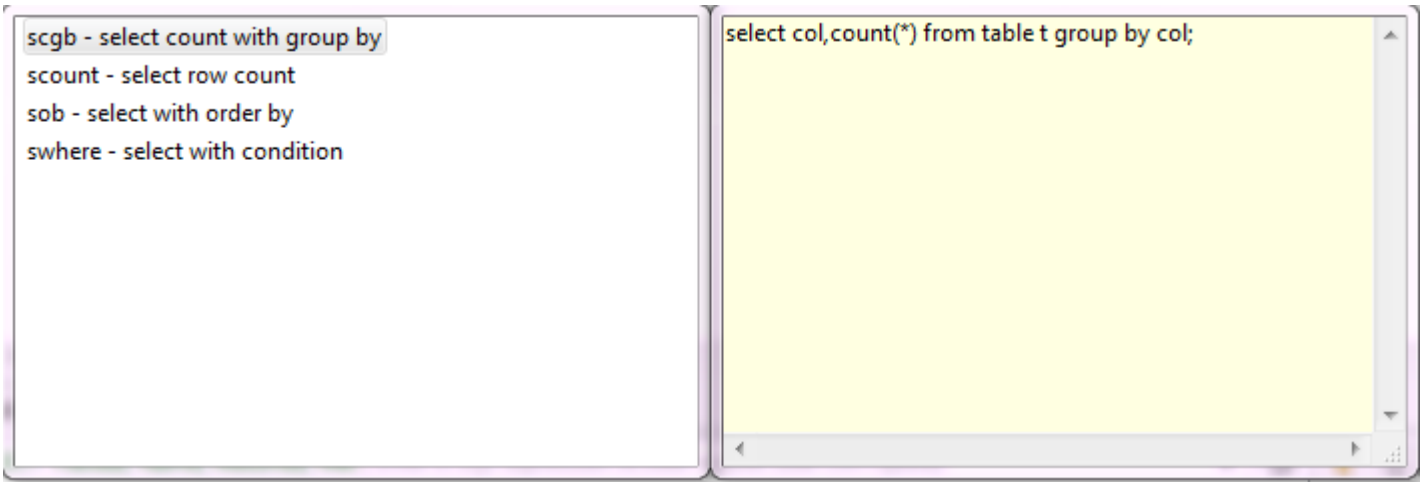


By default, only Execute SQL Statement, Execute SQL Script and Explain Execution plan are visible.

SQL Templates

Templates allow you to insert frequently used SQL statements into an SQL script.

To see available templates, press `Ctrl+Alt+SPACE` or right-click the line in the script pane and click **SQL Template** on the context menu. A box with a list of available templates appears:



To apply a template, in the SQL Editor, in the script pane:

- Type the template name and press `Tab`
- Right-click the line where you want to insert a template expression, click **SQL Template** on the context menu, and then, in the list of templates, double-click the required template name.

  The template SQL statement appears in the script.

To edit/add/remove templates, click Configure Icon in the bottom toolbar, then click **Preferences -> SQL Editor -> Templates**. For more information about managing templates, please visit Eclipse Website.

Standard Eclipse templates:

Variable	Description
${cursor}	Specifies the cursor position when the template edit mode is left. This is useful when the cursor should jump to a different place than to the end of the template upon leaving the template edit mode.
${year}	Takes the current year value
${date}	Takes the current date value
${time}	Takes the current time value

${dollar}	Takes the dollar sign $. Alternatively, two dollar signs can be used: $$.
${user}	Takes the user name
${word_selection}	Takes the content of the current text selection
${line_selection}	Takes content of all currently selected lines

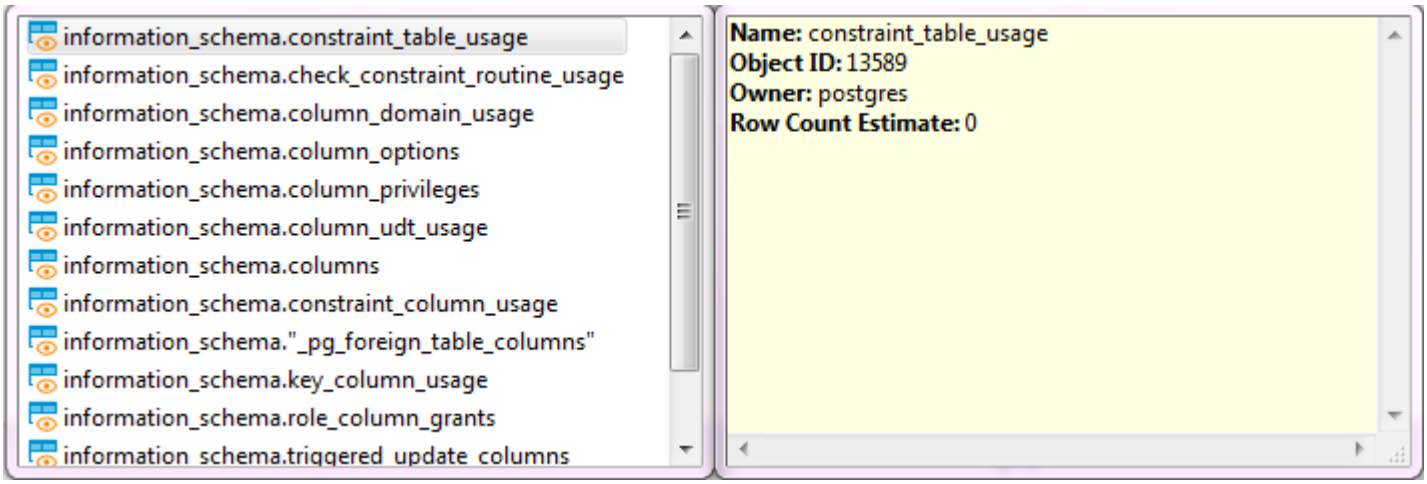
DBeaver-specific templates:

Variable	Description
${schema}	Takes the current schema name
${catalog}	Takes the catalog name
${table}	Takes the current table name (from the active catalog/schema)
${column}	Takes the column name (from the current table)

SQL Assist and Auto-Complete

The SQL Assist feature provides auto-completion of database object names and SQL commands, and other keywords in queries.

To perform some object name auto-complete, press `Ctrl+Space` or right-click the desired location in the query and click **SQL Assist** on the context menu. DBeaver searches for objects in a database by their names and/or descriptions.



When you start typing an SQL keyword in a statement, DBeaver offers auto-complete options as well.

Another auto-complete function is that it searches for the last entered identifier - type the first letter and press `Ctr`

.

You can also press `Ctrl+Space` after the asterisk in the query similar to `SELECT * FROM tableName` or similar to `INSERT INTO tableName (*)` (brackets are important) (you can use ()[]{} brackets) - the asterisk will be replaced with a list of all the table columns.

Hippie Engine

Hippie Engine provides autocompletion based on information from the current script file. It scans files looking for words and adds proposals based on similarity with the given string.

It can be toggled on or off in the Code Completion preferences.

Customize

You can customize your SQL Assist and Auto Complete by navigating to **Window** -> **Preferences** -> **Editor** -> **SQL Editor** -> **Code Completion**, optimizing the performance of the auto-completion feature.

AI SQL Assistance (ChatGPT)

In DBeaver, you can build working SQL queries using human language thanks to integration with OpenAI (ChatGPT, to be more precise - the GPT-3 language model).

Note:

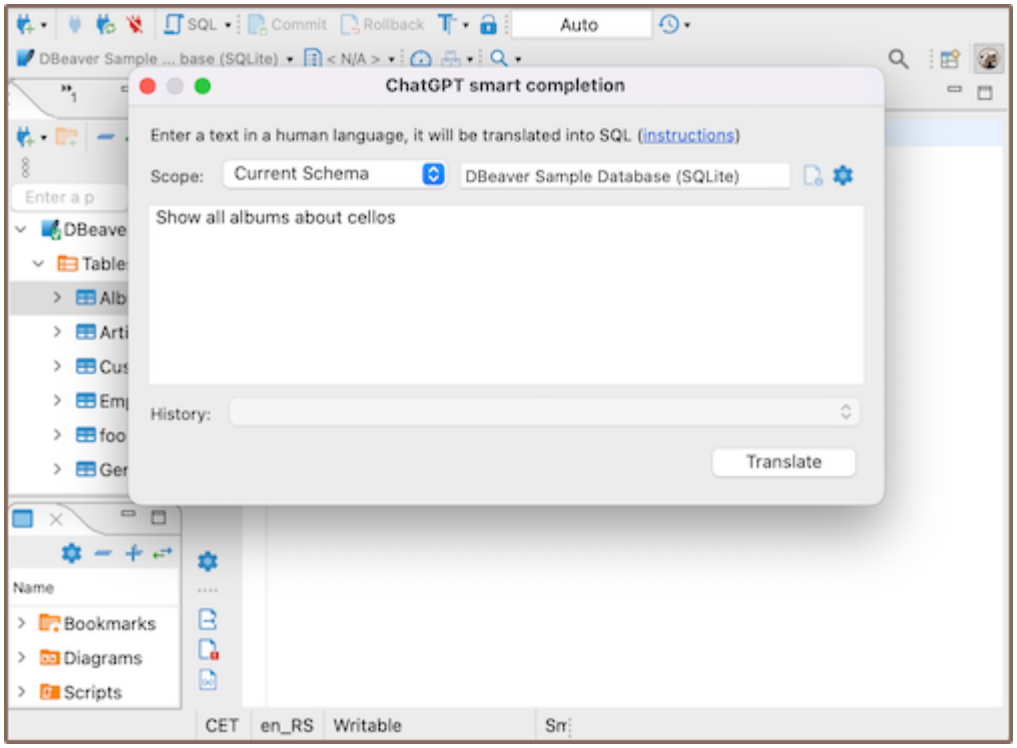
- **DBeaver Corp has nothing to do with OpenAI company. We use public API to integrate with the GPT completion engine.**
- **To use this feature, you need to register on the OpenAI platform and receive a secret key.**
- **To use this feature in the DBeaver Community version, you need to install the GPT extension.**
- **This feature isn't available in the Lite version.**

How it works | Get started | Write questions | Configure | Disable | Receive API key | Install GPT extension | Data privacy

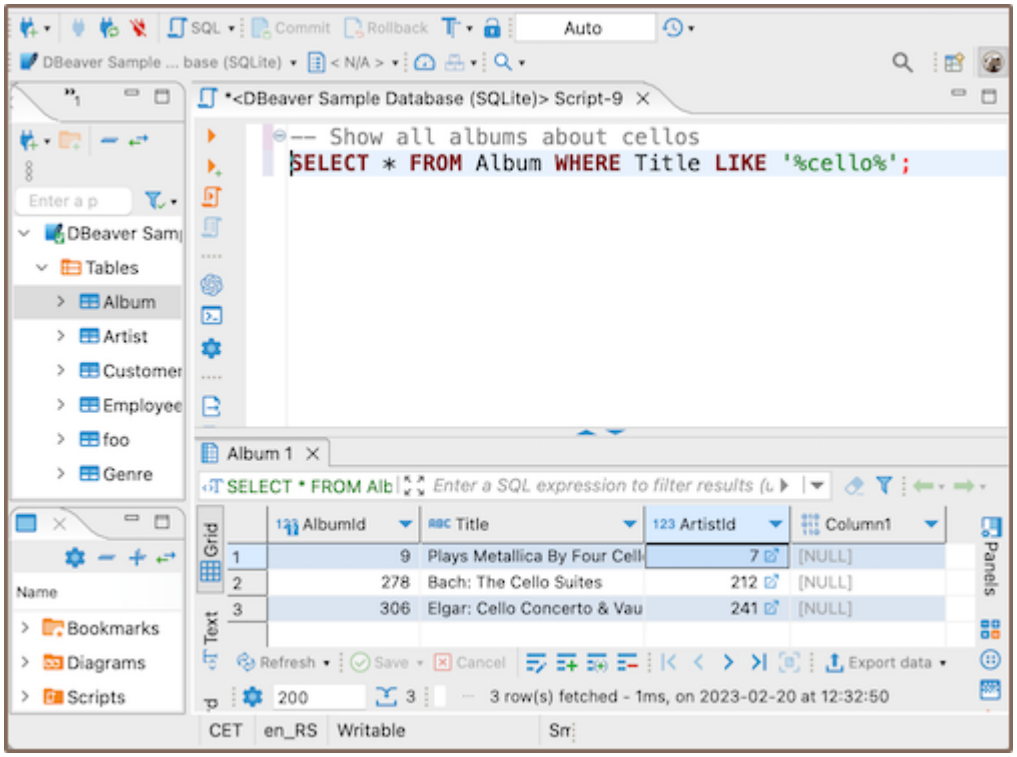
How it works

You write what you want to get from the database in the **ChatGPT smart completion** window, and DBeaver translates your phrase into the correct SELECT query.

Enter human language request:



Generated SQL query:



Note: DBeaver must send the database metadata to the OpenAI platform to translate a phrase into a query.

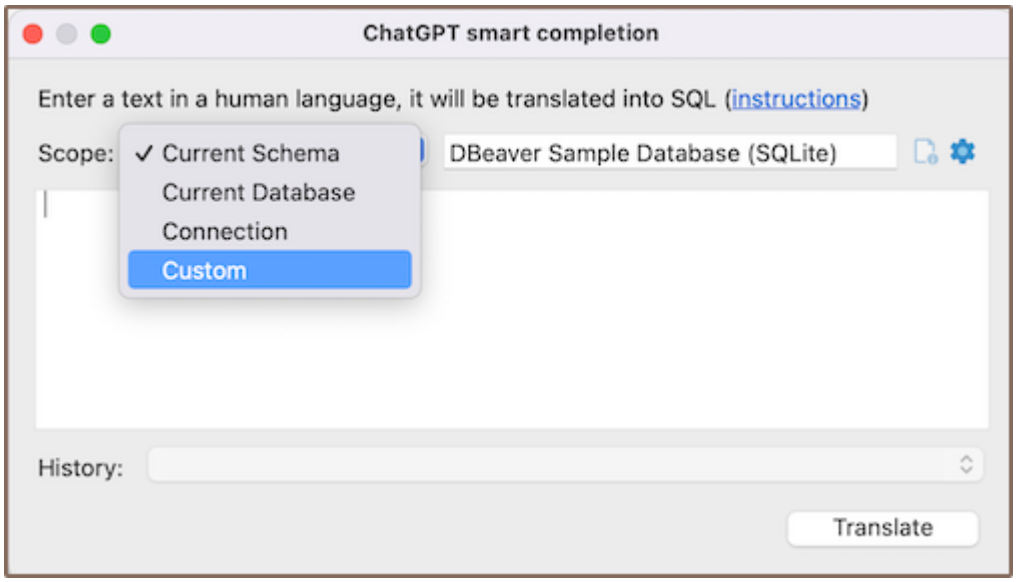
OpenAI will know table and column names in your database. DBeaver will ask about this on the first smart completion for every connection. Learn more about data privacy

Additional settings

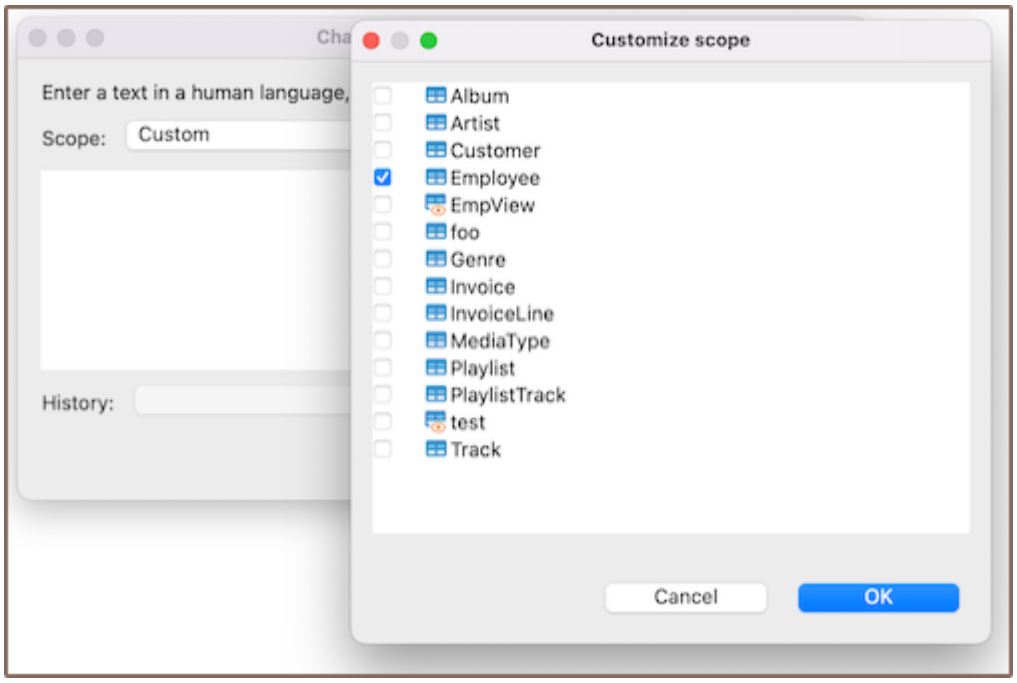
Scope

You can specify the scope used to generate queries in the **Scope** field. You may need this if your database schema is large enough to eliminate problems when generating queries.

If you want to choose a table, select the **Custom** option.



Then select a table.

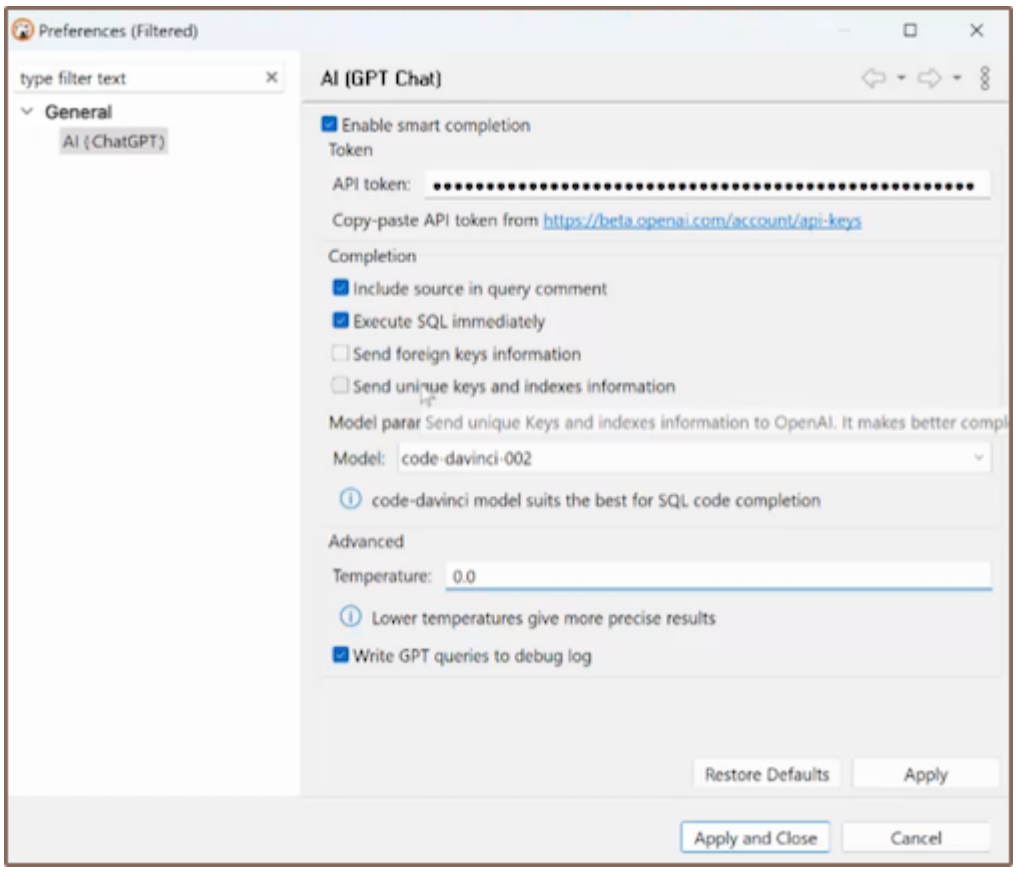


History

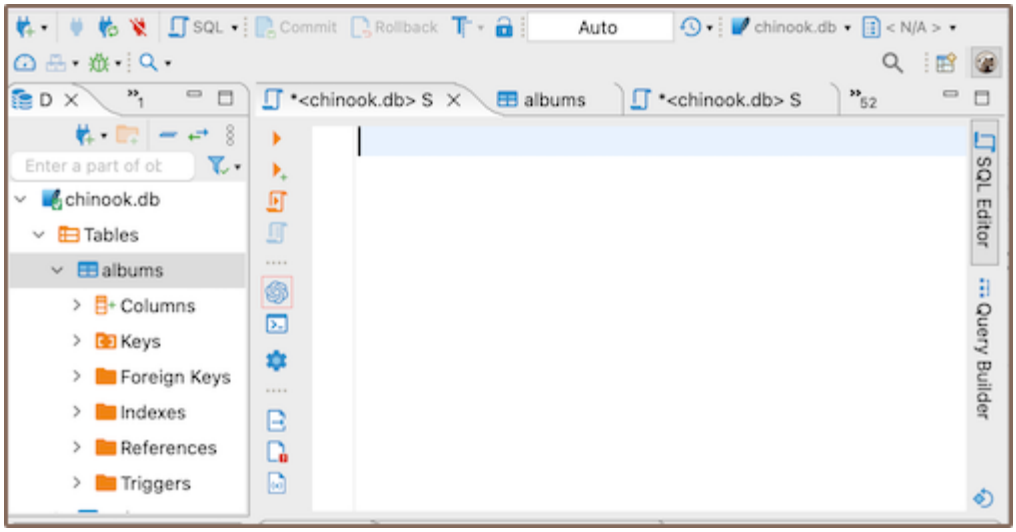
You can see the request history in the **History** field. If you are using the PRO version, you can see the request history for previous sessions.

Getting started

1. If using the Community edition, install the extension.
2. Open **Window** -> **Preferences** -> **General** -> **AI (ChatGPT)**. Click **Enable smart completion** if it's not on (as in the PRO versions).
3. Copy the available AI secret key to the **API token** field and apply changes. Where to find a secret key.



4. Open **SQL Editor** (F3) and click the **GPT icon** in the left toolbar. (Note: toolbar is customizable. See Toolbar Customization)



5. Confirm the metadata transfer to Open AI.

6. Write your question in the **ChatGPT smart completion** window and press **Translate**.

You can also select database objects to work within the **Scope** field. This is useful if you have a large schema with many objects.

How to write questions

You need to write a phrase in English or any other language describing what you want to get from the database. You should know at least something about your database structure. You can correct the query step by step, add details and make it more accurate. You can add specific details, such as how to build a query or which tables or columns to use.

For example, using DBeaver sample SQLite database, you could write something like this:

Example 1: "show customers from Italy"

```
SELECT * FROM customers WHERE country = 'Italy';
```

Example 2: "montre les clients de France"

```
SELECT * FROM customer WHERE country = 'France';
```

Example 3: "show customers who purchased blues tracks, use joins"

```
SELECT c.FirstName, c.LastName, t.Name, g.Name
FROM Customer c
JOIN Invoice i ON c.CustomerId = i.CustomerId
JOIN InvoiceLine il ON i.InvoiceId = il.InvoiceId
JOIN Track t ON il.TrackId = t.TrackId
JOIN Genre g ON t.GenreId = g.GenreId

WHERE g.Name = 'Blues'
ORDER BY c.LastName, c.FirstName;
```

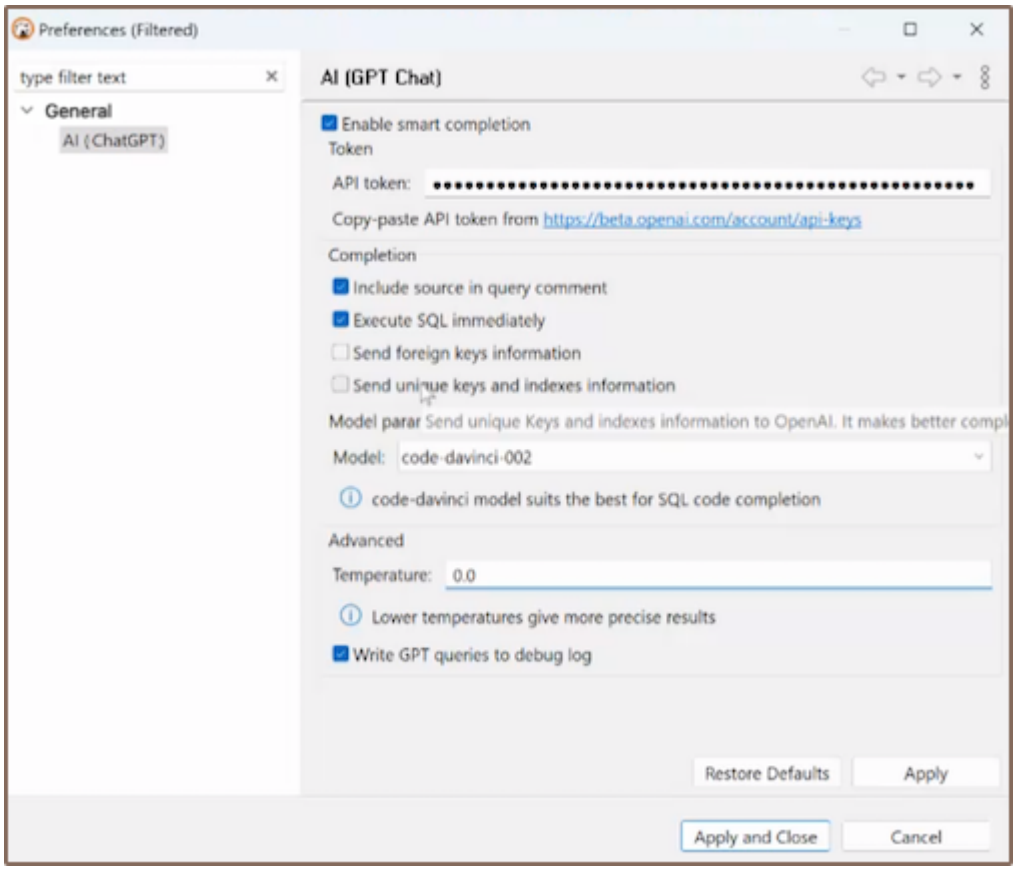
Example 4: "get names of customers who purchased blues tracks, use joins"

```
SELECT DISTINCT c.FirstName, c.LastName
FROM Customer c
JOIN Invoice i ON c.CustomerId = i.CustomerId
JOIN InvoiceLine il ON i.InvoiceId = il.InvoiceId
JOIN Track t ON il.TrackId = t.TrackId
JOIN Genre g ON t.GenreId = g.GenreId
WHERE g.Name = 'Blues';
```

Configure

You don't need to configure ChatGPT smart completion to use it. It's enough to specify the API key, and everything will work. However, if you have problems generating SQL queries or want to experiment, you can try changing some settings.

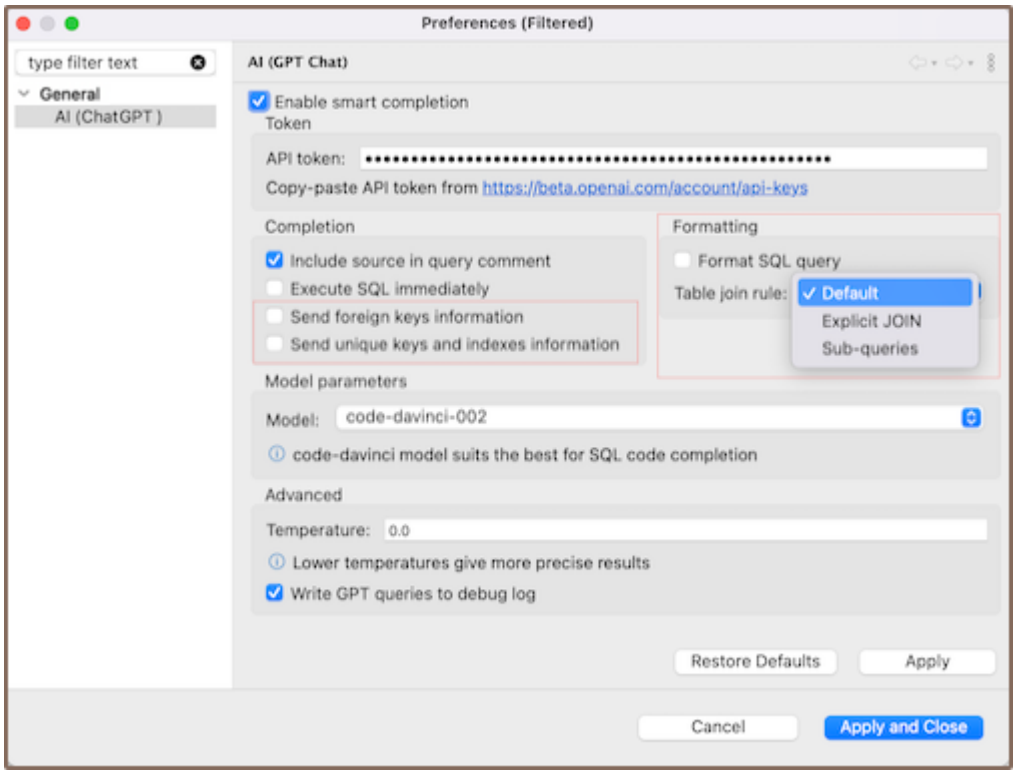
To configure this feature navigate to **Window** -> **Preferences** —> **General** —> **AI (ChatGPT)**.



- **Enable smart completion** shows the ChatGPT icon in SQL Editor. If this option is not selected, you can enable ChatGPT in Preferences.
- **API token** it's a secret key from the OpenAI platform. Where to find a secret key
- **Include source in query comment** — select this option if you want to see the request in human language in the SQL Editor before the generated query.
- **Execute SQL immediately** — select this option if you want to run SQL query just after translation.
- **Model** is a GPT-3 tool for understanding and generating natural language. The best model for generating SQL queries is **gpt-3.5-turbo**.
- **Temperature** sets the level of creativity of the translation results. If you need accurate results, use 0.0. For less common and more creative output, use 0.9.
- **Write GPT queries to debug log** — select this option if you want to see requests to ChatGPT in the log files.

Settings in PRO versions

There are some additional settings available in Enterprise and Ultimate editions only.



- **Send foreign keys information** and **Send unique keys and indexes information** allow to understand the relationships between tables, which can help create complex queries to a database with a complex structure.
- **Format SQL query** add formatting.
- **Table join rule** allows explicit JOIN or JOIN with sub-queries.

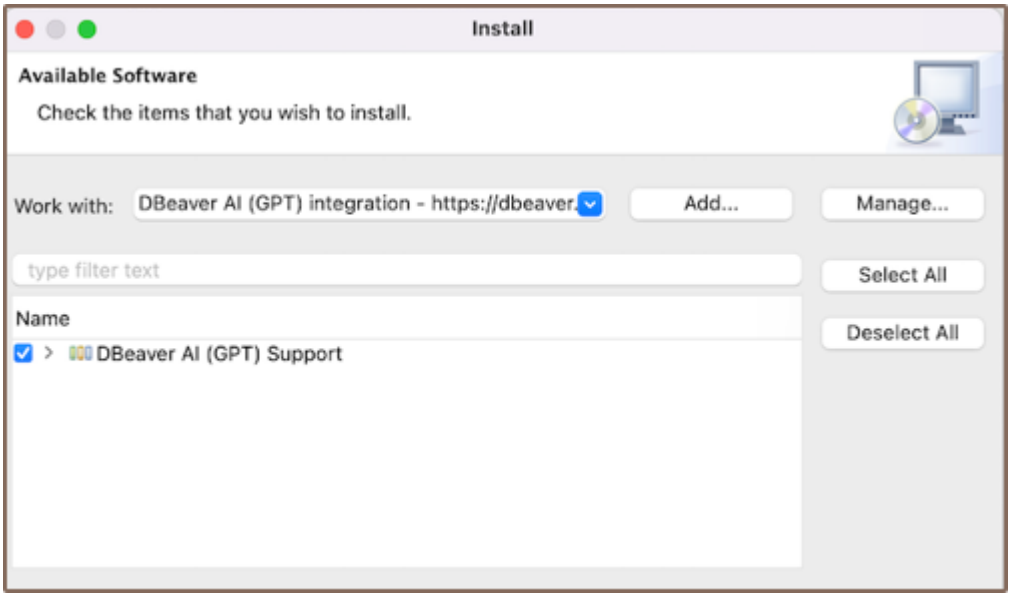
Receive API key

1. Register on the OpenAI platform.
2. Open API Keys section in your profile, and click Create new secret key button.
3. You'll see the new secret key, copy it and paste it into the **API token** field in Preferences.

Install GPT Extension

You only need to install this extension in the DBeaver Community version. In other editions, it is already installed.

1. From the main menu, select **Help —> Install New Software**.
2. In the installation window, in the **Work with** field select **DBeaver AI (GPT) integration** .
3. Then select **AI (GPT) Support**, press **Next** and follow the installation process.



4. Restart DBeaver.

That's all done. Open the SQL editor; in the left toolbar, you'll see the ChatGPT icon.

Disable

You can temporarily remove the ChatGPT icon from the SQL Editor toolbar. Move on **Window** -> **Preferences** —> **General** —> **AI (ChatGPT)** and uncheck the **Enable smart completion** option. You can always enable it again.

You can permanently disable ChatGPT smart completion only in PRO versions. In this case, it cannot be enabled in Preferences.

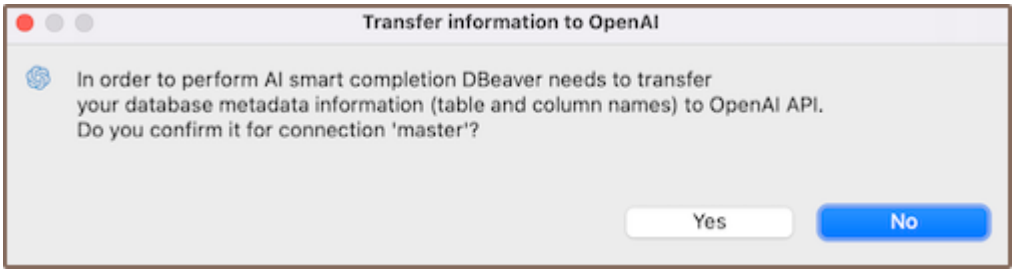
How to do it:

- set system variable `DBEAVER_AI_DISABLED` to `true`.
- or add the string `Dai.disabled=true` to the end of `dbeaver.ini`.

Data privacy

DBeaver needs to send the tables and column names of the current database schema to the OpenAI platform to convert your request into an SQL query. DBeaver does not send any data from tables. The full text of the request can be seen in the log file, which you can enable in **Preferences** by selecting the **Write GPT queries to debug log** option.

DBeaver will ask for confirmation the first time you use ChatGPT completion for each connection. You must confirm the metadata sending before using this feature.



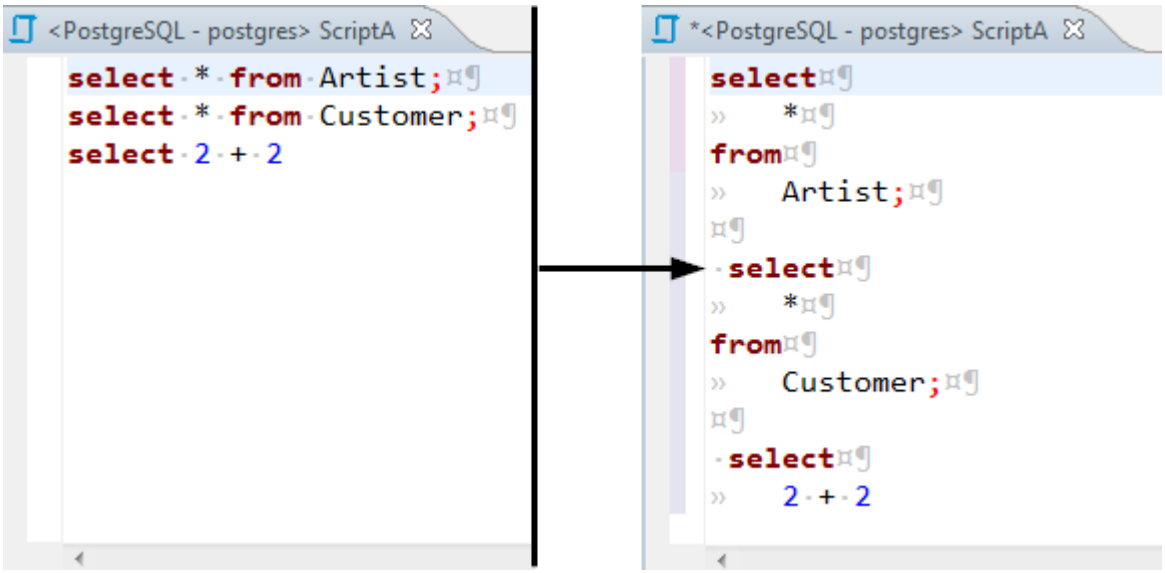
If you don't want to send information about some tables, you can choose which tables to use in the **Scope** field.

You can completely disable the ChatGPT feature in PRO versions. How to disable it

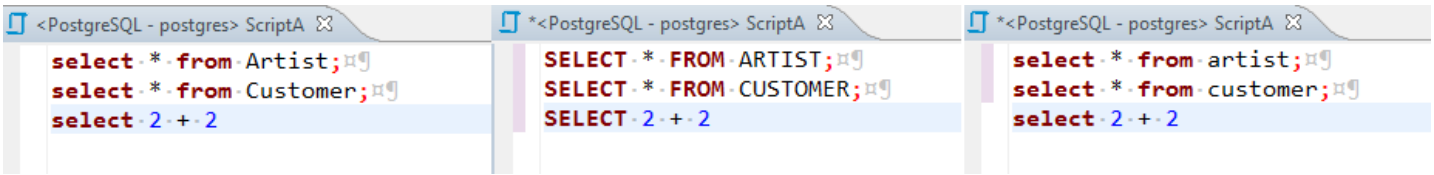
If you don't want to use ChatGPT in the CE version, you can choose not to install the plugin, and it will not be visible at all.

SQL Formatting

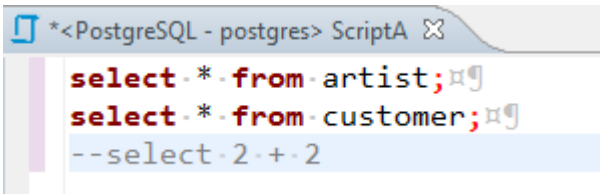
To format an SQL text, select it and press `Ctrl+Shift+F` or right-click the selected text and click **Format -> Format SQL** on the context menu.



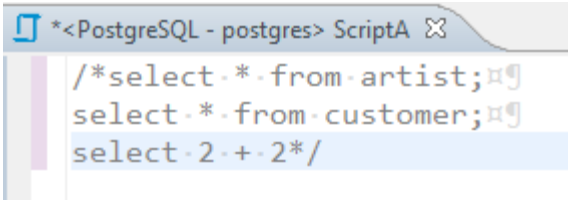
To format a script to upper or lower case, highlight the SQL text, then right-click it and click **Format -> To Upper Case / To Lower Case**, respectively, on the context menu.



To comment out an SQL line, press `Ctrl+/` or right-click the line and click **Format -> Toggle Line Comment** on the context menu. To uncomment a commented line, manually remove the commenting syntax, or press the same button combination, or right-click the line and click the same item on the context menu.

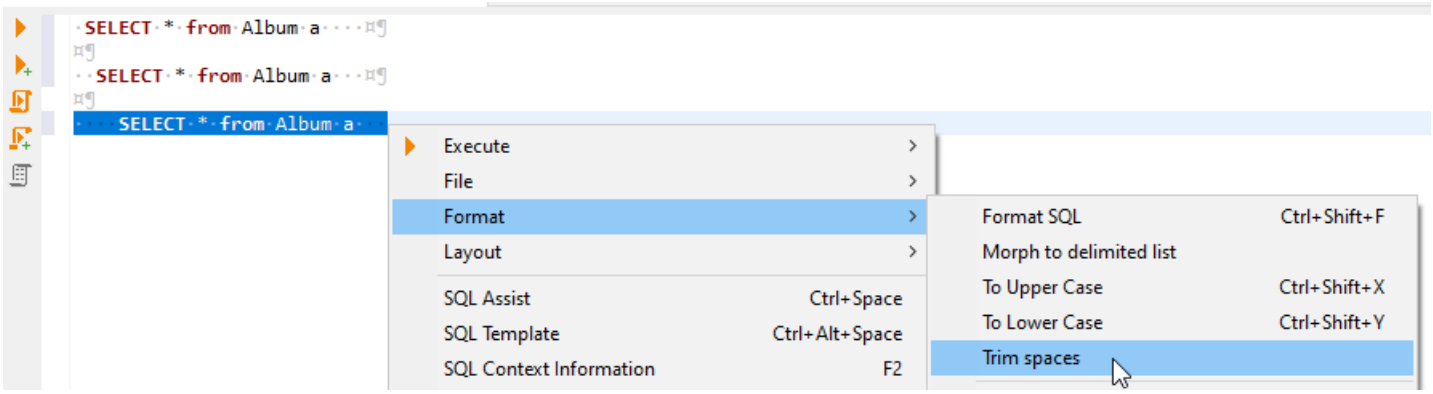


To comment out a block of text, select the text, then press `Ctrl+Shift+/` or right-click it and click **Format -> Toggle Block Comment** on the context menu. To uncomment a commented block of text, you can either manually remove the commenting syntax or select the same block of text, right-click it and click the same item on the context menu or press the same button combination.

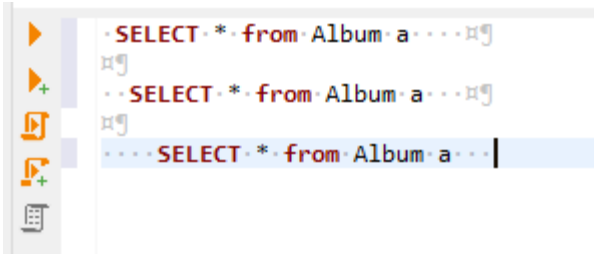


To trim spaces (leading and trailing) SQL text right-click the selected text or end of string and click **Format -> Trim spaces** on the context menu.

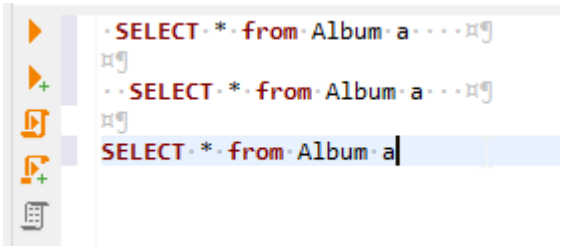
You can choose a part of the text



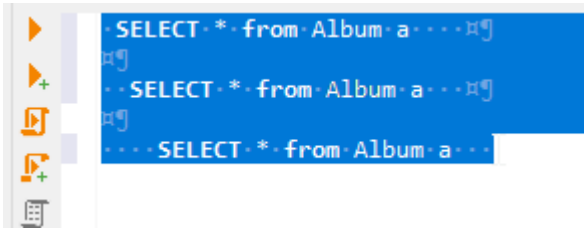
or put the cursor at the end of the row you want to trim



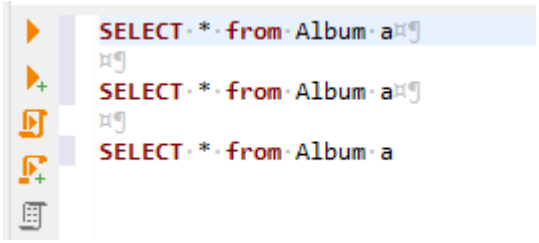
You will get the following result:



Or you can choose a part of the text from many lines



Then each of the selected rows will trim, and the final result will be like this:


SELECT * from Album a
SELECT * from Album a
SELECT * from Album a

SQL Execution

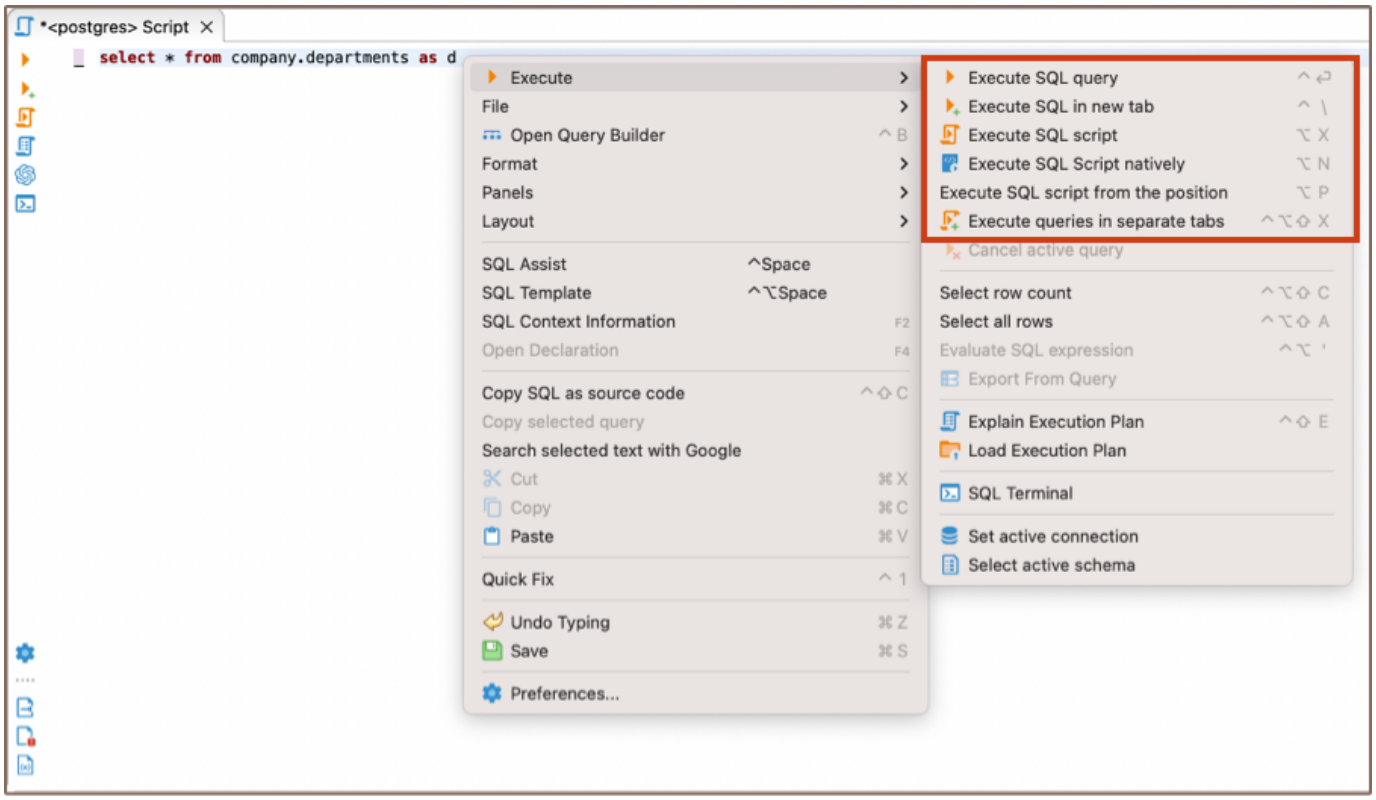
You can execute one query, a highlighted portion of a script, or a whole script. You can execute them using the following:

- Shortcut key combinations (see details further in this article)
- Tools in the main toolbar:

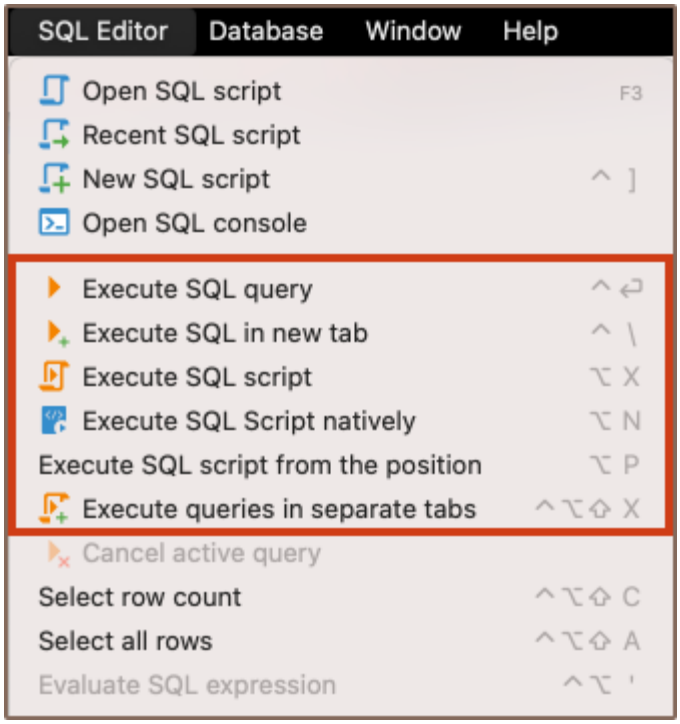


Note: toolbar is customizable. See Toolbar Customization

- Context menu (right-click the query):



- DBeaver main menu:



To execute a query under the cursor or selected text, press `Ctrl+Enter` or right-click the query and click **Execute** -> **Execute SQL Statement** on the context menu. You can do the same using the main toolbar or main menu: **SQL Editor** -> **Execute SQL Statement**. This executes the SQL query under the cursor or selected text and fills the results pane with the query results.

To execute a query under the cursor in a separate tab, press `CTRL+\` or right-click the query and click **Execute** -> **Execute SQL in new tab** on the context menu. The same can be done using the main toolbar or the main menu: **SQL Editor** -> **Execute SQL in new tab**. This executes the SQL query under the cursor or selected text and creates a new results tab.

To execute the whole script, press `Alt+X` or click **Execute** -> **Execute SQL Script** on the context menu or **SQL Editor** -> **Execute SQL Script** on the main menu or in the main toolbar. This executes all queries in the current editor (or selected queries) as a script. DBeaver parses queries one by one using a statement delimiter (";" by default) and executes them consecutively. You can configure the script execution behavior in the SQL editor preferences (Right-click the script and click **Preferences** on the context menu).

To execute the script natively, press `Alt+N` or click **Execute** -> **Execute SQL Script natively** on the context menu or **SQL Editor** -> **Execute SQL Script natively** on the main menu or in the main toolbar. Upon activation, a setup wizard is launched, which allows you to configure the parameters for script execution before the script is launched in the native client like PLSQL, MySQL, or SQLPlus. The results are displayed in a text field in the format of console output. It is handy when functions are not supported by DBeaver drivers and require more specialized clients or when the function is weighty, and a faster client is needed.

Note: This function is available for MySQL/Maria, Oracle, and PostgreSQL and may require additional software installation for each database.

To execute a script opening, each query results in a separate tab, press `Ctrl+Alt+Shift+X` or click **Execute** -> **Execute Statements In Separate Tabs** on the context menu or **SQL Editor** -> **Execute Statements In Separate Tabs** on the main menu or in the main toolbar. It executes all queries in the script but opens multiple result tabs. Each script query is executed in a separate thread (that is, all queries are executed simultaneously).
Important: Executing a massive script with numerous queries can result in unforeseen problems.

Result tabs

A single query may generate several result sets represented by tabs. These tabs are linked to the query they are executed from.

- To close an individual tab, press `CTRL+Shift+\` or middle-click on a tab header.
- To close all tabs expect current, click **Close all result tabs except this** on the context menu of this tab.
- To close all tabs of the desired query, click **Close all result tabs of same query** on the context menu of this tab.

Naming

A tab is often named after the primary table of your query. For example, after executing the following query you will see a single tab called `Album` (assuming that your database has a table called `Album`):

```
SELECT * FROM Album;
```

If a query has joins or, in other words, has multiple source tables, a `(+)` is shown right to the table name. The following query will result in a tab called `Album(+)`:

```
SELECT * FROM Album al, Artist ar WHERE al.AlbumId = ar.ArtistId;
```

Additionally, you can change the name of a given tab via its context menu or by using a special comment:

```
-- title: DBeaver is cool
SELECT * FROM Album;
```

In other cases, tabs are named in the form of `Results <A> (<B>)`, where:

- **A** is *an index of query*
- **B** is *an index of the result set of this query*

Pinning

Tabs can be moved around by dragging them with a mouse and pinned using the **Pin tab** on the context menu of the desired tab. Pinned tabs are stacked on the left. They can be moved among other pinned tabs but can't be mixed with unpinned tabs. Pinned tabs cannot be closed without being unpinned first and cannot be overwritten by executing a query (by making this tab active).

Detaching

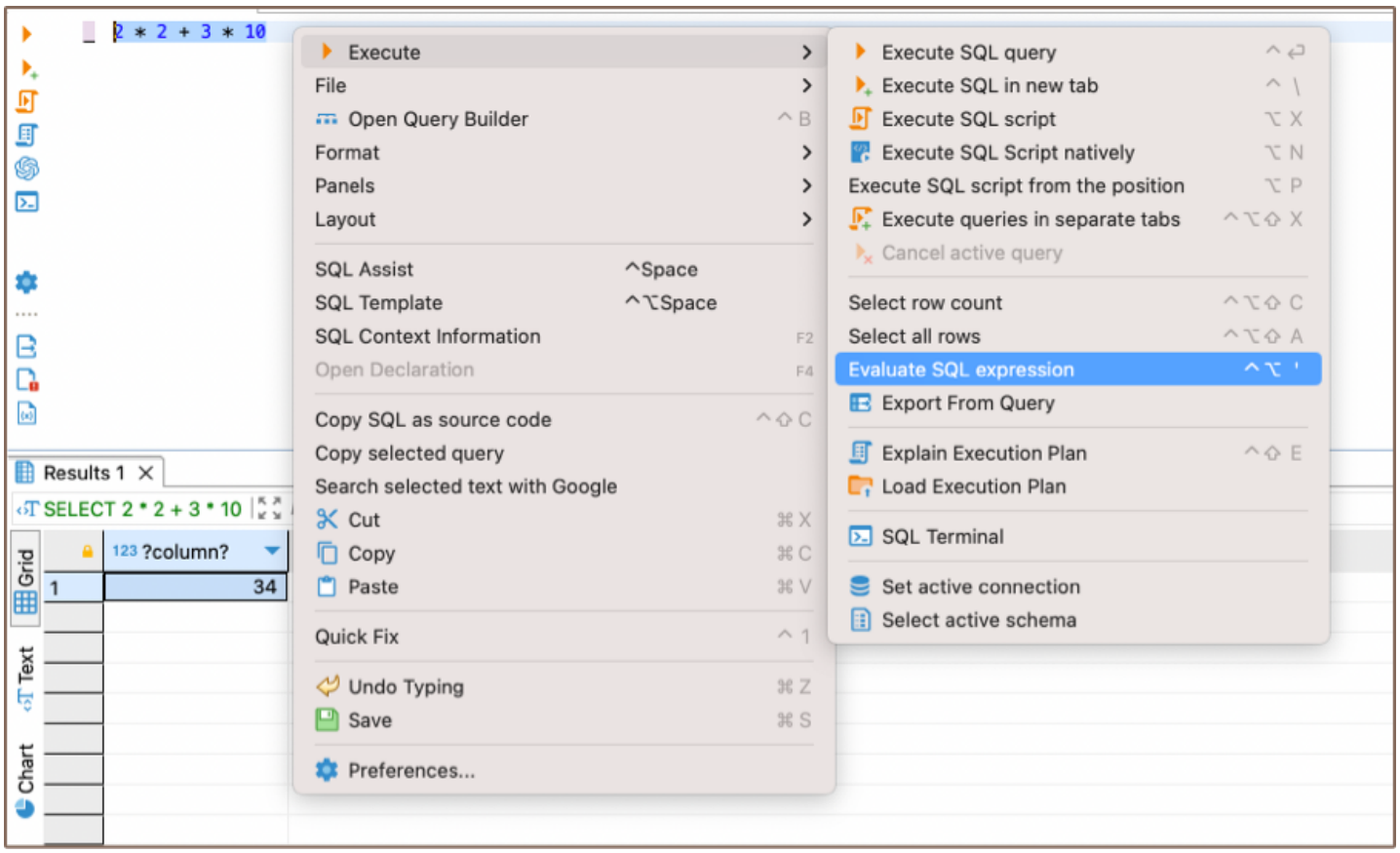
Tabs can be detached from the SQL editor into a separate view using the **Detach Tab** action found in the context menu of the desired tab. After the tab is detached, you can rearrange and move it anywhere you want (for example, you can put two tabs side-by-side for comparison).

Additionally, you can detach it from the application window using **Detach** found in the context menu of an already detached tab.

After the tab is detached, it's still synchronized with the SQL editor, meaning you can edit and refresh data as long as the SQL editor that produced that tab is open. Once you close it, tabs become read-only.

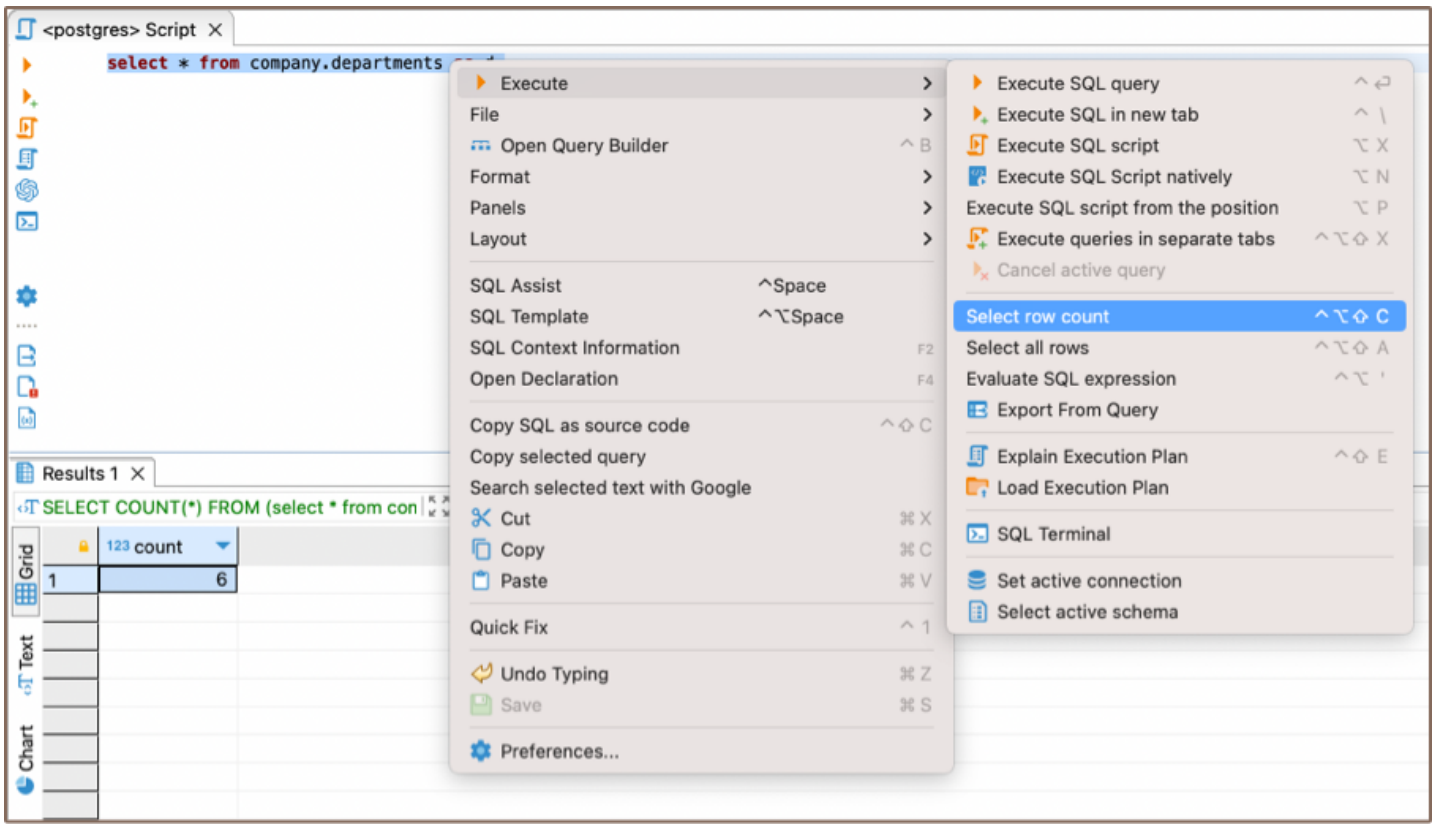
SQL Expression Evaluation

To evaluate an SQL expression, right-click the expression and click **Execute -> Evaluate SQL expression** on the context menu. This command performs a query of **SELECT [expression] FROM DUAL** type:



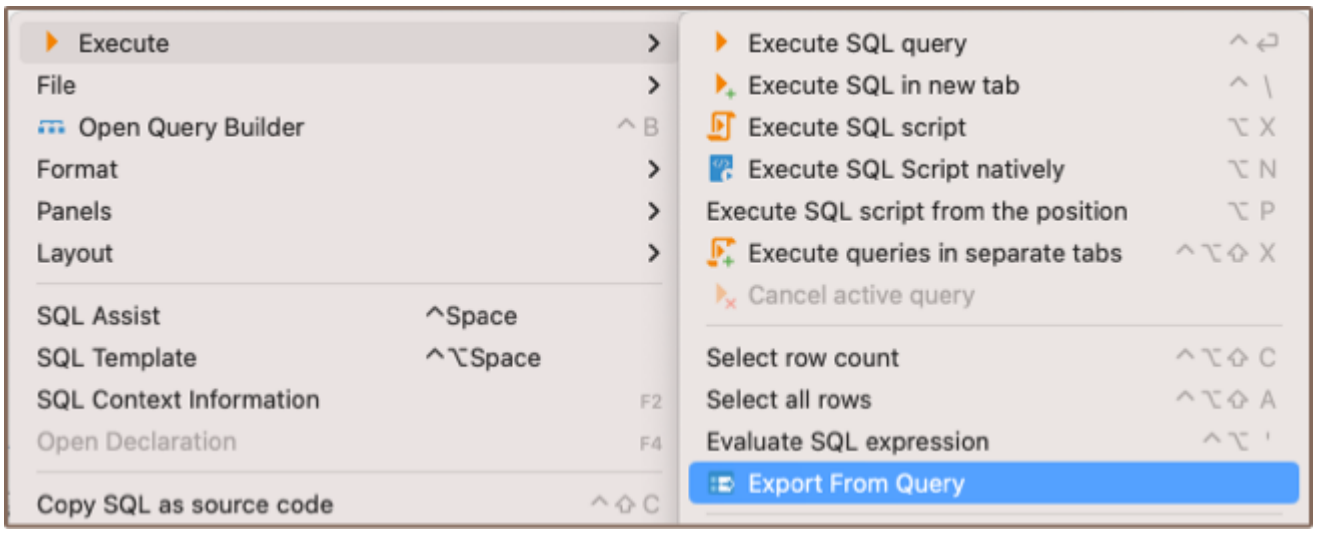
Row Count

If you want to know how many rows an SQL query will produce, you need to apply the Row Count feature – highlight and right-click the SQL text and then click **Execute** -> **Select row count** on the context menu:

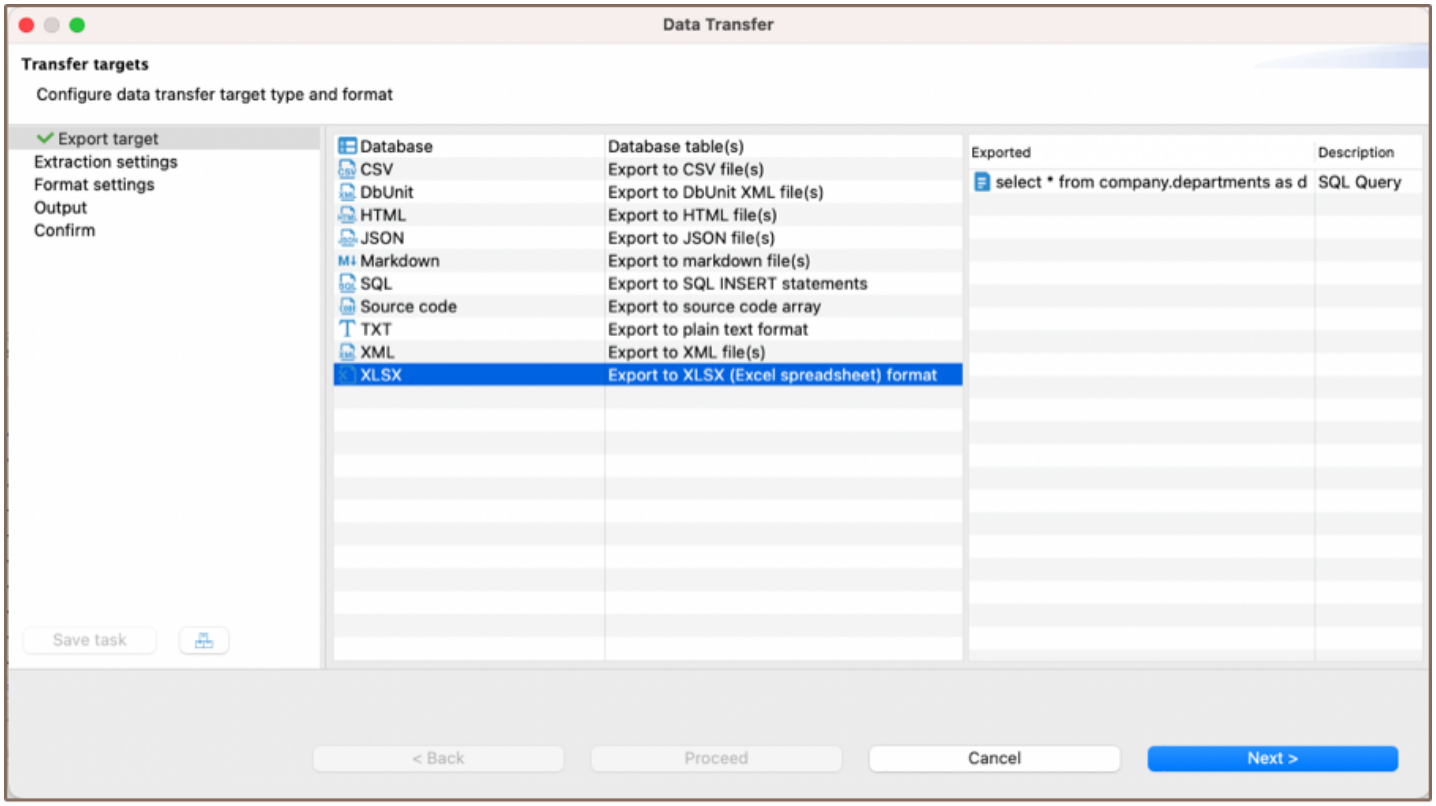


Query Export

It might be useful to export a query if you have a long-running query and you do not need to see its results in the results panel. You can directly export the current query results to a file/table by right-clicking the query and then clicking **Execute** -> **Export From Query** on the context menu:

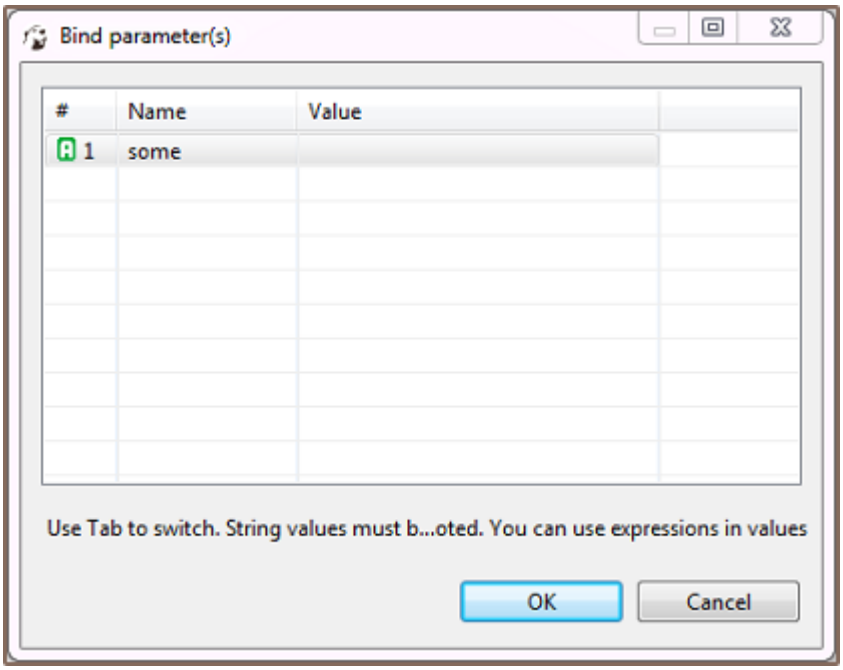


The Data transfer wizard opens. Go through its steps to complete the export of the query.



Dynamic Parameter Bindings

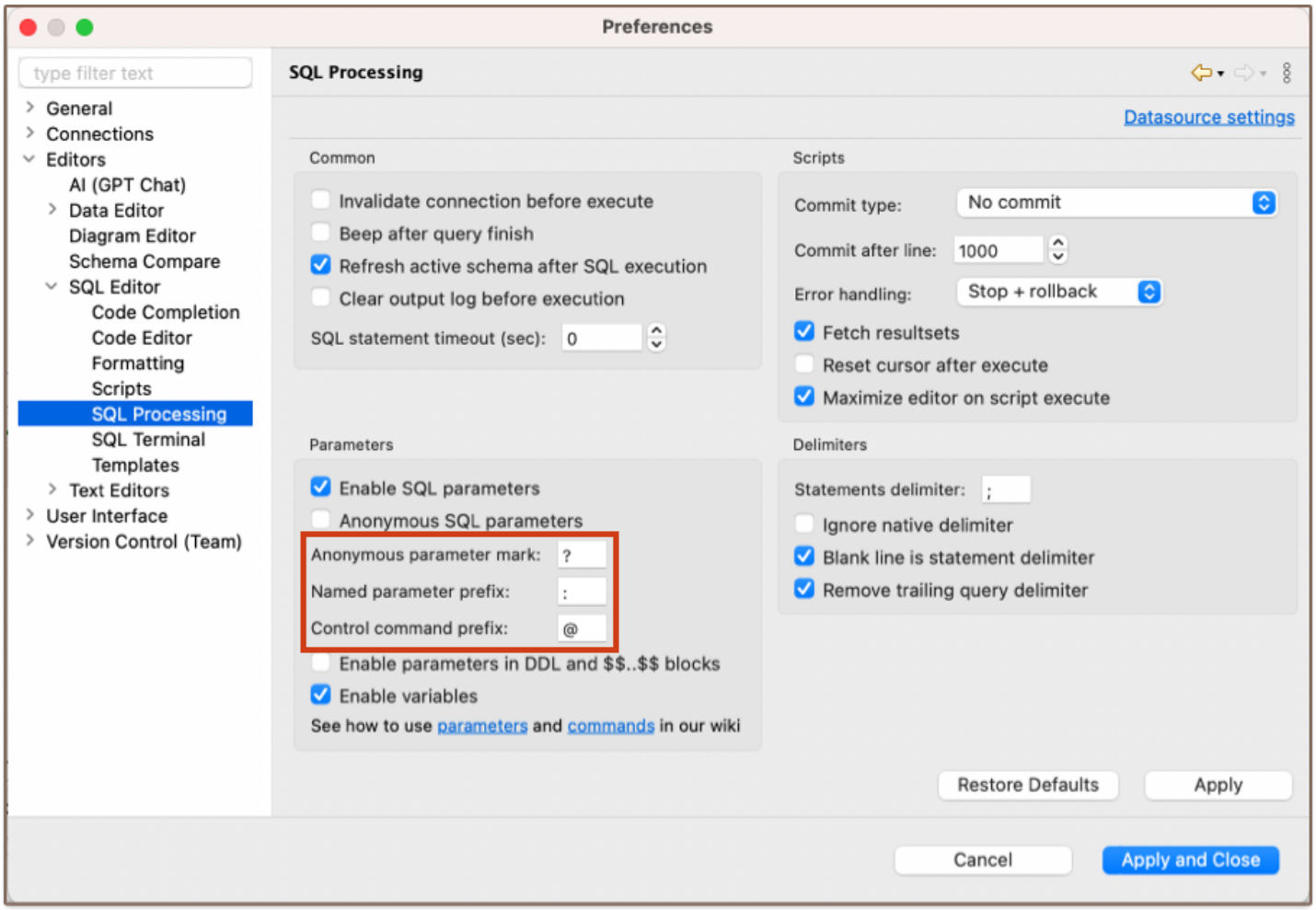
You can use dynamic parameters and variables in your SQL queries. The parameter format is `:name`. Also, variables syntax could be used as `${varname}`. When you execute a query that contains dynamic parameters, DBeaver displays a dialog box in which you can fill the parameter values:



This dialog appears every time you execute the query with parameters.

If you want to set the value once, you can use @set command or add the variable and its value in the Variables panel. Then Bind parameter(s) dialog will not appear.

You can also use anonymous parameters (?), but you will need to enable them in the SQL editor preferences:



You can open the SQL editor preferences by pressing `Alt+Enter`.

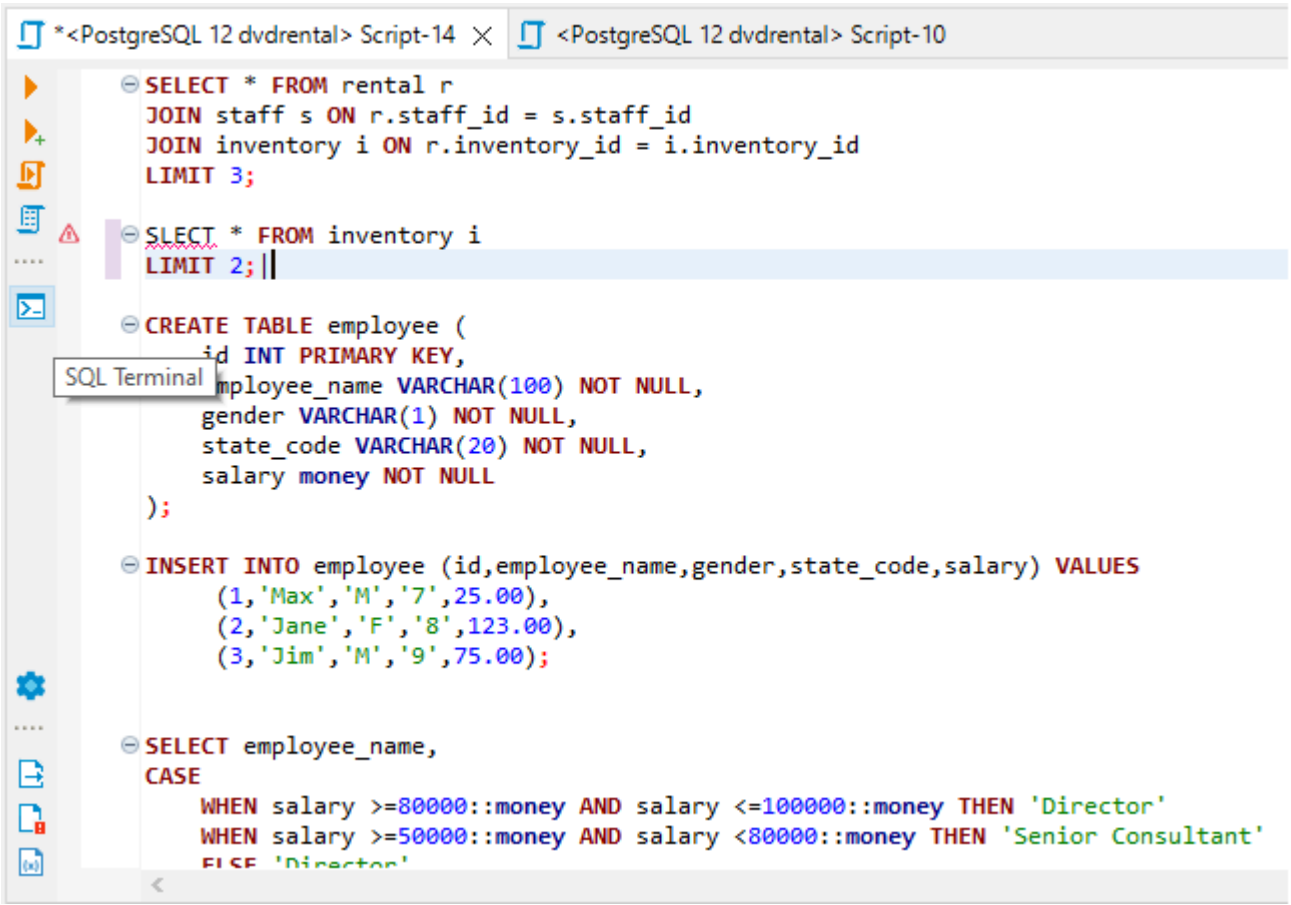
Miscellaneous

- To select the current query row count, press `Ctrl+Alt+Shift+C`.
- To open the definition of the database object currently in focus (under cursor) in a viewer/editor, press `F4`.

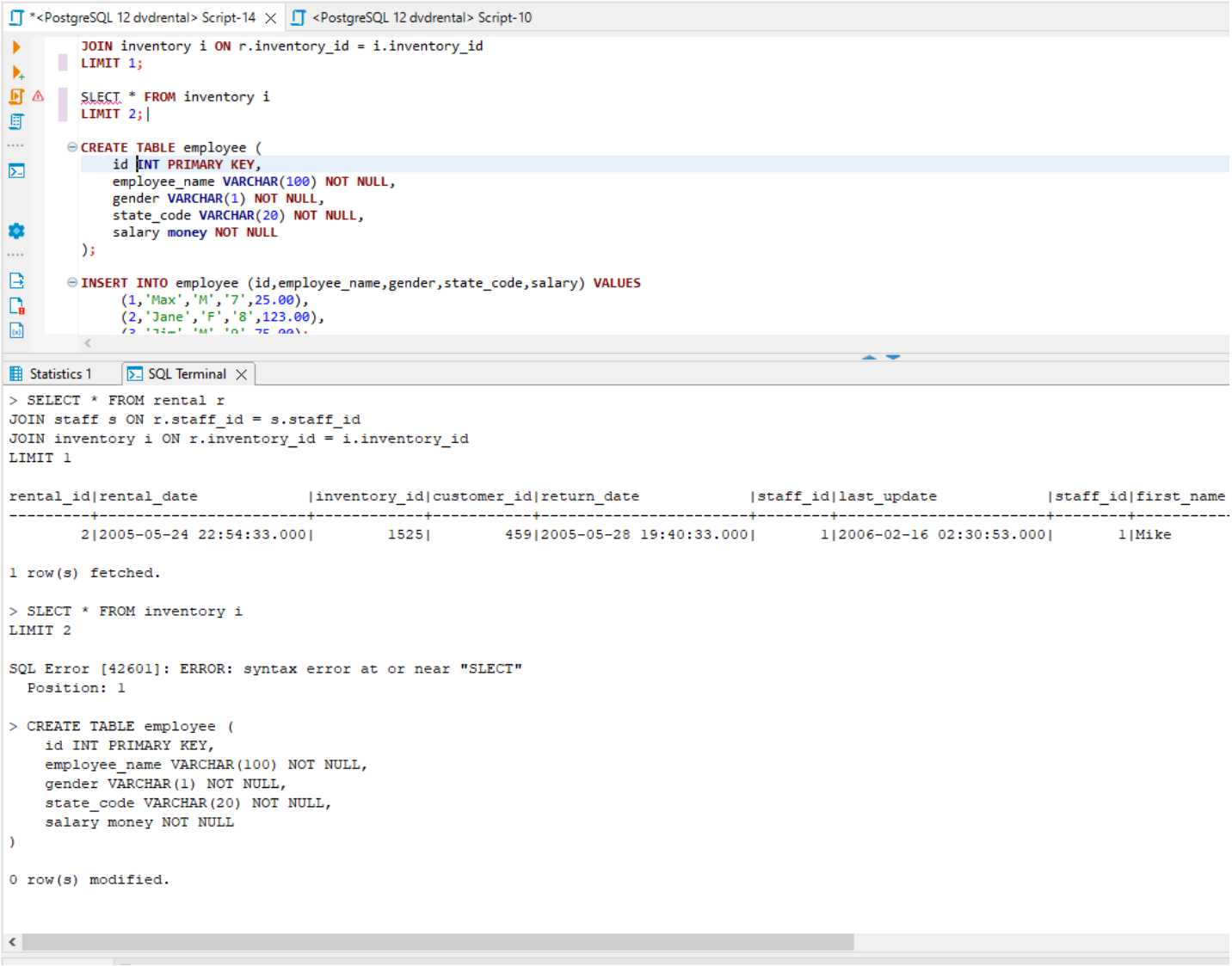
SQL Terminal

SQL Terminal is an SQL Editor result tab where you can find the results for all executed queries in a text format.

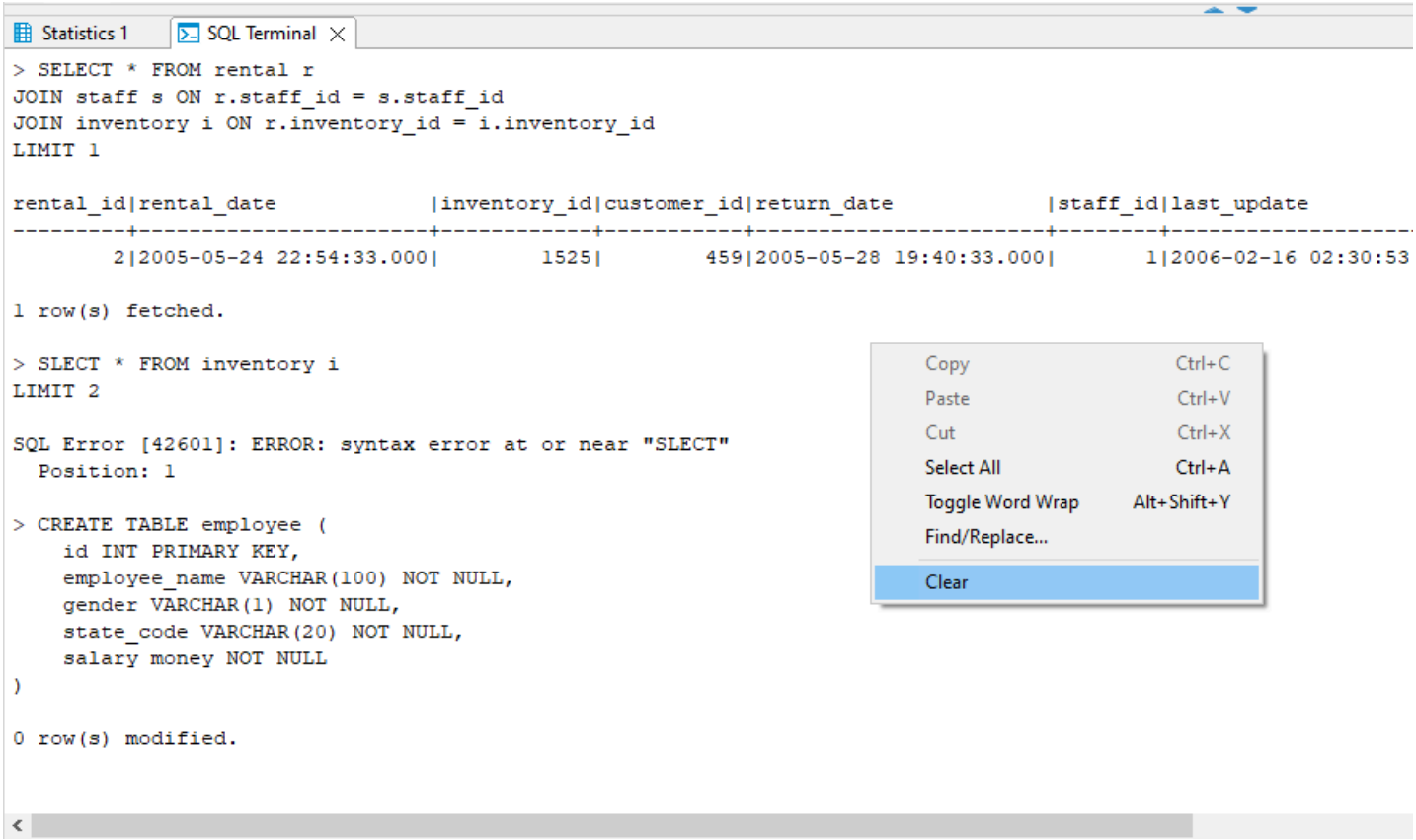
To open SQL Terminal you should press the corresponding button on the left SQL Editor toolbar. (Note: toolbar is customizable. See Toolbar Customization)



You can execute a single statement or a sql script and see the result set with data, errors, and statistical information along with the corresponding query text.

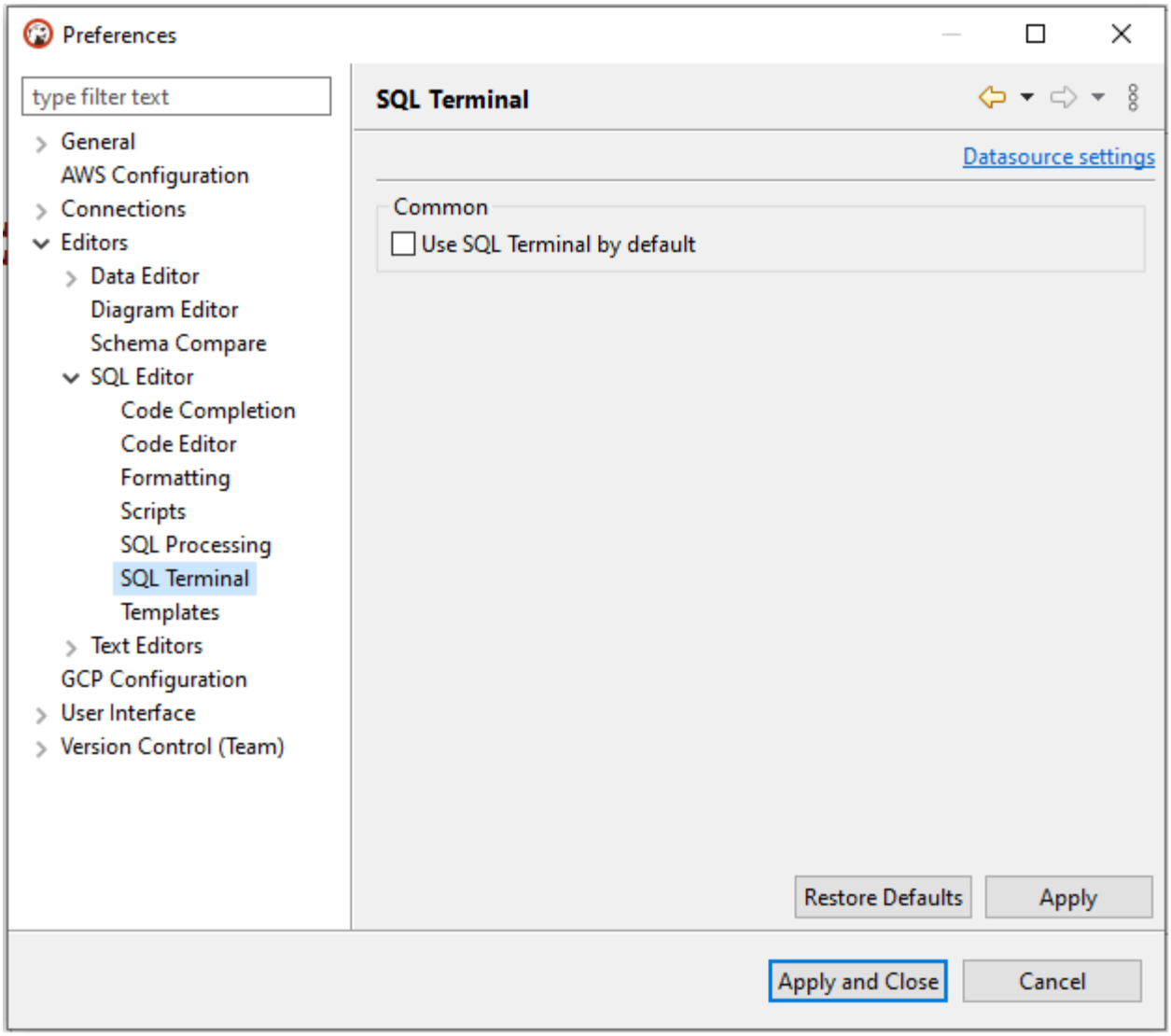


The context menu is available by right-clicking on the SQL Terminal area.



For each sql script file, it will be saved whether the SQL Terminal is enabled, and the state of the SQL Terminal button will be restored when the file is opened.

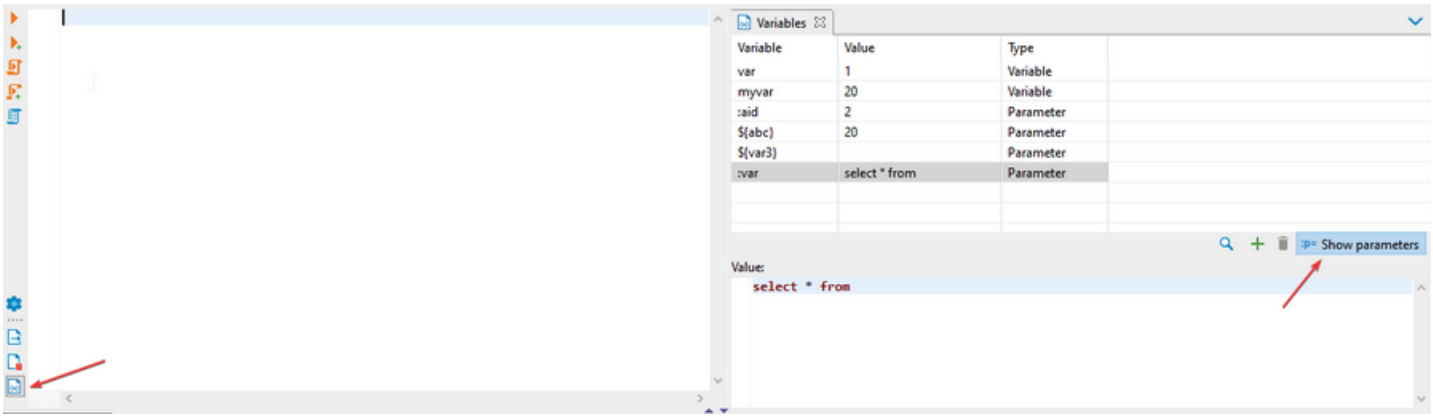
You can set up whether to enable or disable the SQL Terminal for new scripts by default in `Preferences->Editors->SQL Editor->SQL Terminal`.


Preferences
type filter text
General
AWS Configuration
Connections
Editors
Data Editor
Diagram Editor
Schema Compare
SQL Editor
Code Completion
Code Editor
Formatting
Scripts
SQL Processing
SQL Terminal
Templates
Text Editors
GCP Configuration
User Interface
Version Control (Team)
SQL Terminal
Datasource settings
Common
Use SQL Terminal by default
Restore Defaults
Apply
Apply and Close
Cancel

Variables panel

You can see all of the currently assigned local variables for SQL Editor. You need to click a "Show SQL variables" button in the SQL editor.

A new tab alongside Output and Execution log panels will be opened with a list of assigned variables. On this panel, you can also show assigned parameters by clicking the corresponding button. (Note: toolbar is customizable. See Toolbar Customization)



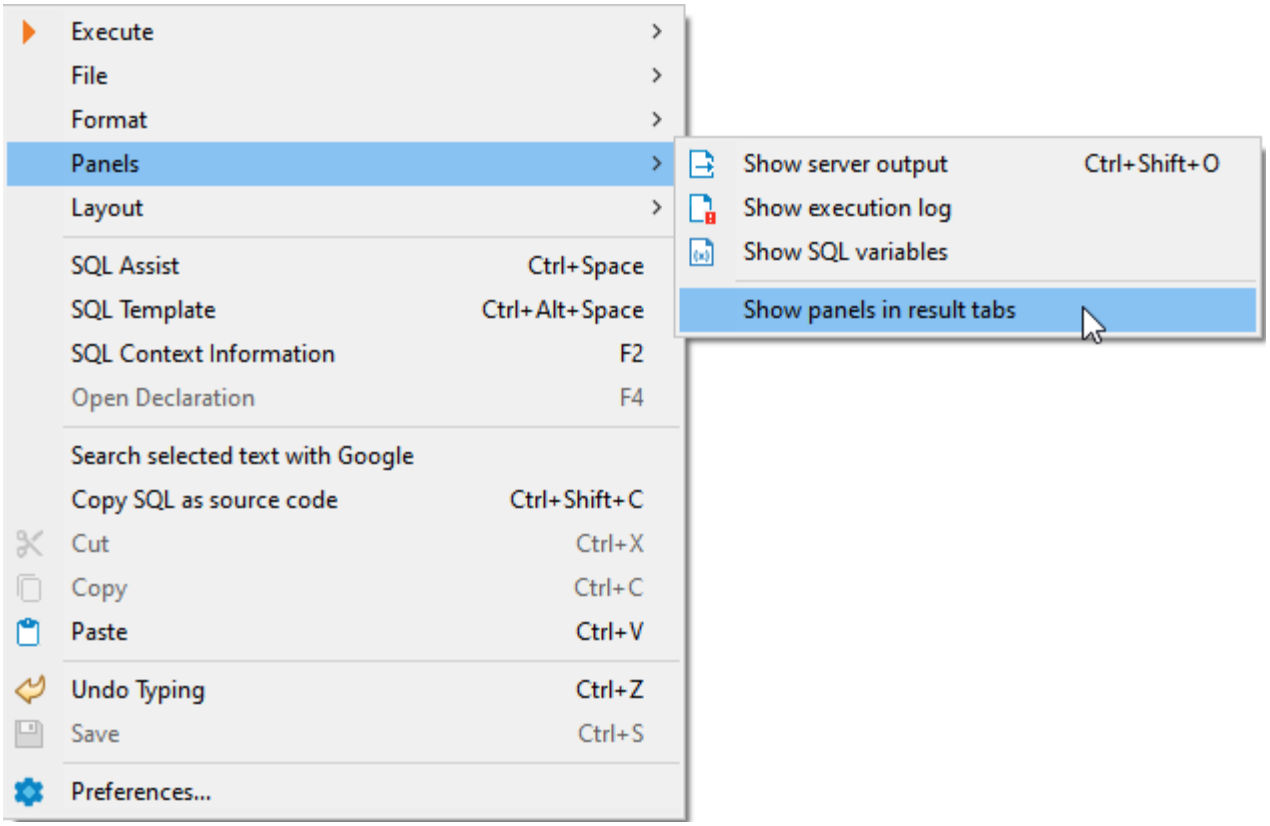
Manipulating variables

You can change values for the current variable using the variables tab. Simply click on a variable's row and edit its value in the window below. To delete or add a variable or parameter, you can use a corresponding button. Instead of typing @set or @unset, you can use these buttons in the script.

If you have a long list of variables, you can click a magnifying glass to initiate a search bar. Start typing either a variable's name or its value to filter the list.

Moving a panel

Variables panel is always connected to output and execution log panels, but this group of tabs can be configured to be shown either at the right side of the editor or at the bottom alongside the results panel. By default, the panels are shown on the right side. To change their location you need to either check or uncheck `Show panels i` in the context menu.


Execute
File
Format
Panels
Layout
SQL Assist Ctrl+Space
SQL Template Ctrl+Alt+Space
SQL Context Information F2
Open Declaration F4
Search selected text with Google
Copy SQL as source code Ctrl+Shift+C
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Undo Typing Ctrl+Z
Save Ctrl+S
Preferences...
Show server output Ctrl+Shift+O
Show execution log
Show SQL variables
Show panels in result tabs

Query Execution Plan

Execution Plan

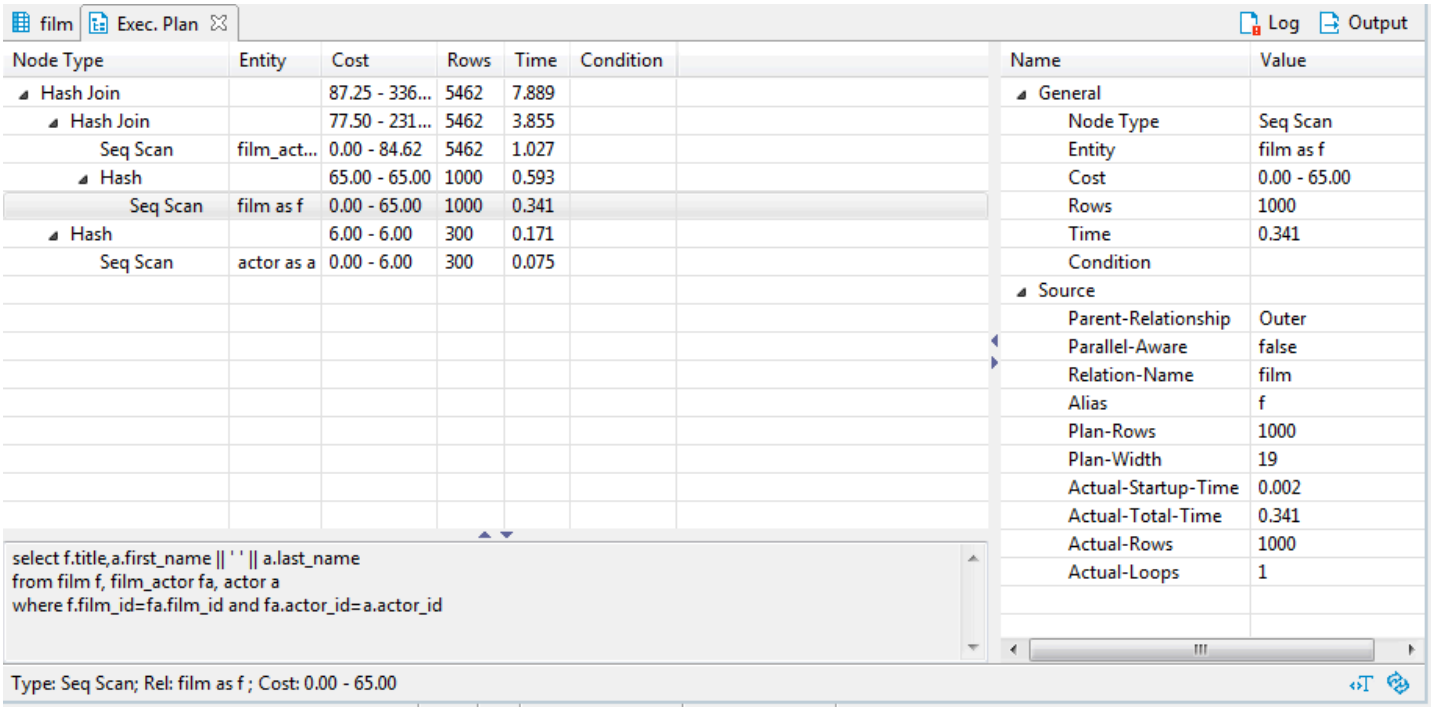
This feature is supported for the following data sources:

The Execution plan for databases marked with a star is supported only in Lite, Enterprise and Ultimate editions

- ClickHouse ★
- Couchbase ★
- DB2 LUW
- Exasol
- Firebird
- HSQLDB
- Microsoft SQL Server ★
- MySQL
- Netezza ★
- Vertica ★
- OceanBase
- Ocient
- Oracle
- PostgreSQL
- SAP HANA

Simple plan view

If a database driver supports the visualization of the execution plan, you can see the execution plan of the current query (under cursor) by pressing `Ctrl+Shift+E` or clicking **Explain execution plan** on the context menu or in the SQL Editor toolbar: (Note: toolbar is customizable. See Toolbar Customization) The execution plan command generates a tree of query execution as one of the result tabs and is convenient in estimating if the query /script is quick/optimal enough:



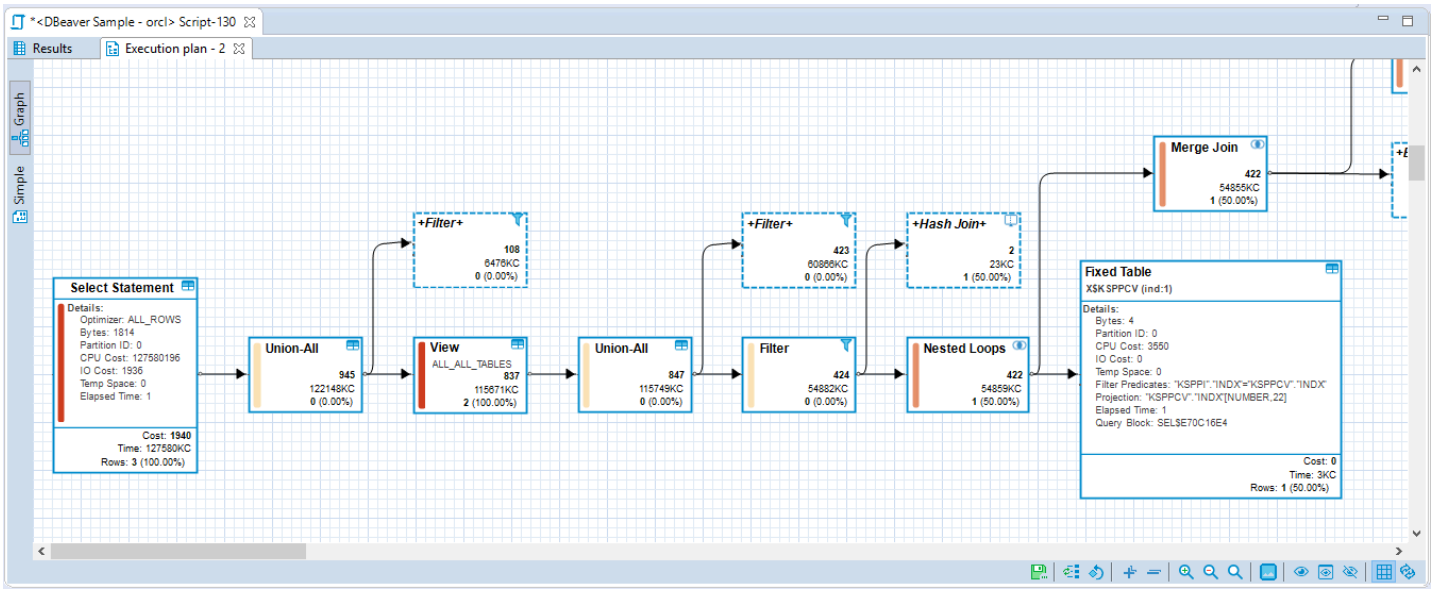
You can click the rows of the execution plan to see their details (statistics) in the panels below and to the right of the plan.

To reevaluate the plan, click the **Reevaluate** button ( ). To see the source script on which the plan is based, click the **View Source** button ( ).

Advanced plan view ★

In DBeaver Lite, Enterprise, and Ultimate editions you can use an advanced (graph) visualization of the execution plan.

This visualization shows the most expensive (cost-based) plan nodes. You can hide all irrelevant nodes (see node details), use horizontal or vertical plan layouts, export it to an image or save it as JSON to send to a colleague.


*<DBeaver Sample - orcl> Script-130
Results
Execution plan - 2
Graph
Simple
Select Statement
Details:
Optimizer: ALL_ROWS
Bytes: 1814
Partition ID: 0
CPU Cost: 127580196
IO Cost: 1936
Temp Space: 0
Elapsed Time: 1
Cost: 1940
Time: 127580KC
Rows: 3 (100.00%)
Union-All
945
122148KC
0 (0.00%)
+Filter+
108
6476KC
0 (0.00%)
View
ALL_ALL_TABLES
837
115671KC
2 (100.00%)
Union-All
847
115749KC
0 (0.00%)
+Filter+
423
60866KC
0 (0.00%)
Filter
424
54882KC
0 (0.00%)
+Hash Join+
2
23KC
1 (50.00%)
Nested Loops
422
54859KC
1 (50.00%)
Merge Join
422
54855KC
1 (50.00%)
Fixed Table
X$KSPPCV (ind:1)
Details:
Bytes: 4
Partition ID: 0
CPU Cost: 3550
IO Cost: 0
Temp Space: 0
Filter Predicates: "KSPPI"."INDX"="KSPPCV"."INDX"
Projection: "KSPPCV"."INDX"[NUMBER,22]
Elapsed Time: 1
Query Block: SEL$E70C16E4
Cost: 0
Time: 3KC
Rows: 1 (50.00%)

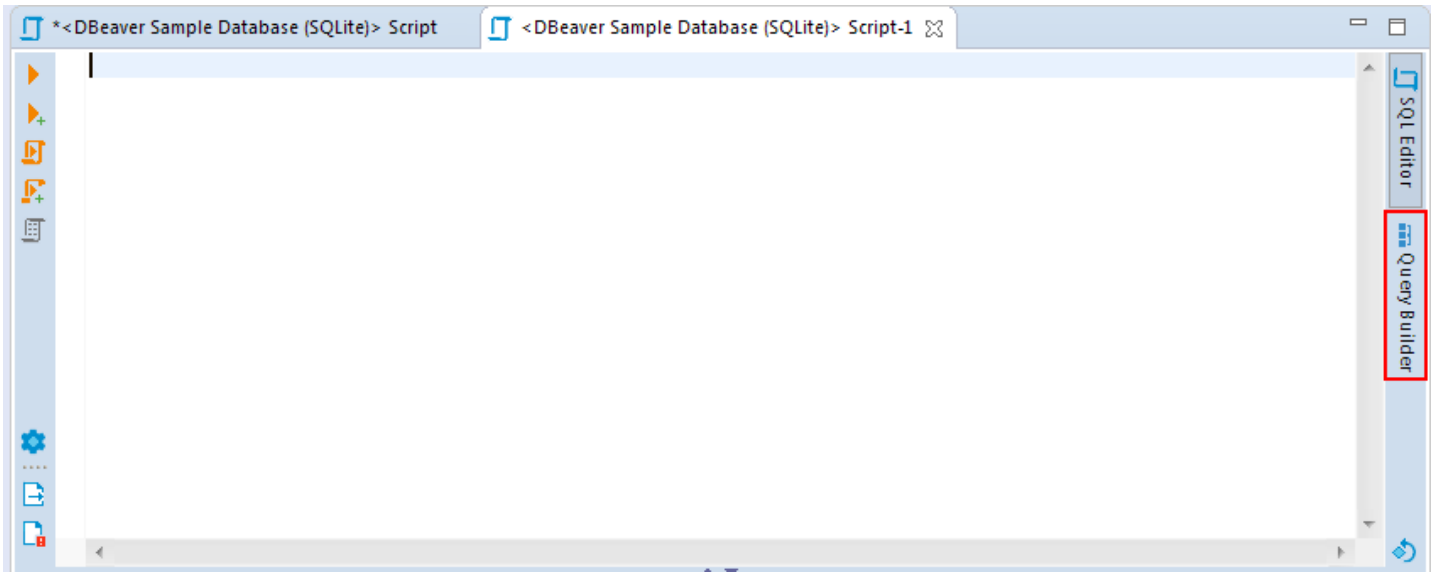
Visual Query Builder

Note: This feature is available in Lite, Enterprise, Ultimate and Team editions only.

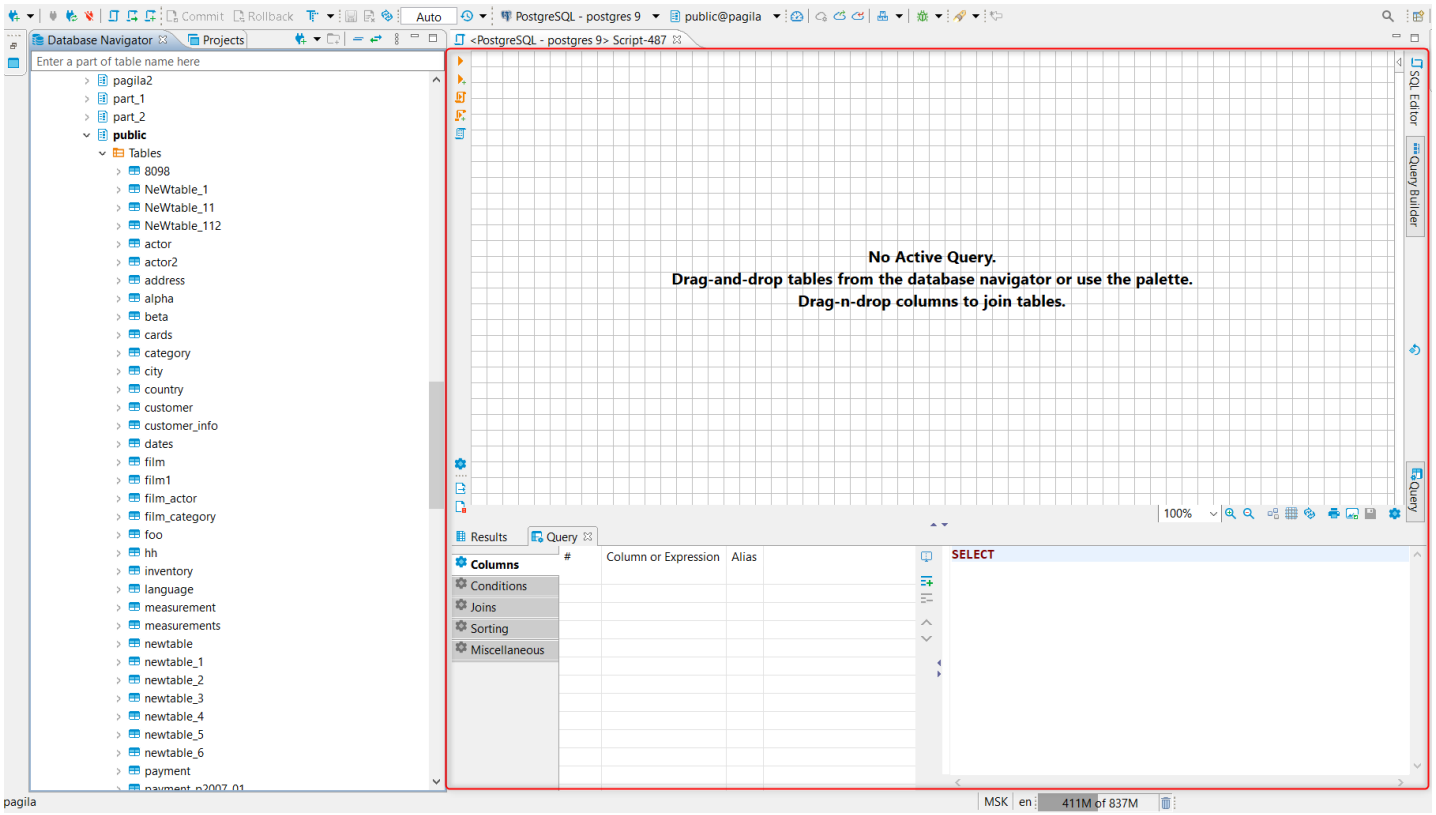
Query Builder is a user-friendly visualization tool that will help you make sense of your complex database designs. It can be useful when you need to understand the various relationships between different tables. Also, it can be helpful for those who are not very familiar with SQL scripting or if you do not want to insert script commands manually. The tool creates SQL scripts automatically based on the visual schema you have created.

Opening Visual Query Builder

To open **Visual Query Builder** click the **Open Query Builder** button in the SQL Editor tool bar.

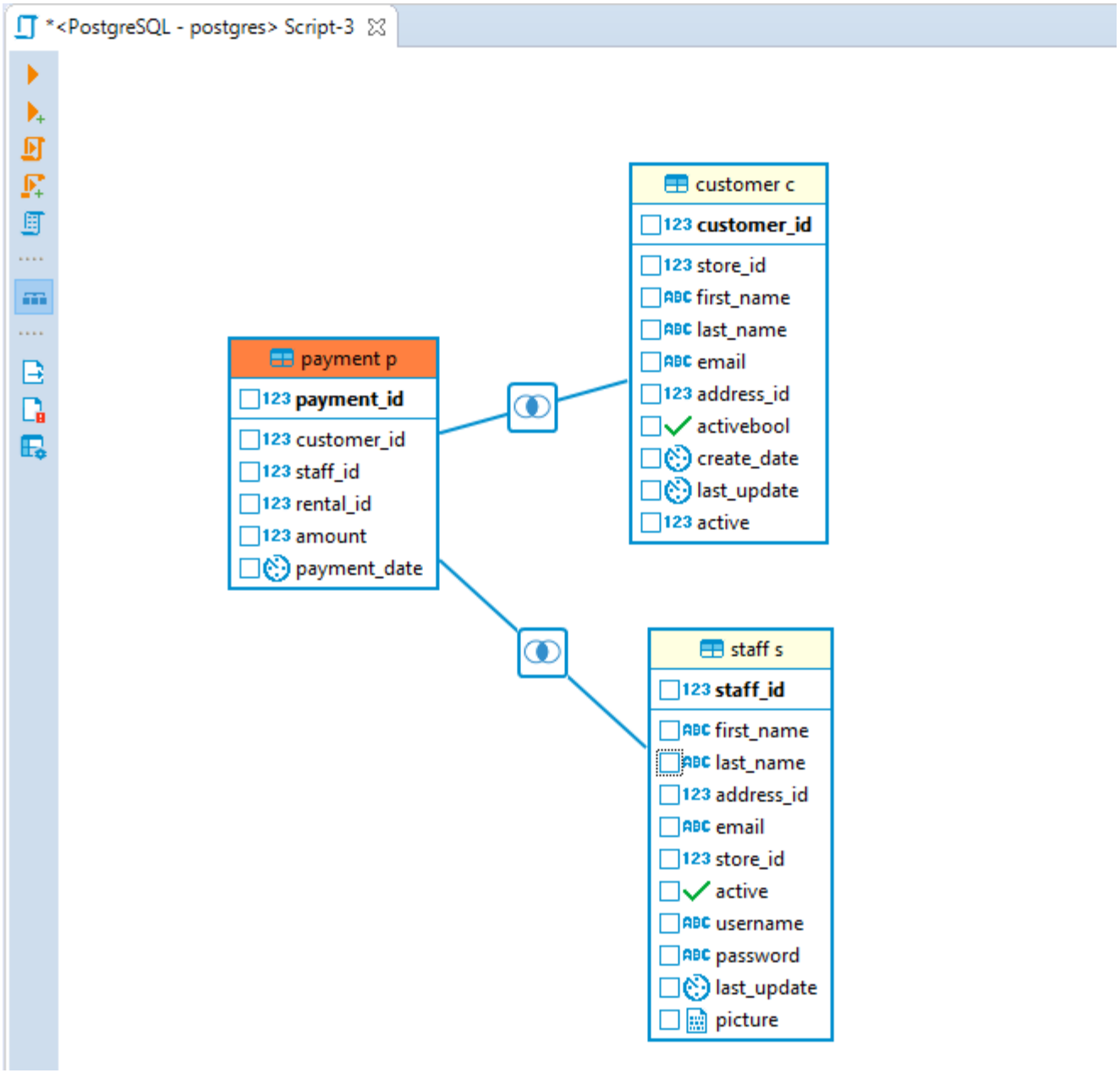


The **Visual Query Builder** will appear on the right.

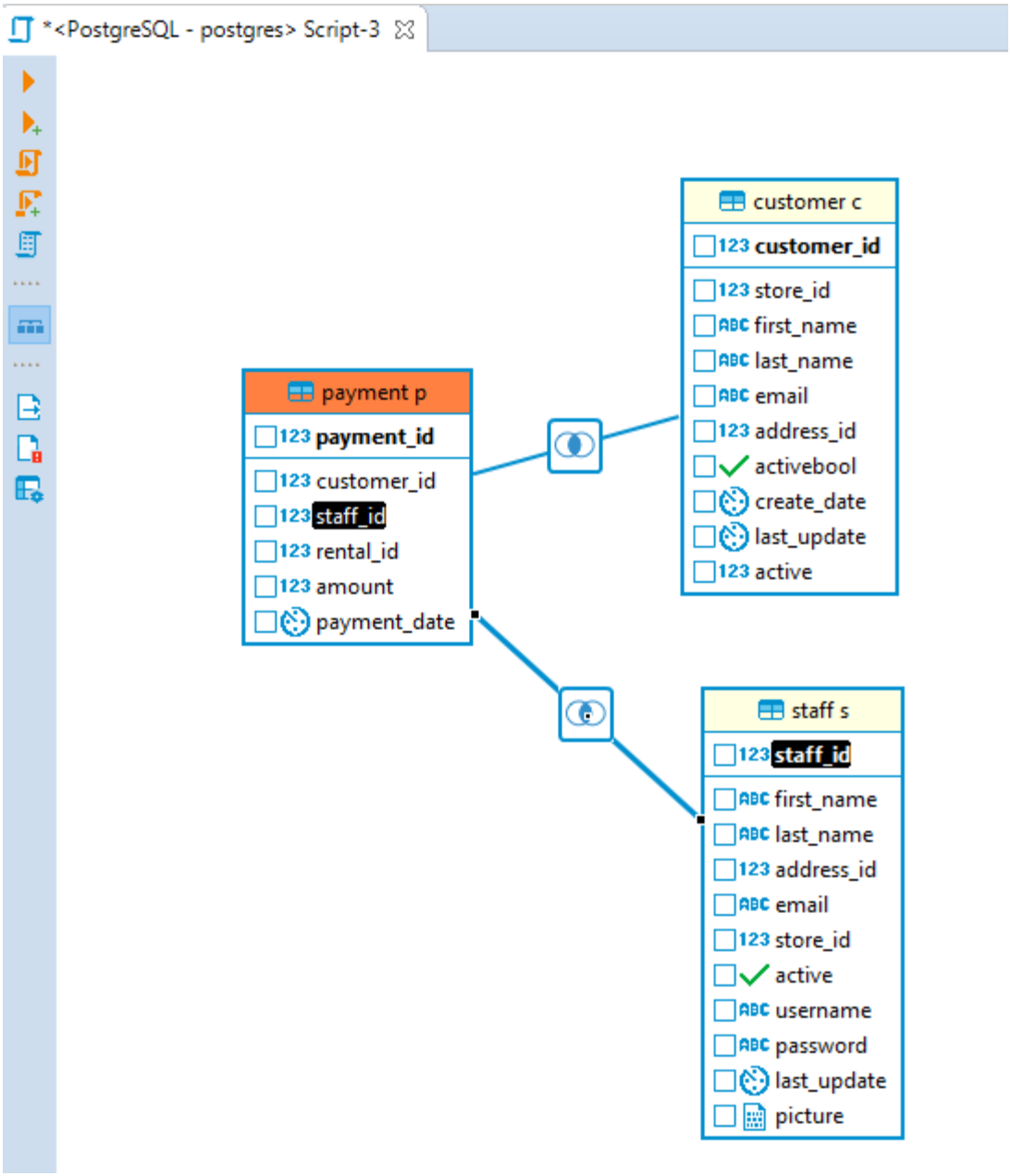


Creating Visual Query

Start creating a query by selecting a query data source: drag-and-drop tables you want to work with from the **Database Navigator** pane into the **Visual Query Builder** area. All the connections existing between the tables will be shown automatically.



To create a new join between the tables, press the left mouse button when the cursor is over the column of one table, holding the right mouse button drag the cursor to the column of another table and release the right mouse button. The connection between the selected columns of the tables will be created visually and in the SQL script a new join will be added.



To remove a join between the tables, click on it. The connection will be highlighted. Then, press `Delete` or use the **Delete** option in the context menu. The visual connection will be removed and the corresponding join will be automatically removed from the SQL script area.

To build a SELECT query you need to select columns in the tables you added. To select a column, click the check-box next to its name - the column will be added to the **Columns** tab of the **Query Settings Editor** and SELECT query will be added to the SQL script area automatically.

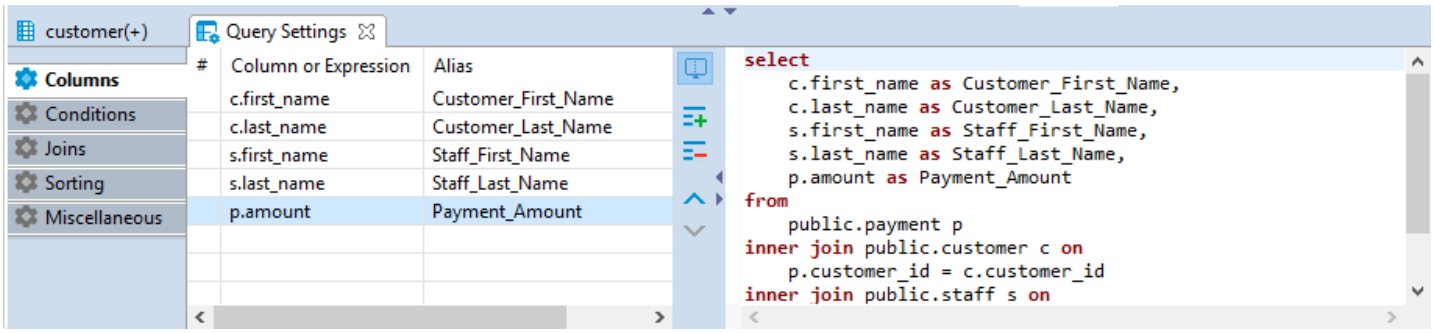
Adjusting Query Settings

Visual Query Builder also allows you to set the query conditions and adjust the representation of query results by means of **Query Settings Editor**.

To open **Query Settings Editor** use **Visual builder query settings** button in the vertical tool bar on the left.

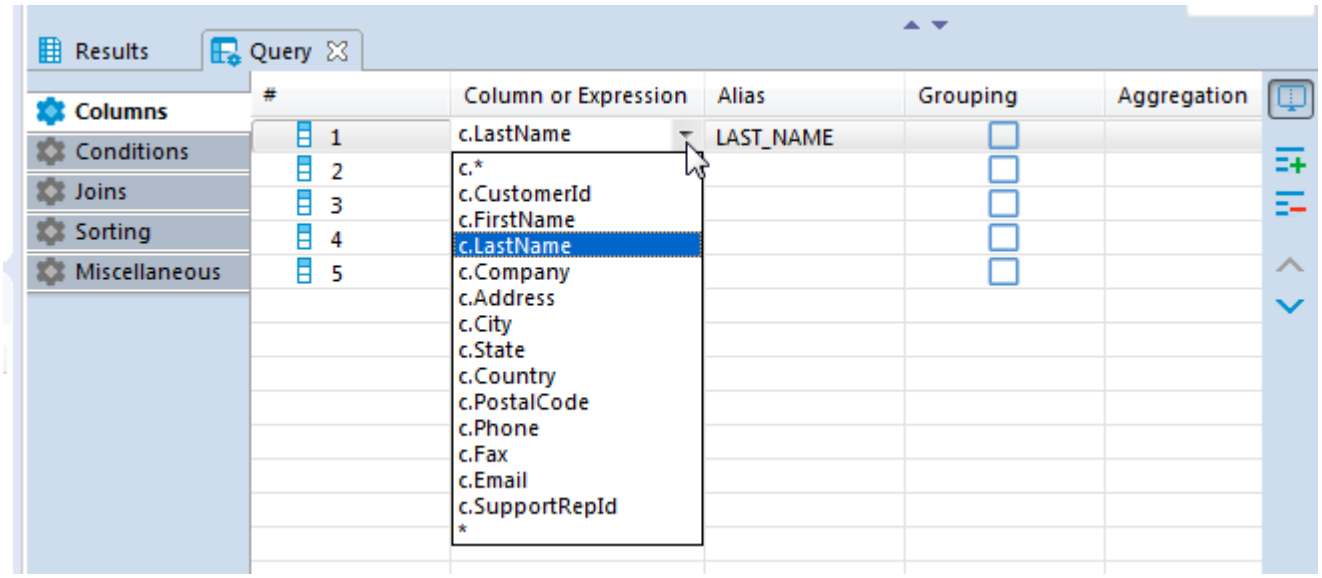
Query Settings window contains five tabs described below.

Columns



Columns tab of the **Query Settings Editor** contains all the columns you added by selecting column names in **Visual Builder** main window. In this tab you can add and remove columns using **Add** and **Remove** buttons correspondingly.

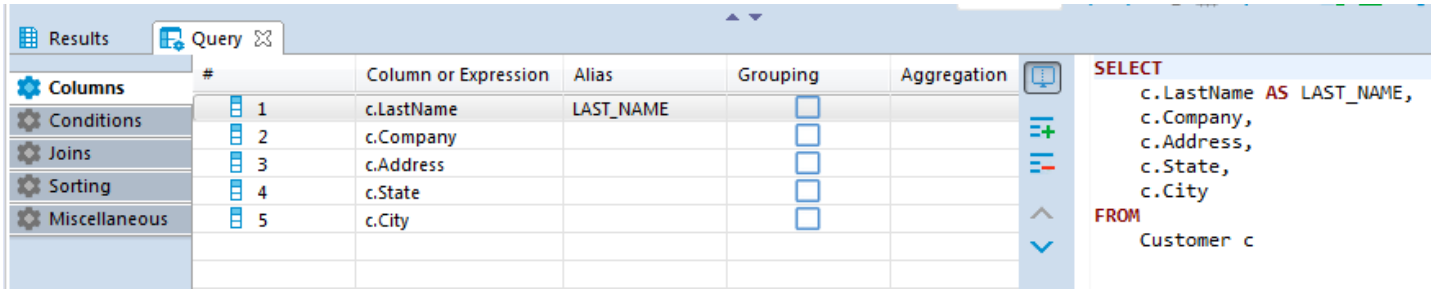
To add a column, press **Add** button and a new instance will be added to the table. Click on the first cell in **Column or Expression** column and select a column from the list of available columns displayed in the dropdown list appeared.



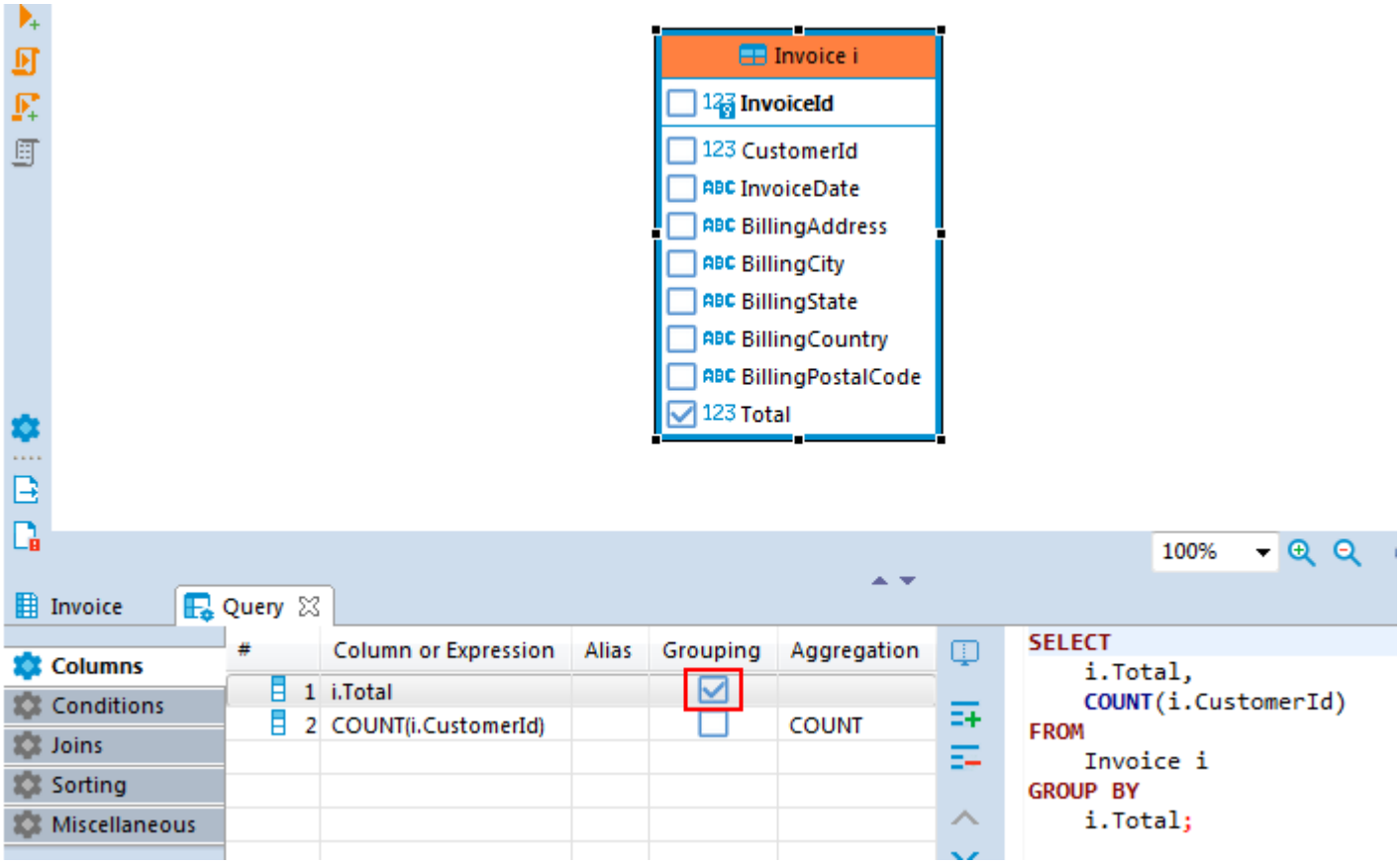
To remove a column, click on the row containing its name and press the **Remove** button on the right.

To change the display order of columns in the result table use **Move Up/Down** buttons .

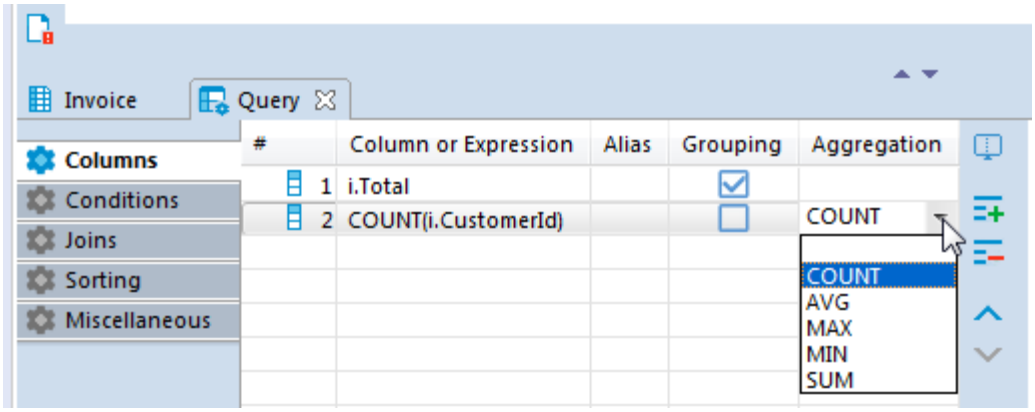
You can also define a user-friendly name of the column to be displayed in the result table. To set a user-friendly name, click on a cell in **Alias** column and insert the name. The change will be immediately displayed in the SQL script area.



If you want a **grouping** condition to be added to your expression, you can click on the checkbox in the column row. The expression will update automatically. The other previously selected columns will become aggregate. If there are no other columns, then the expression COUNT(*) will be automatically added.



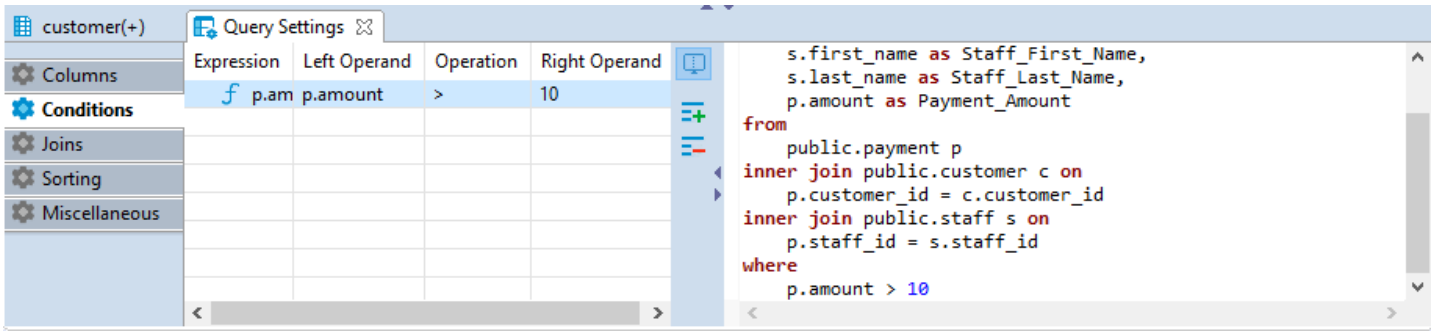
You can select other aggregation functions from the drop-down list. Or enter your own version in the cell.



When removing columns from the list, they will be removed from the grouping expression as well. When adding new columns to the list, it is added to the grouping expression.

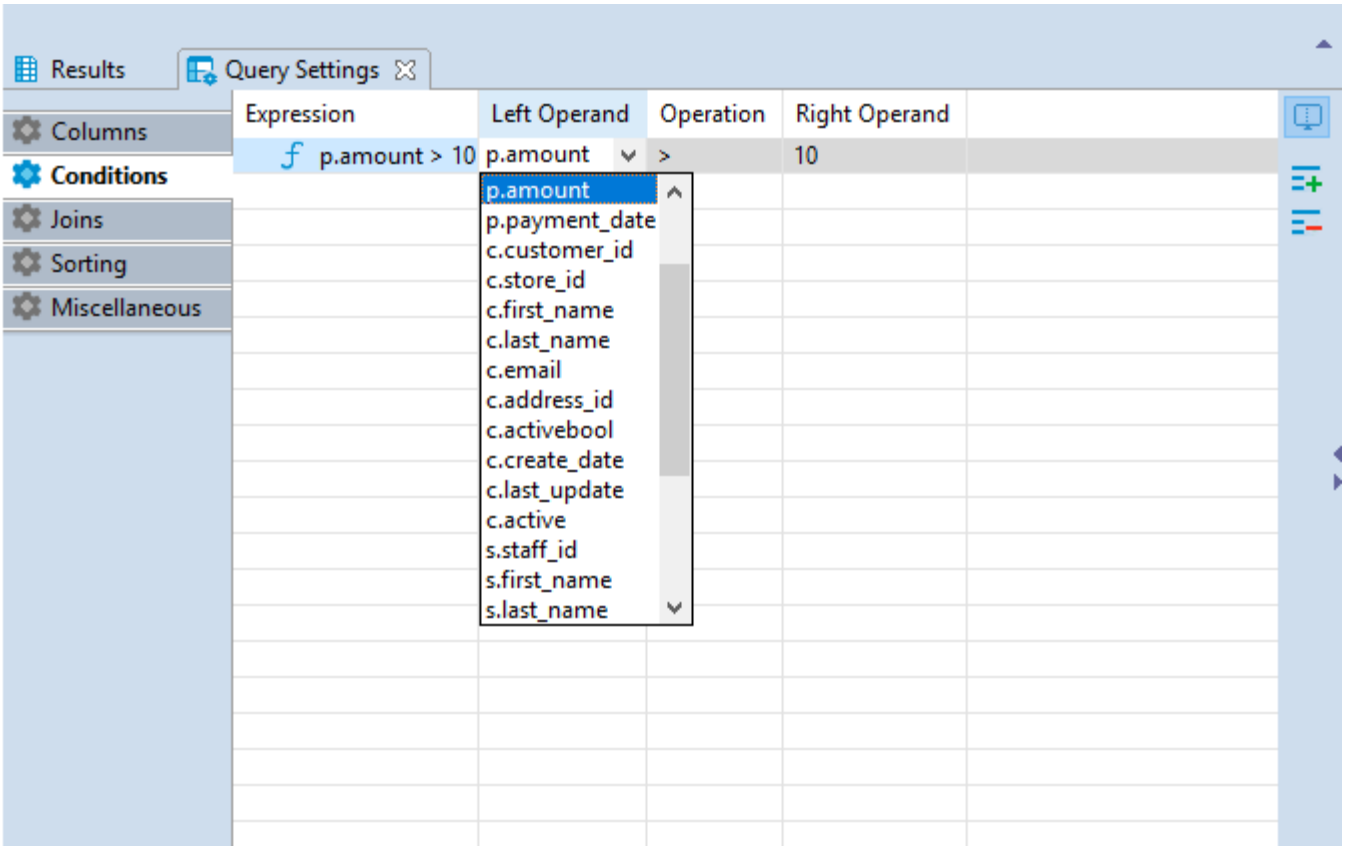
Conditions

The **Conditions** tab is used for managing query conditional expressions.

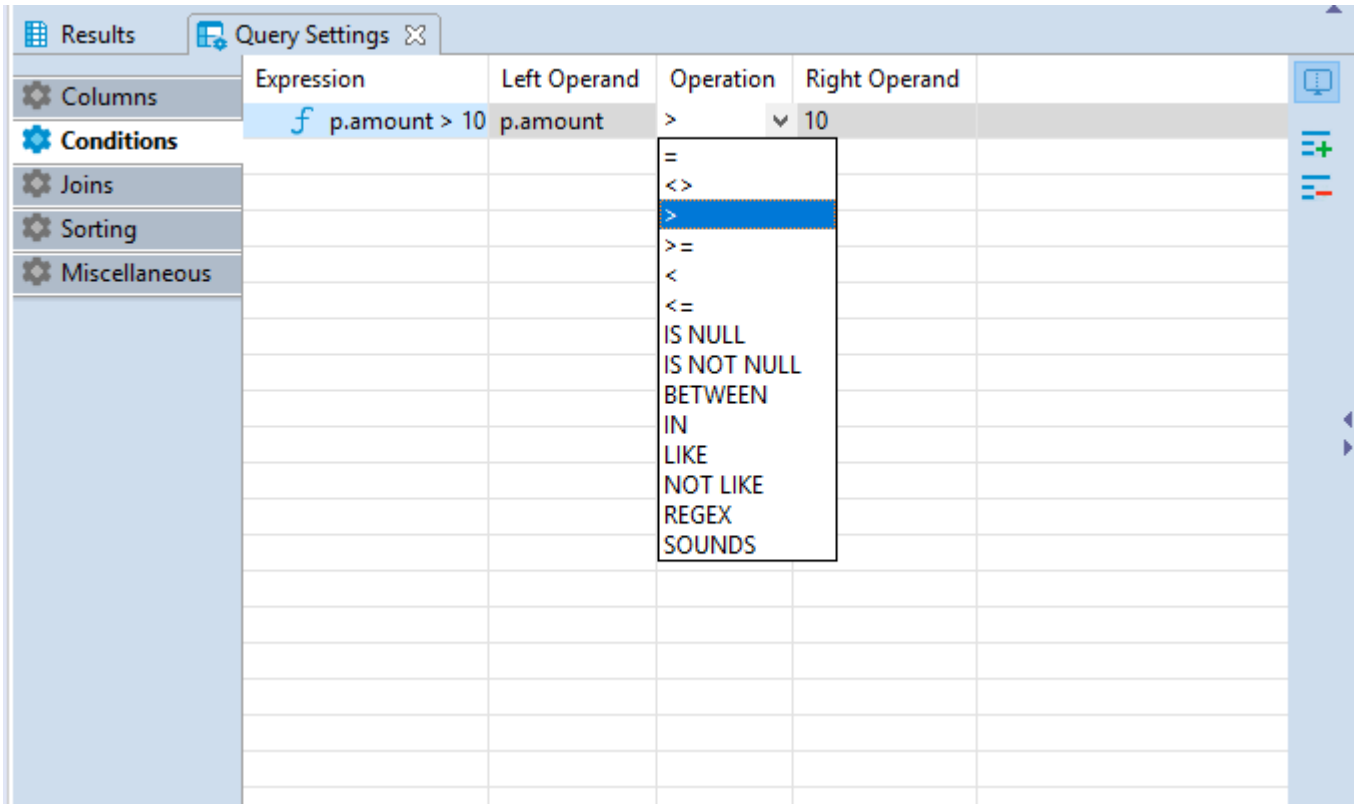


To add a new conditional expression, use the **Add** button on the right - a new instance will be added and the default conditional expression WHERE will be added to the SQL script area automatically. This default conditional expression can then be adjusted to the one you need:

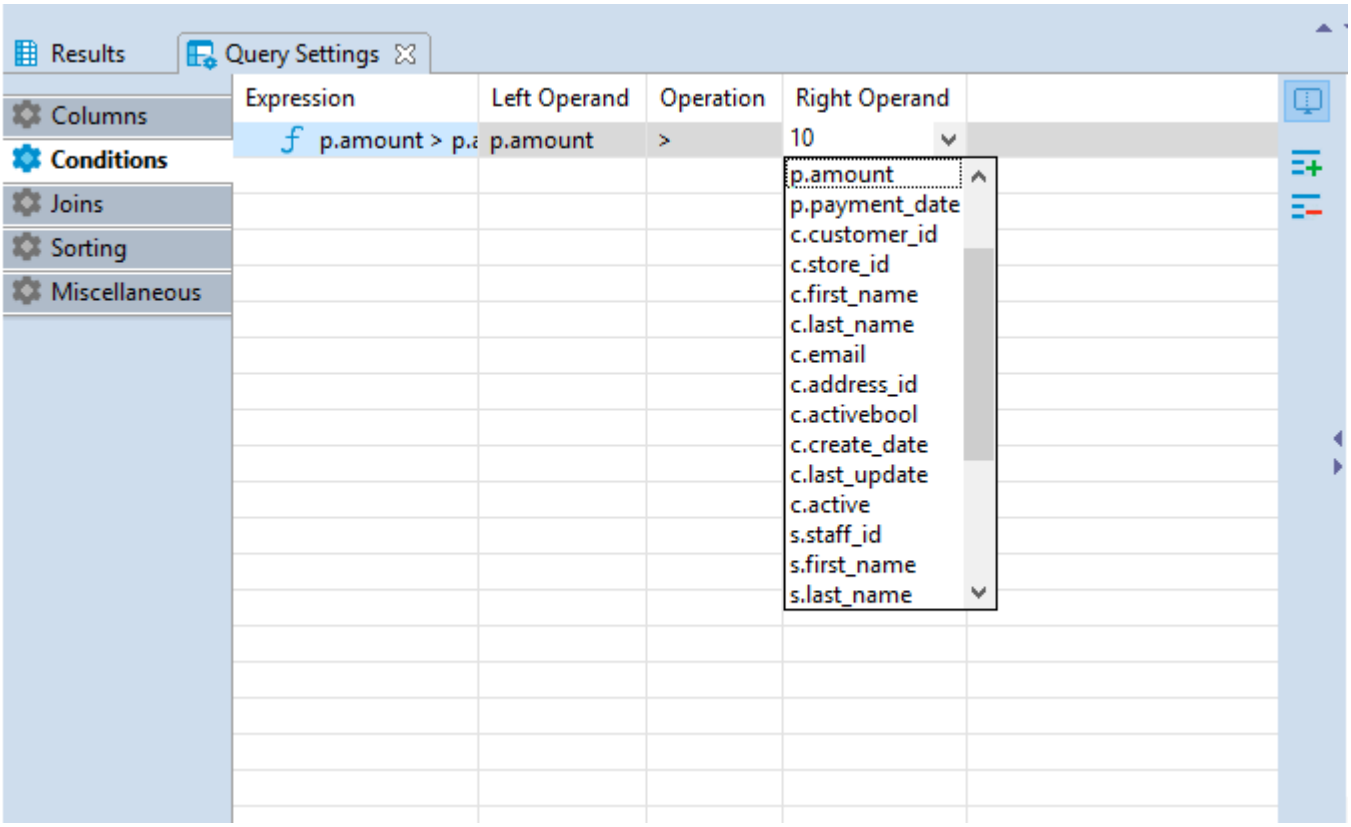
- **Left Operand** setting defines the left operand of the conditional expression. To set the left operand, click the cell in the **Left Operand** column and a drop down list of all available columns will be displayed. Select a column you want to use as the left operand in your conditional expression or insert a digit.



- **Operation** setting defines the comparison rule between the left and the right operands of the conditional expression. To set a comparison rule, click the cell in the **Operation** column and select the rule you need from the drop down list which will appear.



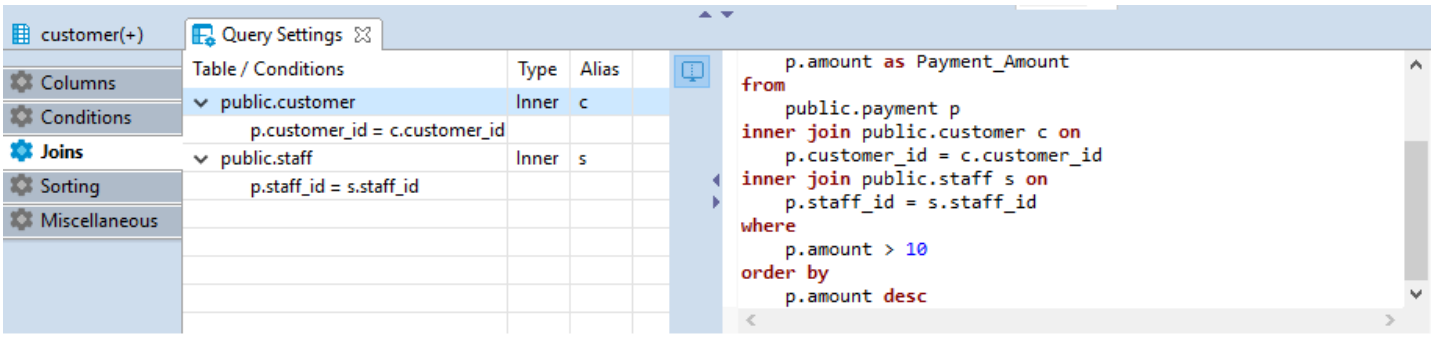
- **Right Operand** setting defines the right operand of the conditional expression. To set the right operand, click the cell in the **Right Operand** column and a drop down list of all available columns will be displayed. Select a column you want to use as the left operand in your conditional expression or insert a digit.



To remove a conditional expression, click on the row containing the expression and press the **Remove** button on the right.

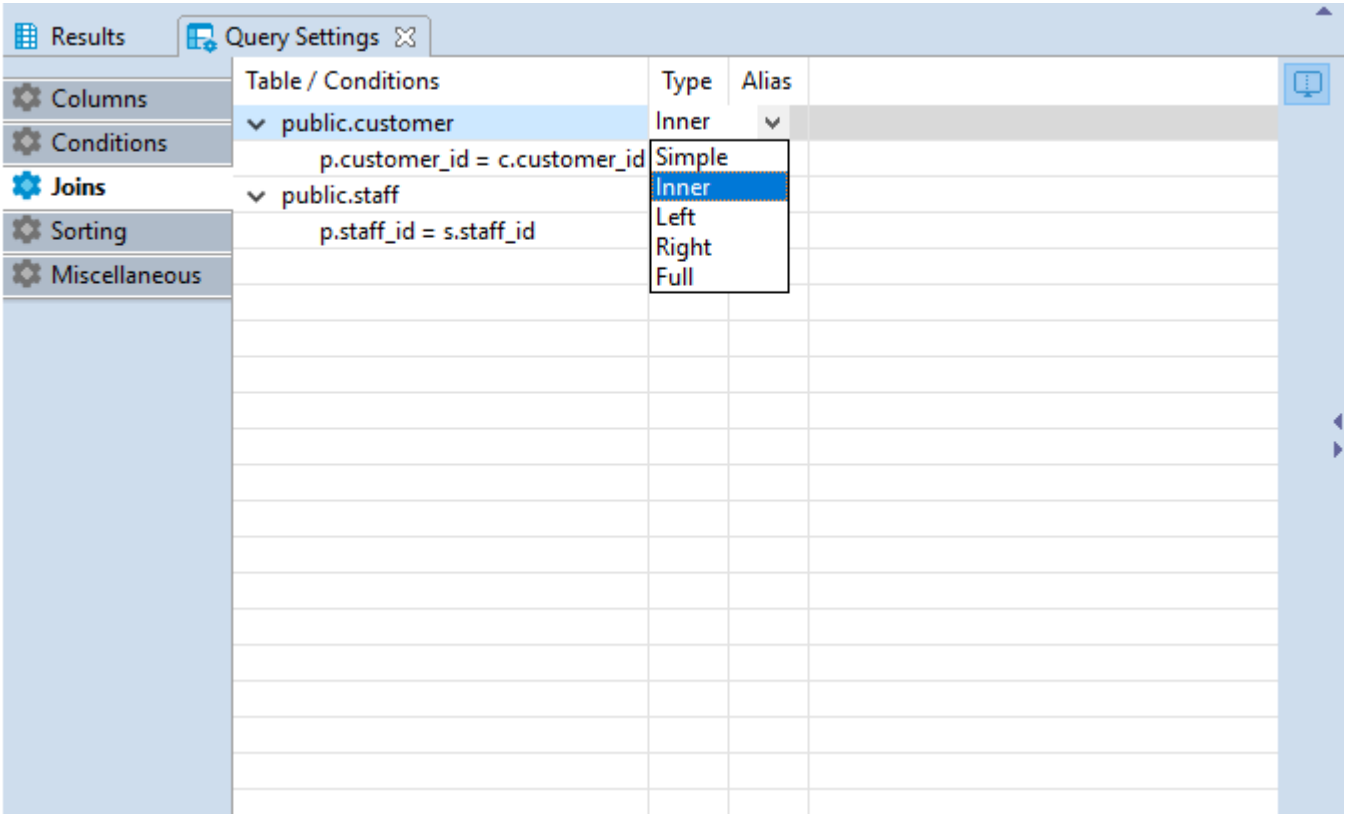
Joins

All the joins existing between the tables in **Visual Query Builder** main window are displayed in the **Joins** tab of **Query Settings Editor**.

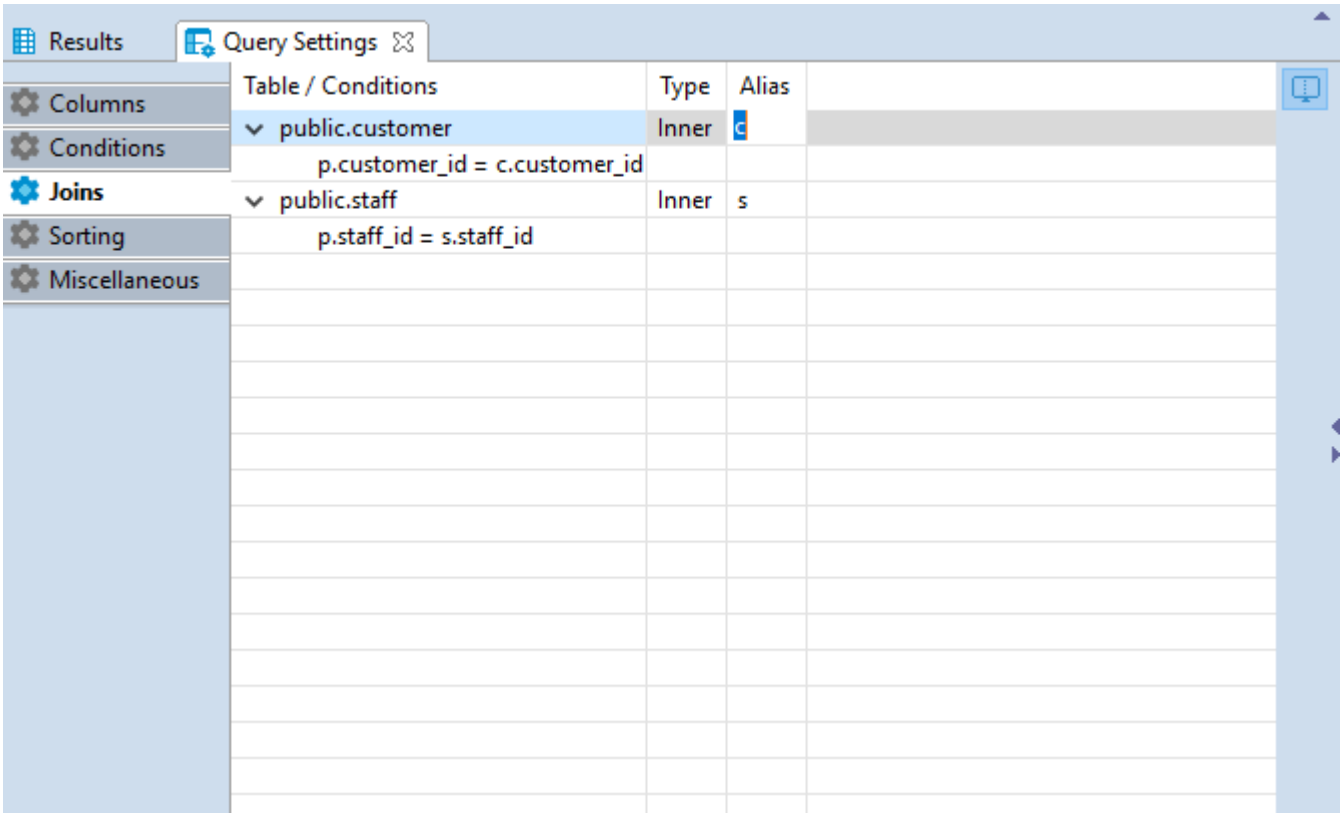


Joins cannot be added or removed by means of **Query Settings Editor**, however, the following join settings can be adjusted here:

- **Type** - defines the type of the join. Click the cell in the **Type** column - a drop down with available join types will be displayed. Select the required option from the list by clicking on it.

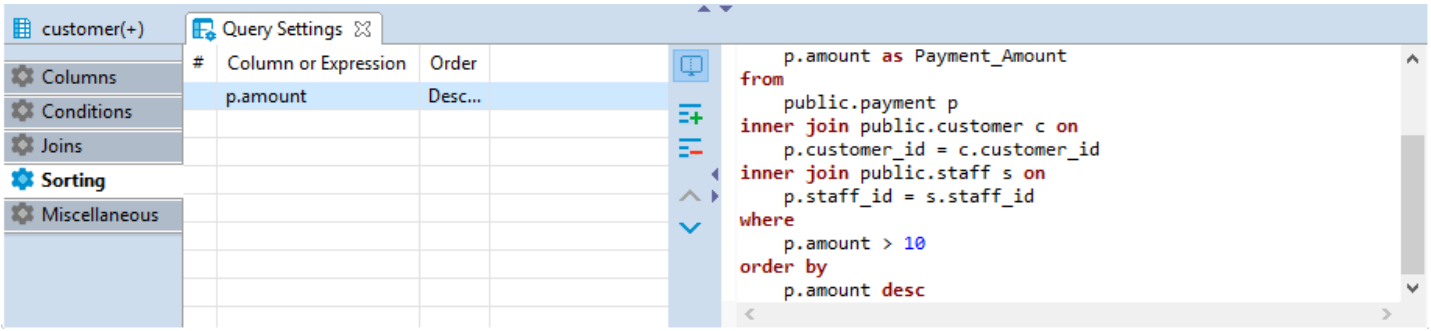


- **Alias** - defines a user friendly name of the join. To define this setting click on the cell in the **Alias** column and input the name.



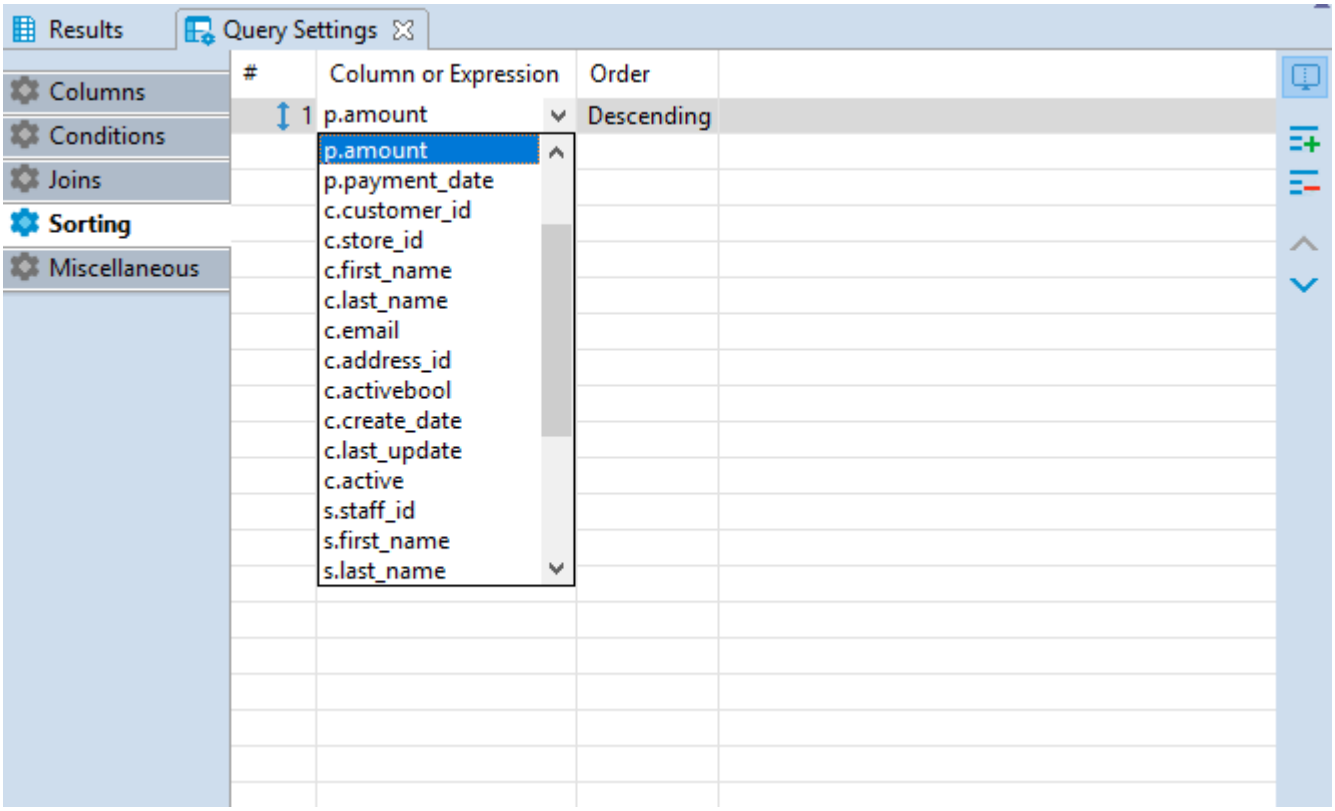
Sorting

In the **Sorting** tab you can set the order of rows in the result table.

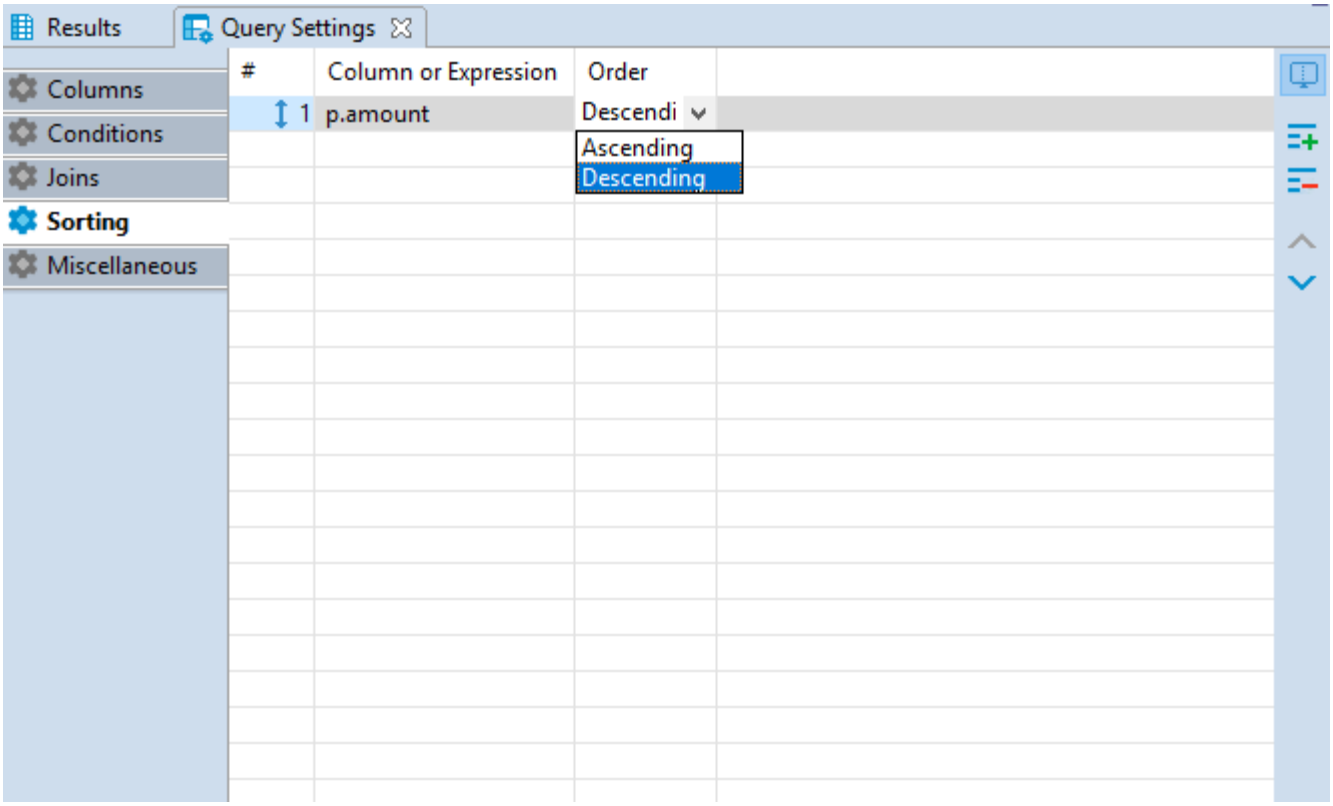


To add a new sorting condition press the **Add** button  on the right and the default conditional expression ORDER BY will be added to the SQL script area automatically. This default conditional expression can then be adjusted to the one you need:

- Once a new condition is added, click the first cell in **Conditions or Expressions** column and a drop down list of all available columns will appear. Select the required column by clicking on its name.



- In the **Order** column you can define whether the rows of the selected column should be sorted in ascending or descending order in the result table. To set the order, click the cell in **Order** column and select the required option from.The order by command will be added to the script.

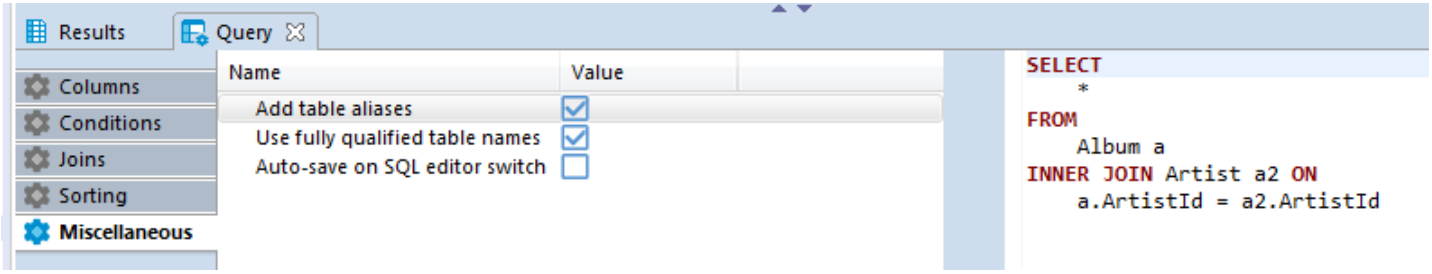


To remove a condition use the **Remove** button  on the right.

Miscellaneous

In the **Miscellaneous** tab it is possible to:

- Enable or disable the automatic generation of aliases for tables by selecting the **Add table aliases** check-box.
- Disable auto-completion for table names by selecting the **Use fully qualified table names** check-box.
- Enable Autosave on SQL-editor switch by selecting the **Autosave on SQL-editor switch** check-box.



Executing Visual Query

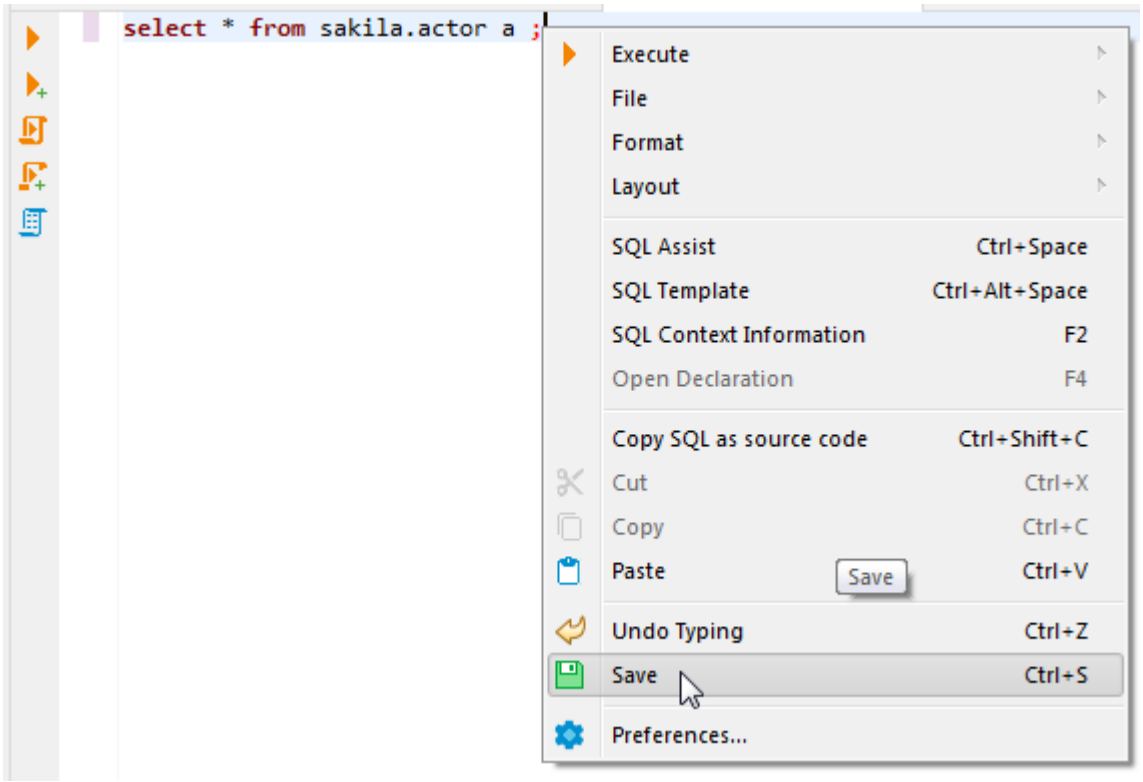
To execute a query, use the **Execute SQL statement** button  to get the results in the same tab or **Execute SQL statement in new tab** button  to get the results in a new tab. Both buttons are located in the **Visual Query Builder** vertical toolbar.

Script Management

Saving Scripts

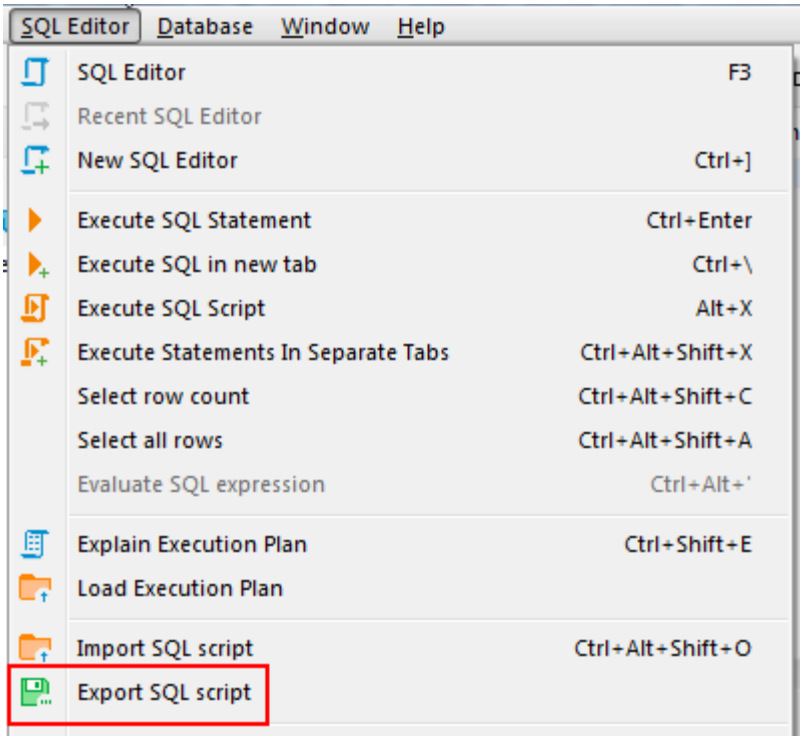
You can save scripts to a predefined space in the currently active project or somewhere in the file system.

To save a script to the current project space, just press `Ctrl+S` or right-click the script and click **Save** on the context menu:

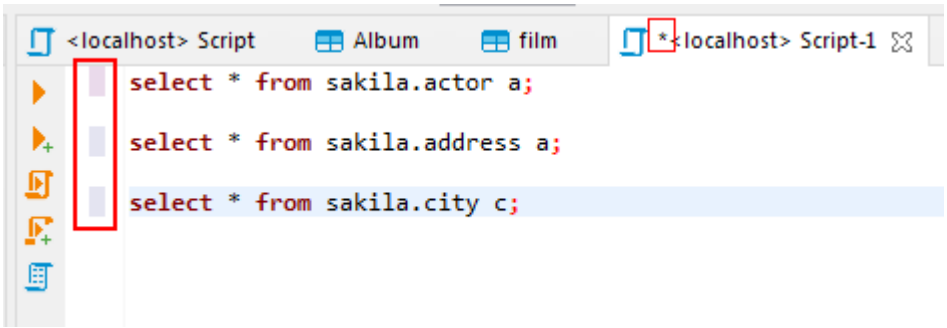


You can find the script saved this way in the Project Explorer view in the **Scripts** folder.

To save a script to the file system, right-click the script, click **File -> Export SQL script** on the context menu and then select the folder in the file system. You can also click **SQL Editor -> Export SQL script** on the main menu:

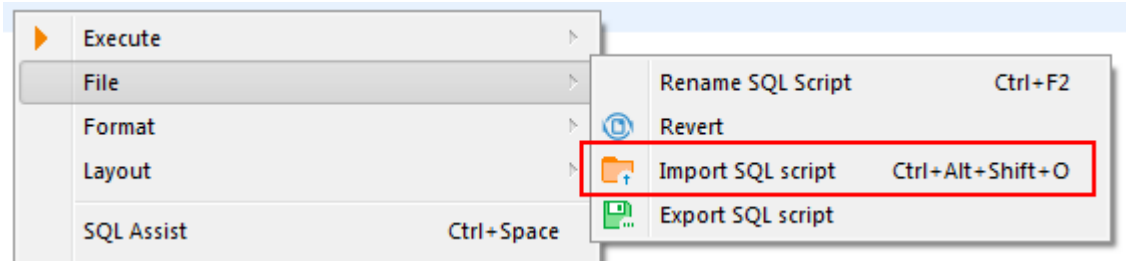


Unsaved data is highlighted in color on the left side of the editor, in addition to having an asterisk in the name of the script.



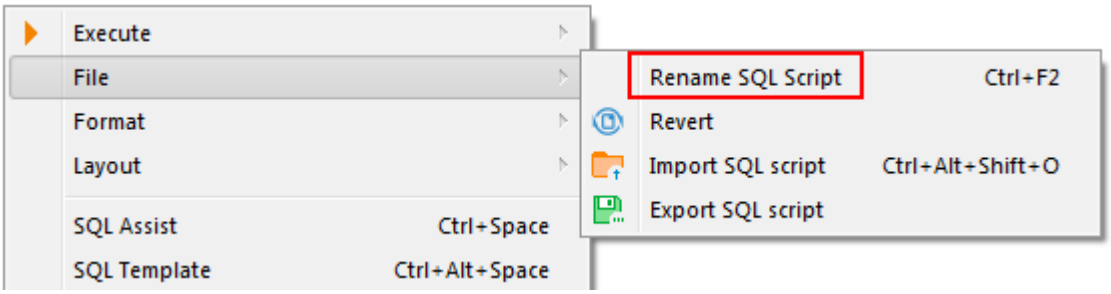
Loading Scripts

To load a script stored in the file system to the SQL Editor, press `CTRL+SHIFT+O` , or click **SQL Editor -> Import SQL script** on the main menu, or right-click the script panel and click **File -> Import SQL script** on the context menu:

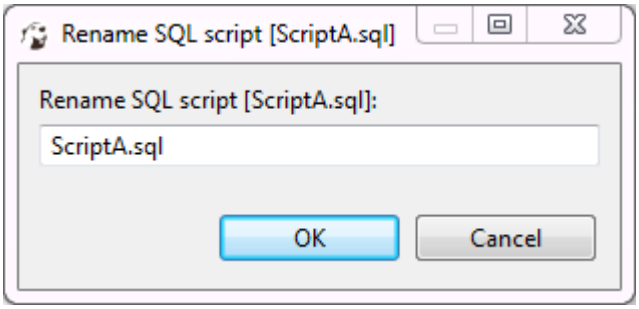


Renaming Scripts

To rename a script, right-click anywhere in the script panel, click **File -> Rename SQL Script** on the context menu or press `CTRL+F2` :



Then enter the new name in the Rename SQL script dialog box and click **OK**:

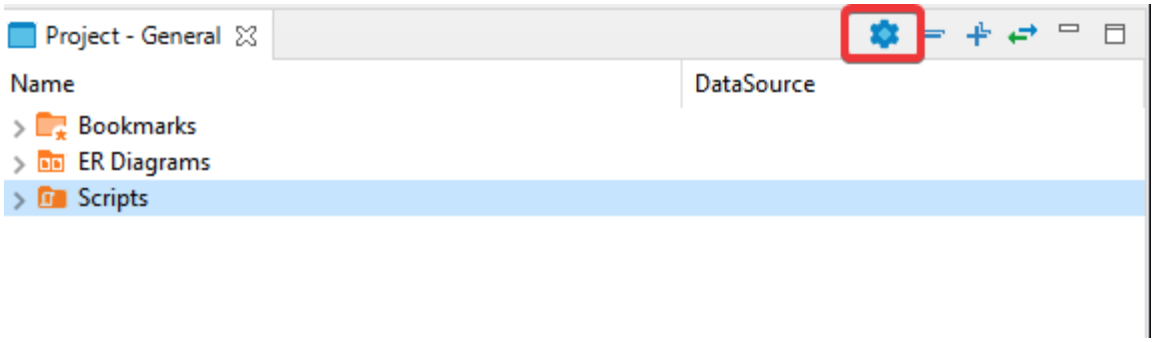


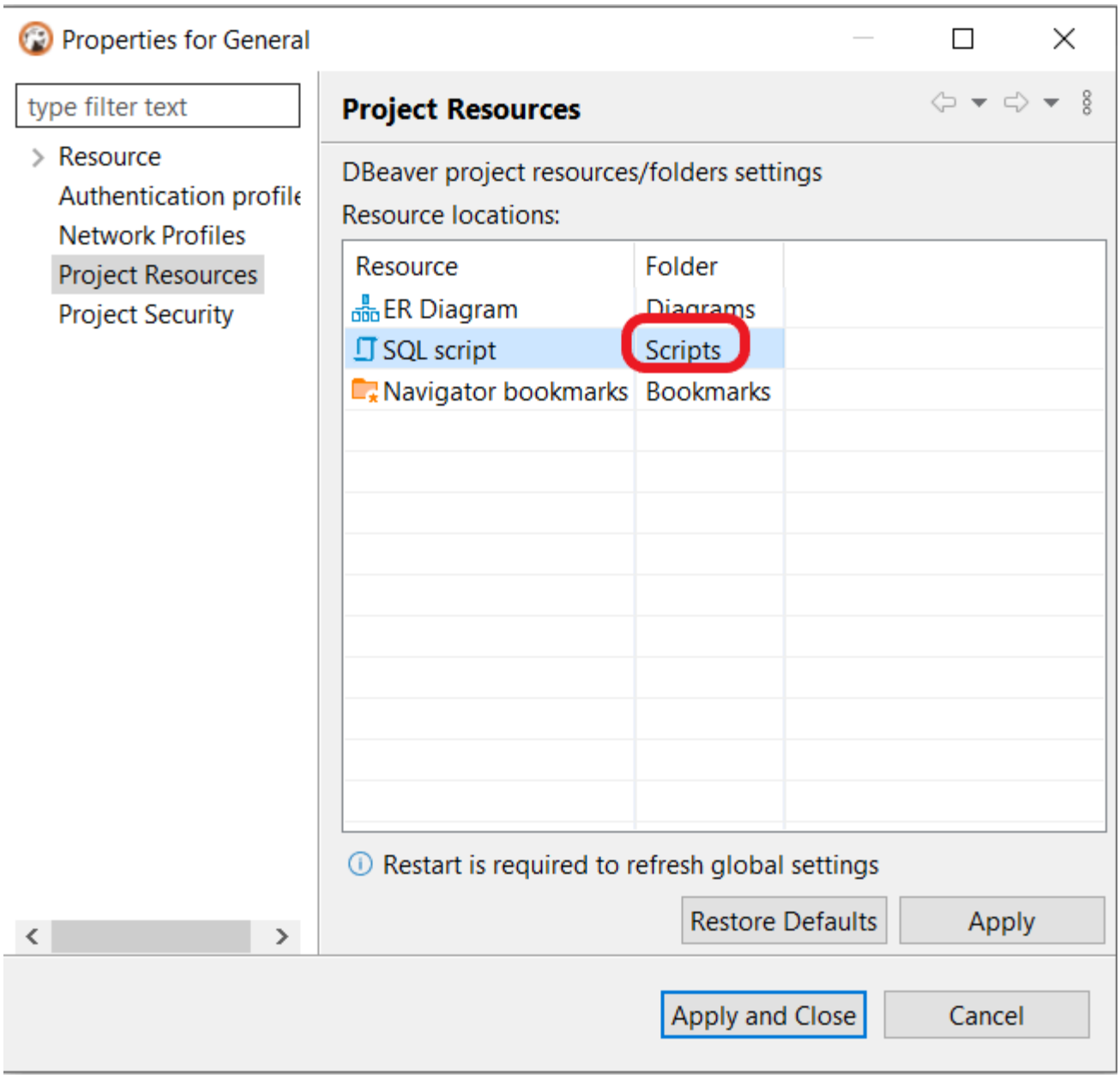
Reverting Changes

If you want to revert all changes made to the current SQL script and return it to its initial state (reload from disk), right-click anywhere in the script panel and click **File -> Revert** on the context menu.

Changing default scripts directory

By default all of the scripts are saved to a "Scripts" folder located in your project inside the workspace directory. This can be changed by clicking the **Configure** button in Project Explorer view. There you can click on a folder's name an pick any other folder inside the Project.





Adding external directory

You can also link an external directory to your project to either save your scripts into it, or to access scripts that were created outside of DBeaver.

To link an External directory right-click anywhere in the Project Explorer and pick **Create -> Link Folder**. There you

can link any directory on your drive to a project. This will allow you to open any externally created scripts through Project explorer and to set this folder as default to save new scripts into.

SQL Console

In some cases you might want to execute a query and not save it in a script. For example, when you read table data using "Read data in SQL console" or open procedure/function source from DDL editor. SQL console does not have an associated .sql file. Its contents will be lost when you close it.

Client Side Commands

You can use special commands in the SQL scripts.

These commands are executed on DBeaver's side, not on the server-side.

DBeaver supports the following commands:

Command	Database	Description
`@set var = value`	All	Sets a script variable. You can use expressions as a value. Variables can be used as SQL queries input parameters. For more information see Dynamic parameter bindings
`@unset var`	All	Unsets a script variable.
`@echo message`	All	Prints message to output log. You can use a macro in a message (for example `${var}`).
`@include fileName`	All	- Executes a specified file name, - Can be used in scripts, - Opens a new SQL console with the specified file and processes SQL queries as in a regular SQL editor.
`@export { ... }`	All	Opens the data transfer wizard with predefined settings. For more information see the main article.
`source fileName`	MySQL	The same as `@include` but in MySQL CLI syntax
`define var = value`	Exasol	The same as `@set` but in Exasol EXAPlus syntax.

Export Command

The `@export` command allows you to open the data transfer wizard with prefilled settings.

It may be helpful in case you're editing several SQL queries and want to quickly perform the export of the produced results without creating any additional data transfer tasks.

Disclaimer: This article describes supported settings by the `@export` command, their purpose, and allowed values. Generally, this article contains every setting accessible in the data transfer wizard. Settings are written in the order they appear in the wizard, so you can always look at the wizard to quickly locate any of these settings.

Usage

The body of the command consists of JSON text, which looks like this:

```
{
    "type": <ID of the processor>,
    "producer": {
        <producer settings>
    },
    "consumer": {
        <consumer settings>
    },
    "processor": {
        <processor-specific settings>
    },
}
```

Due to certain limitations, it must be written on a single line, without line delimiters:

```
@export { "type": "csv", "producer": { ... }, "consumer": { ... }, "processor": { ... } }
```

The command itself doesn't do anything. It must be followed by any other query:

```
@export { "type": "csv", "producer": { ... }, "consumer": { ... }, "processor": { ... } }
SELECT * FROM Album;
```

You can either execute each line separately, or execute the entire script at once.

Settings

Here's the description of each attribute:

Attribute	Description
`type`	Type of the processor.
`producer`	Settings that affect how the data is extracted. See the full table of supported settings in the main section.
`consumer`	Settings that affect how the data is transformed before processing. See the full table of supported settings in the main section.
`processor`	Settings that affect how the data is processed. This includes formatting, transformations, etc. These settings are specific to the processor specified by the `type` attribute. See the full table of supported processors in the main section.

Producer Settings

Id	Name	Description	Type	Default Value	Allowed Values
`extractType`	Extract type	Data extraction mode. Denotes whether a single query or multiple segmented queries should be used to extract data.	String	`SINGLE_QUERY`	`SINGLE_QUERY`, `SEGMENTS`
`segmentSize`	Segment size	Specifies how many rows are read per segment during data extraction. See `extractType`	Integer	`100000`	*Any*
`fetchSize`	Fetch size	Number of rows to fetch per one server round trip. May greatly affect extraction performance.	Integer	`10000`	*Any*
`openNewConnections`	Open new connection (s)	Open new physical connection for data reading. Makes great sense if you are going to continue to work with your database during the export process.	Boolean	`true`	*Any*

`queryRowCount`	Select row count	Query row count before performing export. This will let you track export progress but may cause performance faults in some cases.	Boolean	`true`	*Any*

Consumer Settings

Id	Name	Description	Type	Default Value	Allowed Values
`formatterProfile`	Formatting Profile	Specifies the profile used for formatting data.	String		*Any*
`valueFormat`	Value Formatting	Specifies how the data is interpreted.	String	`UI`	`UI`, `EDIT`, `NATIVE`
`lobExtractType`	Binaries Policy	Specifies how binaries are processed.	String	`INLINE`	`SKIP`, `FILES`, `INLINE`
`lobEncoding`	Binaries Encoding	Specifies how binaries are encoded.	String	`BINARY`	`BASE64`, `HEX`, `BINARY`, `NATIVE`
`outputClipboard`	Copy to Clipboard	Specifies that the data should be copied to the clipboard rather written to files on a disk.	Boolean	`false`	*Any*
`outputFolder`	Output Directory	Output directory pattern. Specifies there the output files should be located.	String	*N/A*	*Any*
`outputFilePattern`	Output Filename	Output filename pattern.	String	`${table}_${timestamp}`	*Any*

`outputEncoding`	Output Encoding	Specifies the file encoding.	String	`UTF-8`	*Any*
`outputEncodingBOM`	Insert BOM	Specifies whether the byte order mark should be written to the output file. Common for encoding such as `UTF-16LE`, `UTF-16BE`, `UTF-32LE`, and `UTF-32LE`.	Boolean	`false`	*Any*
`outputTimestampPattern`	Timestamp Pattern	Pattern used for the `${timestamp}` variable in `outputFolder` and `outputFilePattern`.	String	`yyyyMMddHHmm`	*Any*
`appendToFile`	Append to the end of the file	If file already exists, appends data at end of it. Only works against compatible processors.	Boolean	`false`	*Any*
`useSingleFile`	Write to the single file	Write all streams to the single file. Only works against compatible processors.	Boolean	`false`	*Any*
`compressResults`	Compress	Specifies whether the output file should be compressed using ZIP.	Boolean	`false`	*Any*
`splitOutFiles`	Split output file	Specifies whether the output file should be split using the `maxOutFileSize` threshold. If size exceeds this threshold, a separate file is created and so on.	Boolean	`false`	*Any*

`maxOutFileSize`	Maximum file size	Maximum size of a single file. See `splitOutFiles`	Integer	`10000000`	*Any*

Processor Settings

CSV (`csv`)

Id	Name	Description	Type	Default Value	Allowed Values
`extension`	File extension		String	`csv`	*Any*
`delimiter`	Delimiter	Column delimiter. You can use special characters \ + t,n,r	String	`,`	*Any*
`rowDelimiter`	Row delimiter	Row delimiter. Default is system-specific line feed delimiter. You can use special characters \ + t,n,r	String	`default`	`default` , `\n` , `\r` , `\r\n` , `\n\r`
`header`	Header	CSV header settings	String	`top`	`none` , `top` , `bottom`
`headerFormat`	Header format	Header format	String	`label`	`label` , `description` , `both`
`escape`	Characters escape	Bad characters escaping model (surrounded with quotes or escaped with '\' character)	String	`quotes`	`quotes` , `escape`
`quoteChar`	Quote character	Character which will be used to quote strings (space means no quote)	String	`"`	*Any*
`quoteAlways`	Quote always	Quote all cell values. Cannot be used with "quoteNever"	String	`disabled`	`disabled` , `all` , `strings` , `all but numbers` , `all but nulls`
	Quote	Do not quote cell values. Cannot be			

`quoteNever`	never	used with "quoteAlways"	Boolean	`false`	*Any*
`nullString`	NULL string	String which will be used instead of NULL values	String		*Any*
`formatNumbers`	Format numbers	Format numeric values using locale settings	Boolean	`false`	*Any*

DbUnit (`dbunit`)

Id	Name	Description	Type	Default Value	Allowed Values
`upperCaseTableName`	Force upper case table name		Boolean	`true`	*Any*
`upperCaseColumnNames`	Force upper case column names		Boolean	`true`	*Any*
`extension`	File extension		String	`xml`	*Any*
`includeNullValues`	Include NULL values in export		Boolean	`true`	*Any*
`nullValueString`	Replace NULL values with		String	`[NULL]`	*Any*

HTML (`html`)

Id	Name	Description	Type	Default Value	Allowed Values
`extension`	File extension		String	`html`	*Any*
`tableHeader`	Output table header	Output query or table name as first row in generated table	Boolean	`true`	*Any*
`columnHeaders`	Output column headers	Output column names as extra row in generated table	Boolean	`true`	*Any*
`extractImages`	Images	Extract images to graphic files	Boolean	`true`	*Any*

JSON (`json`)

Id	Name	Description	Type	Default Value	Allowed Values
`printTableName`	Print table name		Boolean	`true`	*Any*
`formatDateISO`	Format dates in ISO 8601		Boolean	`true`	*Any*

`extension`	File extension		String	`json`	*Any*

Markdown (`markdown.table`)

Id	Name	Description	Type	Default Value	Allowed Values
`extension`	File extension		String	`md`	*Any*
`nullString`	NULL string	String which will be used instead of NULL values	String		*Any*
`formatNumbers`	Format numbers	Format numeric values using locale settings	Boolean	`false`	*Any*
`showHeaderSeparator`	Show header separator	Print header separator (---). Required for GitHub markdown.	Boolean	`true`	*Any*
`confluenceFormat`	Confluence format	Use Confluence format (special format of header and no separator line)	Boolean	`false`	*Any*

SQL (`sql`)

Id	Name	Description	Type	Default Value	Allowed Values
`includeAutoGenerated`	Include generated columns	Include auto-generated columns (e.g. auto-increment) in SQL INSERT	Boolean	`false`	*Any*
`extension`	File extension		String	`sql`	*Any*
`nativeFormat`	Native date/time format	Use native date/time format in INSERT statements	Boolean	`true`	*Any*
`omitSchema`	Omit schema name	Omit schema/catalog name in INSERT statements	Boolean	`false`	*Any*
	Data rows				

`rowsInStatement`	per statement	Number of data rows per single insert statement	Integer	`10`	*Any*
`lineBeforeRows`	Insert line before rows	Insert line feed before values (for multi-row inserts)	Boolean	`true`	*Any*
`keywordCase`	Keyword case	You can choose lower or upper keyword case	String	`upper`	`upper` , `lower`
`identifierCase`	Identifier case	You can choose lower or upper keyword case for table and column names	String	`as is`	`as is` , `upper` , `lower`
`upsertKeyword`	Upsert keyword	You can choose different upsert keywords	String	`INSERT`	`INSERT` , `INSERT ALL` , `UPDATE OR` , `UPSERT INTO` , `REPLACE INTO` , `ON DUPLICATE KEY UPDATE` , `ON CONFLICT`
`insertOnConflict`	On conflict expression	Expression for the end of the statement. Enter the required value in this field. This is database specific setting	String		*Any*

Source code (`source.code`)

Id	Name	Description	Type	Default Value	Allowed Values
`language`	Language	Programming languages	String	`PHP < 5.4`	`PHP < 5.4` , `PHP 5.4+`
`formatDateISOPHP`	Format dates in ISO 8601		Boolean	`true`	*Any*
`extension`	File extension		String	`php`	*Any*
`quoteChar`	Quote character	Character which will be used to quote strings	String	`"`	`"` , `'`

`rowDelimiter`	Row delimiter	Row delimiter. Default is system-specific line feed delimiter. You can use special characters \ + t,n,r	String	`default`	`default` , `\n` , `\r` , `\r\n` , `\n\r`

TXT (`txt`)

Id	Name	Description	Type	Default Value	Allowed Values
`extension`	File extension		String	`txt`	*Any*
`batchSize`	Batch size		String	`200`	*Any*
`minColumnLength`	Min column length		String	`3`	*Any*
`maxColumnLength`	Max column length		String	`0`	*Any*
`showNulls`	Show NULLs		Boolean	`false`	*Any*
`delimHeader`	Show header delimiter		Boolean	`true`	*Any*
`delimLeading`	Show leading delimiter		Boolean	`true`	*Any*
`delimTrailing`	Show trailing delimiter		Boolean	`true`	*Any*
`delimBetween`	Show in-between delimiter		Boolean	`true`	*Any*

XML (`xml`)

Id	Name	Description	Type	Default Value	Allowed Values
`extension`	File extension		String	`xml`	*Any*

XLSX (`xlsx`)

Id	Name	Description	Type	Default Value	Allowed Values
`extension`	File extension		String	`xlsx`	*Any*
`rownumber`	Row number	Set row index as first column	Boolean	`false`	*Any*

	(s)				
`border`	Border style	Cell borders style	String	`THIN`	`NONE` , `THIN` , `THICK`
`nullString`	NULL string	String which will be used instead of NULL values	String		*Any*
`header`	Column names as header	Use column name as first row	Boolean	`true`	*Any*
`headerfont`	Header row font	First row font properties	String	`BOLD`	`NONE` , `BOLD` , `ITALIC` , `STRIKEOUT` , `UNDERLINE`
`trueString`	Boolean string TRUE	String which will be used instead of TRUE boolean values	String	`true`	*Any*
`falseString`	Boolean string FALSE	String which will be used instead of FALSE boolean values	String	`false`	*Any*
`exportSql`	Export SQL	Export SQL to a second sheet	Boolean	`false`	*Any*
`splitSqlText`	Split SQL Text	Split exported SQL on rows by CR	Boolean	`false`	*Any*
`splitByRowCount`	Max row on sheet	Split by row count	Integer	`1048575`	*Any*
`splitByColNum`	Column group	Column number for grouping rows on sheet by column value	Integer	`0`	*Any*
`dateFormat`	Excel date format	Excel date and time format (e.g. m/d/yy h:mm) it can be changed in Excel application	String	`m/d/yy`	`m/d/yy` , `d-mmm-yy` , `d-mmm` , `mmm-yy` , `h:mm AM/PM` , `h:mm:ss AM/PM` , `h:mm` , `h:mm:ss` , `m/d/yy h:mm`

PostgreSQL Debugger

Prerequisites for Debugging

To enable interactive debugging of PL/pgSQL procedures on a Postgres server, you need to use the _plugin *debugger*. The _plugin*debugger* is a typical interactive debugger delivered as an extension. It requires a shared library preload in Postgres to operate the shared_preload_libraries parameter in the settings. The debugger is developed and maintained by EDB. Its source code is available for examination and improvement.

The debugger provides the required server API for debugging PL/SQL procedures with:

- Breakpoint management;
- Step-by-step tracing;
- Variable acquisition and management.

Installation

PostgreSQL 12 on Ubuntu-based distros

If you happen to have a PostgreSQL 12 installed via *apt*, then the procedure is quite straightforward:

```
sudo apt install postgresql-12-pldebugger
sudo service postgresql restart
```

After that, run the following command in the database or databases that you wish to debug functions in:

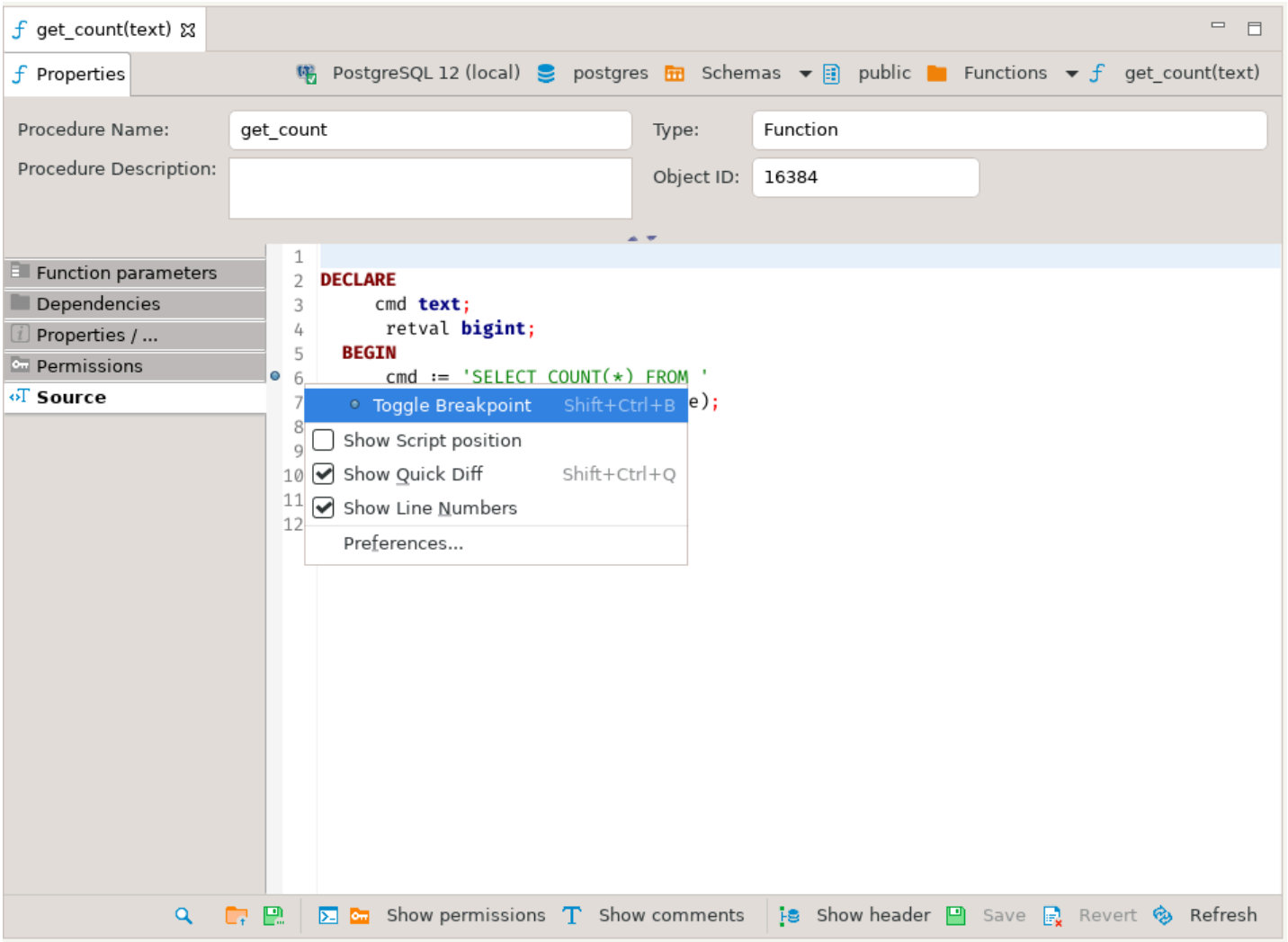
```
CREATE EXTENSION pldbgapi;
```

Installation from source code

You can find the source code in this repository. Installation instructions are located in the README file.

Running debugger in the DBeaver interface

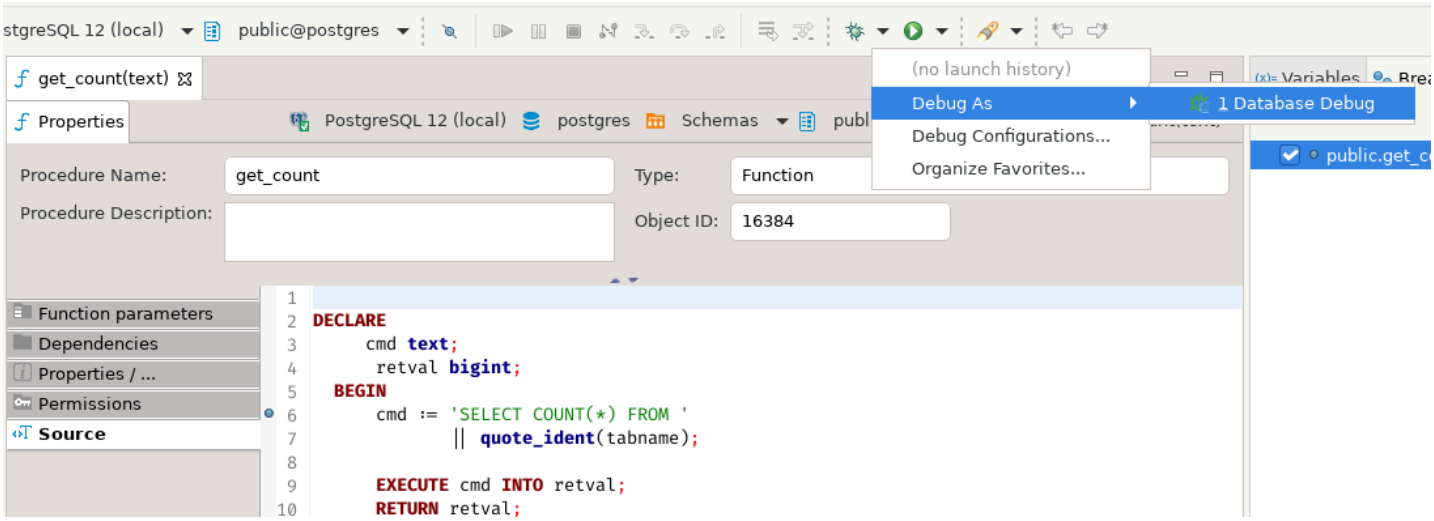
Open the source code of the function you want to debug. To toggle breakpoints, place the caret on the line you want the function to be stopped at and use a shortcut `Shift + Control + B`. Alternatively, you can toggle the breakpoint with your mouse by clicking on a ruler, as demonstrated in the screenshot below:



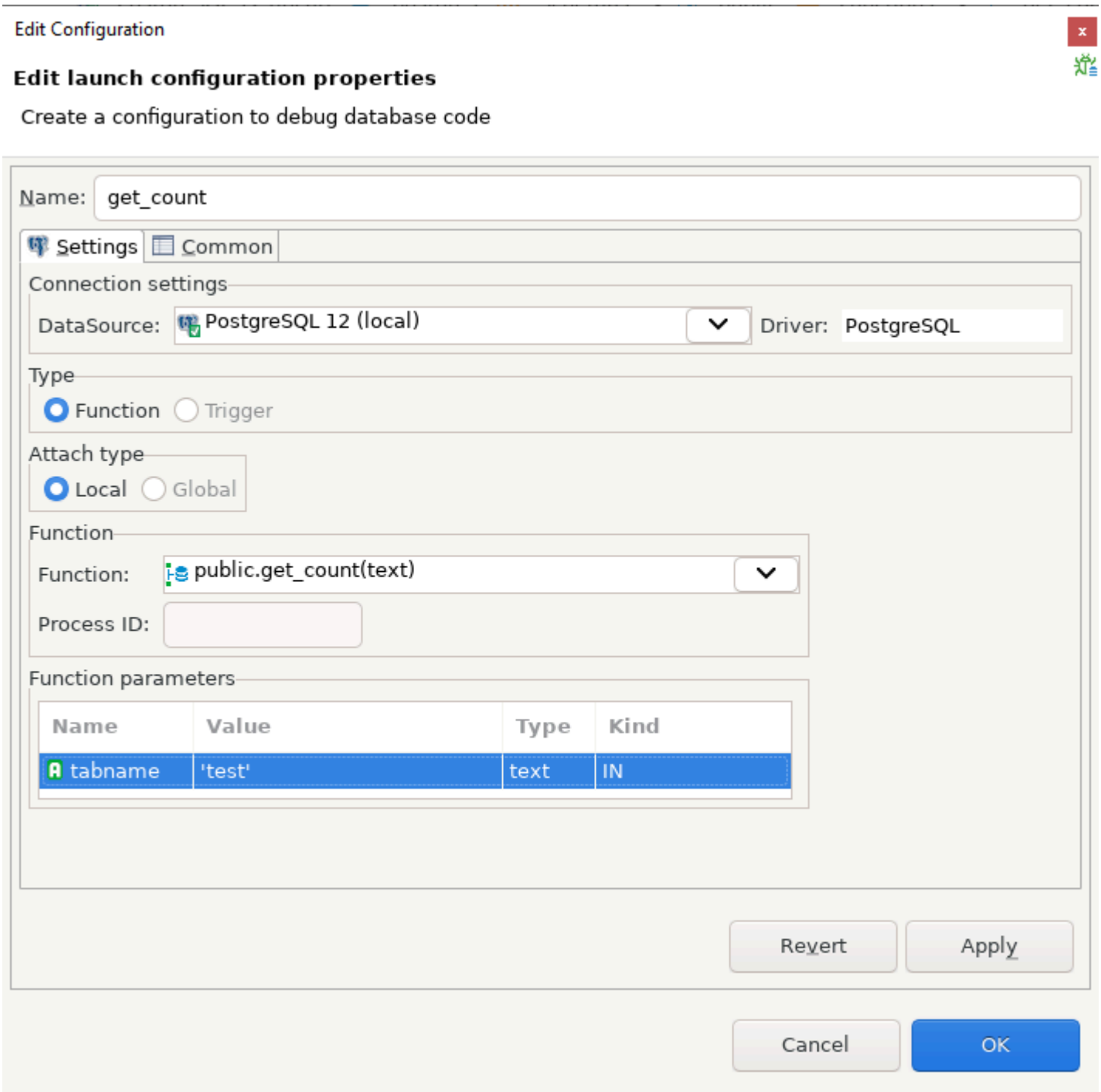
NB. You must only toggle the breakpoints when *Show header* option is **not** on.

Then you need to set up a debugging configuration. Locate the downward-facing arrow right to the bug icon, click on it, then

Debug As -> Database Debug:

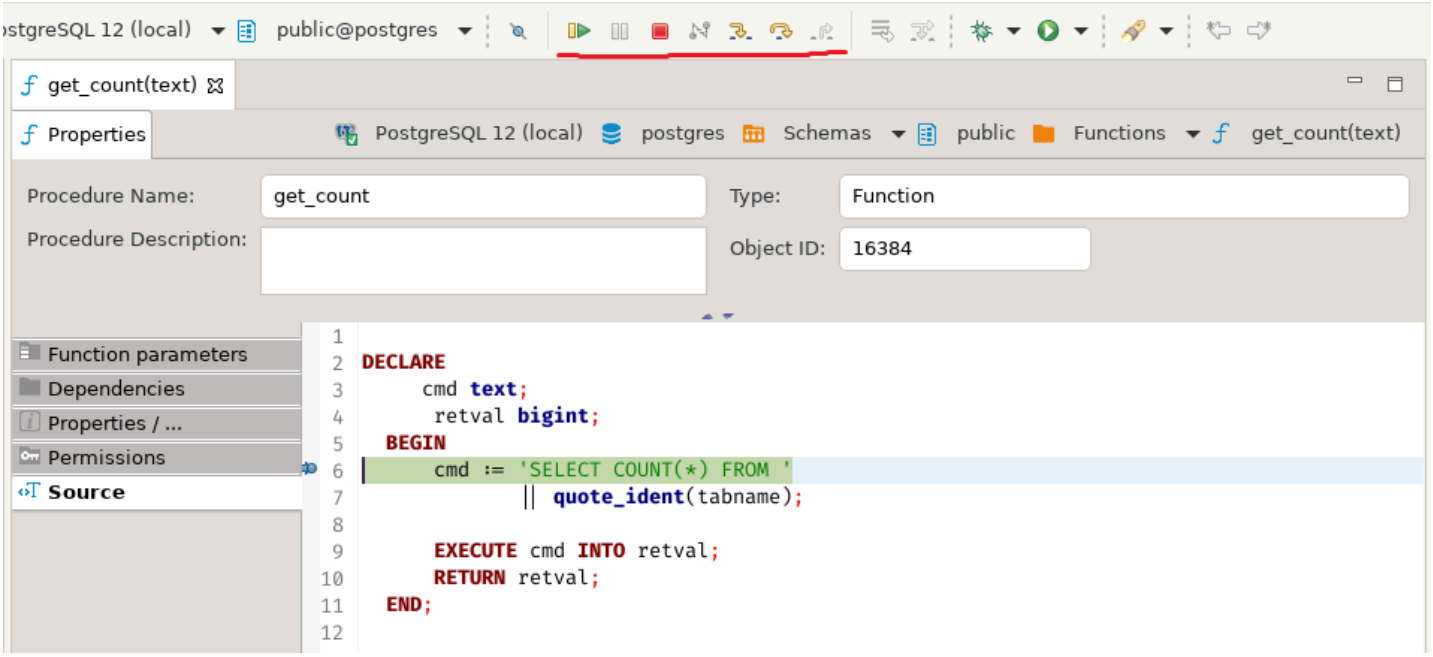


The *Edit Configuration* dialog opens. Set up input values in the table *Function parameters*.



Click on *OK* button, and you are ready to go!

The usual buttons essential for debugging such as *Step Over* and *Continue* are located here:

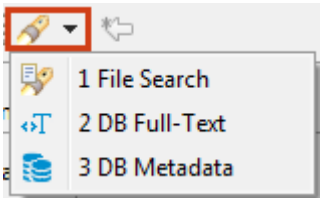


Search

DBeaver provides:

- File search (search among file contents)
- Database full-text search
- Database metadata search

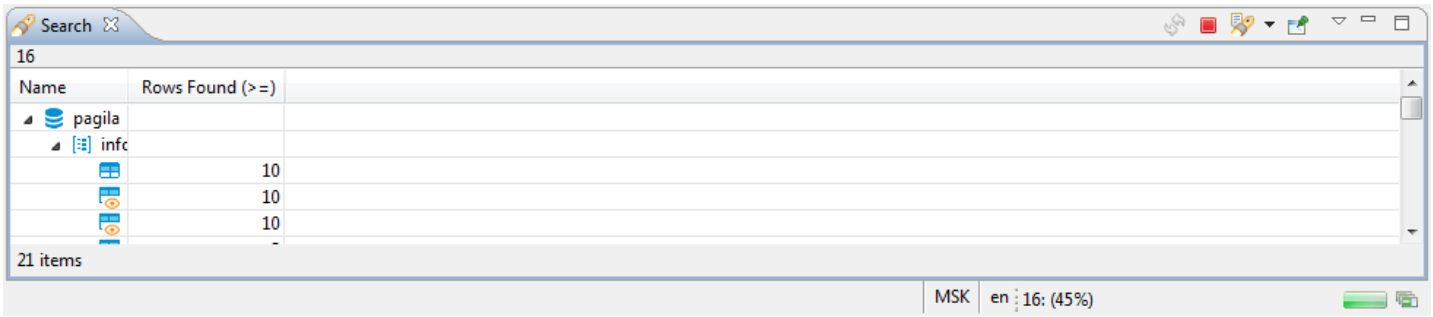
To use search, click the Search button on the main toolbar:



Please see the dedicated articles for information about searching for different types. This article describes common features of the three search types.

Search View

Search results for any of the search types appear in a separate Search view. The following image shows the Search view for the database full text search:



The view contains a toolbar that provides common tools for all types of search as well as specific tools for the File Search type. The following are common tools:

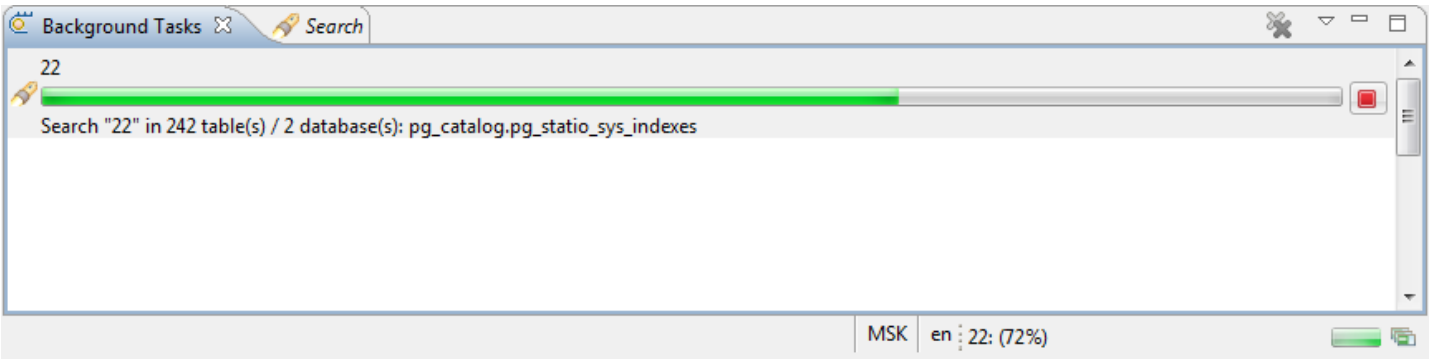
Button	Name	Description
	Run the	

	Current Search Again	Refreshes the search results
/	**Cancel Current Search**	- Active state (red) indicates that the search is still in progress and appears if the search takes some time to complete. Clicking the button in this state stops the current search. - Inactive state (grey) indicates that the search is complete. The button in this state is non-actionable.
	Show Previous Searches	- Clicking the button itself opens the Previous Searches window. - Clicking the arrow next to the button opens a dropdown menu. See the *Search History* section further in this article.
	Pin the Search View	Ties the current search results to the Search view. If you click this button, the current results stay in the view while the results of the next search appear in a new Search view. Otherwise, every new search replaces the previous results with new results.

For information about specific tools of File search, see the File Search article.

If the search is short, the results appear almost instantly. But if it takes some time, the Search view indicates the progress in the following ways:

- The **Cancel Current Search** button in the toolbar has the Active state ( )
- The progress bar appears in the bottom-right corner of the view indicating the process:
- The button to show the search progress in a separate view ( ) appears in the bottom-right corner of the view next to the search progress bar. Clicking the button opens the Background Tasks view:

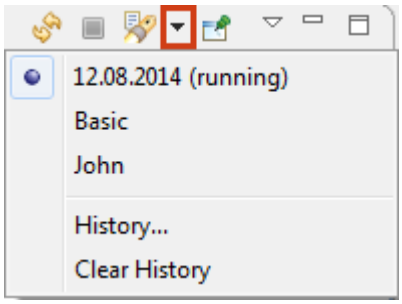


Search History

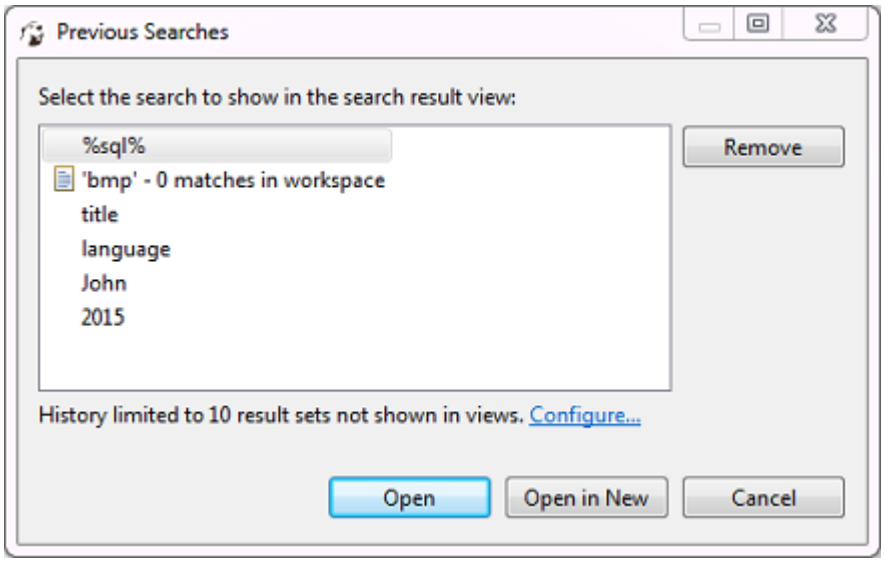
DBeaver stores the history of search queries made during the current session. You can reopen the Search view with results of a previous search query. You can also remove individual queries and clear the history. To manage the search history, use the **Show Previous Searches** button in the toolbar ( ).

To open the results of a previous search query, do one of the following:

- Click the arrow next to the **Show Previous Searches** button in the toolbar and then click the query in the dropdown list:



- Click the **Show Previous Searches** button itself or the arrow next to it and then **History** on the dropdown menu to open the Previous Searches window. Then, in the window, click the query and then either click **Open** to open it in the active Search view or click **Open in New** to open it in a new view:



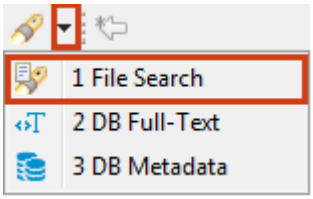
To remove one or more of the previous search queries:

1. Click the **Show Previous Searches** button in the toolbar or click the arrow next to it and then **History** on the dropdown menu. The Previous Searches window opens.
2. Click the query to remove or select several of them by clicking and simultaneously holding the `Ctrl` key.
3. Click **Remove**.

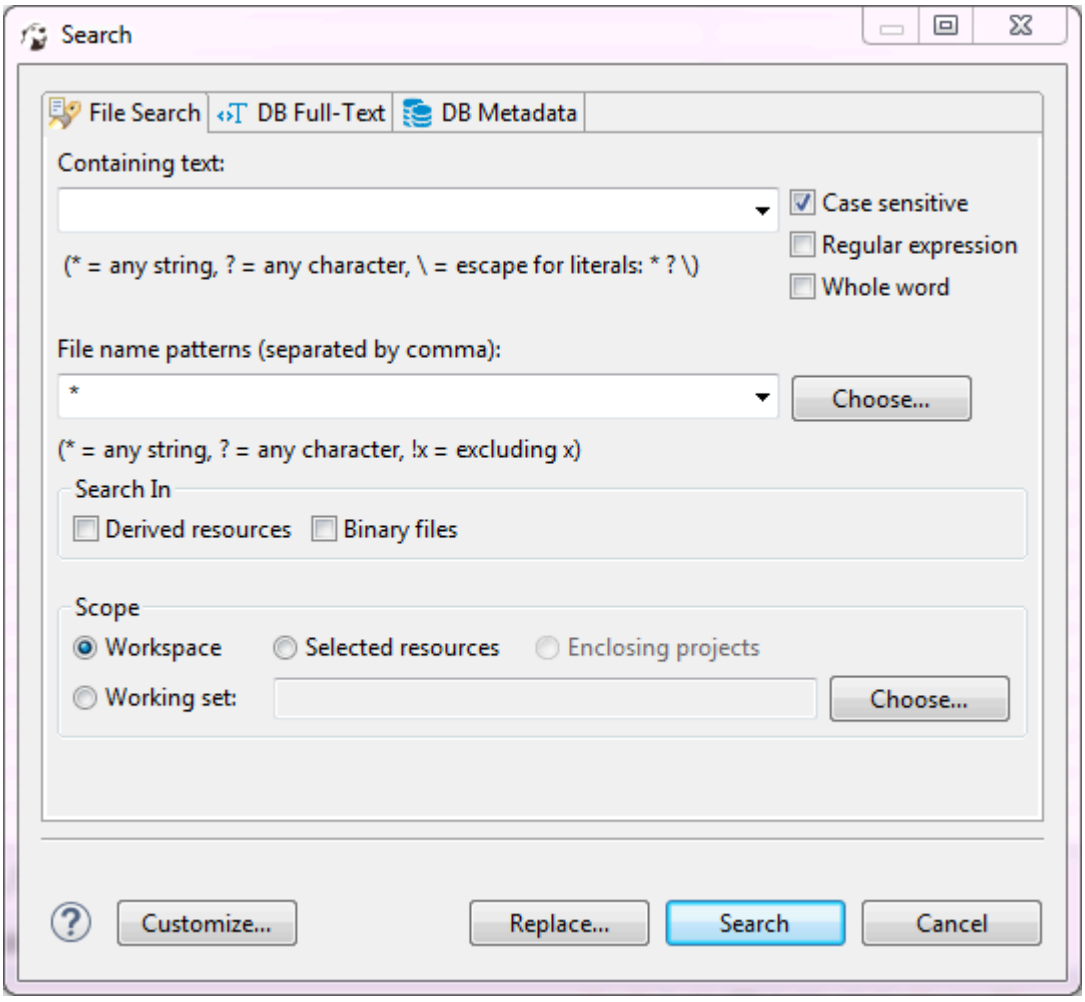
To clear the history by removing all previous queries, click the arrow next to the **Show Previous Searches** button on the toolbar and then click **Clear History** on the dropdown menu.

File Search

To search file contents for a string, click the Search button on the main toolbar or the arrow next to the Search button and then **File Search** on the dropdown menu:

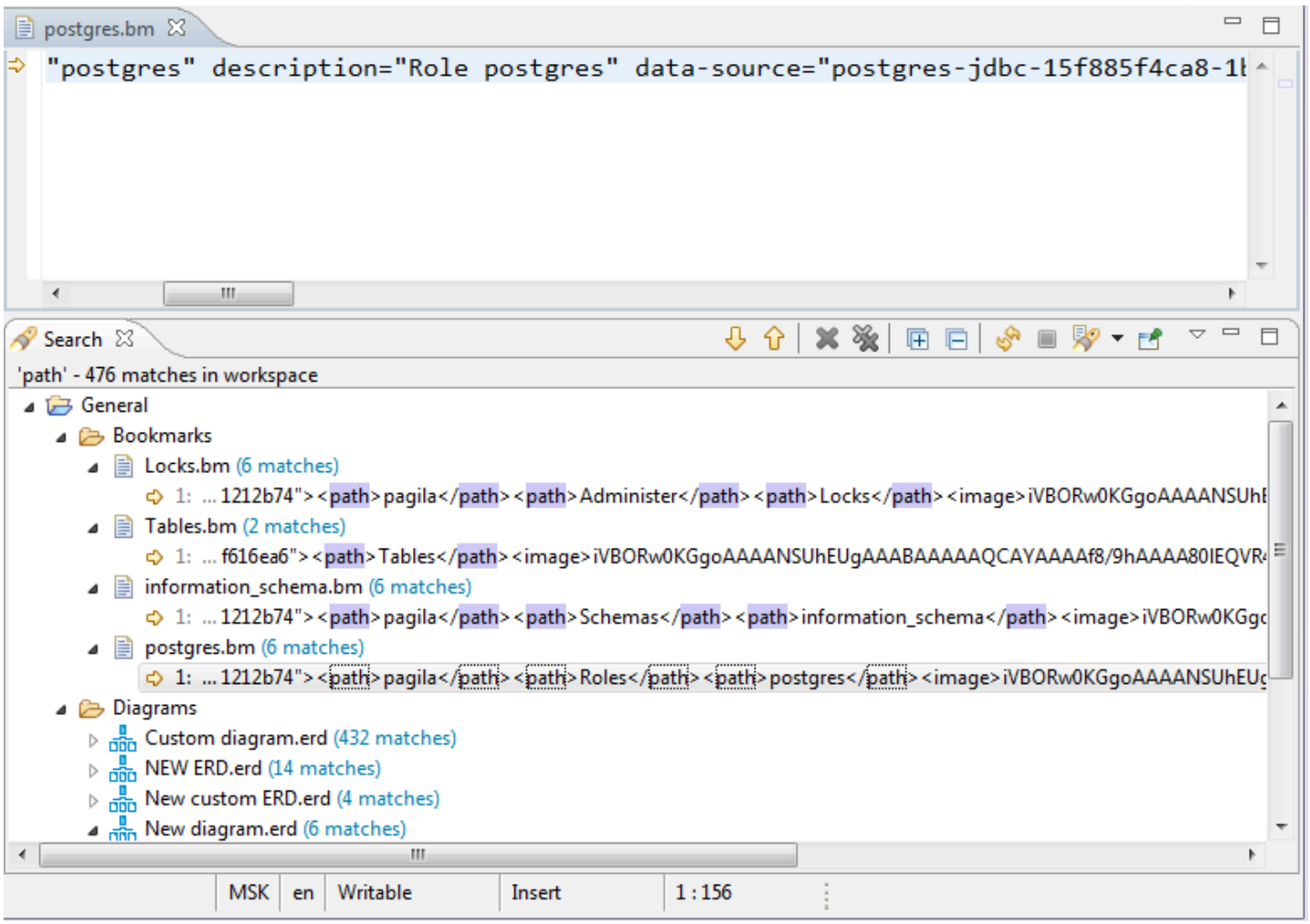


The Search window opens displaying the File Search tab:



You can apply a case sensitive search, search by regular expressions, search among particular file types (**File name patterns** field), and use the find and replace function.

After you click **Search**, the results appear in a Search view. The results represent a tree or list of files with the search combination highlighted:



The toolbar of the Search view for File search provides more tools in addition to those available for all search types:

Button	Name	Description
	Show Next / Previous Match	Open the file in a separate viewer and move the highlight to the next/previous match, respectively
	Remove Selected Matches	Removes selected row (row in focus) of the results
	Remove All Matches	Removes all results in the view
	Expand/Collapse All	Expand/collapse the tree of results

The view also provides a view menu (click the **View Menu** button ( ) in the upper-right corner of the view) that contains the following items:

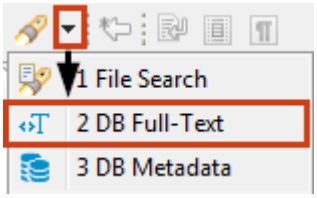
Menu item	Description
Show as List	Presents the results in the form of list
Show as Tree	Presents the results in the form of tree

Filters	Opens Search Filters dialog box
Preferences	Opens the Preferences window on the Search page

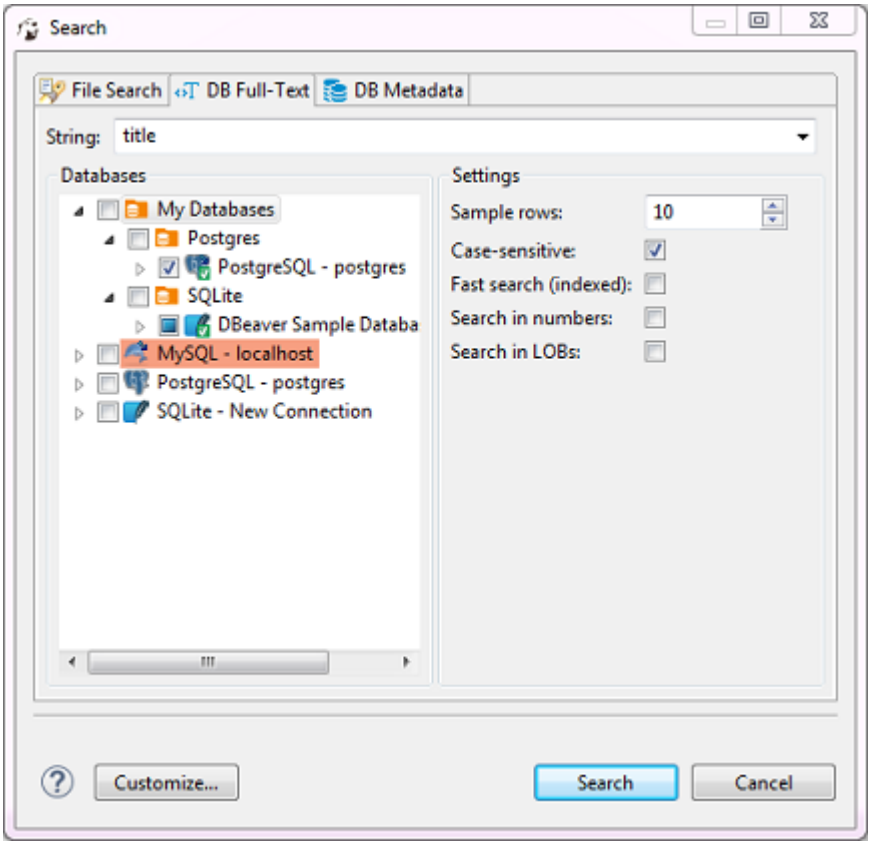
Double-clicking a results row opens it in a separate viewer.

DB Full-Text Search

To do a full text search in the database, click the arrow next to the Search icon in the main toolbar and then click **File Search** on the dropdown menu:



Alternatively, you can click the **Search** button on the main menu and then click the **DB Full-Text** tab in the Search window:



Now you need to choose the database connection or database objects against which to run the search – expand the tree in the **Databases** field to the database connections level or further down and select the checkboxes next to the required connections or database objects.

NOTE: The **Search** button is enabled only when you select the right level of checkboxes – database connections or lower nodes.

You can apply a case-sensitive search, fast search and search in numbers and LOBs.

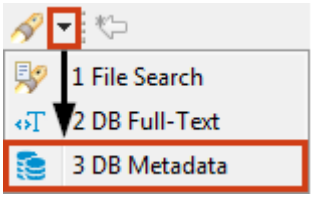
After you click **Search**, the results will open in Search view:



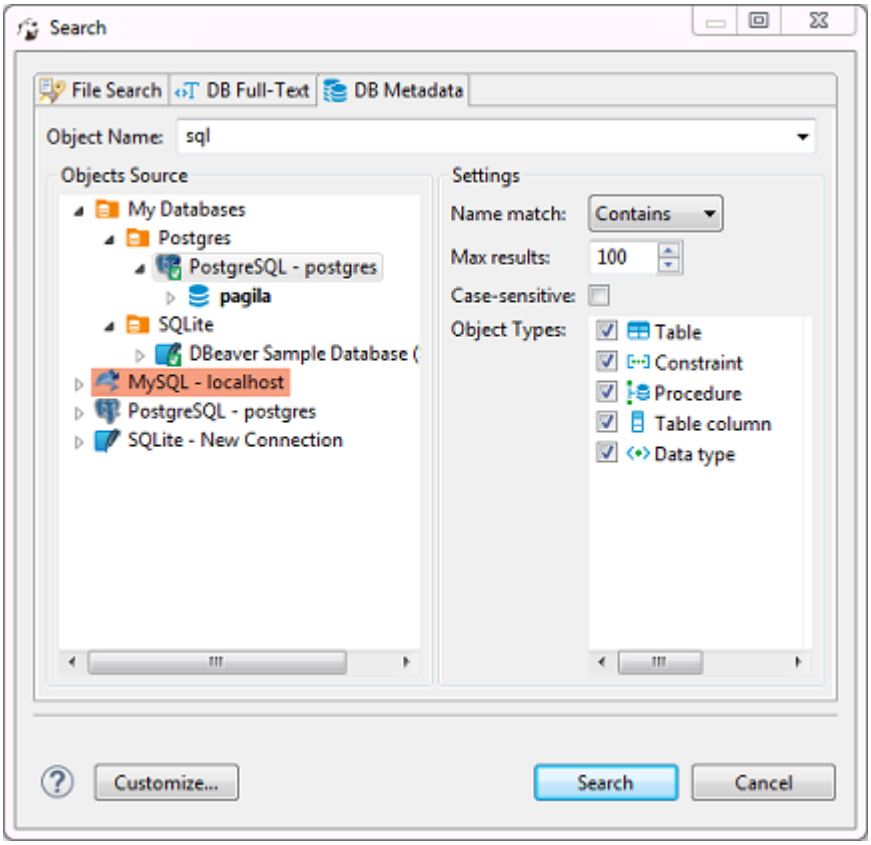
Double-clicking a row in the Search view opens the respective object in a dedicated Database Object editor.

DB Metadata Search

To search for database metadata, click the arrow next to the **Search** button in the main toolbar and then click **DB Metadata Search** on the dropdown menu:



Alternatively, you can click the **Search** button on the main menu and then click the **DB Metadata** tab in the Search window:

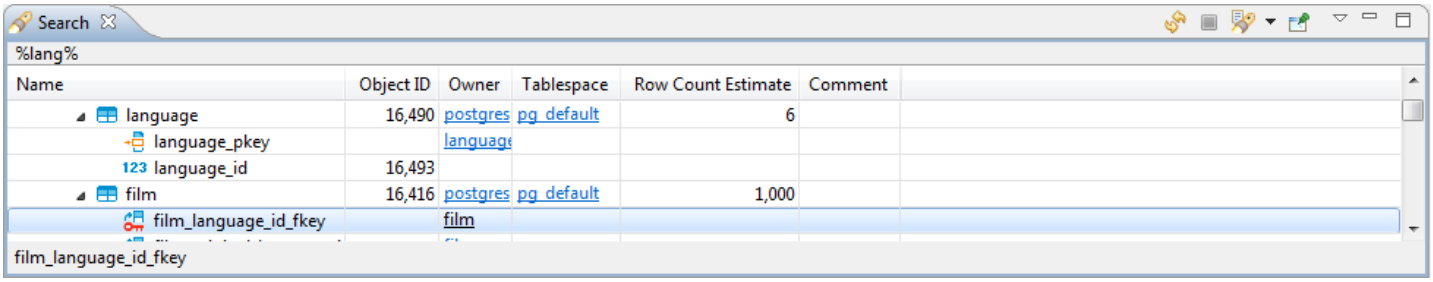


Now you need to choose the database connection against which to run the search. You only need to select the database connection(s) in the **Objects Source** field.

In the **Object Types** field, you can select the database objects among which DBeaver will run the metadata search – select or clear the checkboxes.

You can specify that the object name starts with, contains or is similar to the search combination (**Name match** field). You can also set the maximum number of results to display (**Max results** field) and apply **Case-sensitive** search.

After you click **Search**, the results will open in a Search view:



Double-clicking a row in the Search view opens the respective object in a dedicated Database Object editor.

Schema compare

Note: This feature is available in Enterprise, Ultimate and Team editions only.

- Supported databases
- How it works
- What is possible to compare
- How to use schema compare
- Compare logs
- Using Liquibase in Dbeaver
- Liquibase changelog generation
- Save comparison as a task
- Using schema compare with Liquibase PRO key
- Additional object types in Liquibase PRO

Databases supporting schema comparison

| ---------- | ---------- |
Cockroach	Oracle
Databricks	PostgreSQL
DB2	Redshift
Derby	SAP HANA
EnterpriseDB	Snowflake
Firebird	SQLite
Greenplum	SQLServer

Informix	Teradata
MySQL/MariaDB	Vertica
Netezza	YugabyteDB
DB2i	

How it works

You can compare two schema/database structures and generate a report in the following formats:

- DDL script (series of create/alter/drop statements)
- Diff diagram (sort of ER diagram)
- Liquibase changelog
- Liquibase change report (JSON, YAML, or plaintext)

What is possible to compare

In most cases, you can compare **schemas, databases, or tables**. However, some databases (such as SQLite and Firebird) do not have catalogs and schemes that can be compared. In this case (and only for these databases), it is possible to compare the entire **datasources**.

Usually, you can compare the following database objects: **columns, primary keys, foreign keys, indexes**, and so on (it depends on your database). If you want to compare more objects, such as **check constraints, procedures, functions, triggers**, you need to enable Liquibase PRO.

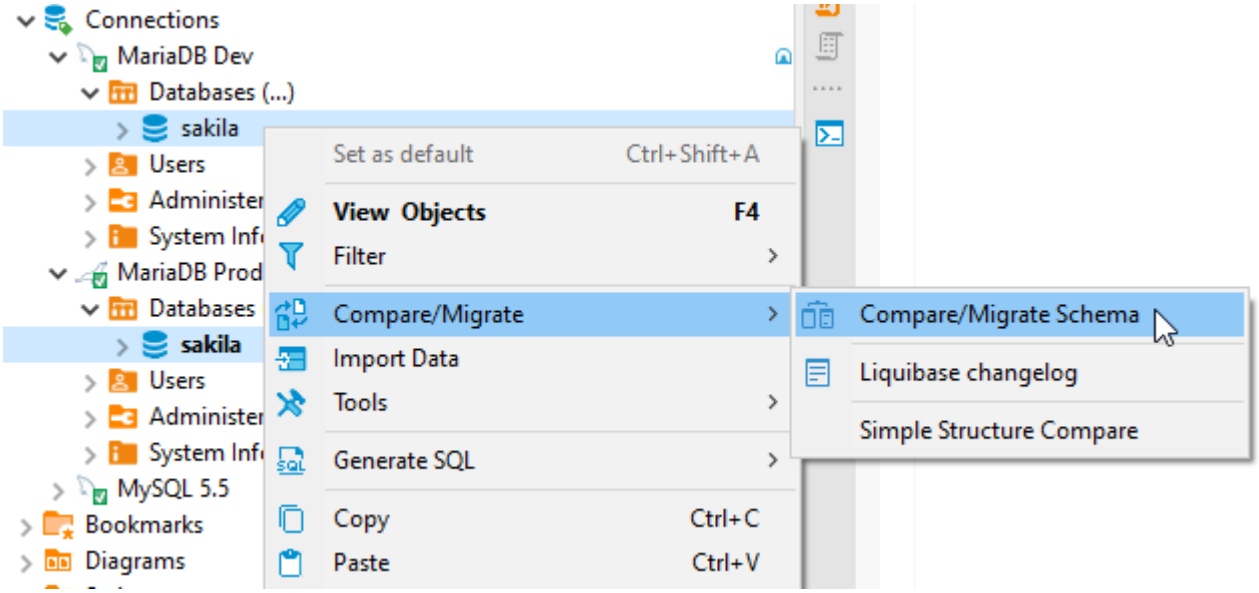
How to use schema compare

Step1. Select two entities to compare

1. Select the two objects (schemas, databases, or tables) you want to compare in the **Database Navigator**.

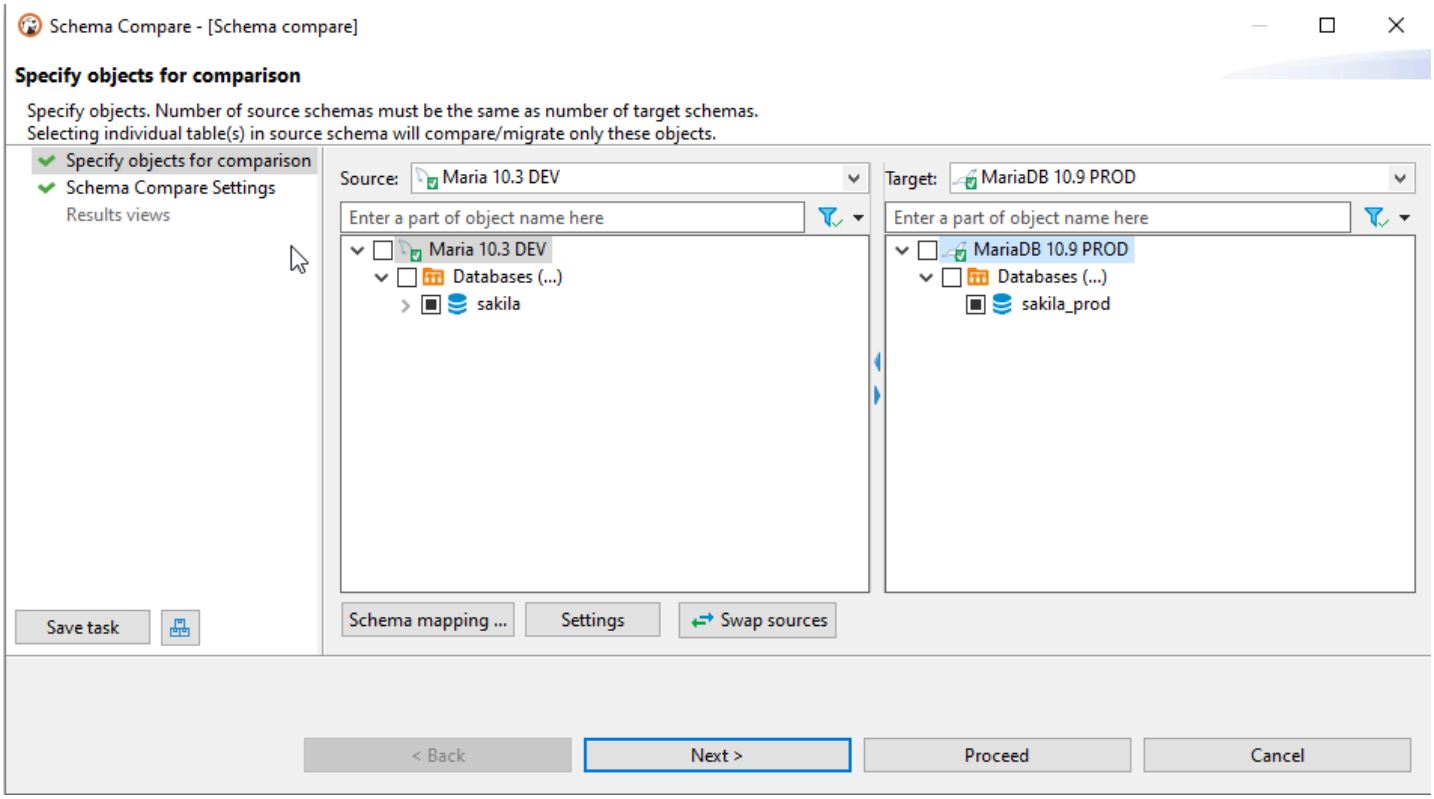
2. Open the context menu.

3. Open the sub-menu **Compare/Migrate** and click the **Compare/Migrate Schema**. You'll see the comparison window.



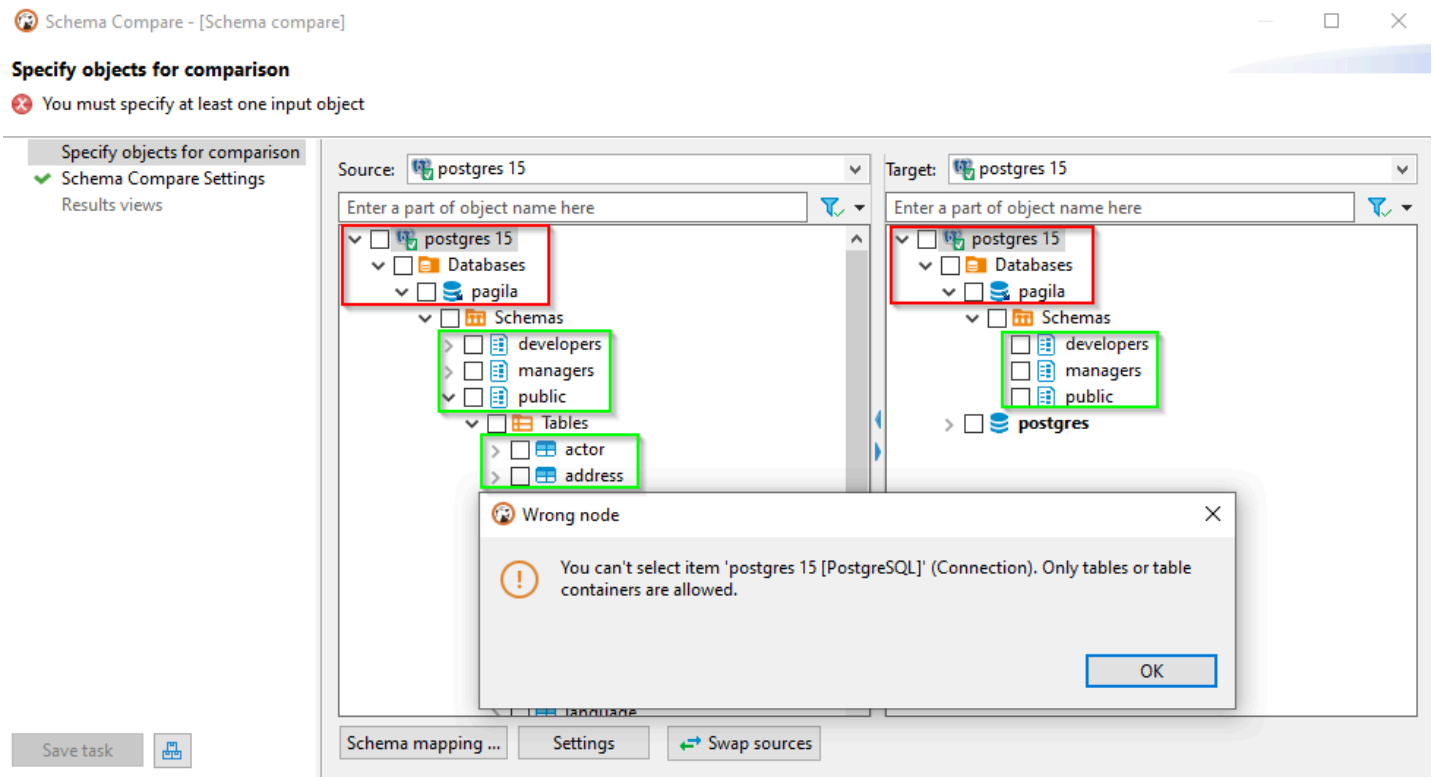
Step 2. Check the selected entities

Re-validate that you have chosen the correct objects to compare. You can change target and source containers by clicking the **Swap sources**.

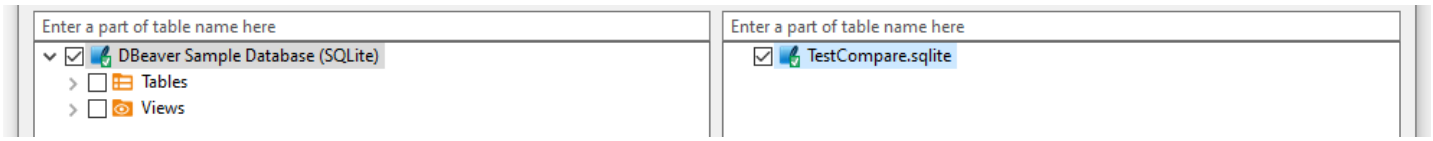


Note: You must select only those schemas/directories/containers that contain tables:

- Schemas – if the database supports schemas.
- Databases – if the database supports catalogs and does not support schemas.



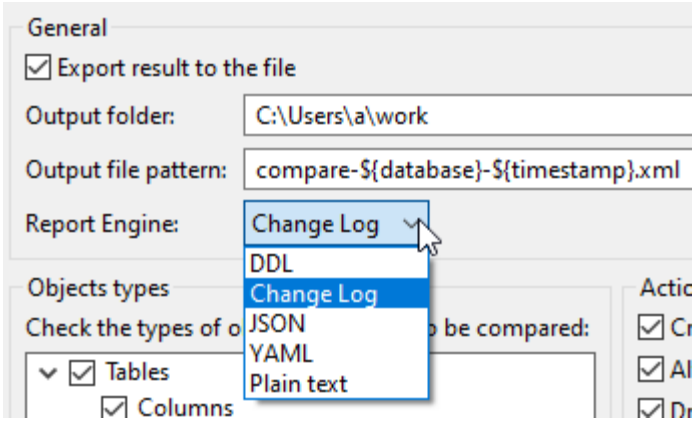
- Datasources – if there is no support for schemas or catalogs.



If everything is correct, click **Next**. You'll see the comparison settings.

Step 3. Specify compare settings

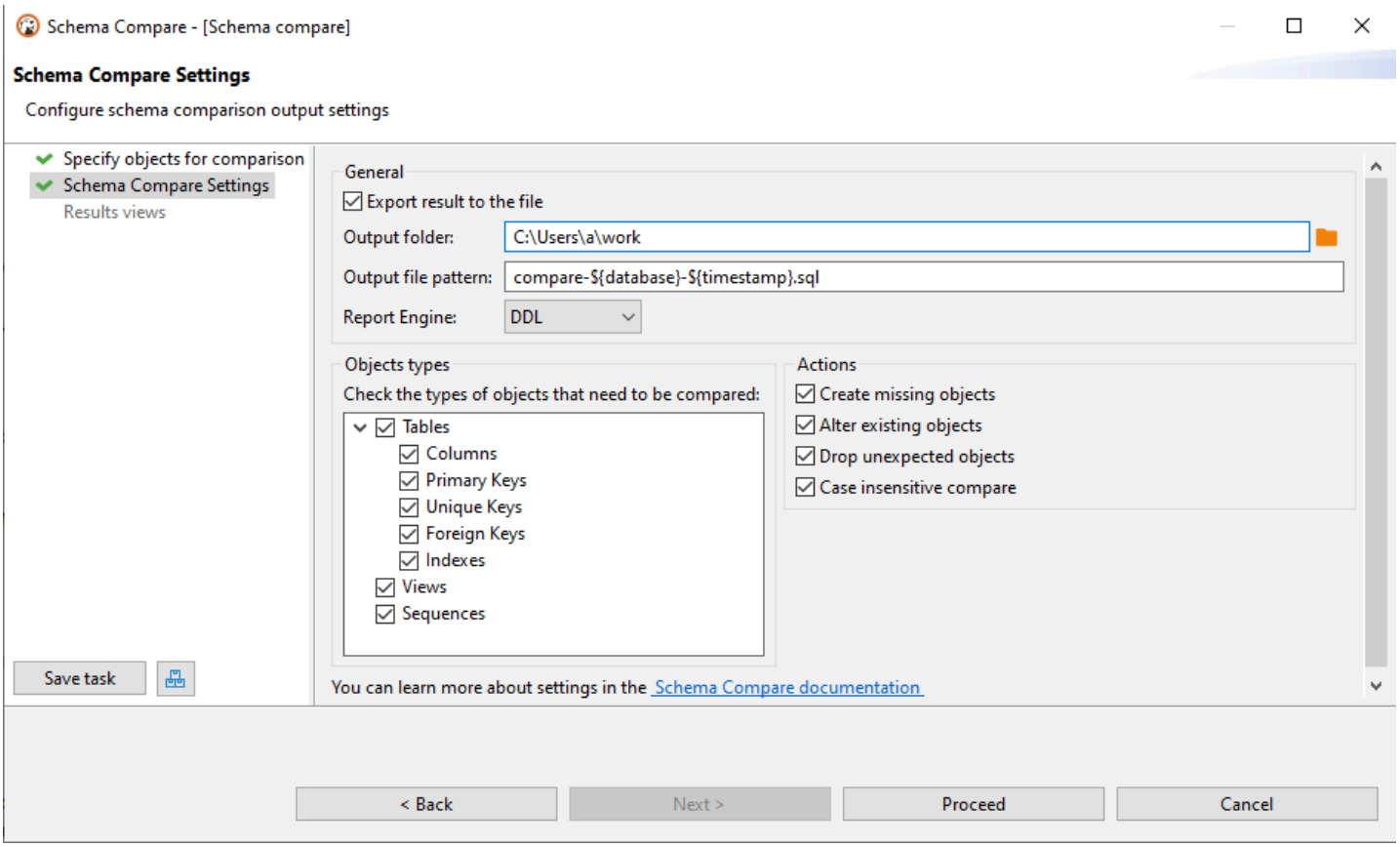
1. If you want to export the comparison result in a file, select **Export result to the file** option, then specify the folder and file name. You can use variables in the file name. Click on the field with a file name to see a list of available variables.
2. Select the report format in the **Report Engine** field.



3. You can simply exclude specific types of objects from comparison.

   For example, it is possible not to show sequences, views, or external keys in the final comparison result.

4. You can also specify which changes should be processed: create, drop, or alter. By default, all kinds of changes are enabled. If you do not want to compare objects with equal names but in different cases (like "test" and "TesT"), enable the **Case insensitive compare**. (Note: This settings section is unavailable for the generation changelog process.)

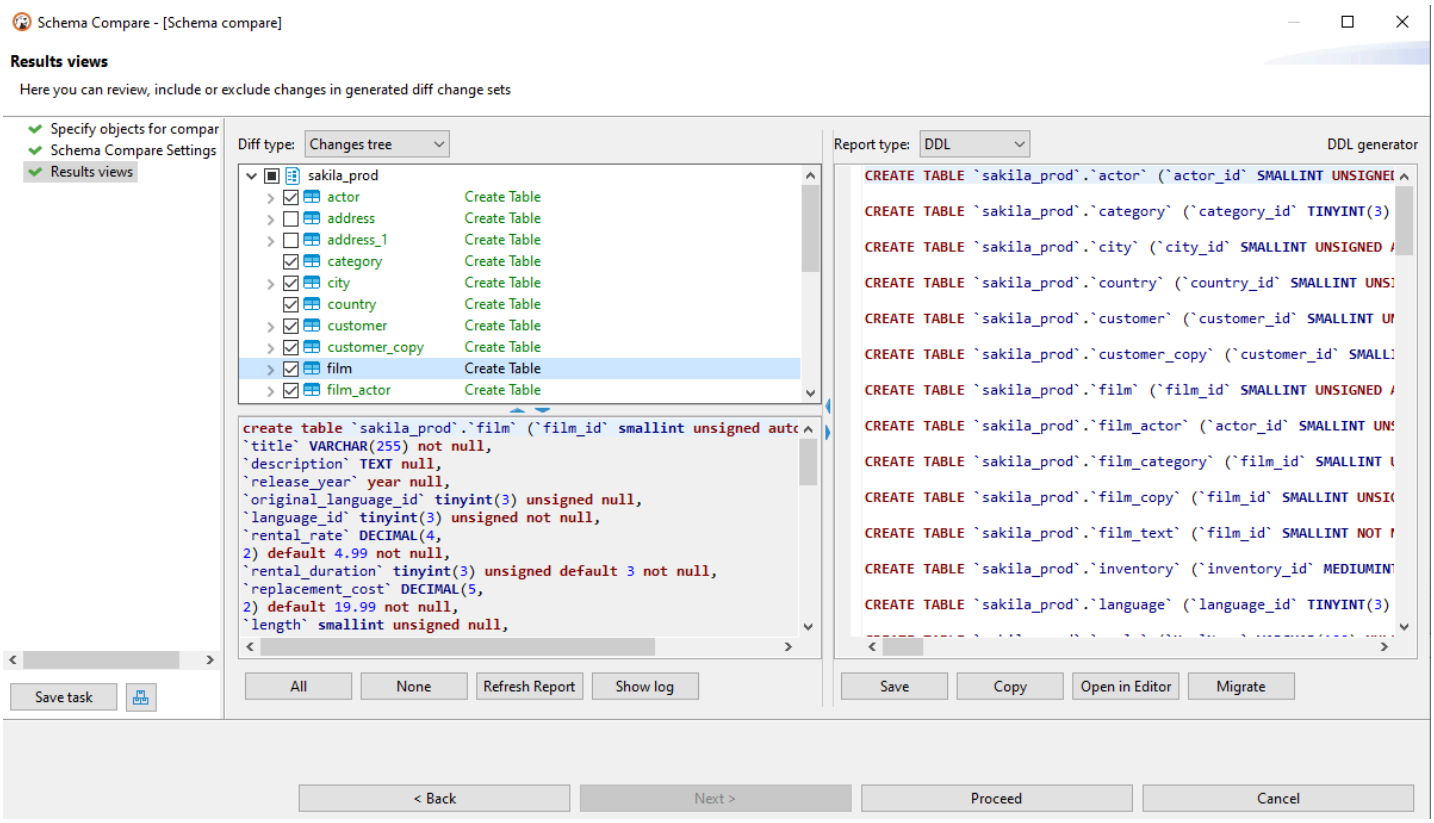


Step 4. Look at the comparison results and save the report

Click **Proceed** to generate a diff report.

By default, DDL diff is generated. It contains a series of creating, altering, and/or dropping statements that will modify the schema on the right side. Thus it will make it identical to the schema on the left side.

You can enable/disable certain changes in the tree on the left side of the diff page:

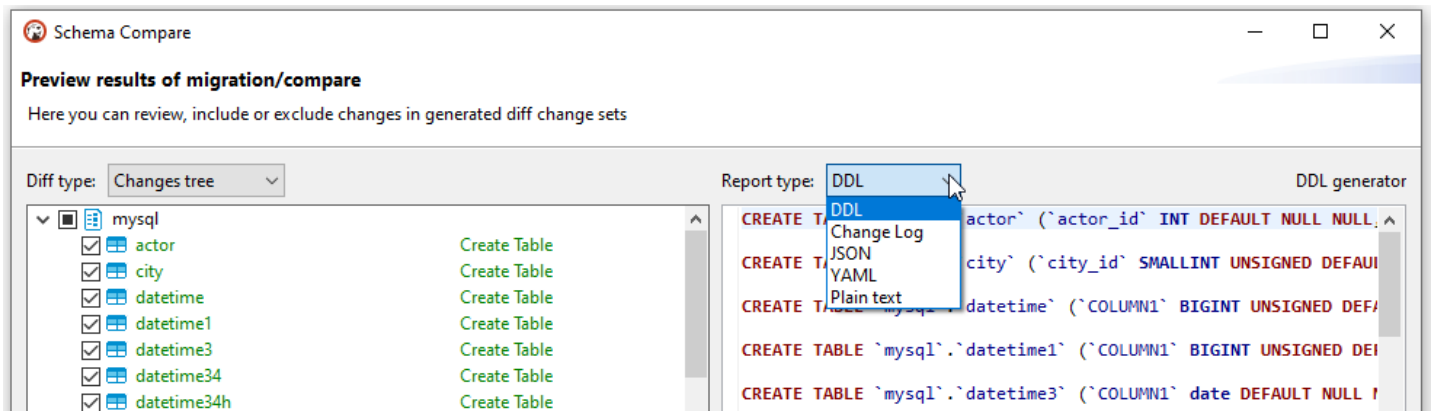


Use the **Refresh Report** button after objects change in the left tree to refresh the report on the right side.

You can view all changes in the SQL Editor with the help of the **Open in Editor** button.

If you are sure that compared changes should be applied to a target container, click the **Migrate** button. And all generated SQL statements will be executed in a target container.

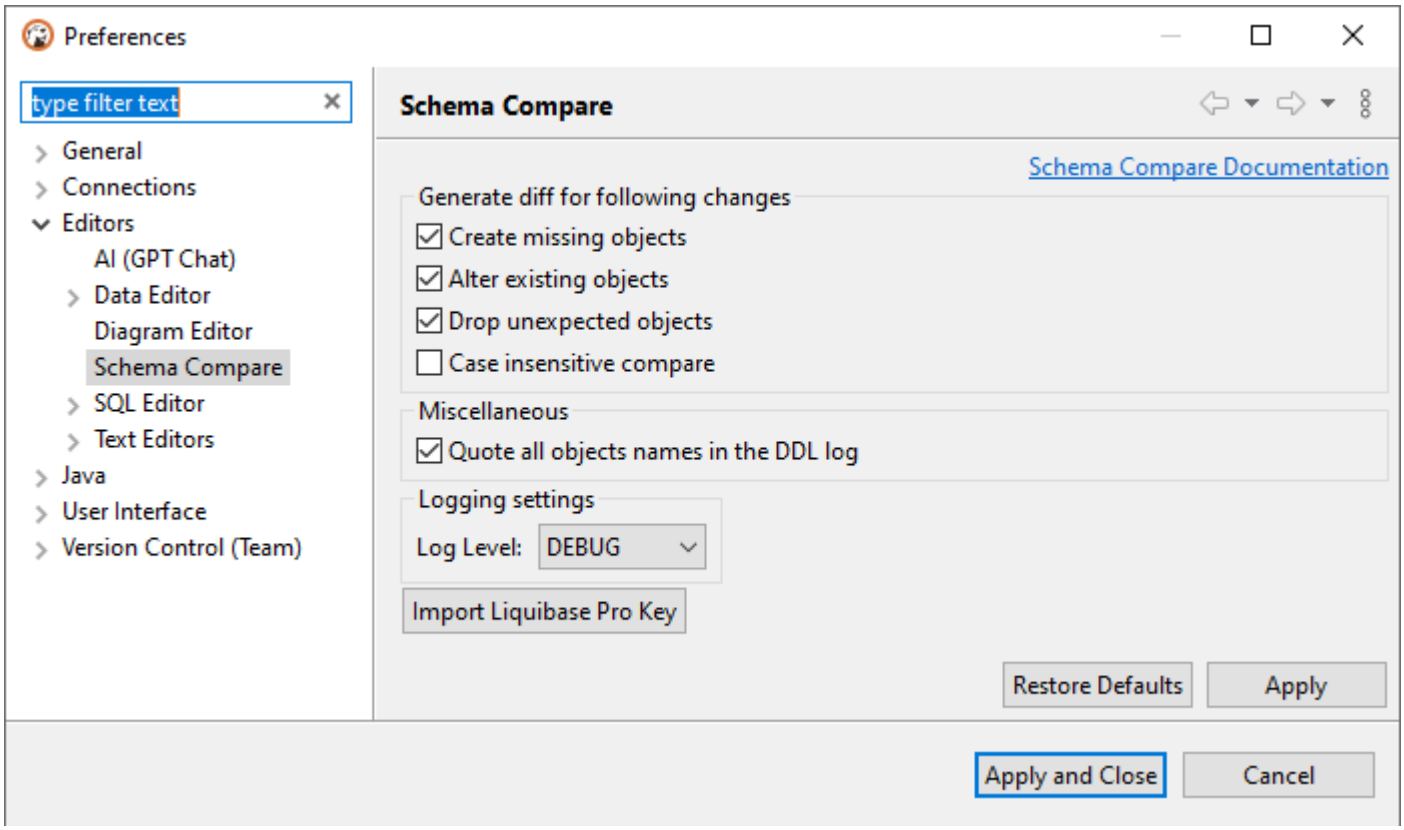
You can select another diff report representation (diagram, JSON, YAML, plaintext) in the **Report type**.



Click **Save**. That's it. Your report is saved.

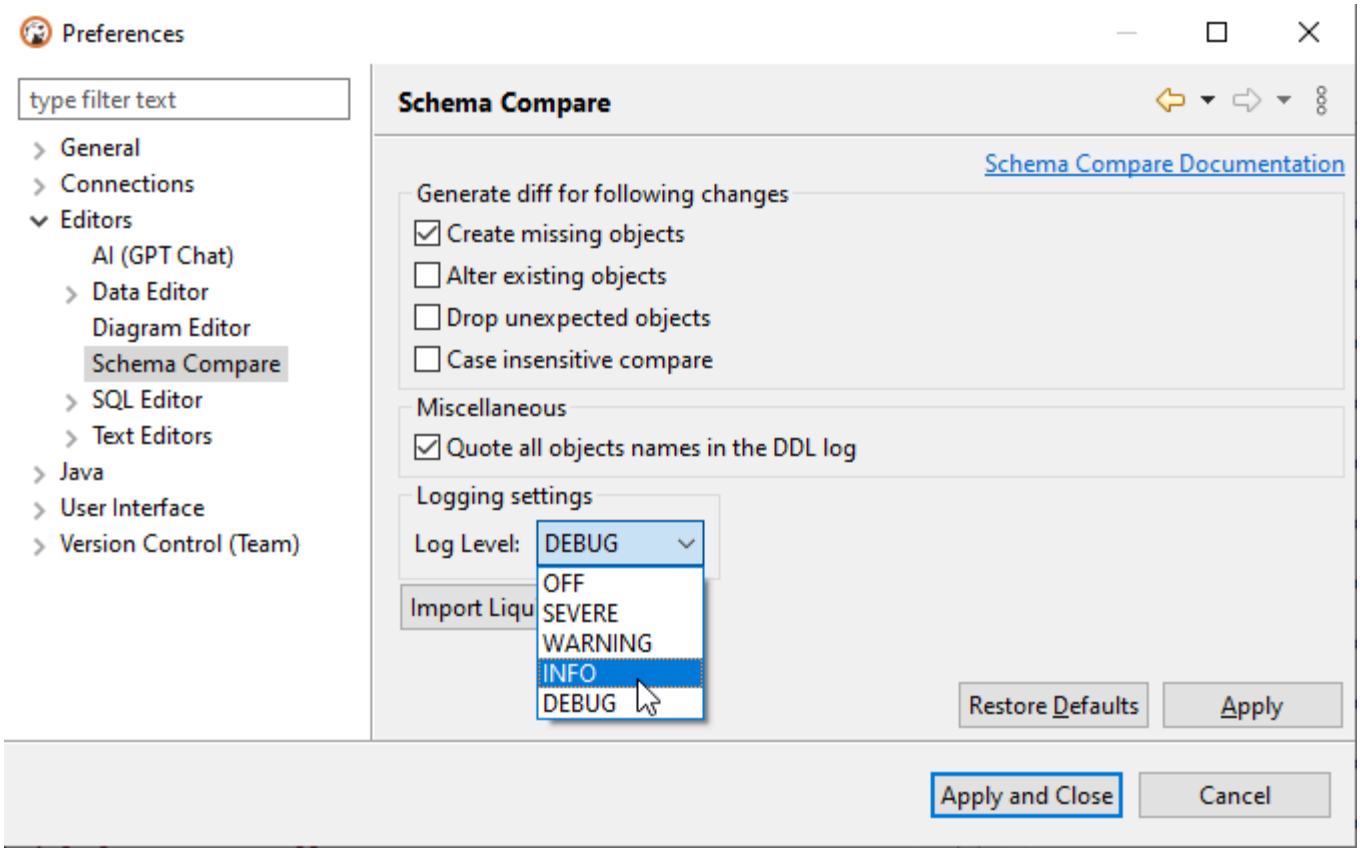
Quote objects names in the report

For the objects in the report to be dressed in quotation marks, select the `Quote all objects names` option in Preferences.

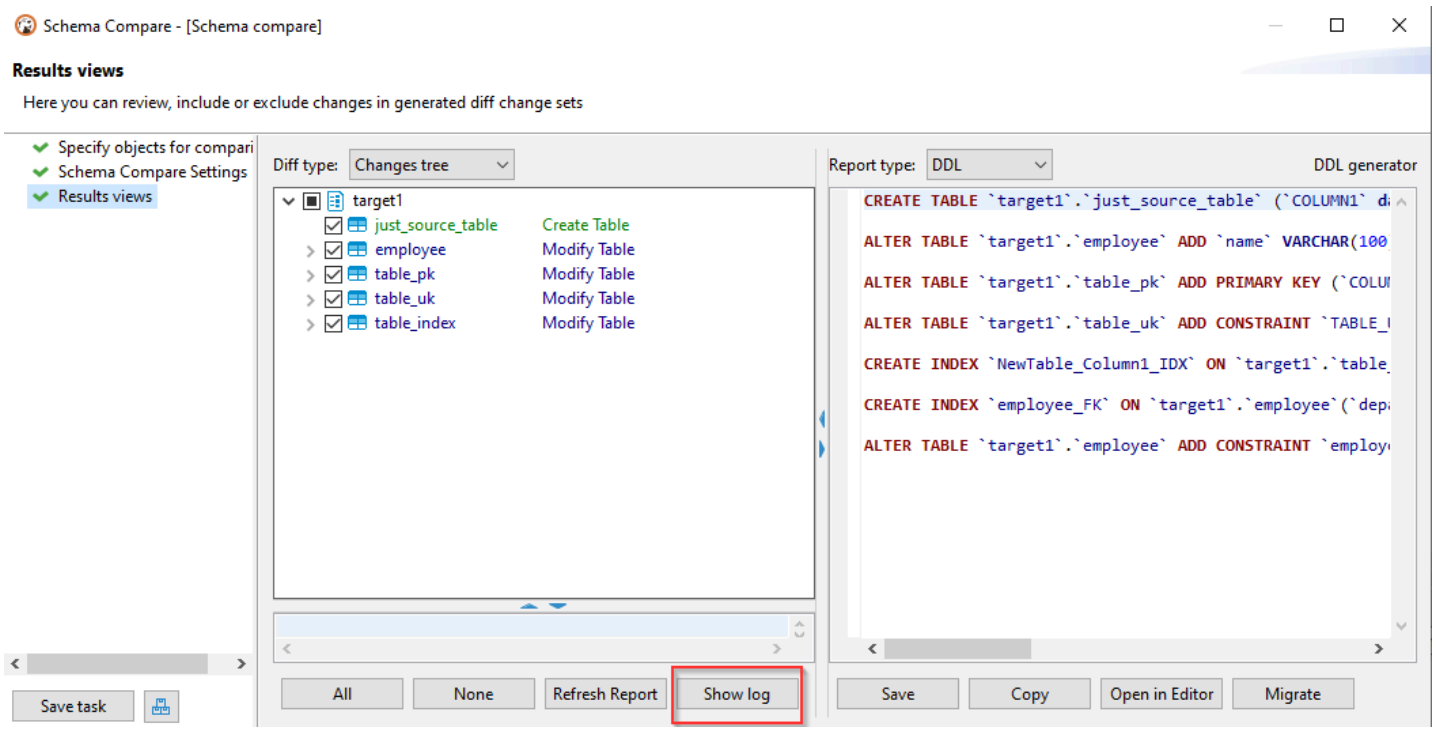


Compare logs

To get acquainted with the comparison logs, you first specify the logging level on the **Preferences-> Editors-> Schema Compare** preference page. Specify one of the logging levels and click on **Apply**. By default, the logging level is the OFF level. To get complete information, you can choose the DEBUG level.

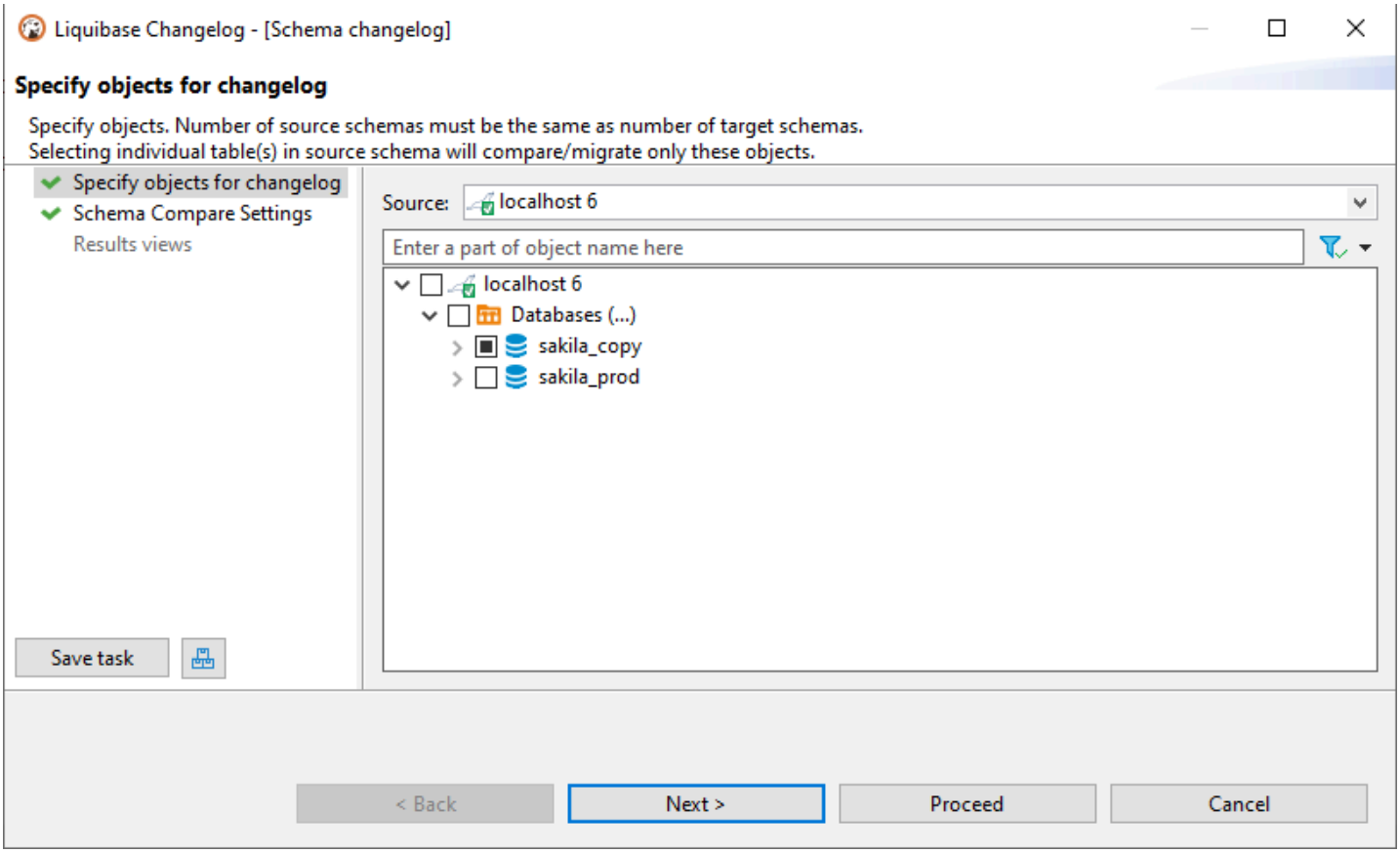


After comparing operations, click the `Show log` button. A log will be open in the Editor, and the content of this log will depend on the logging level you choose in the settings. Log level changes from preferences will not be applied to the comparison wizard if it is already open in another window. Close and open the schema compare wizard in this case.

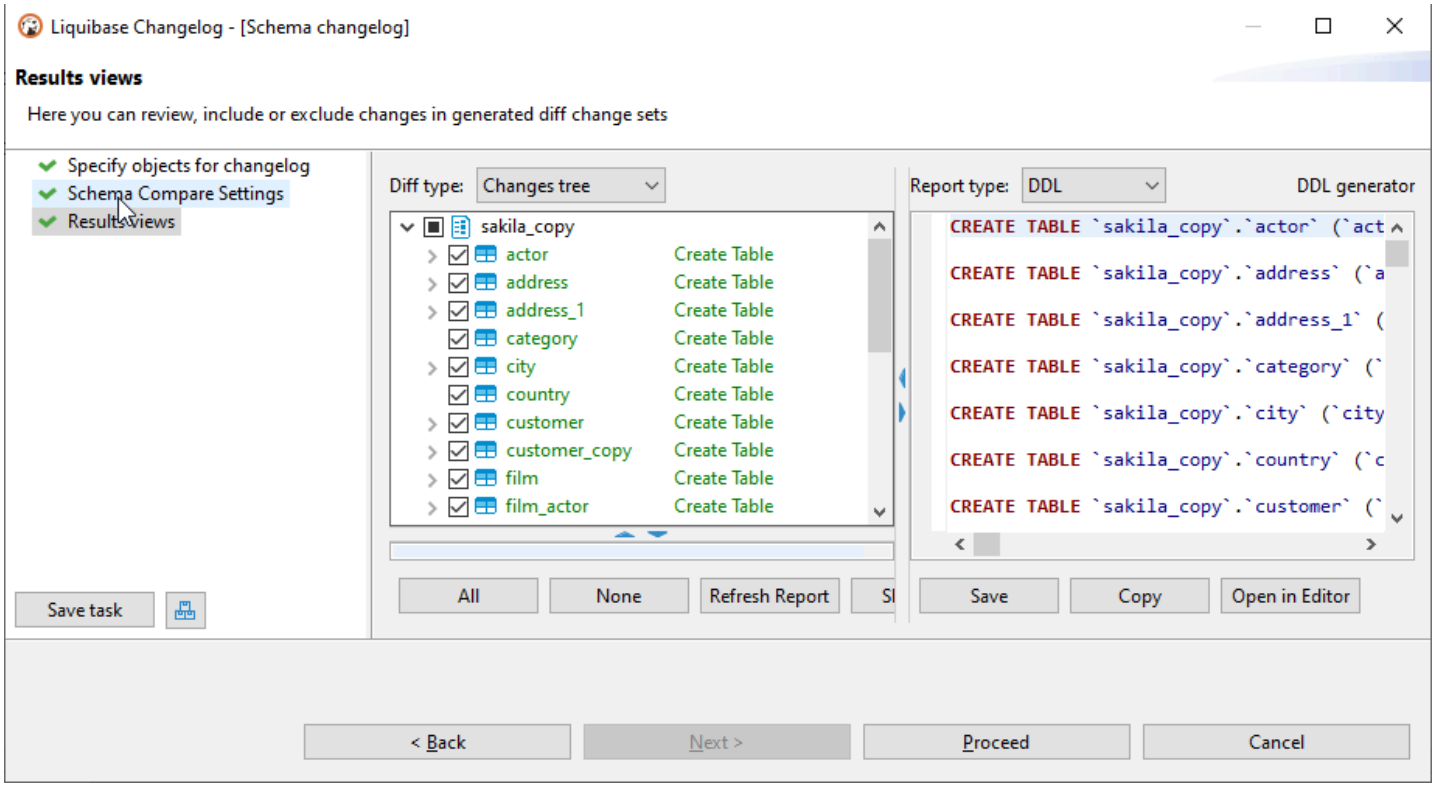


Liquibase Changelog generation

Suppose you want to create a report about the objects in your table container (similar to the metadata dump operation). In that case, you can select in the navigator tree on your container `Compare/Migrate` -> `Liquibase changelog` command.

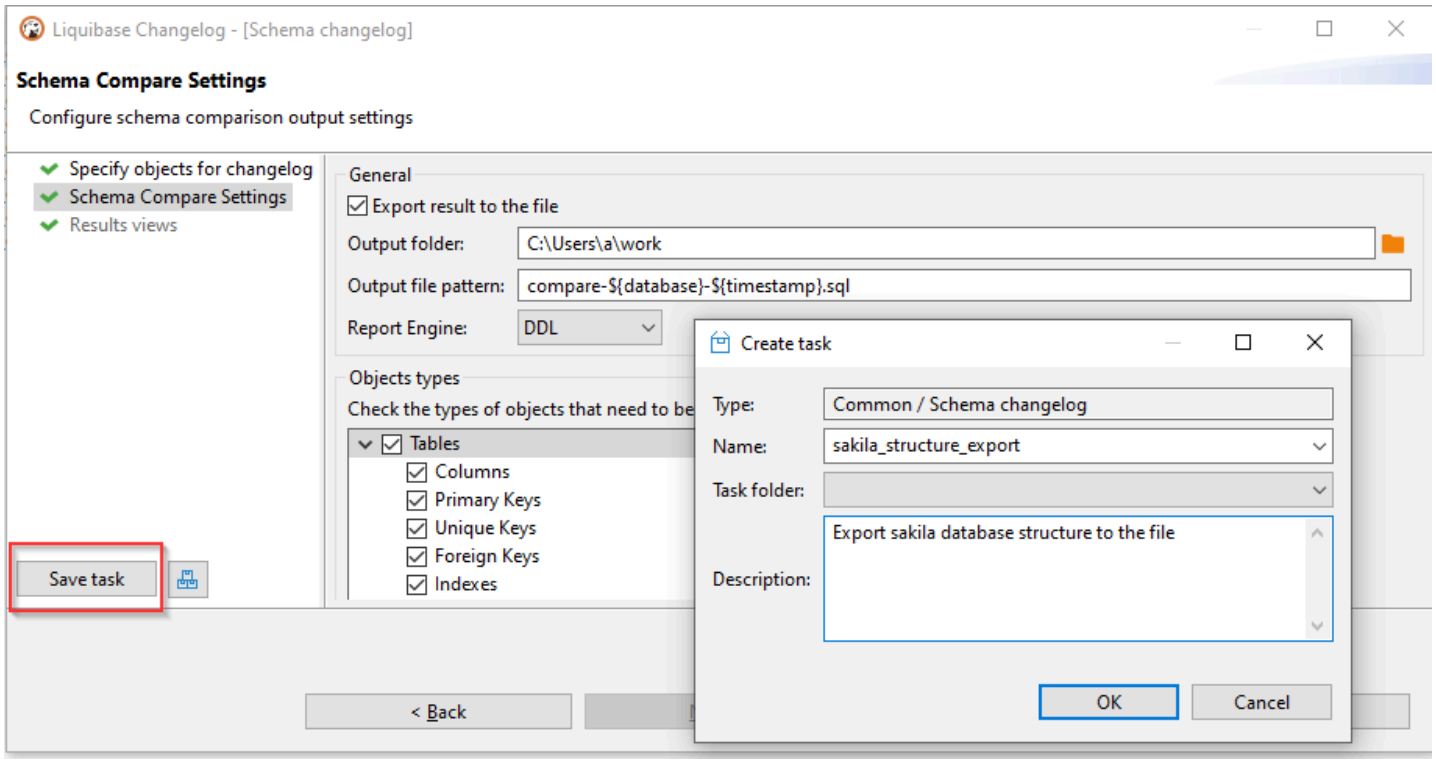


One or several table containers can be chosen. The report will contain creation statements of tables/views/keys /sequences - metadata from the table containers. But without data from tables/views. You can use this report in the future to restore the structure of your database.

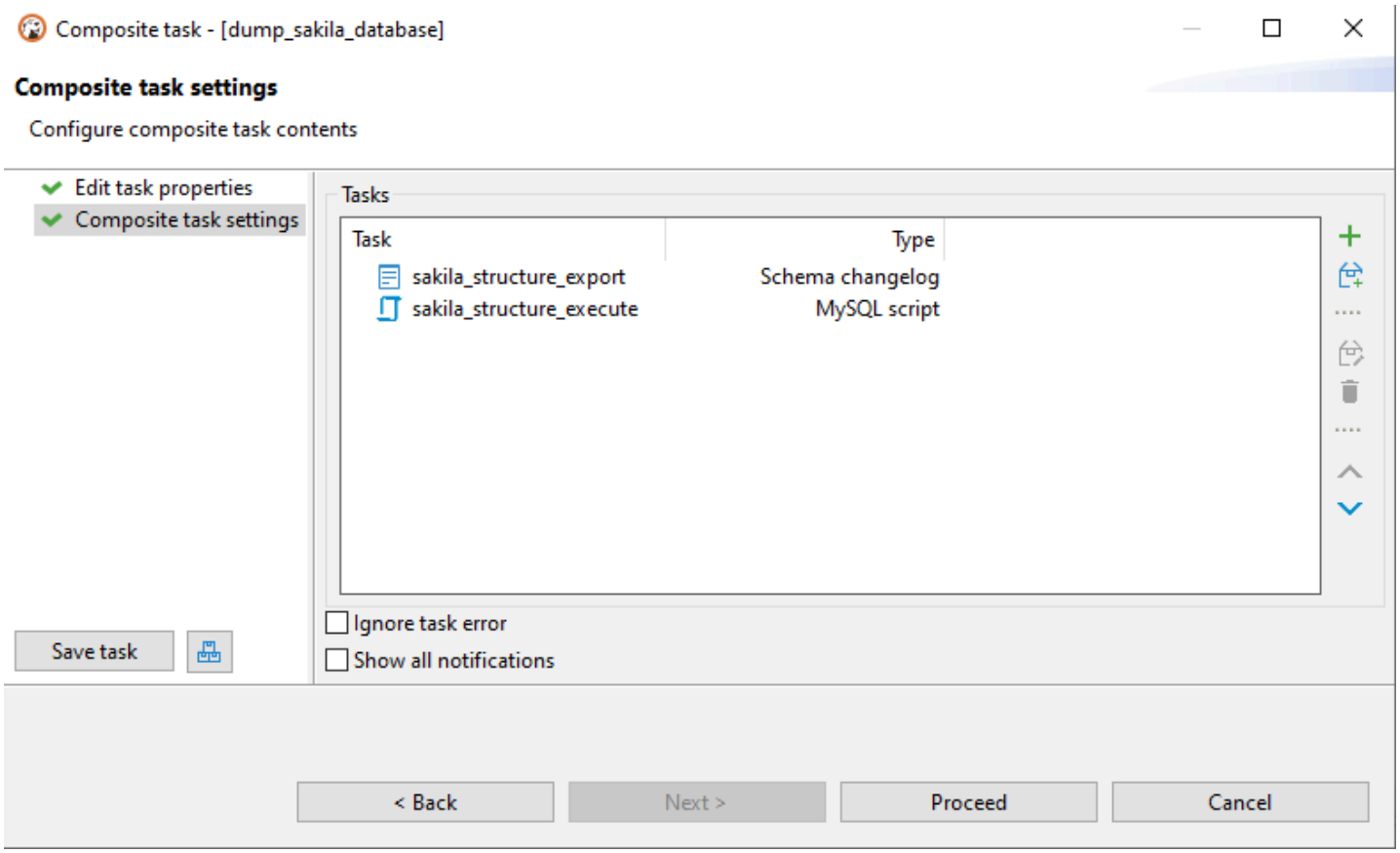


Save operation as a task

If you plan to constantly use the comparison or generation changelog/schema comparison, exporting the result as a file, you can save this as a task. The task can be completed anytime and put on the schedule.



Also, generating a changelog or schema comparison can be part of the composite task. For example, if you keep the log as a script with a certain name, then the implementation of this script can be set by the second operation of the composite task.

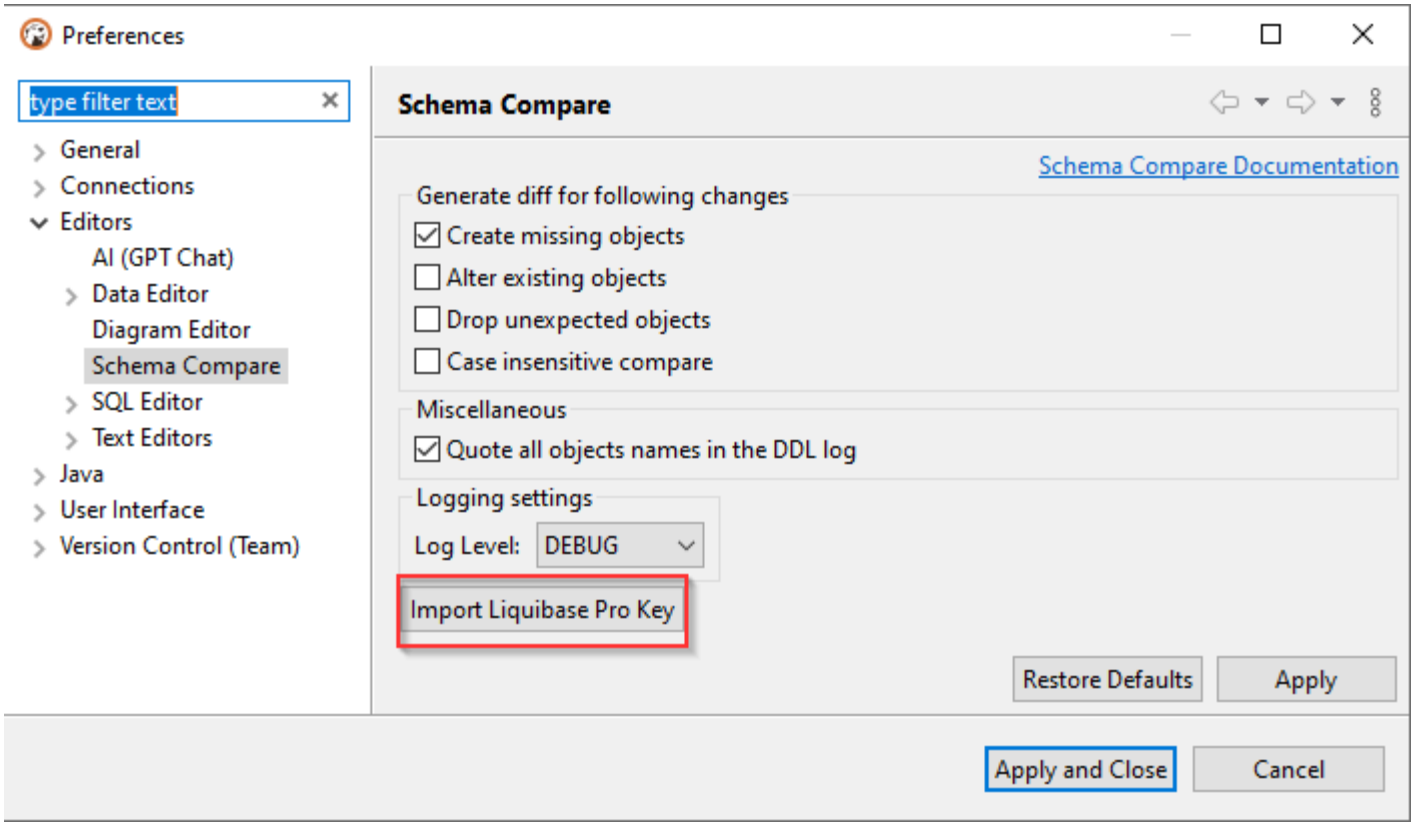


Using schema compare with Liquibase PRO key

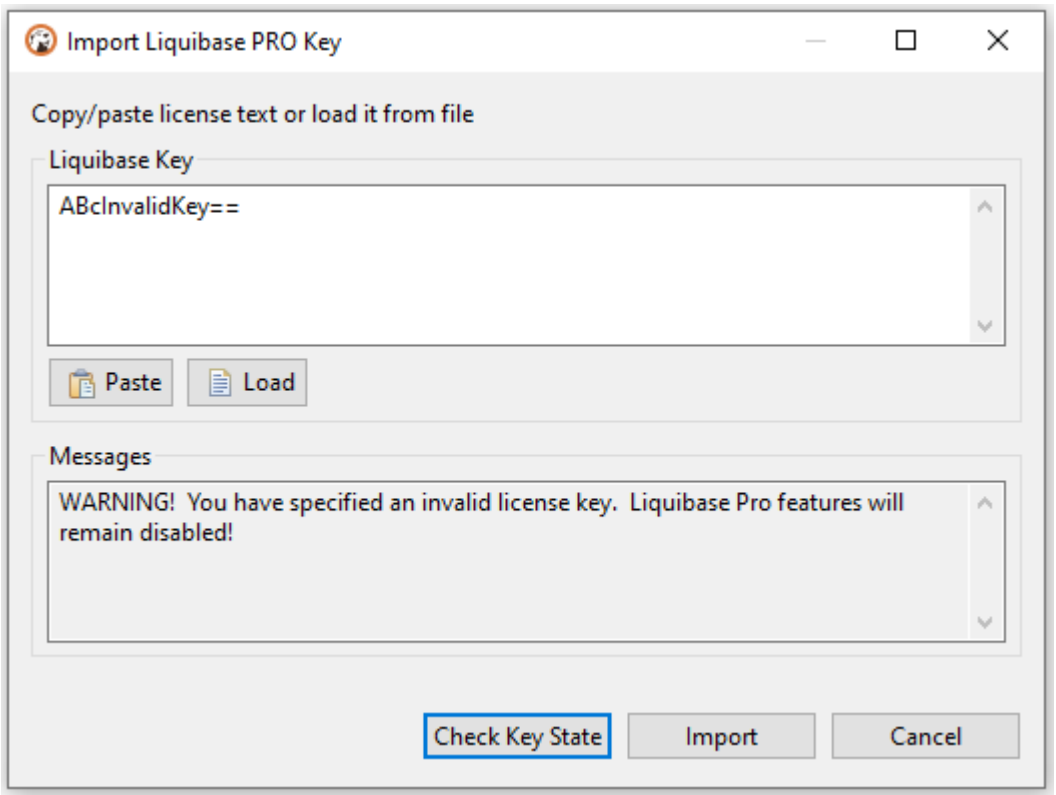
If you have a Liquibase PRO key, then you can use it with DBeaver. Steps you need:

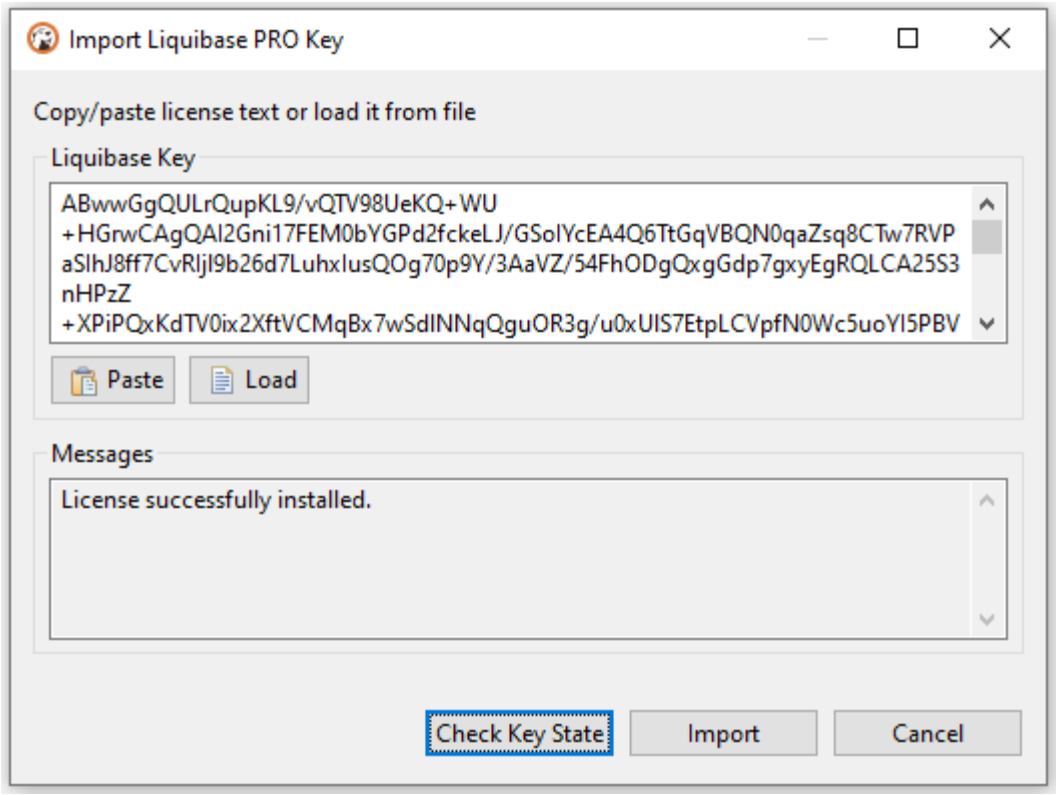
- Find and open your dbeaver.ini file. It is located in the DBeaver root directory.
- Find -vmargs command
- Add a new line after this command: -Dliquibase.license.key=yourKey (example: -Dliquibase.license.key=ABwwGgQU...)
- Open DBeaver and the "Schema compare" window. The key will be checked at this step.

You can also add the Liquibase Pro key via UI in Preferences->Editors->Schema Compare preference page. Use the `Import Liquibase Pro Key` button to open the Import key dialog.



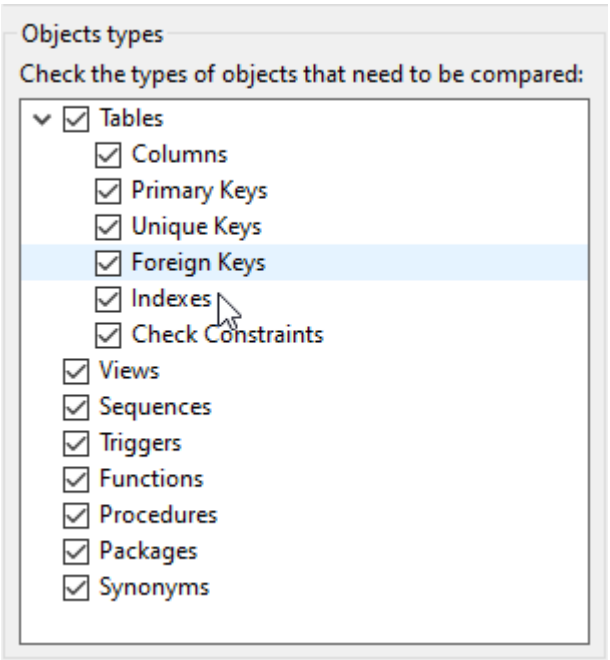
You can manually add your key in the Liquibase Key text field, throw the `Paste` button, or use the `Load` button to download a file. You can check the license state with the `Check Key State` button. After pressing the button, you can see the result of the checking in the `Messages` field.





We suggest you restart the program after adding a key for the correct program work. Settings changes will not be applied to the comparison wizard if it is already open in another window. The key will be saved in the DBeaver settings. If you specify the key in the .ini file and install another key through the Import Key dialog, the key from the .ini file will be in priority.

If the license key is valid, the `Object types` dialog will be extended to PRO objects. (If PRO objects didn't appear in the schema compare changelog - check your logs. Maybe the license expired, or the key is invalid)



Object types being compared by LiquibasePRO

- Check Constraints
- Procedures
- Functions
- Triggers
- Synonyms (Oracle)
- Package with the body (Oracle)

Using Liquibase in DBeaver

Note: This feature is available in Enterprise, Ultimate and Team editions only.

DBeaver integrates with Liquibase, an open-source tool that helps you track, manage, and apply database changes.

Supported Databases

Liquibase works with a variety of databases. The specific databases that DBeaver supports are listed in the Schema Compare article.

Using Liquibase in DBeaver

With DBeaver's Liquibase support, you can efficiently manage your database changes. You can create and manage Liquibase changelogs directly in DBeaver, making tracking and applying database changes easier. With the Liquibase PRO key, you can also use DBeaver's Schema Compare feature for an enhanced comparison of database schemas.

Additional Resources

For more information on Liquibase, refer to these resources:

- How Liquibase Works
- Fundamentals of Liquibase
- Core Liquibase Usage Concepts
- Liquibase Quickstart
- Evolving Your Database Schema Safely with Liquibase

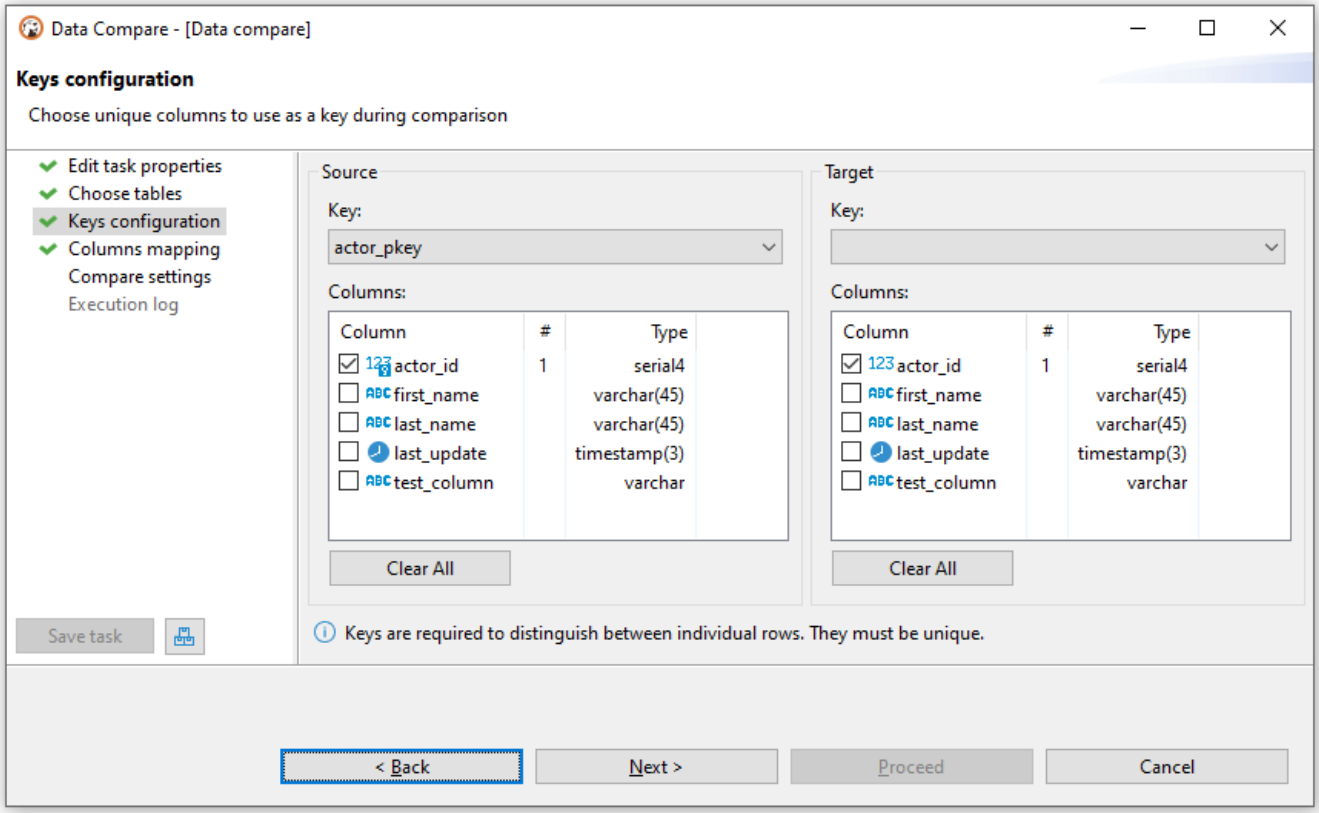
Data compare

Note: This feature is available in Enterprise, Ultimate and Team editions only.

Sometimes you need to compare data from two sources (tables) which have almost identical data with just a few differences. There may be plenty reasons to do so: quickly visualize and navigate through all the differences; copy different rows or individual values; export them using Data Transfer.

Preparing the tool

1. You will need to choose one or two desired tables in the Database Navigator - it may be the tables from the same databases or from different databases or even from different RDBMS (e.g. PostgreSQL and MySQL).
2. Then choose `Database ⇒ Compare/Migrate ⇒ Data Compare` from the menu and the Data Compare Wizard will appear. Here you can preview selected tables or choose other ones. Then you can navigate to the second page.
3. On the second page you should choose the columns that will be used as a **unique key** during the comparison (the amount of columns must be equal). If the *keys* chosen are wrong, it may lead to invalid results. By default, if the table has a unique key in it, it will be chosen automatically during the initial setup:

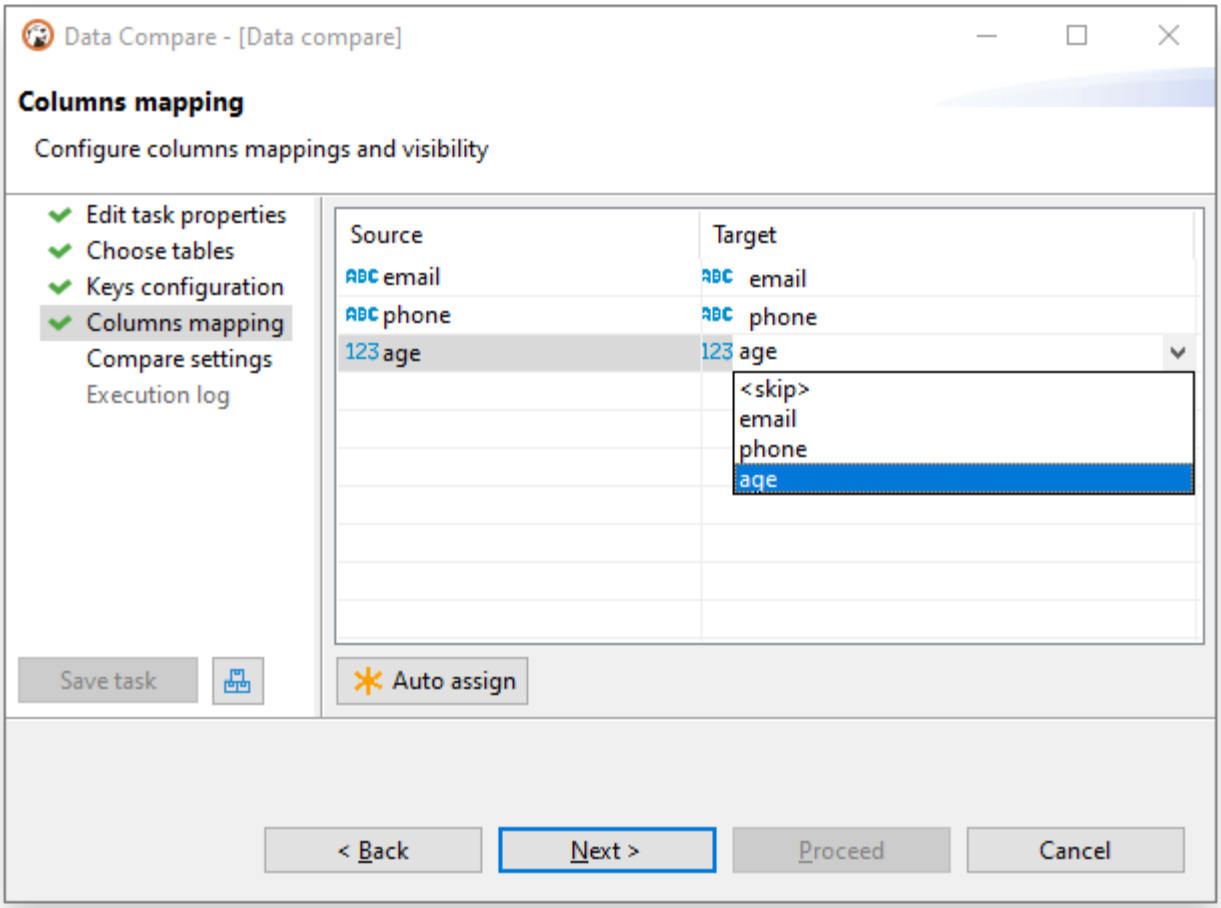


Without unique keys the rows cannot be compared properly, since there is no way to distinguish between individual rows.

4. After reaching the last page, you can tweak limits (e.g. you only care about the `N` first rows) and exclude categories of the resulting rows (e.g. you are not interested in modified rows). Also, you can press the `Save t` button to save the configuration in Task to use it later, or Schedule it.

Changing columns mapping

If tables you want to compare have different ordering, naming or quantity of columns, you can tweak their mappings and/or entirely skip columns you're not interested in:



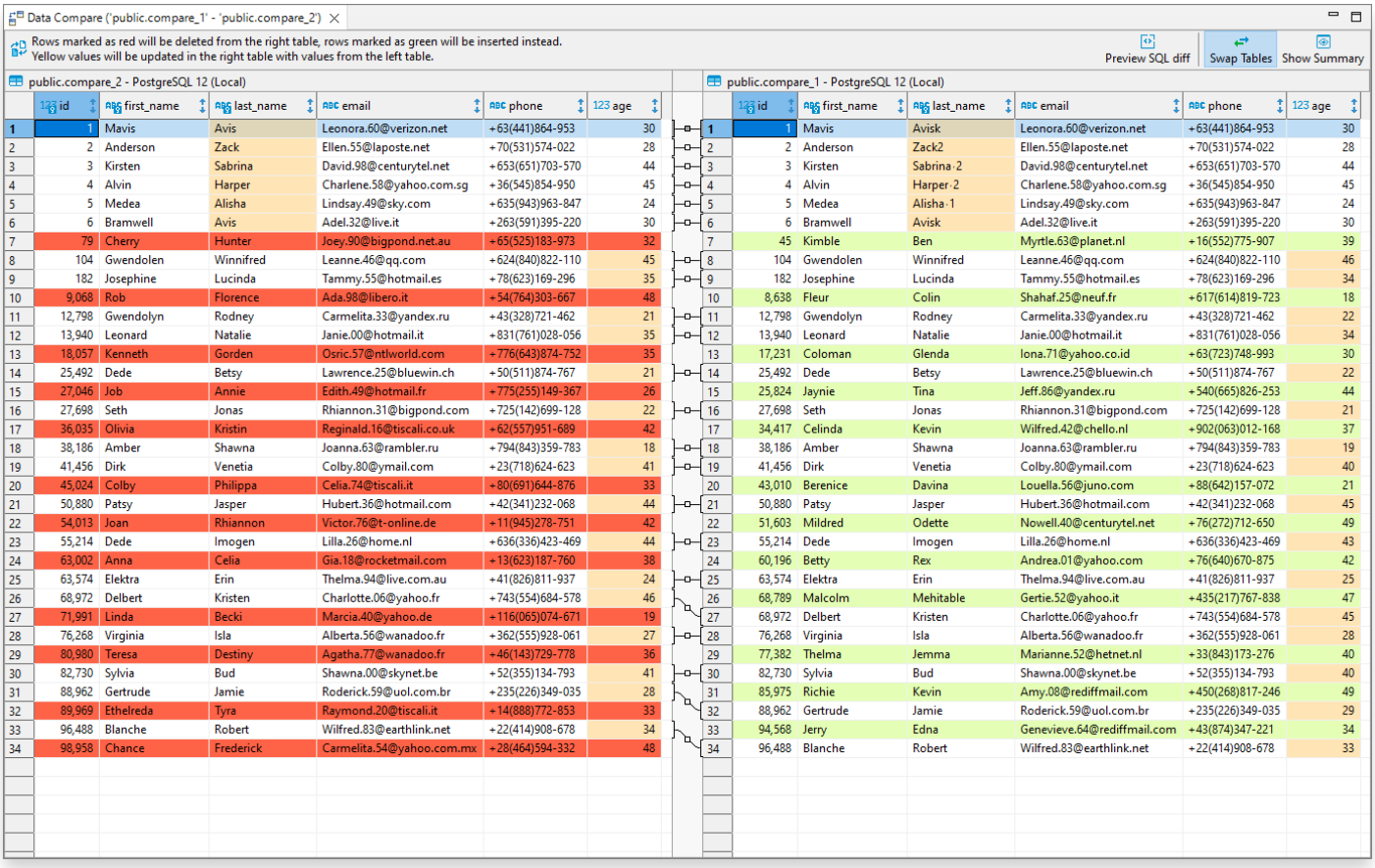
Please note that this page only features columns that aren't used as primary key configured on a previous page.

Pressing the **Auto assign** button will attempt to link columns with same names.

Viewing the results

After finishing the wizard, you can press the `Start` button to begin the actual comparison process - it may take some time depending on the databases you are comparing, the amount of data in them, and your network speed (*).

When the results are ready, DBeaver will play a *beep* sound, and the editor will open:



Here you can examine the differences, swap the panels using the `Swap Containers` button in the toolbar, or preview the summary including the statistics by using the `Show Summary` button in the toolbar too. Everything you can do in the regular data editor is possible to perform here - you can copy data or transfer it to another database, except modify the values.

** - Please note that the actual preview is only available when Data Compare is launched through the wizard - otherwise only the statistics will be written into the Task Log.*

Exporting the results

You can export results as an SQL script containing statements that will turn the second table into the first one by `DELETE`-ing unrelated rows, `INSERT`-in new ones, and `UPDATE`-ing existing ones, so the data will look the same.

You can export results either by:

1. Setting the **Export compare results to file** option found on the last page
2. Pressing the **Preview SQL diff** button found in the compare viewer

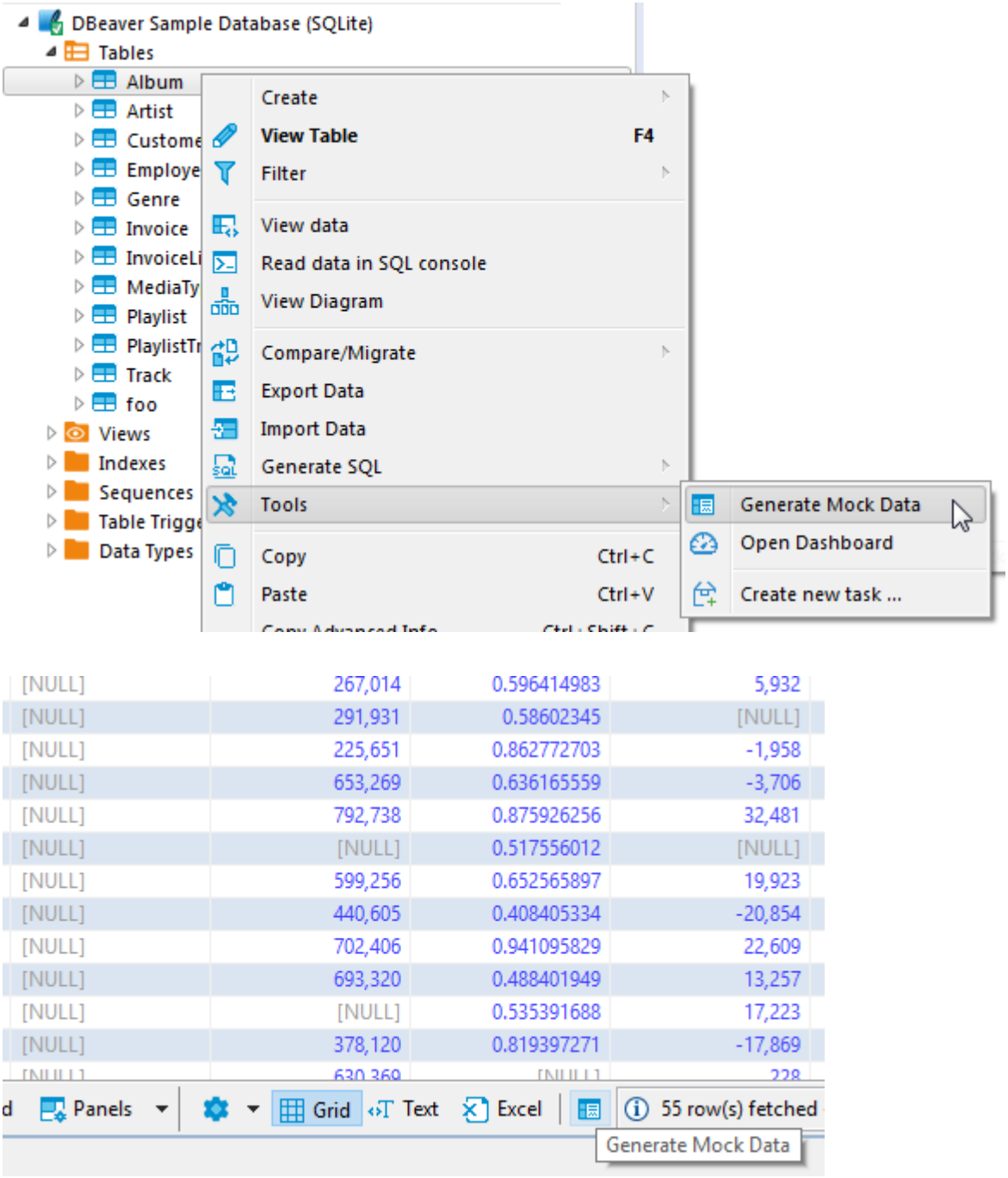
Under the hood

This section is under construction. It will include information about the engine's implementation details and more.

MockData generation

Note: since version 6.2 MockData generator extension is available only in Enterprise, Ultimate, and Team editions.

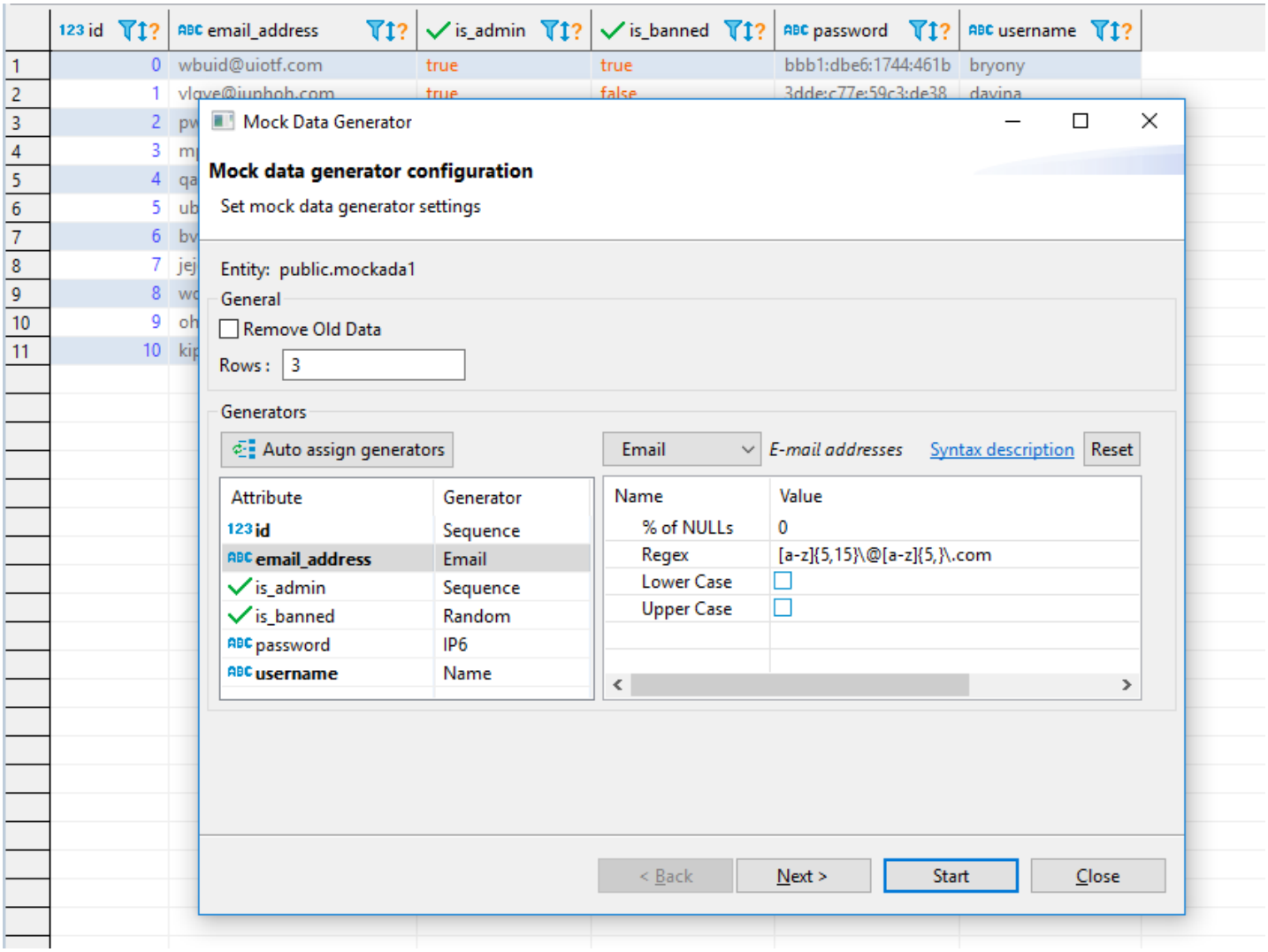
Sometimes in software development we need to generate mock, but valid, data for testing. Populating a database manually is a time-consuming and exhausting process. It can be very complicated when you need to generate not just 5–10 users, but thousands of entities of different types. DBeaver Mock Data generator helps you generate test data much easier.



> *Disclaimer:* The idea behind Mock Data is to generate mock data in a table but it should **NOT TO BE USED IN PRODUCTION ENVIRONMENTS**. Please make sure you have a backup of your database before running the Mock Data generation process.

Th following are features of the DBeaver Mock Data generator:

- Works for all the RDBMS that are supported by DBeaver (DB2, MS SQL Server, MySQL, Oracle, PostgreSQL, SQLite, etc.)
- Generates data that matches your database schema:
  - Generated data matches the database column types.
  - All base data types are supported.
  - Constraints (PK, FK, multi-column FK, unique) are supported.
- Supports over 20 configurable data generators (constants, randoms, sequences, names, domains, addresses, prices, regex based, etc.)
- Automatically associates a column with a generator based on the column characteristics
- Saves or overwrites old database data



The following are mock data generators for data types with their configurable parameters:

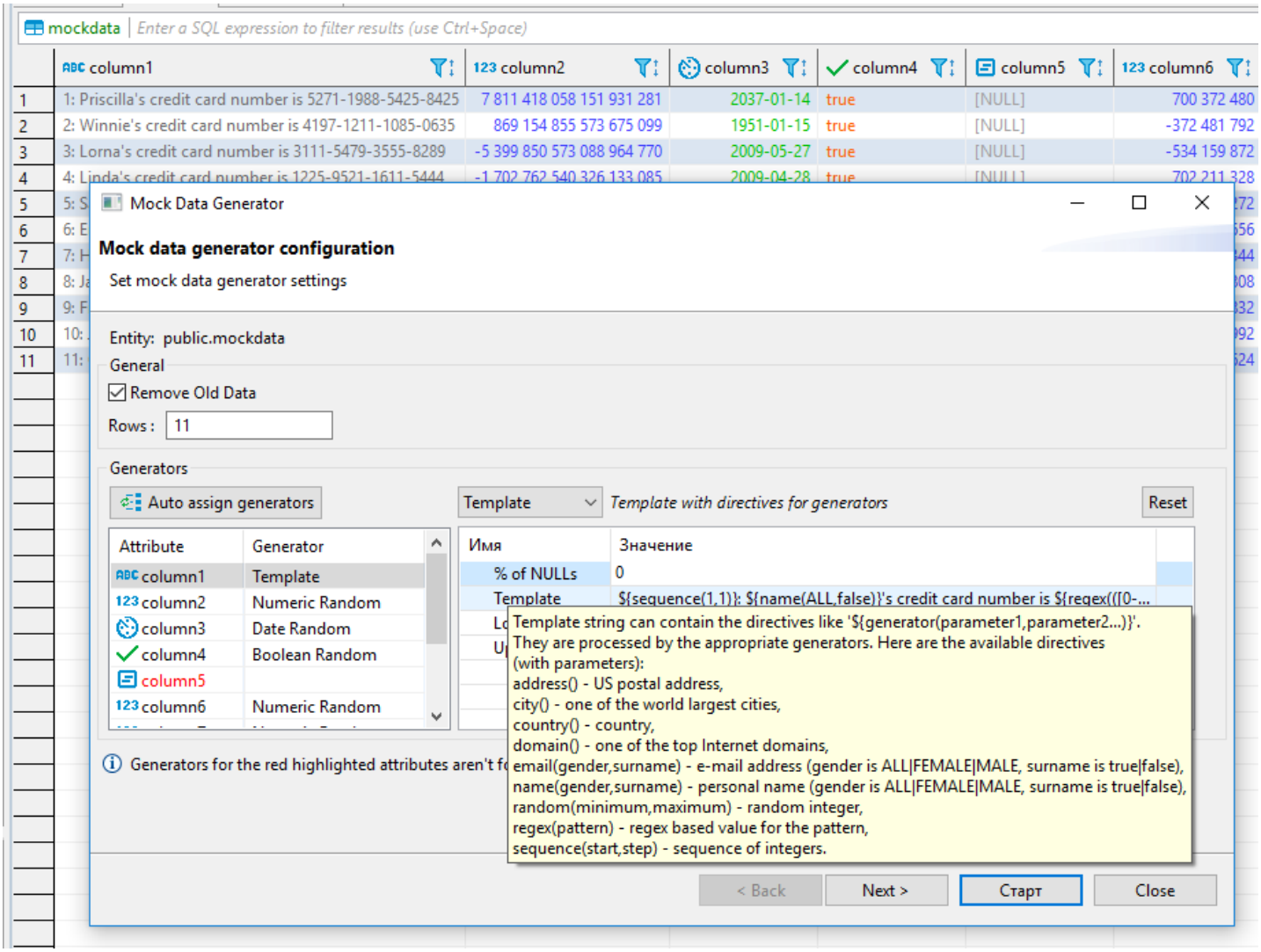
- Boolean
  - Random
  - Sequence (initial, order)
- Date
  - Random (start, end)
  - Sequence (start, step, reverse)
- Numeric
  - Random
  - Sequence (start, step, reverse)
  - Advanced (min, max, precision, scale) *

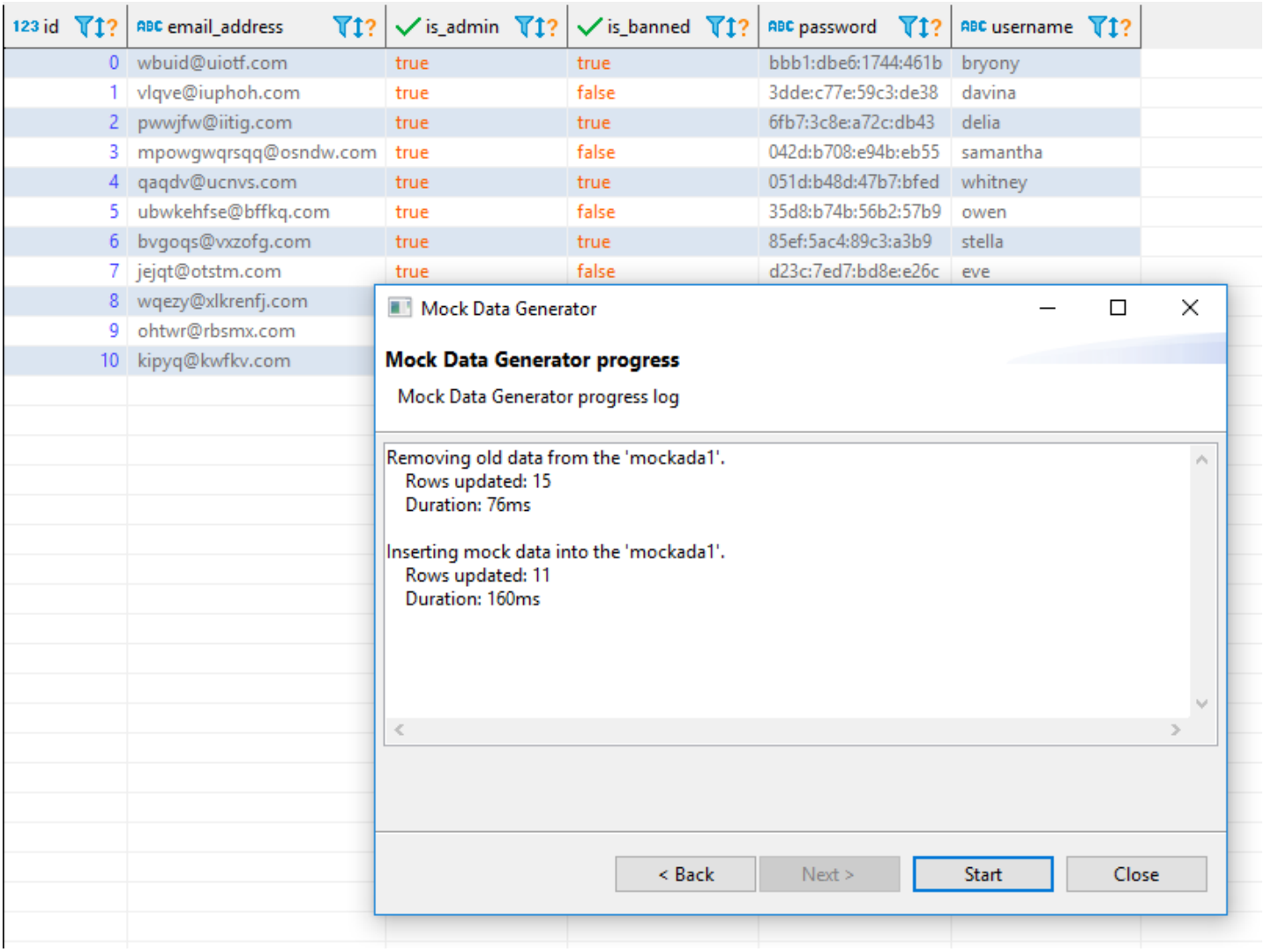
        - Price preset *
        - Coordinate preset *
- String
    - Text (template, min length, max length)
    - UUID
    - Address *
    - City *
    - Country *
    - Domain *
    - Email (gender, with surname, numeric suffix) *
    - Name (gender, with surname) *
    - Price (country, min, max) *
    - Regex based random (regex template) *
        - Credit Card preset *
        - Email preset *
        - Gender preset *
        - HEX Color preset *
        - IP4 address preset *
        - IP6 address preset *
        - Phone Number preset *

- Postal Code preset *
- Price preset *

- Template with parametrized directives for other generators *:
  - address() - US postal address
  - city() - one of the world's largest cities
  - country() - country
  - domain() - one of the top Internet domains
  - email(gender,surname) - e-mail address (gender is ALL|FEMALE|MALE, surname is true|false)
  - name(gender,surname) - personal name (gender is ALL|FEMALE|MALE, surname is true|false)
  - random(minimum,maximum) - random integer
  - regex(pattern) - regex based value for the pattern
  - sequence(start,step) - sequence of integers

- NULL values
- FK - data from the referenced table according to the constraint


mockdata
Enter a SQL expression to filter results (use Ctrl+Space)
column1
column2
column3
column4
column5
column6
1: Priscilla's credit card number is 5271-1988-5425-8425
7 811 418 058 151 931 281
2037-01-14
true
[NULL]
700 372 480
2: Winnie's credit card number is 4197-1211-1085-0635
869 154 855 573 675 099
1951-01-15
true
[NULL]
-372 481 792
3: Lorna's credit card number is 3111-5479-3555-8289
-5 399 850 573 088 964 770
2009-05-27
true
[NULL]
-534 159 872
4: Linda's credit card number is 1225-9521-1611-5444
-1 702 762 540 326 133 085
2009-04-28
true
[NULL]
702 211 328
Mock Data Generator
Mock data generator configuration
Set mock data generator settings
Entity: public.mockdata
General
Remove Old Data
Rows : 11
Generators
Auto assign generators
Template
Template with directives for generators
Reset
Attribute
Generator
column1
Template
column2
Numeric Random
column3
Date Random
column4
Boolean Random
column5
column6
Numeric Random
Имя
Значение
% of NULLs
0
Template
${sequence(1,1)}: ${name(ALL,false)}'s credit card number is ${regex(([0-...
Template string can contain the directives like '${generator(parameter1,parameter2...)}'.
They are processed by the appropriate generators. Here are the available directives
(with parameters):
address() - US postal address,
city() - one of the world largest cities,
country() - country,
domain() - one of the top Internet domains,
email(gender,surname) - e-mail address (gender is ALL|FEMALE|MALE, surname is true|false),
name(gender,surname) - personal name (gender is ALL|FEMALE|MALE, surname is true|false),
random(minimum,maximum) - random integer,
regex(pattern) - regex based value for the pattern,
sequence(start,step) - sequence of integers.
Generators for the red highlighted attributes aren't f
< Back
Next >
Старт
Close



* These features are available in Enterprise and Ultimate editions only.

Spelling

Note: This feature is available in Lite, Enterprise, Ultimate and Team editions only.

DBeaver's spelling function is a feature designed to detect spelling errors. It allows you to change the incorrect words, upload custom dictionaries in any language and it provides you with advanced options for a more refined text management experience.

Usage

The spelling function is accessible in your SQL script files. Words that are not spelled correctly are underlined, which will alert you to potential inaccuracies.

To change these incorrect words, open the context menu by right-clicking on the word and selecting **Quick Fix** or by using the `Ctrl+1` (or `#1` for Mac OS).



The context menu offers several options:

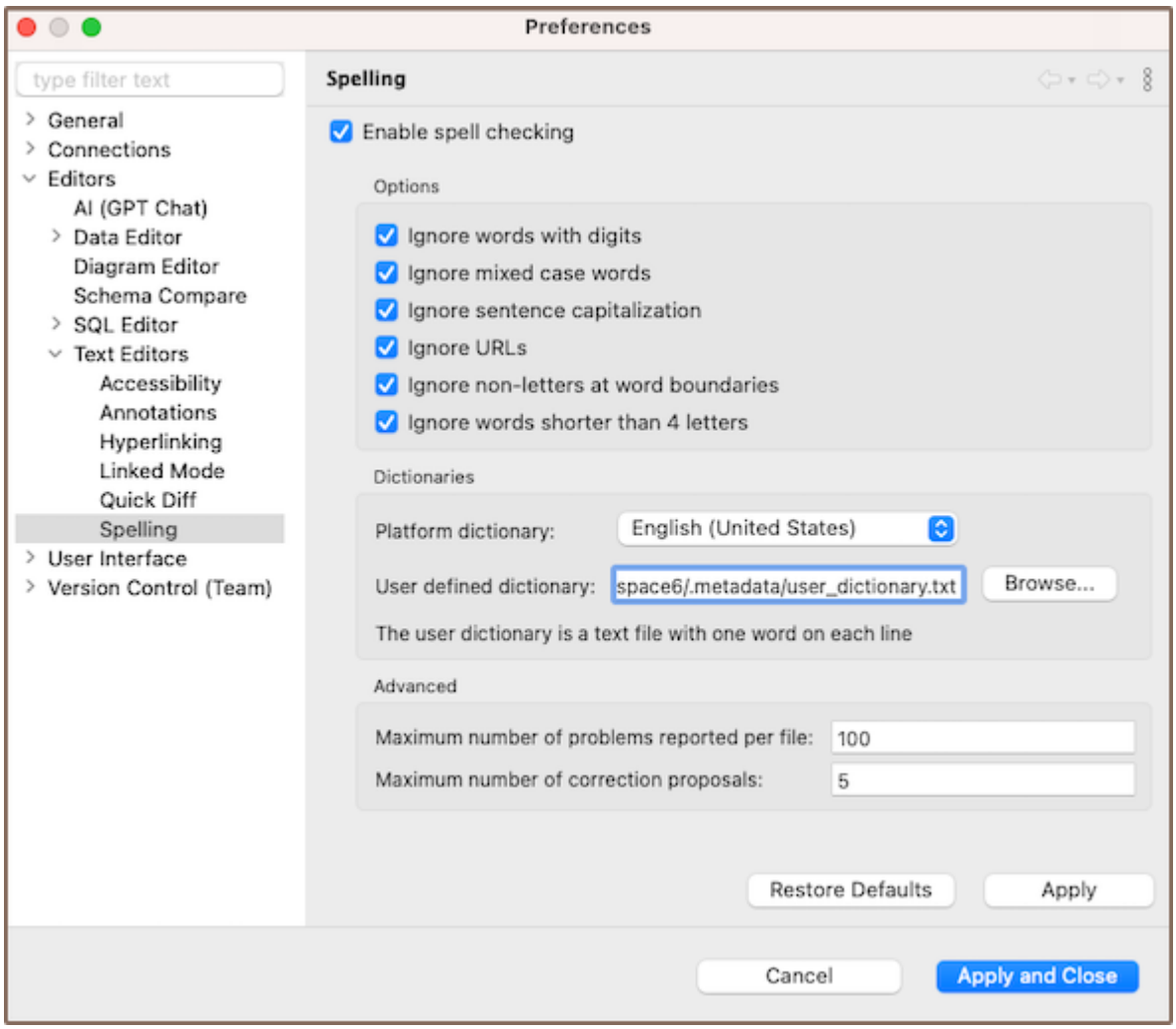
- Change the incorrect word with to a suggested word.
- Add the word to the dictionary.
- Ignore the word during the current session.

- Disable the spell checking entirely.

To apply any of these changes, you need to double-click on the chosen option with the left mouse button or press the `Tab` key.

Settings

To open the settings for the spelling page, navigate through the following steps: **Window** -> **Preferences** -> **Editors** -> **Text Editors** -> **Spelling**.



Activation

The spelling function is enabled by default but can be customized to your requirements.

You can disable the spelling function by unchecking the box next to **Enable spell checking** in Preferences.

Options

DBeaver's spelling settings provide specific options for this feature that can be tailored to your needs.

Option name	Description
Ignore words with digits	Skips words containing numbers.
Ignore mix case words	Skips words with combined upper and lower case letters.
Ignore sentence capitalization	Skips words starting with a lowercase letter at sentence beginnings.
Ignore URLs	Excludes web addresses from spell checking.
Ignore words shorter than 4 letters	Disregards shorter words from spell checking.

Custom Dictionaries

DBeaver includes an integrated English dictionary. However, you can choose to disable it and connect your custom dictionary, or use both simultaneously, according to your needs.

In DBeaver's spelling settings you can add your own custom dictionary by specifying the path to it in the **User defined dictionary** field. This dictionary should be a **.txt** file with one word per line.

To disable the custom dictionary, clear the **User defined dictionary** field and apply the changes.

Advanced Settings

You can adjust the **Maximum number of correction proposals**. This setting controls the number of suggested substitutions that appear in the context menu for each misspelled word.

Also, you can specify the **Maximum number of problems reported per file**. This setting determines the total number of words, starting from the first one, that the system will flag for replacement in each file.

Dashboards, DB monitoring

Dashboards tool allows DBAs and programmers to quickly identify performance, disk space issues, the number of connections and other important KPIs associated with a single database connection. To learn more about database connections, see Database Connections.

By default, DBeaver is delivered with a number of predefined sets of dashboards for such data bases as PostgreSQL, MySQL, Oracle and Exasol. Custom dashboards are also supported. To learn more about custom dashboards, see the Managing Dashboards section below.

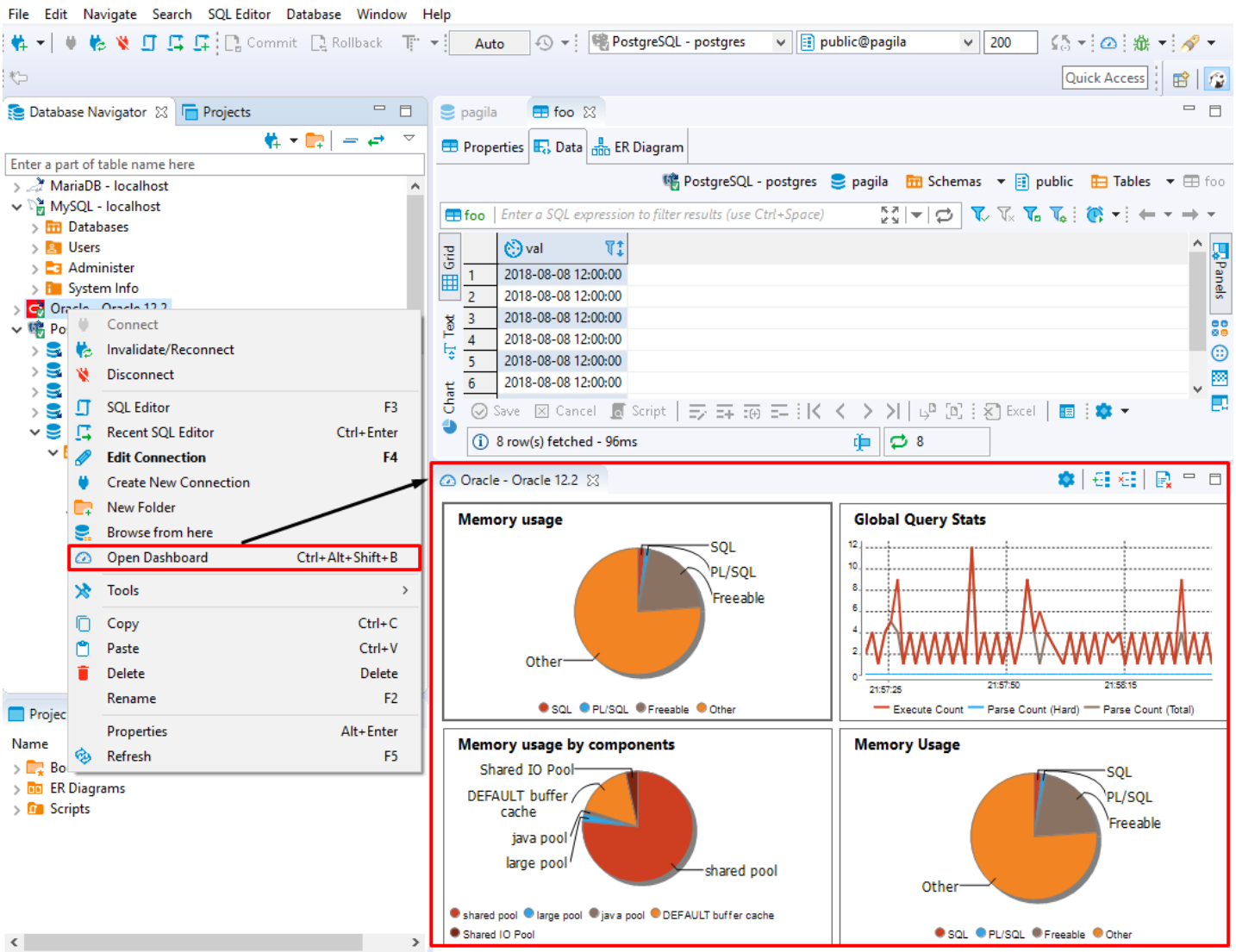
Managing Dashboards Panel

Dashboards panel is a collection of real-time dashboards, which are dashboards that are updated continuously. Dashboards displayed on the dashboards panel are actually a combination of continiously run SQL SELECT queries and charts continiously built on the data fetched.

Opening Dashboard Panel

To open the dashboards panel press the **Open Dashboard** button  on the main toolbar. The default configuration of the dashboards panel for the current database connection will appear. To learn more about database connections, see Database Connections.

You can also right-click a connection name in the **Database Navigator** editor and select the **Open Dashboard** menu option or use the keyboard shortcut `Ctrl + Alt + Shift + B` and the dashboards panel will be opened.

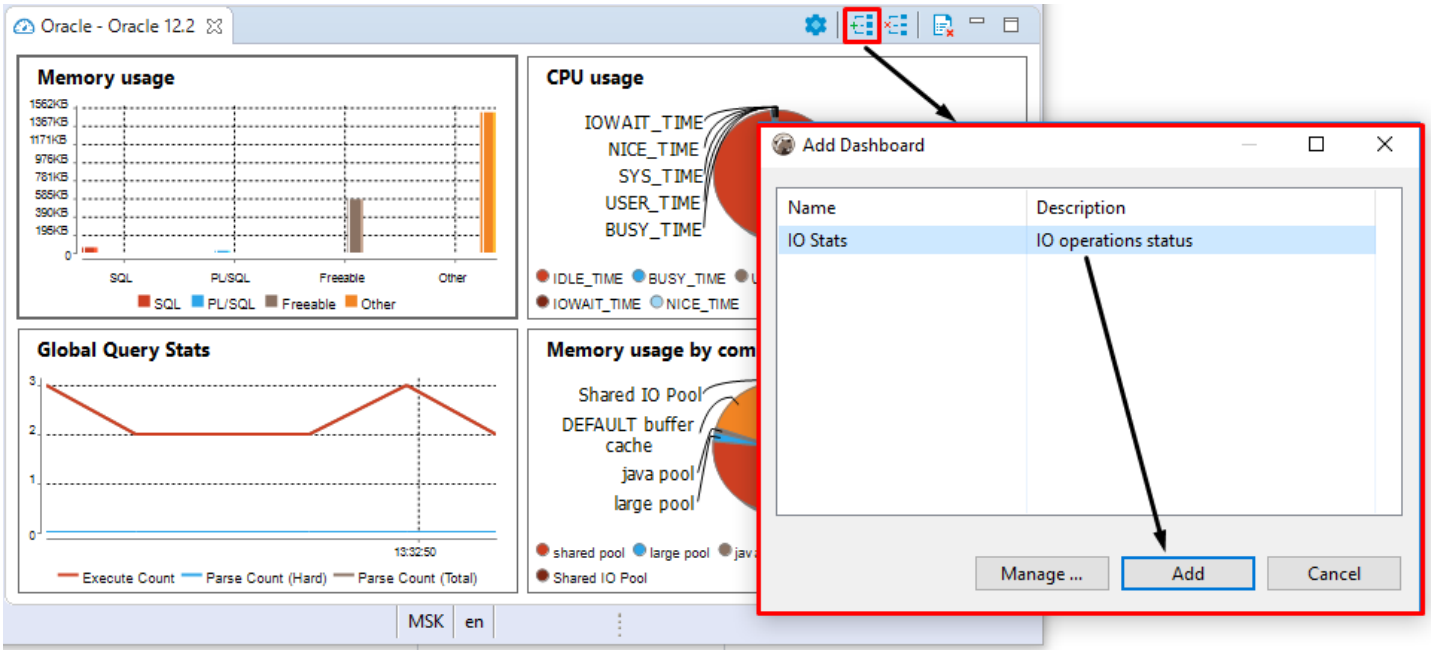


The following controls are available on the dashboards panel toolbar:

Icon	Name	Description
	Settings	Opens dashboard's configuration.
	Add dashboard	Adds dashboard to the dashboard panel.
	Remove dashboard	Removes dashboard from the dashboard panel.
	Reset dashboards	Restarts dashboard's calculation.

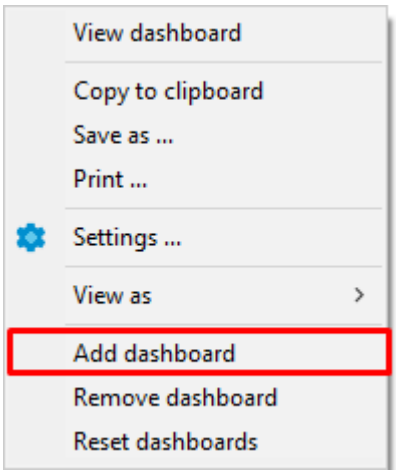
Adding Dashboards

To add a dashboard to the dashboards panel, press **Add dashboard** button on the dashboards panel's toolbar, choose one of the dashboards from the list of available dashboards and press the **Add** button.



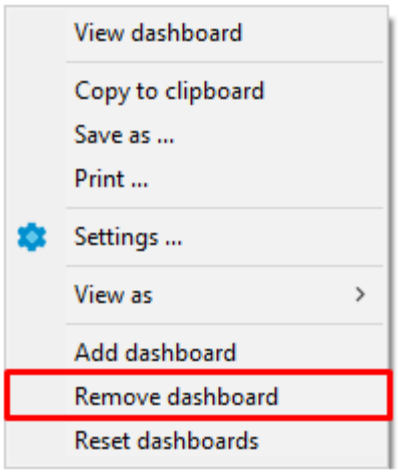
Note: Different databases have different sets of predefined dashboards. DBeaver is delivered with sets of predefined dashboards for such databases as Postgress SQL, MySQL, Oracle, and Exasol. It is also possible to create new custom dashboards, for more details see Managing Dashboards.

You can also add a dashboard by right-click in any place of the dashboards panel and then select the **Add dashboard** menu option.



Removing Dashboards

To remove a dashboard from the dashboards panel, click on the dashboard you want to remove and press button **Remove dashboard** in the dashboards panel toolbar or select **Remove dashboard** option in the dashboard's context menu.

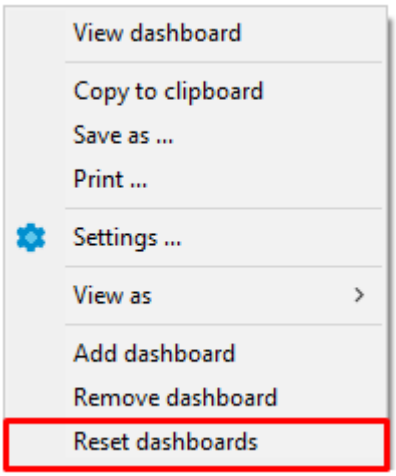


Resetting Dashboards

If you want to restart the dashboard's calculation you can reset it.

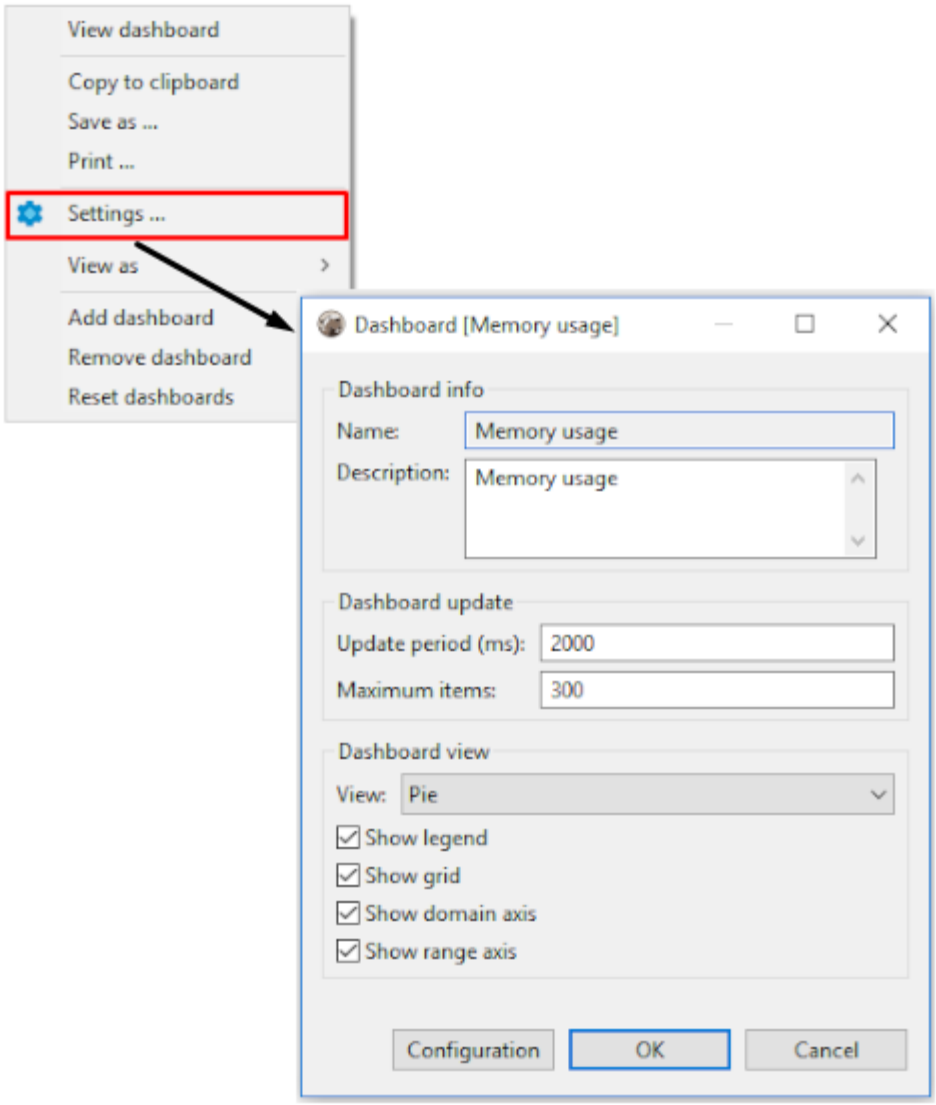
You can reset all the dashboards displayed on the dashboards panel by a single click on **Reset dashboards** button on the dashboard panel's toolbar.

To reset a particular dashboard right-click on it and select **Reset dashboards** menu option or left click a dashboard and press **Reset dashboards** button on the dashboards panel's toolbar.



Changing Dashboard Representation

To adjust dashboard representation settings, right click on a dashboard and select the **Settings** menu option, then, in the opened dialog change the parameters you want.



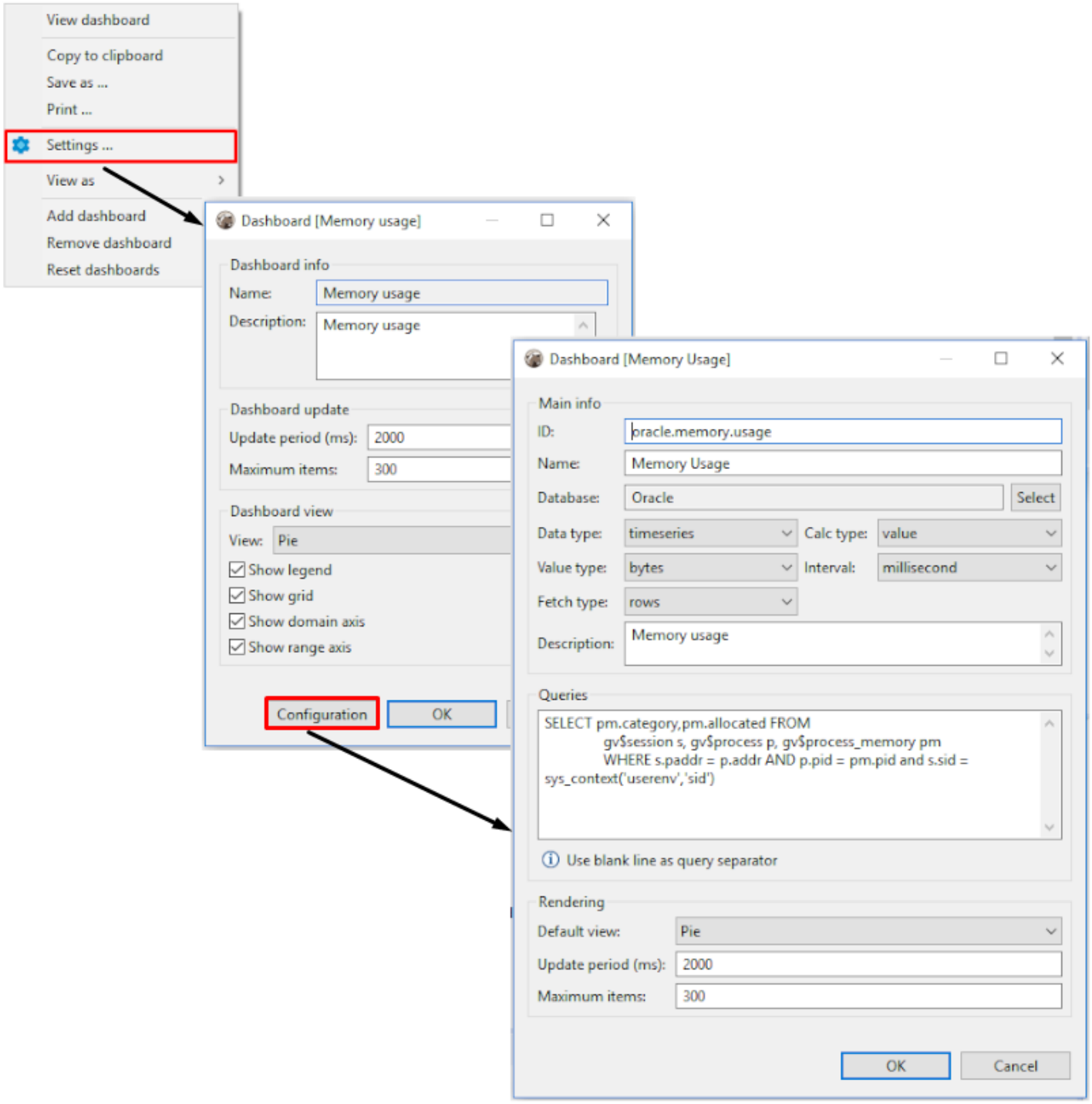
The following dashboard representation parameters can be adjusted:

Parameter	Description
Name	Defines a name of a dashboard.
Description	Defines dashboard's description. Use this field to make it easy to understand what kind of information the dashboard represents.
Update periods (ms)	Defines how often dashboard's rendering should be updated. The default value is 1000 ms.
Maximum items	Defines maximum number of fetched items. The default value is 300.
View	Defines visual representation of the dashboard. The following options are available: Bar, Pie, Time series.
Show legend	If this check-box is selected, the legend will be displayed on the dashboard.
Show grid	If this check-box is selected, the grid will be displayed on the dashboard.

Show domain axis	If this check-box is selected, the domain axis will be displayed on the dashboard.
Show range axis	If this check-box is selected, the range axis will be displayed on the dashboard.

Adjusting Dashboard Configuration

To adjust dashboard's configuration settings right-click on a dashboard, select the **Settings** menu option, then, in the opened dialog box press the **Configuration** menu option.



The following dashboard parameters can be configured:

Parameter	Description
ID	Defines dashboard's ID. Make sure that ID has numeric values in it.
Name	Defines dasboard's name.
Database	Defines the database driver. To learn more about database drivers, see Database Drivers.
Data type	Defines the data type. The following options are availabe: timerseries (the default option) and statistics. Select timeseries type if you want to track the actual value returned by the server. Select the statistics type if your dashboard will show historical data.

Calc type	Defines how the data should be calculated. The following options are available: value (the default option) and delta. Select value if you are interested in the current value. Select delta if you want to track the difference between the current value and the previous one. This may be very useful when you work with statistics data, for example.
Value type	Defines the value to be shown on the range domain. The following options are available: decimal (the default option), integer, percent, bytes. Choose the value type in accordance with your data, for example, memory usage is convinient to be tracked in KBytes.
Interval	Defines time interval to be shown on the domain axis. The following time intervals are available: millicecond(the default option), second, minute, hour, day, week, month, year.
Fetch type	Defines whether the query should fetch data from rows or columns.
Description	Defines the description of a dashboard. Use this field to make it easy to understand what kind of information the dashboard represents.
Queries	Defines an SQL query whose fetched data will be used to build the chart displayed on the dashboard.
Default view	Defines the default visual representation of a dashboard on the dashboard panel. The following options are available: Bar, Pie, Time series(the default option).
Update period(ms)	Defines how often the dashboard's rendering should be updated.
Maximum items	Defines the maximum number of items to be fetched for the dashboard.

Note: Predefined dashboards are read-only and cannot be re-configured, but you can copy them and use them templates to create new dashboards with any query and other settings. To learn about creating new dashboards, see the Managing Dashboards section.

Setting Connection Prefereces

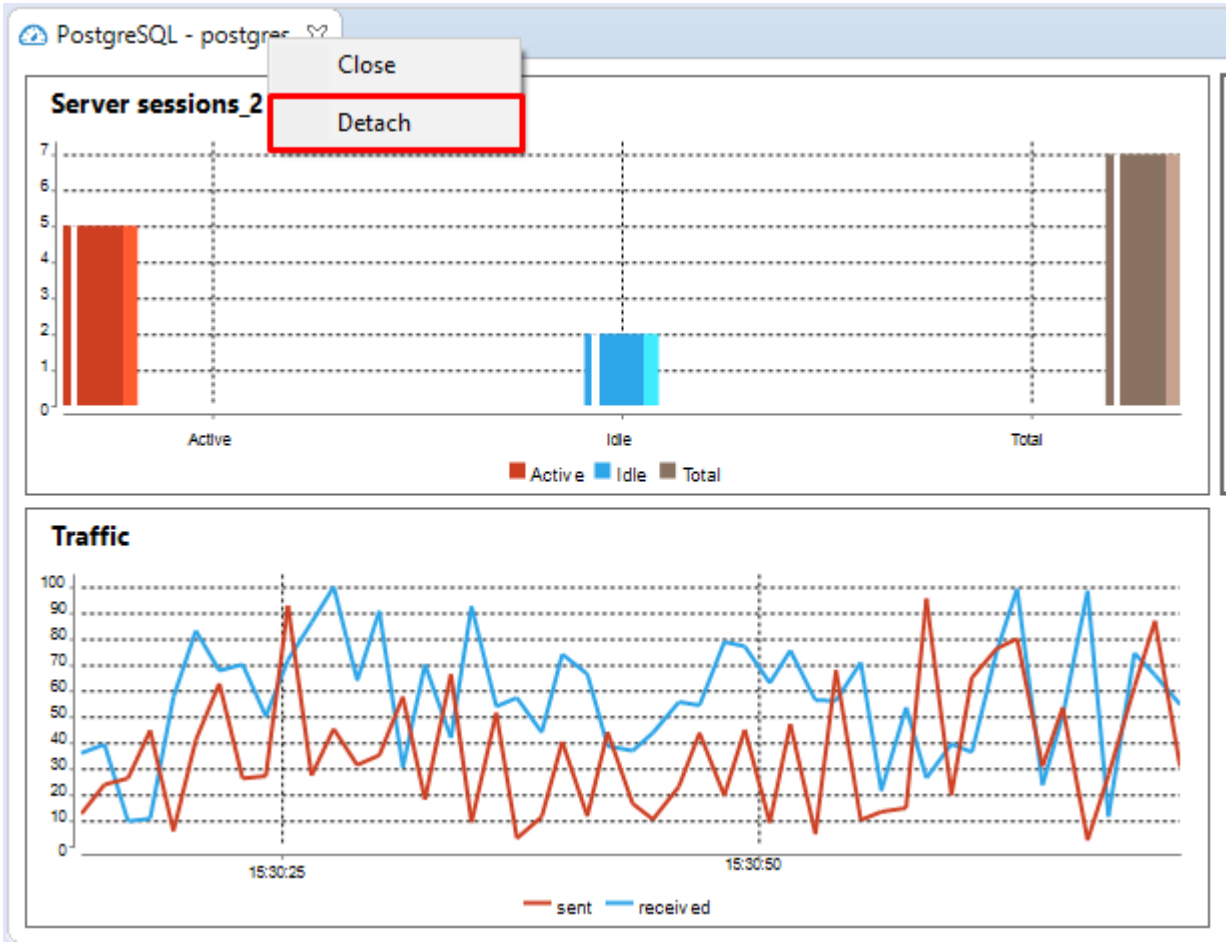
By default, if there is no active connection to the database and you open its dashboards panel, all the dashboards on the panel will be empty.

You can force a database connection on the dashboard panel's activation by pressing the **Settings** button  on the dashboards panel's toolbar and then selecting the **Connect on activation** check-box.

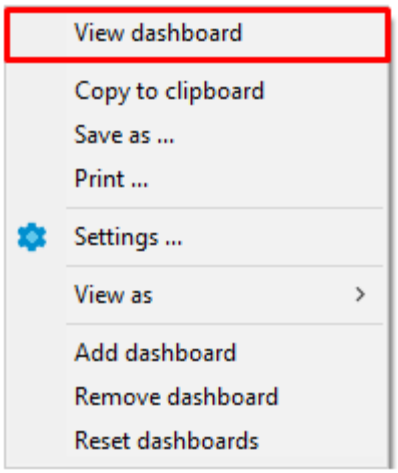
Detaching Dashboards

If you have several monitors and would like to place a dashboard into a separate screen, you can either detach the whole dashboards panel or a single dashboard, and drag-and-drop them to any place you want.

To detach the whole dashboard panel, right click on the dashboard's tab name and select the **Detach**menu option.

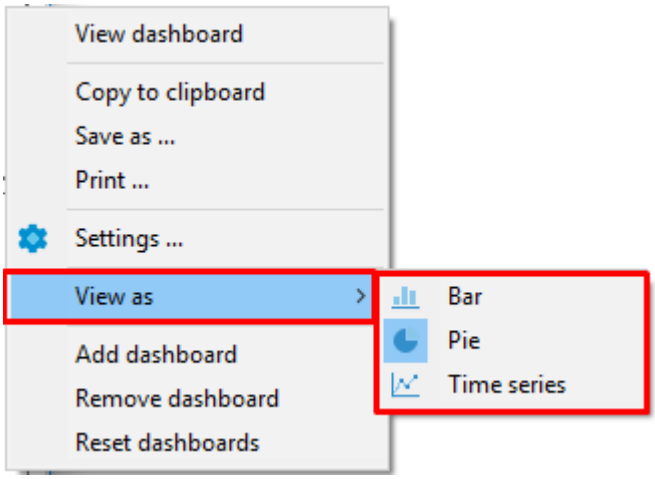


To detach a single dashboard, double left click over it. You can also right click the dashboard and then, select the **View Dashboard** menu option, the dashboard will be detached from the panel and you will be able to move it to any place on your screen.



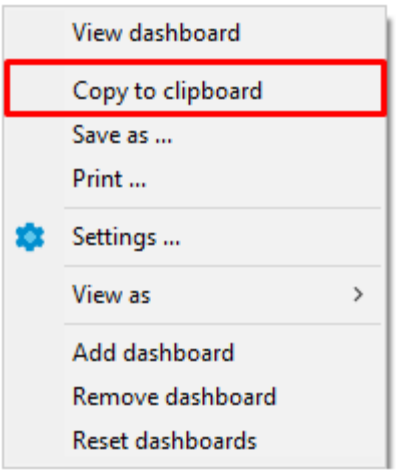
Changing Dashboard View

You can change the representation of a dashboard and view it as a Pie, Bar or Time series. To change the dashboard view, right click on it and select the **View as** menu option.



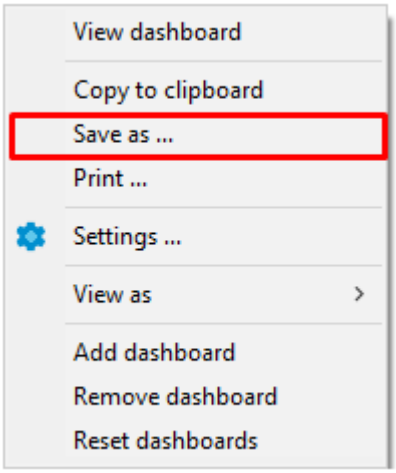
Copying Dashboards to Clipboard

To copy a dashboard onto the clipboard, right click on the dashboard and use the **Copy to Clipboard** menu option. The screenshot of the dashboard will be placed onto the clipboard.



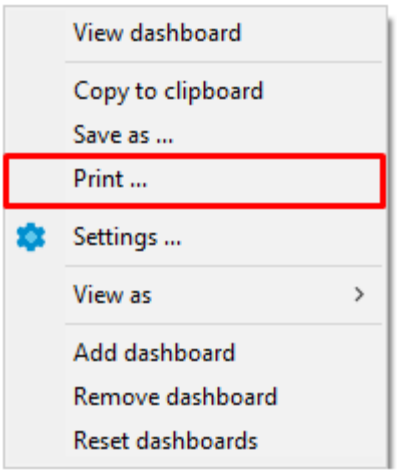
Saving Dashboards

If you want to save a screenshot of a dashboard locally in PNG format, right click on it and select the **Save as ...** option in the context menu displayed.



Printing Dashboards

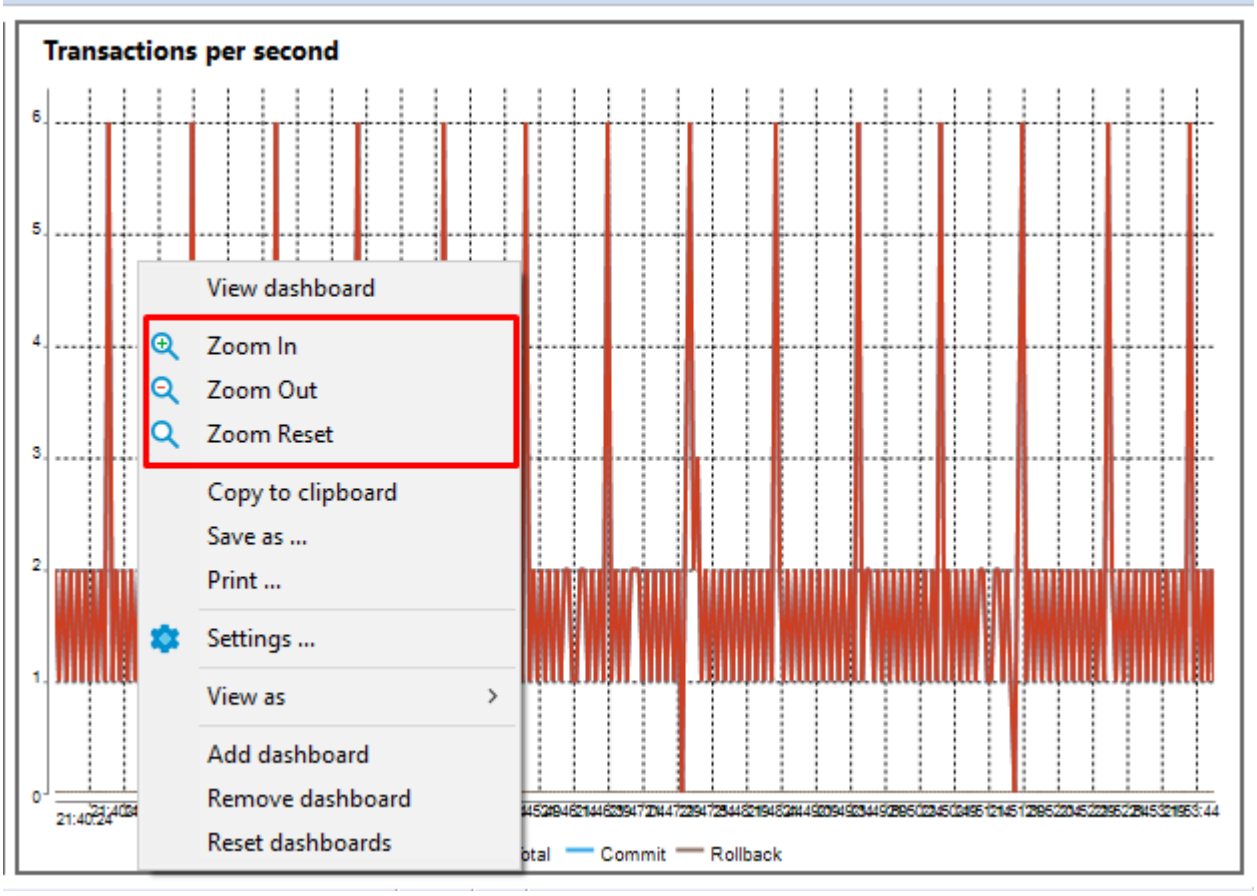
If you want to print out a screenshot of a dashboard, right-click the dashboard to be printed and select the **Print...** option.



Zooming

For Time series and Bar dashboard representations the following zooming options are available on the dashboard's context menu:

- Zoom In
- Zoom Out
- Zoom Reset



Managing Dashboards

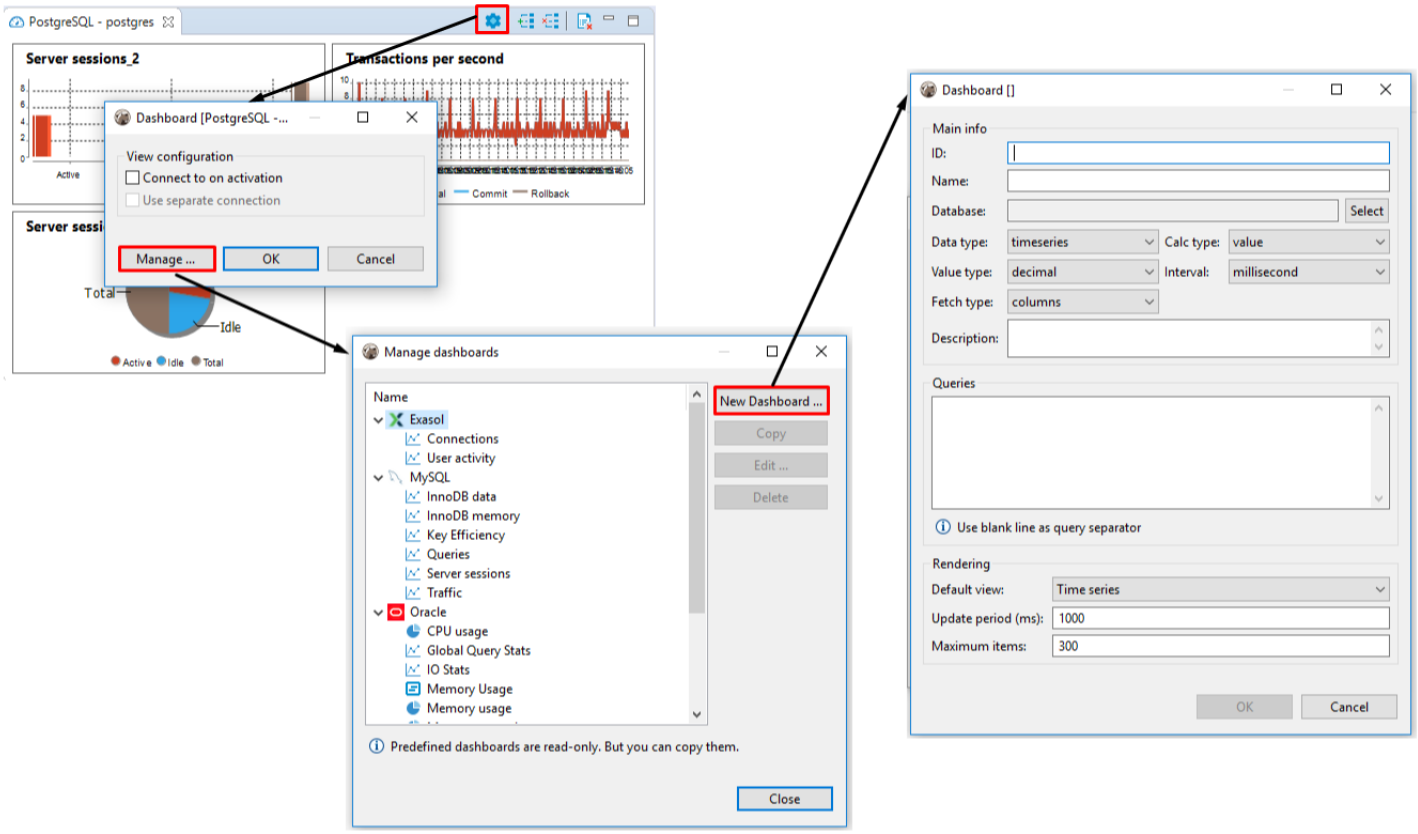
You can extend the list of predefined default dashboards by creating your own custom dashboards. This section describes dashboards' list management.

Creating Dashboards

You can create a new custom dashboard either from scratch or from any existing dashboards.

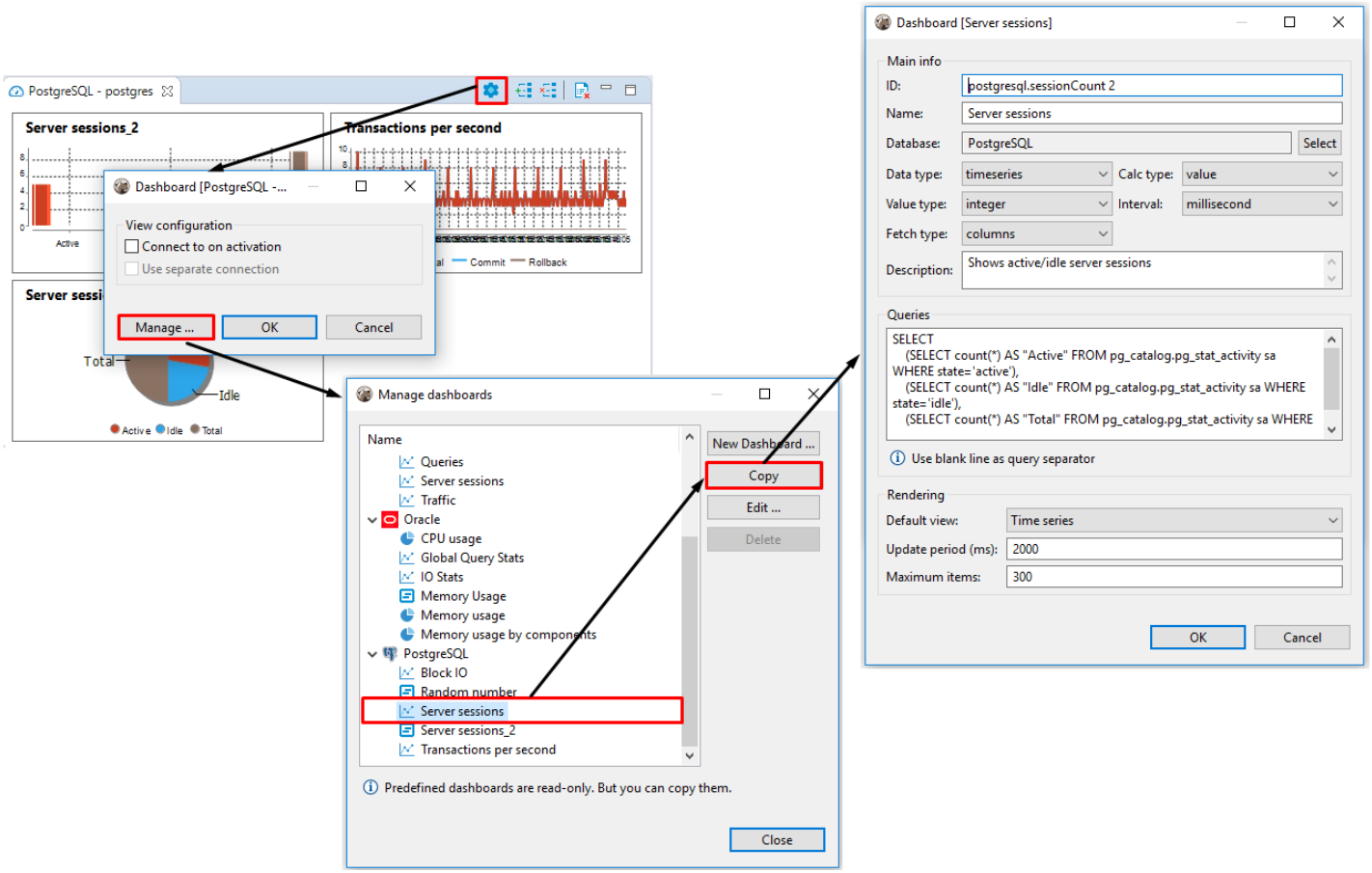
To create a dashboard from scratch:

1. Press the **Settings** button  on the dashboards panel toolbar.
2. In the opened dialog box click the **Manage...** button.
3. In the **Manage dashboards** window click the **New dashboard...** button.
4. Set up all configurational parameters as required and press **OK**. To learn more about the dashboard's configuration parameters, see Adjusting Dashboard Configuration.



To create a dashboard from a template:

1. Press the **Settings** button  on the dashboards panel toolbar.
2. In the opened dialog box click the **Manage...** button.
3. In the **Manage dashboards** window select any of the existing dashboards from the list and click **Copy**.
4. Adjust all configurational parameters as required and press **OK**. To learn more about the dashboard's configuration parameters, see Adjusting Dashboard Configuration.



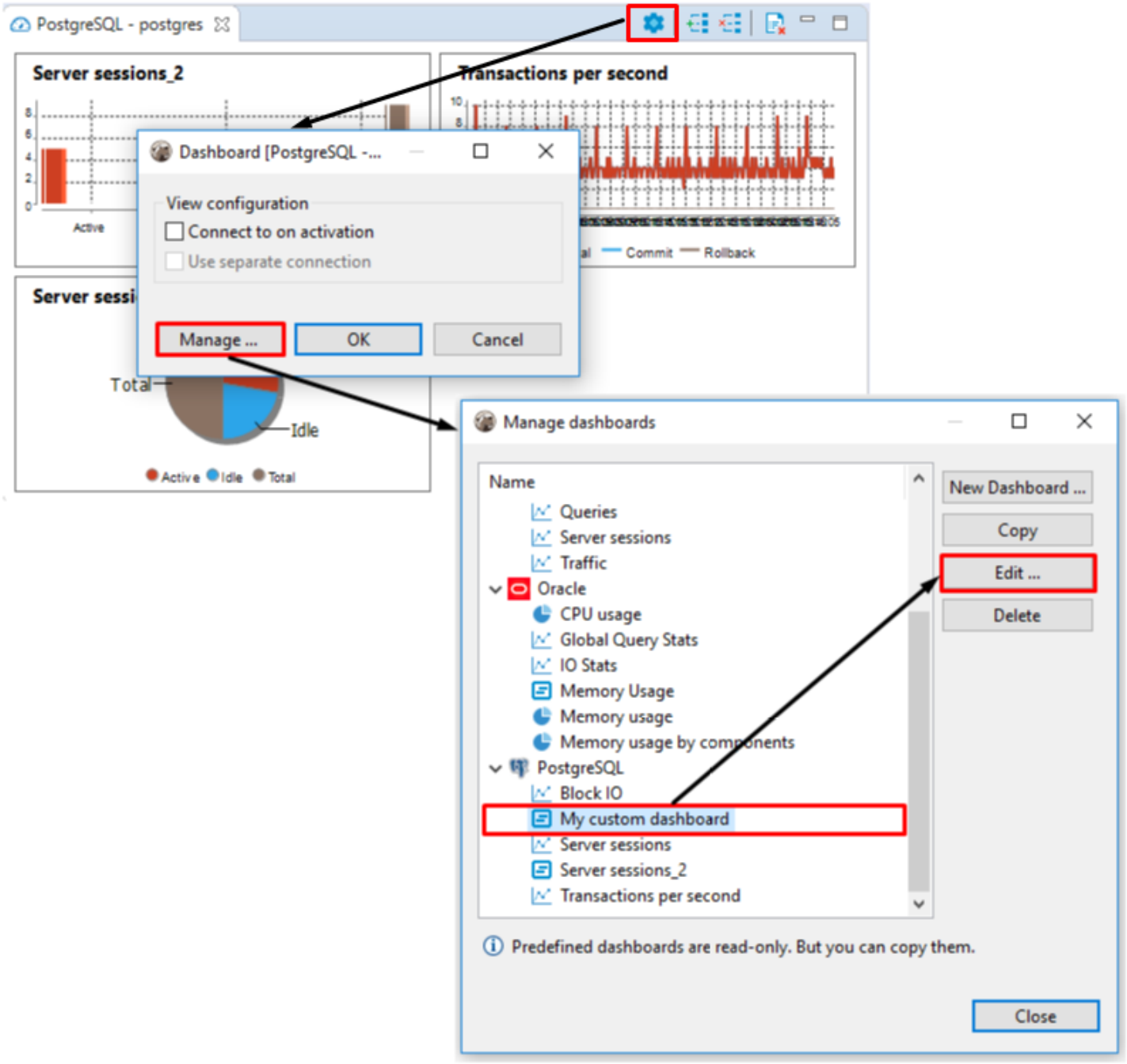
Editing Dashboards

If you need to change the dashboard's name , ID or any other configurational setting, you can edit a dashboard.

Note: Only custom dashboards can be edited, predefined dashboards are read-only, but you can use them as templates and create a custom dashboard whose parameters will be editable. To learn how to create dashboards from templates, see Creating Dashboards.

To edit dashboard's configuration:

1. Press the **Settings** button on the dashboards panel toolbar.
2. In the opened dialog box click the **Manage...** button.
3. In the **Manage dashboards** window select any of the custom dashboards from the list and click **Edit...**.
4. Adjust all configurational parameters as required and press **OK**. To learn more about the dashboard's configuration parameters, see Adjusting Dashboard Configuration.



Deleting Dashboards

Note: Predefined dashboards cannot be deleted, but any custom dashboards can be deleted.

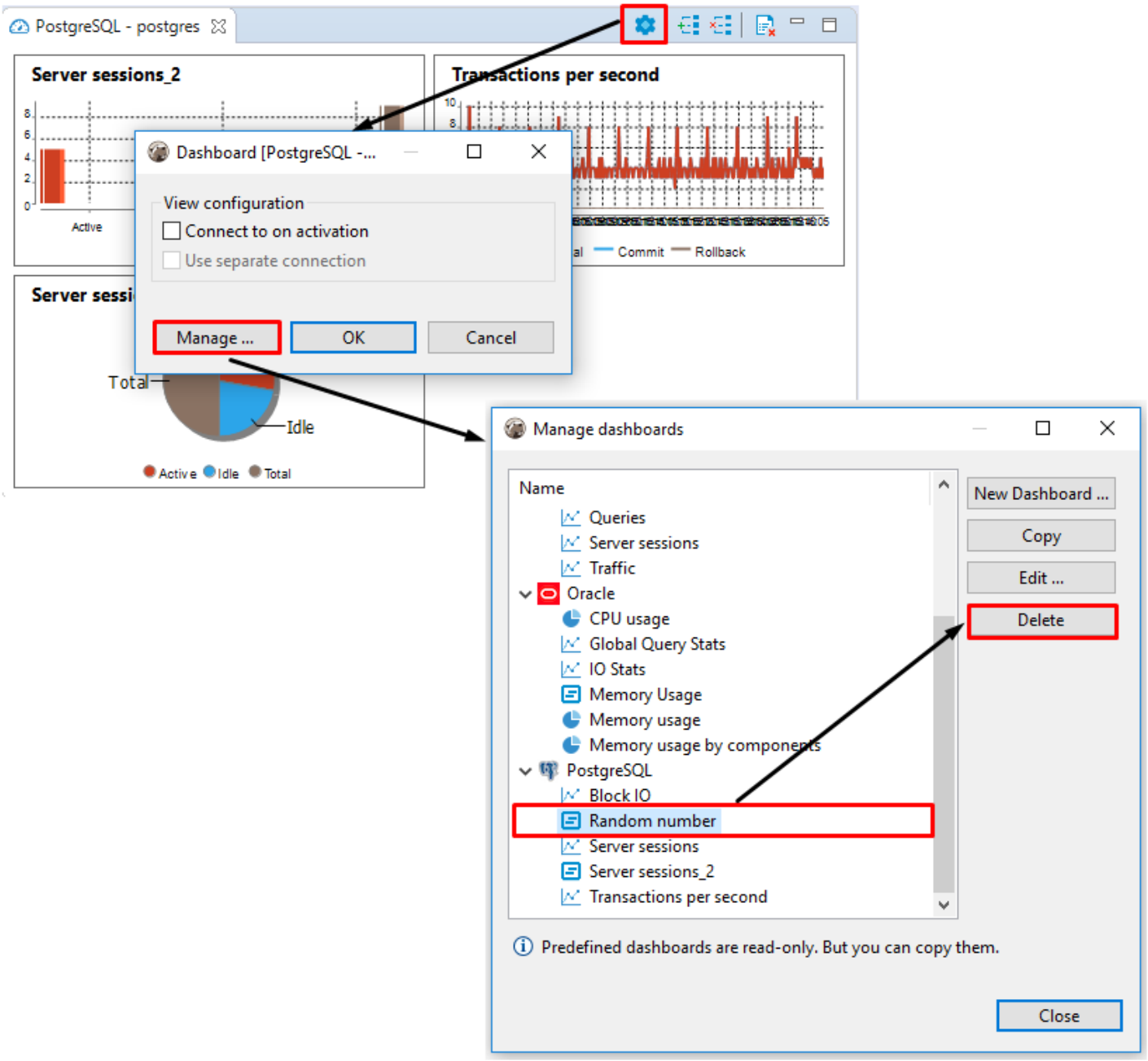
If you want to delete a dashboard, follow the steps described below.

To delete a dashboard:

1. Press the **Settings** button on the dashboards panel toolbar.

2. In the opened dialog box click **Manage...** dashboards.

3. In the **Manage dashboards** window select any of the custom dashboards from the list and click **Delete**.



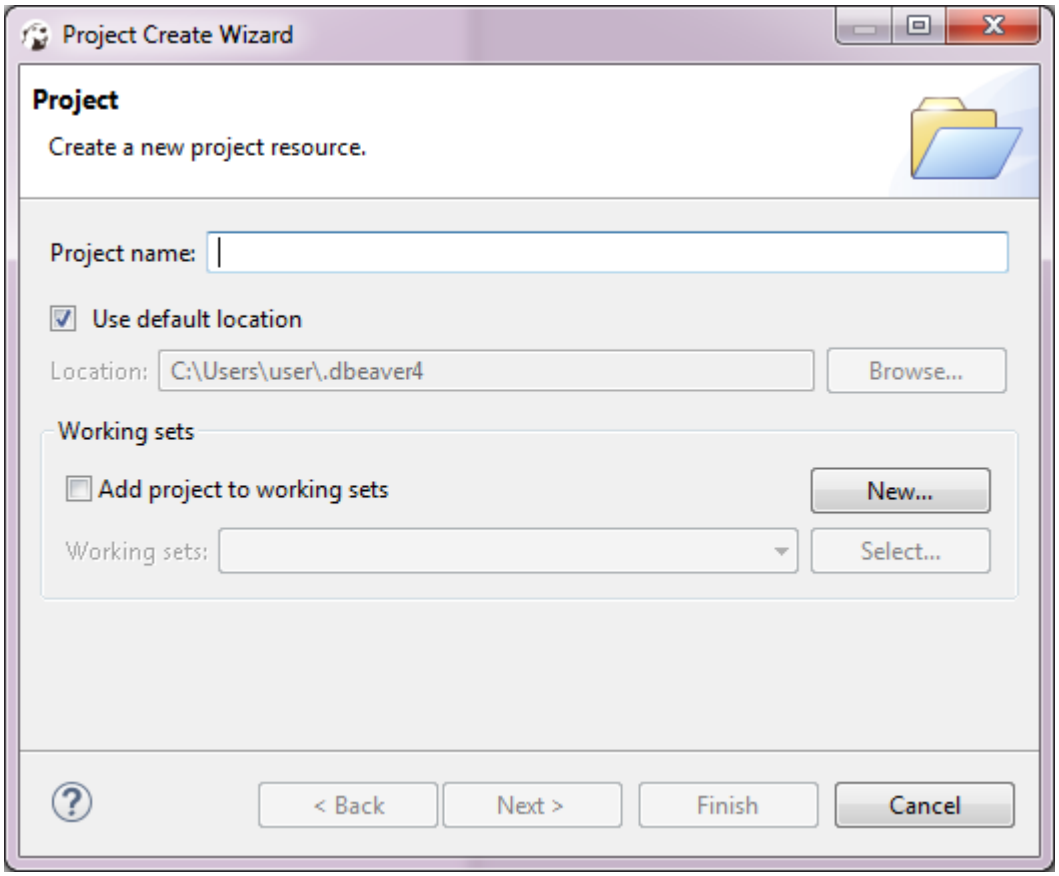
Projects

The Projects view allows the creation of new projects as well as renaming and deleting projects that are not active.

NOTE: You cannot rename or delete a project that is set as active.

Creating Project

To create a project, in the Projects view, in the toolbar, click **Create Project** ( ). The Project Create Wizard opens.

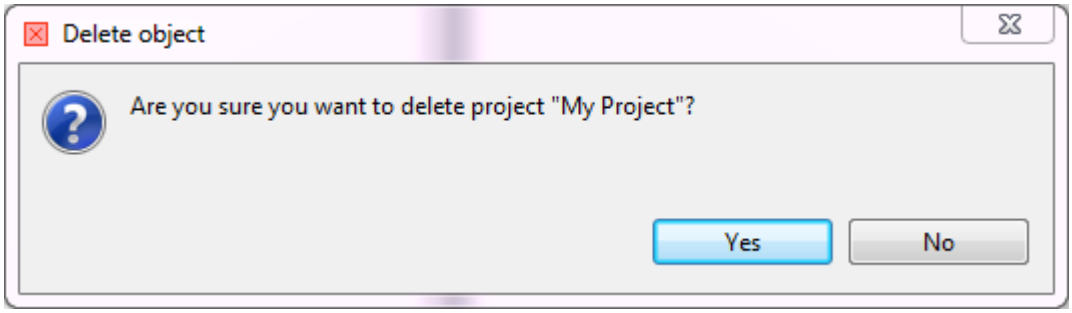


1. In the Project screen, in the **Project name** field, specify the name of the project.
2. To keep the default location to store the project, leave the **Use default location** checkbox selected. If you want to change the location, clear the checkbox and enter the name of the new directory into the **Location** field or click **Browse** and select the directory in the folder tree.
3. Click **Finish**. The new project appears in the projects tree.

Deleting Project

To delete a project, in the Projects view, right-click its name in the tree and click **Delete** on the context menu. Two confirmation dialog boxes will appear one after another:

1. **Delete object** dialog box is to confirm the deletion of the project itself. Click **Yes** if you are sure you want to delete it. Otherwise, click **No**.



2. **Delete project** dialog box is to confirm the deletion of the project's contents. These are the project configuration files and scripts stored in the file system. Click **Yes** if you want the contents to be deleted as well. To keep the contents, click **No**.

NOTE: If you have deleted a project and then re-create it with the same name, the new project picks up all the database connections of the deleted project.

Project security

Note: This feature is available in Lite, Enterprise, Ultimate and Team editions only.

DBeaver supports local storage for connection secure data. It includes:

- Database server user credentials
- SSH tunnel user credentials
- Proxy user credentials

By default, user names and passwords are stored in file `credentials-config.json`. This file is encrypted using the AES key. However the key is insecure because it is found in the DB open-sources and thus this file can be un-encrypted by 3rd party people using some 3rd party software.

In the DBeaver Enterprise, the security support is much safer because of its strong encryption.

Master password for local configuration

It is possible to set a master password for all projects in a local workspace. Go to Preferences->Database->Security and enable the option `Use secure passwords storage`. There are several password storage providers (you can see them on page General->Security->Secure Storage), `DBeaver Enterprise Password Provider` is the default one (in standalone DBeaver). It will ask you to specify master password. DBeaver doesn't store this password anywhere, it only encrypts user credentials in a special local storage. It is not possible to decrypt this password without a password (at least easily).

The side effect of this configuration is that you cannot share your connections (with password) between different users because user credentials are stored in a completely separate location and they are protected by a local user password.



Use Windows Integration password provider

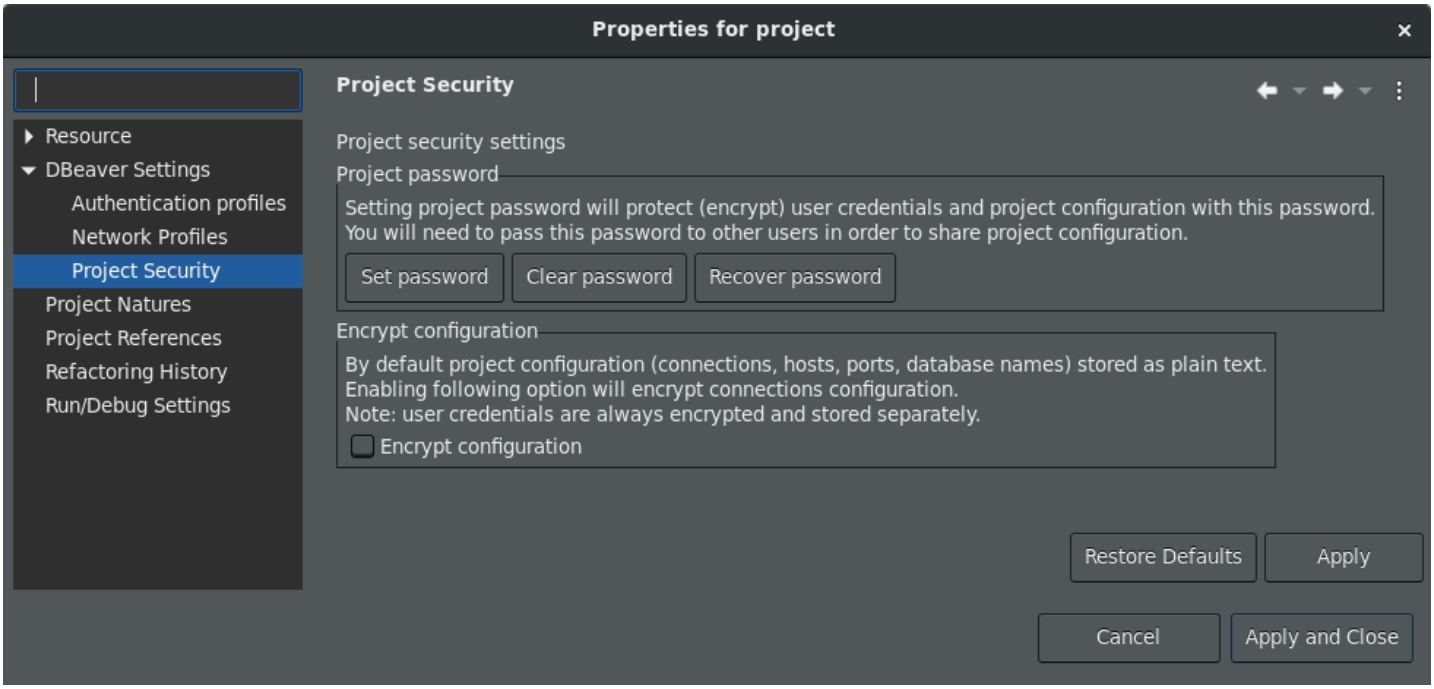
You can disable the default password provider and enable a "Windows Integration" provider. This provider does not need a master password but it uses a randomly generated password stored in a local user secure storage (in Windows). This is easier (as you don't need to remember the master password) but less secure (anybody who has access to your Windows user account will have access to DBeaver's stored credentials).

Project password

You may specify a password for a project. It will encrypt all the project's configurations with this password. Also, you will be able to share your project settings with other users (you will need to pass the project password as well).

In order to enable a project password open the project properties. You can do this by:

- Clicking on main menu File->Project security
- Clicking on "Configure" icon in the project explorer view toolbar and switching to the Project Security tab
- Pressing `ALT+Enter` on a project element in `Projects` view and switch to Project Security tab



On the project security page click on the "Set Password" button to enable the project password. Click on Clear to disable it (you will need to enter a current project password to clear it).

Team work (Git)

Note: This feature is available in Lite,Enterprise and Ultimate editions only.

DBeaver is fully integrated with the Git team work/version control system. You can keep your project configuration, scripts, diagrams, bookmarks and other artifacts in a Git repository.

This extension is included in the EE version. It can also be installed manually to the CE version.

Marketplace URL:https://marketplace.eclipse.org/content/dbeaver-git-support

P2 repository URL:https://dbeaver.io/update/git/latest/

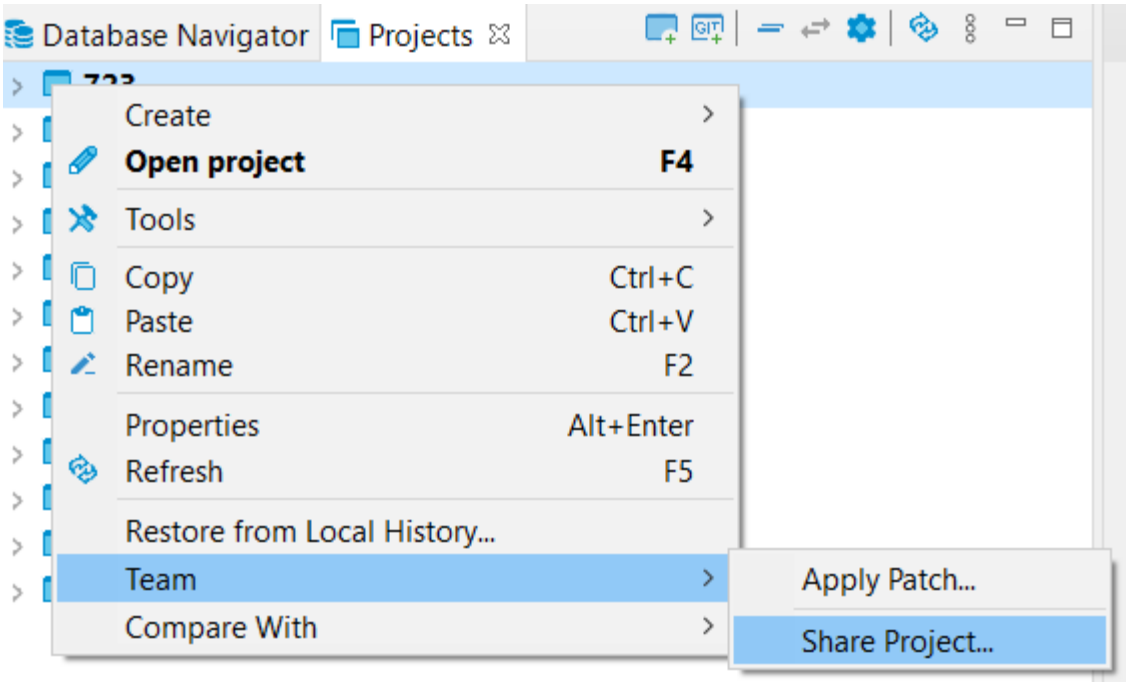
Create Repository

Share Existing Project

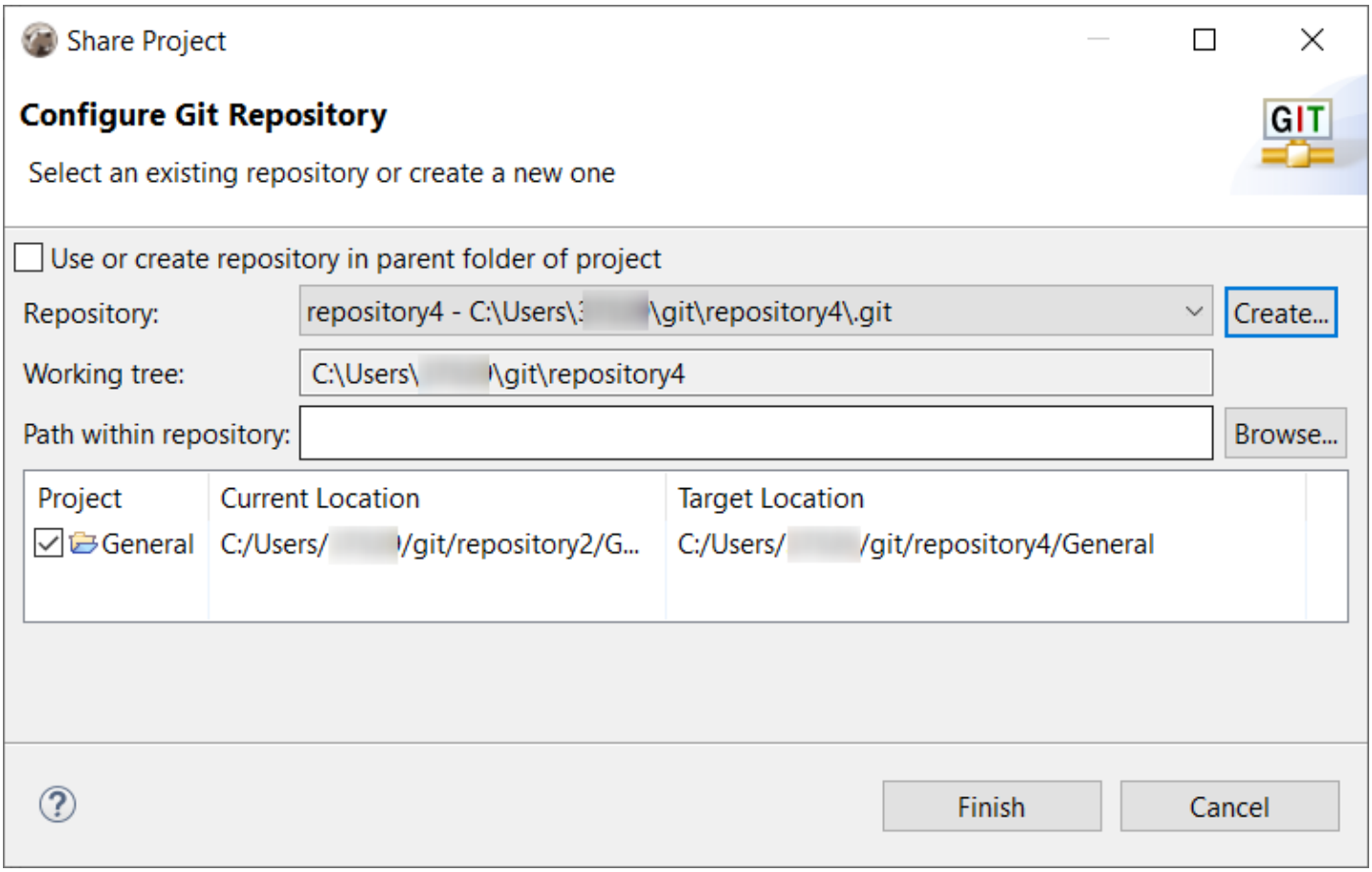
This includes three main steps: 1) creating a local git repository 2) creating a remote git repository 3) sharing your project

Here is how to do these steps in detail:

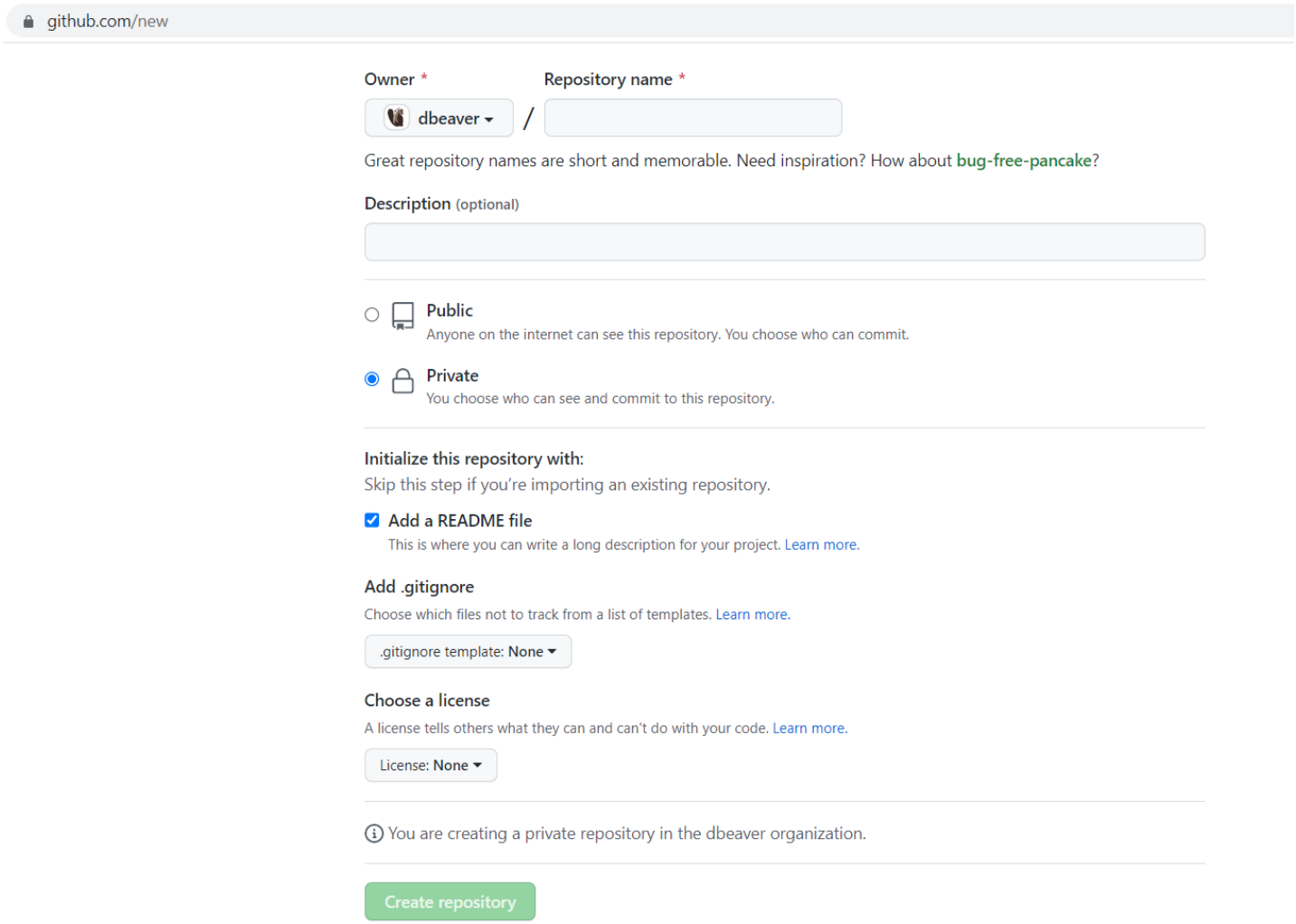
1. Open **Projects tab**, then choose the Project that you need to share and choose `Team - Share project` from the context menu:



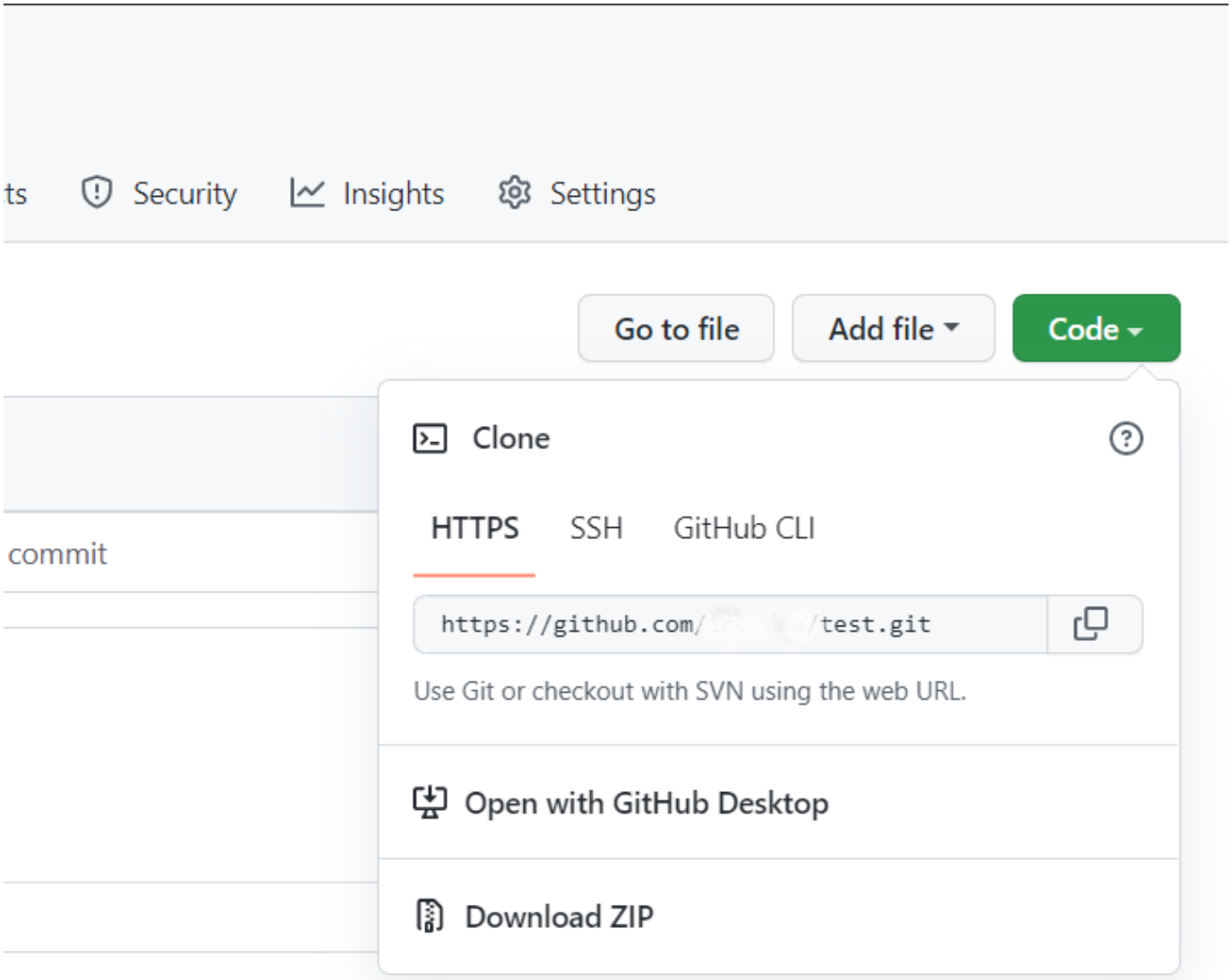
2. Create a new local repository or choose an existing one:



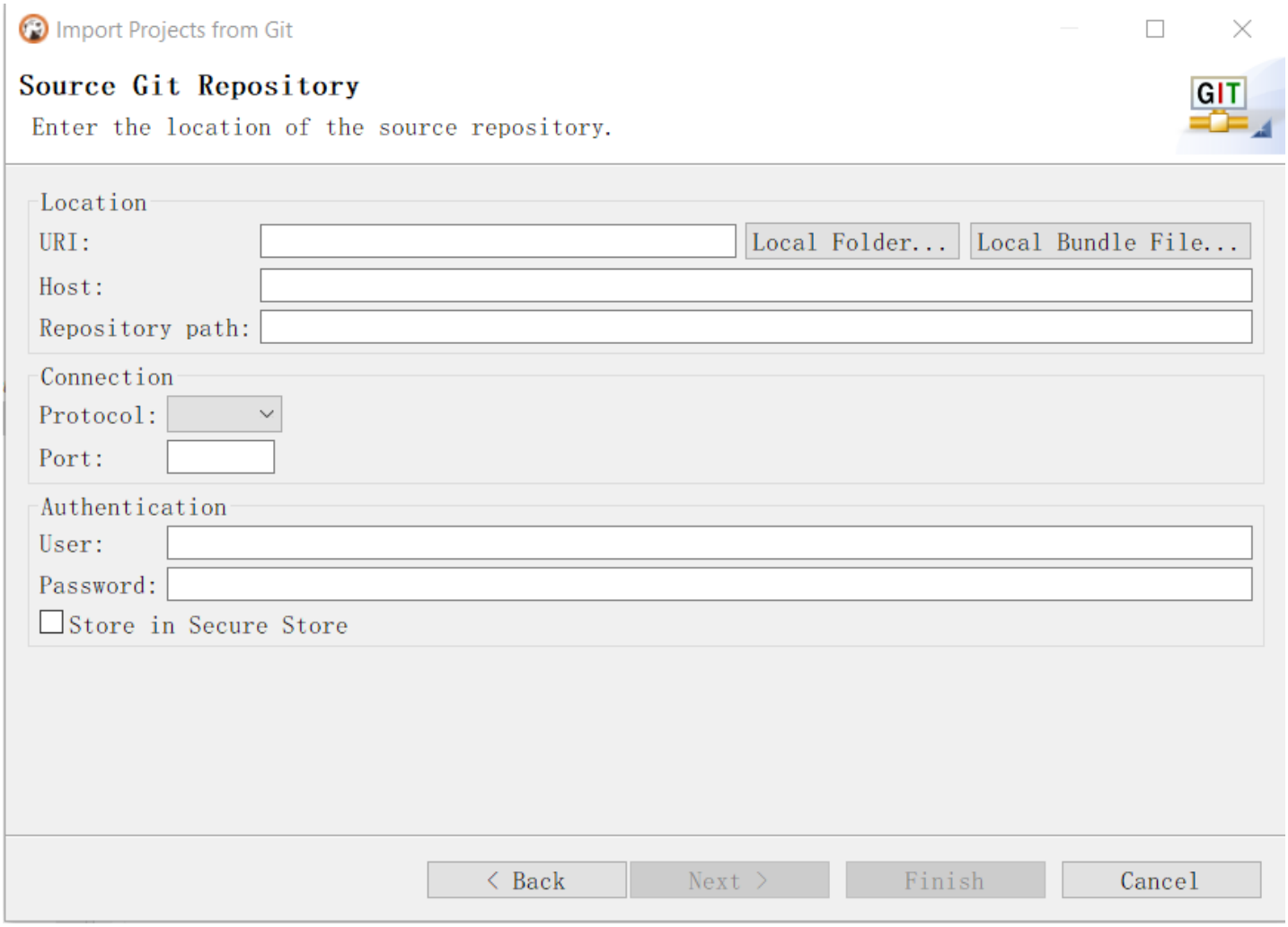
3. Create a new GitHub repository. Make sure to pay attention to the security. The repository should not be **public** as there are credentials to the databases you are working with. Here you fill in the `Repository name`, `Description`, `Accessibility` and click **Create repository**:


github.com/new
Owner *
Repository name *
dbeaver
/
Great repository names are short and memorable. Need inspiration? How about bug-free-pancake?
Description (optional)
Public
Anyone on the internet can see this repository. You choose who can commit.
Private
You choose who can see and commit to this repository.
Initialize this repository with:
Skip this step if you're importing an existing repository.
Add a README file
This is where you can write a long description for your project. Learn more.
Add .gitignore
Choose which files not to track from a list of templates. Learn more.
.gitignore template: None
Choose a license
A license tells others what they can and can't do with your code. Learn more.
License: None
You are creating a private repository in the dbeaver organization.
Create repository

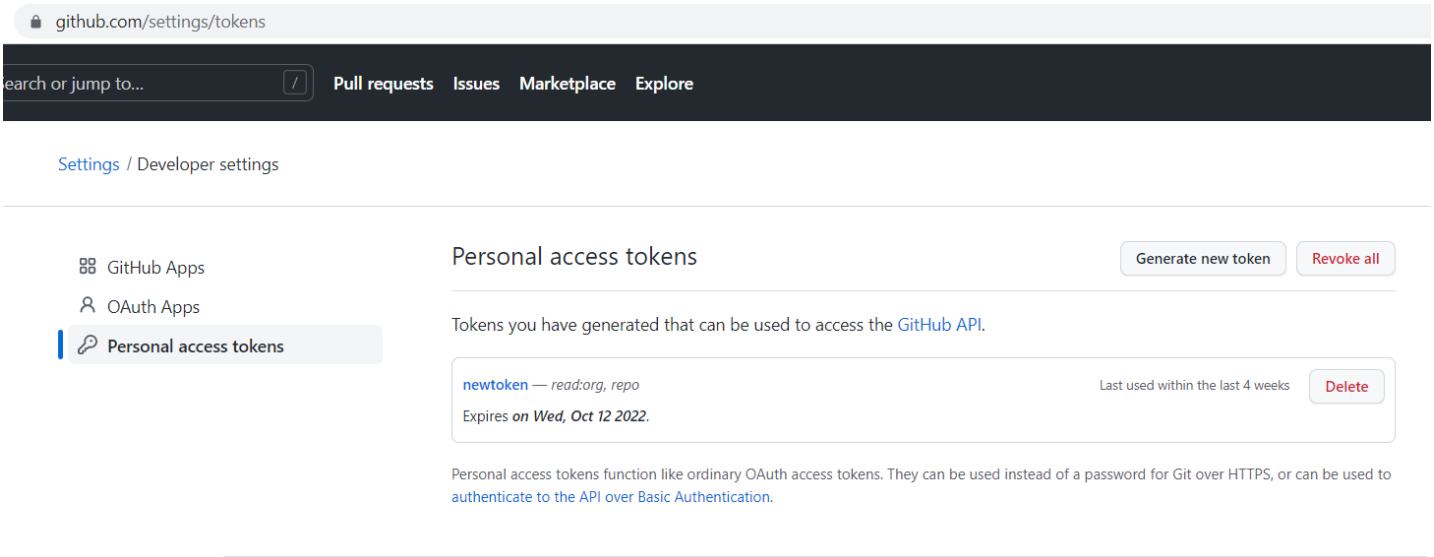
4. Then, on the page of the repository at GitHub, click on the **Code** button and copy the URL:



5. Paste the URL of the GitHub repository on DBeaver's side. You have to specify the GitHub `User` and `Password`. You have to use your GitHub username. However, instead of a password you have to specify the GitHub **access token**. We also recommend ticking `Store in Secure Store` in DBeaver below:



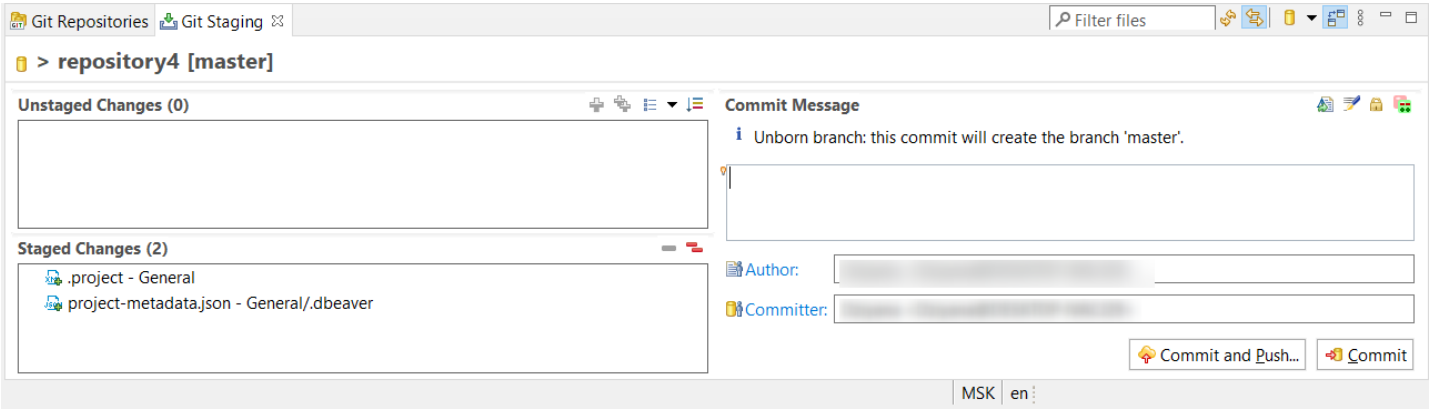
6. The GitHub access token is located at `Settings/Developer Settings/Personal access tokens`. You need to generate the new one and tick some options: all from the `'repo'` section (`repo:status`, `repo_deployment`, etc.). You should also tick `'read:org'` in section `admin:org`. You will see the **only once** token:



7. When you click **Finish** on the Source Git Repository form, you will have to generate a password for all the secure data in DBeaver. You must remember this password.

8

8. Commit the project by right-clicking the project node and selecting `Team - Commit...` or `Ctrl+#` from the context menu. Enter a commit message (the first line should be headline-like, as it will appear in the history view) and hit the **Commit** button. If the commit was successful, the question icons will have turned into repository icons:



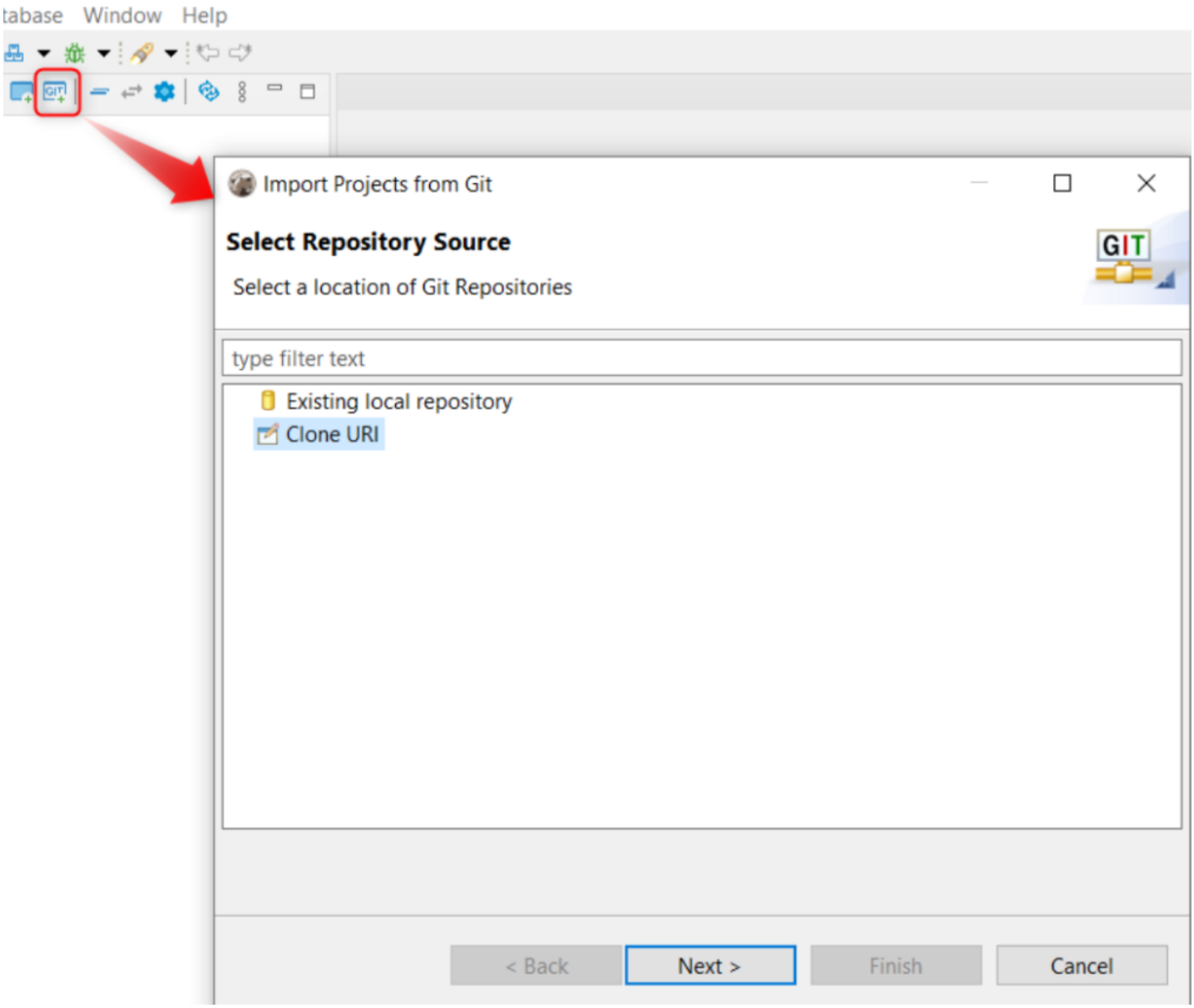
9. Then push changes to a remote repository using **Push HEAD** button:

You can learn more here

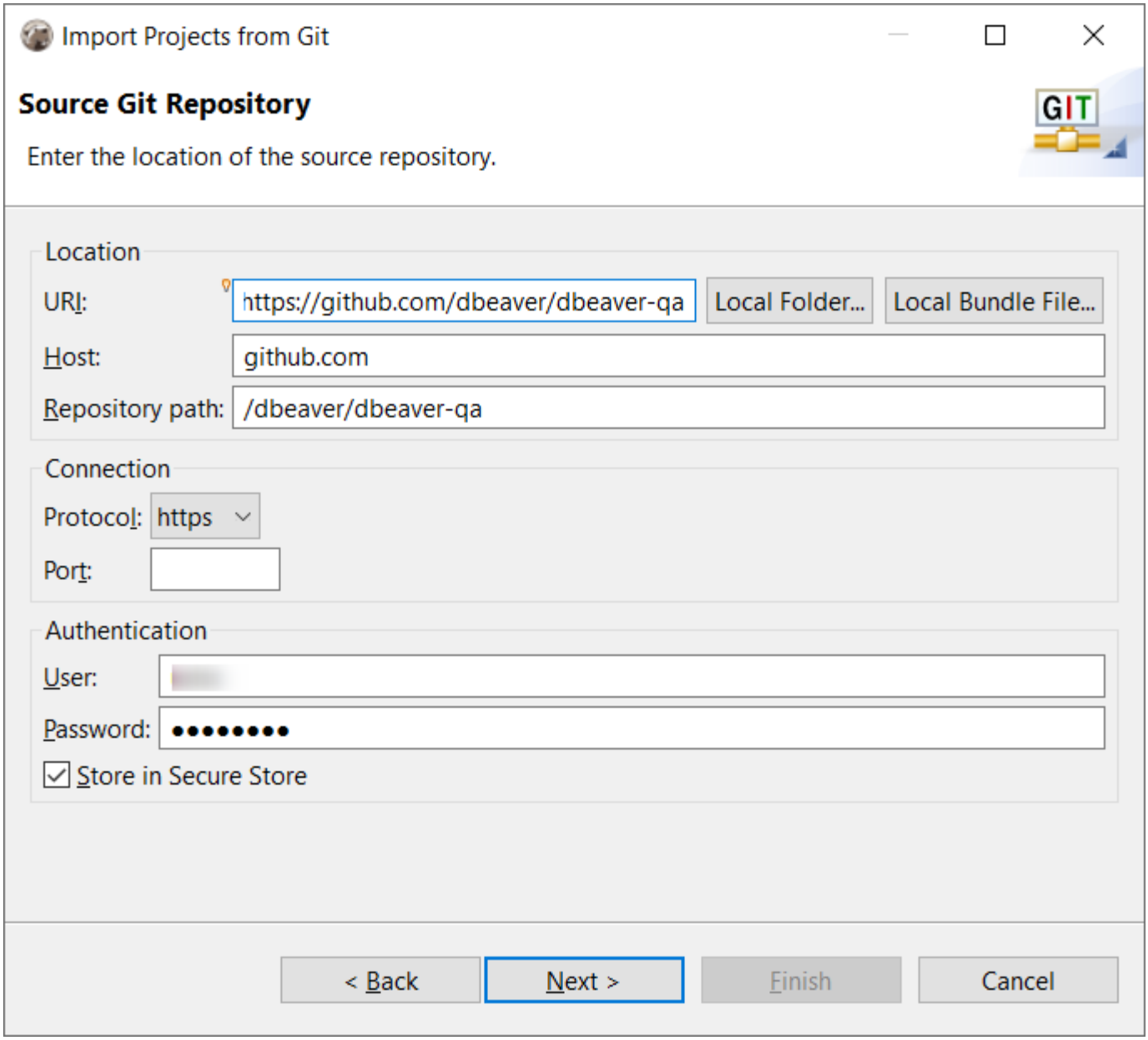
Cloning existing repositories

You need to have a GitHub account.

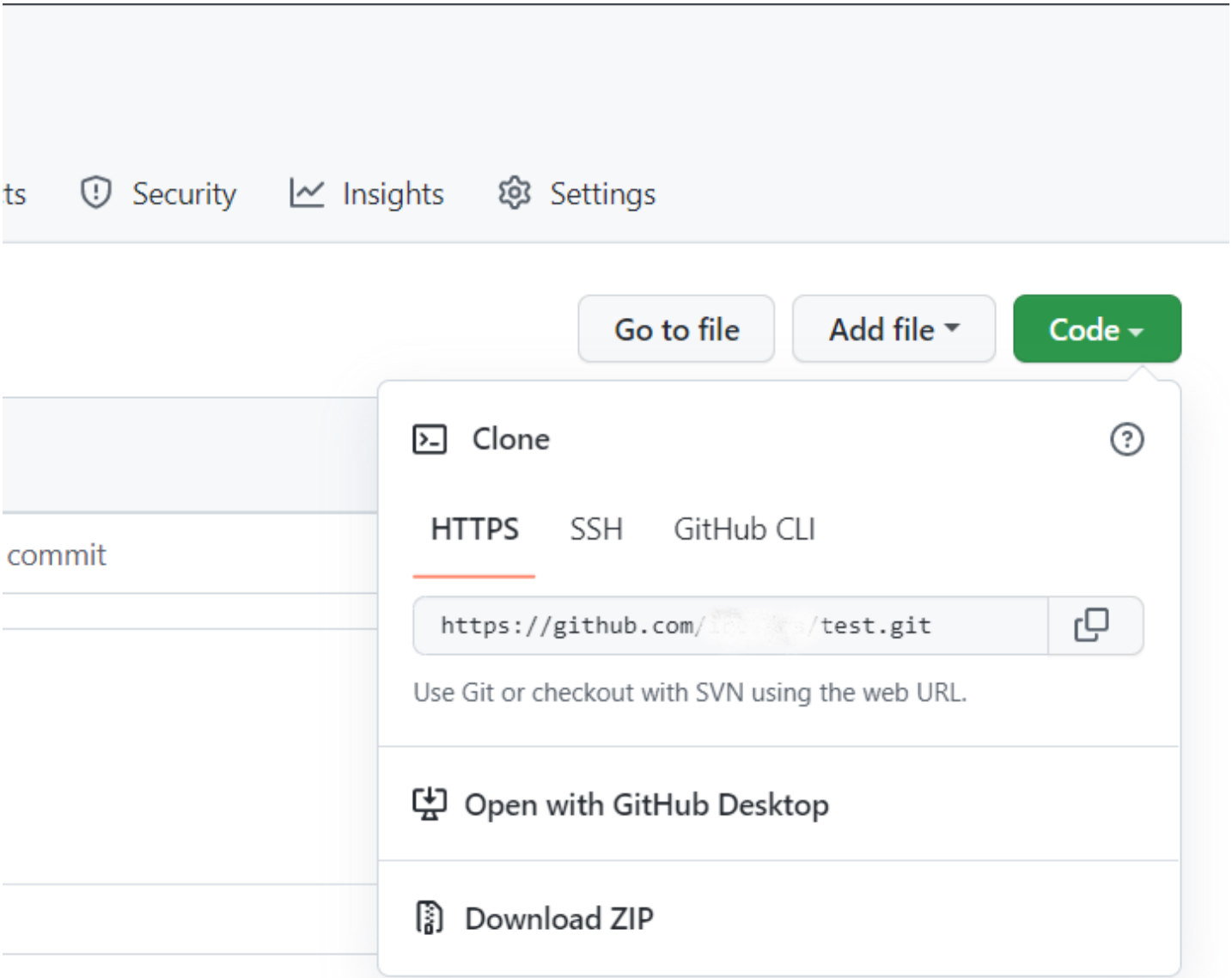
1. Choose **Create Project from Git** button at **Projects** Tab or `File - Git - Create Project` in the Main menu:



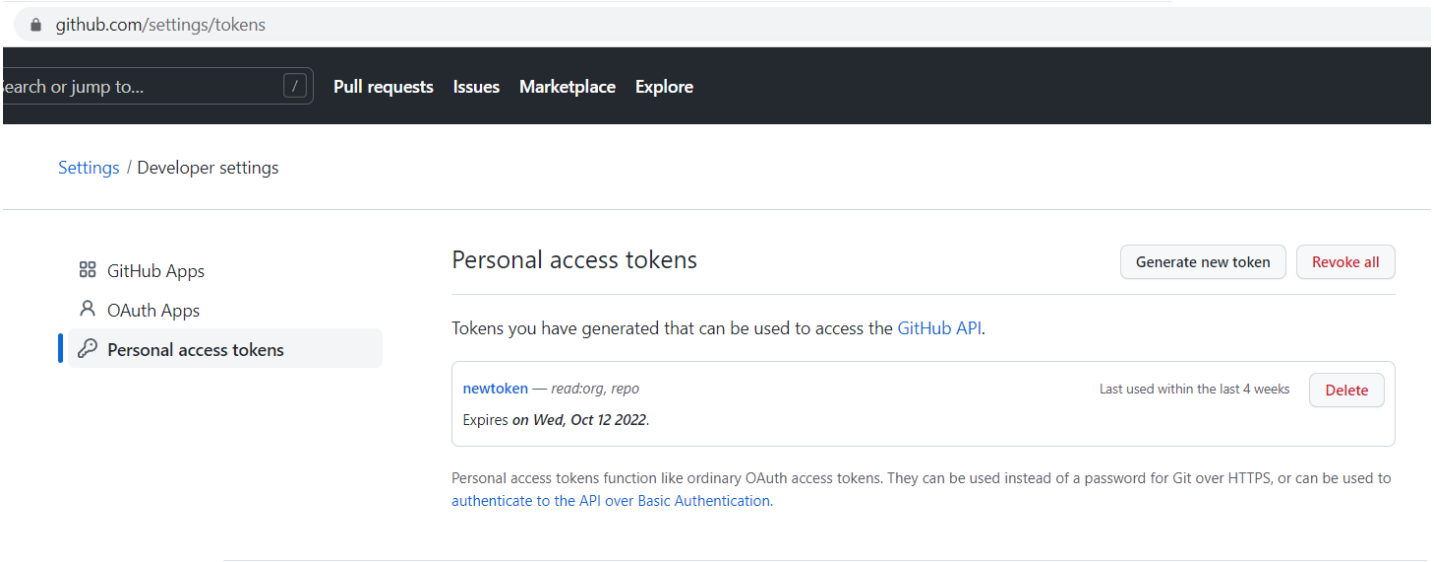
2. On the first page of the wizard, enter the location of the remote repository. Make sure the repository's owner has given you access to it. Here you have to specify the GitHub `User` and `Password`. You have to use your GitHub username, but instead of a password, you specify the GitHub **access token**. We also recommend ticking `Store in the Secure Store` in DBeaver below:



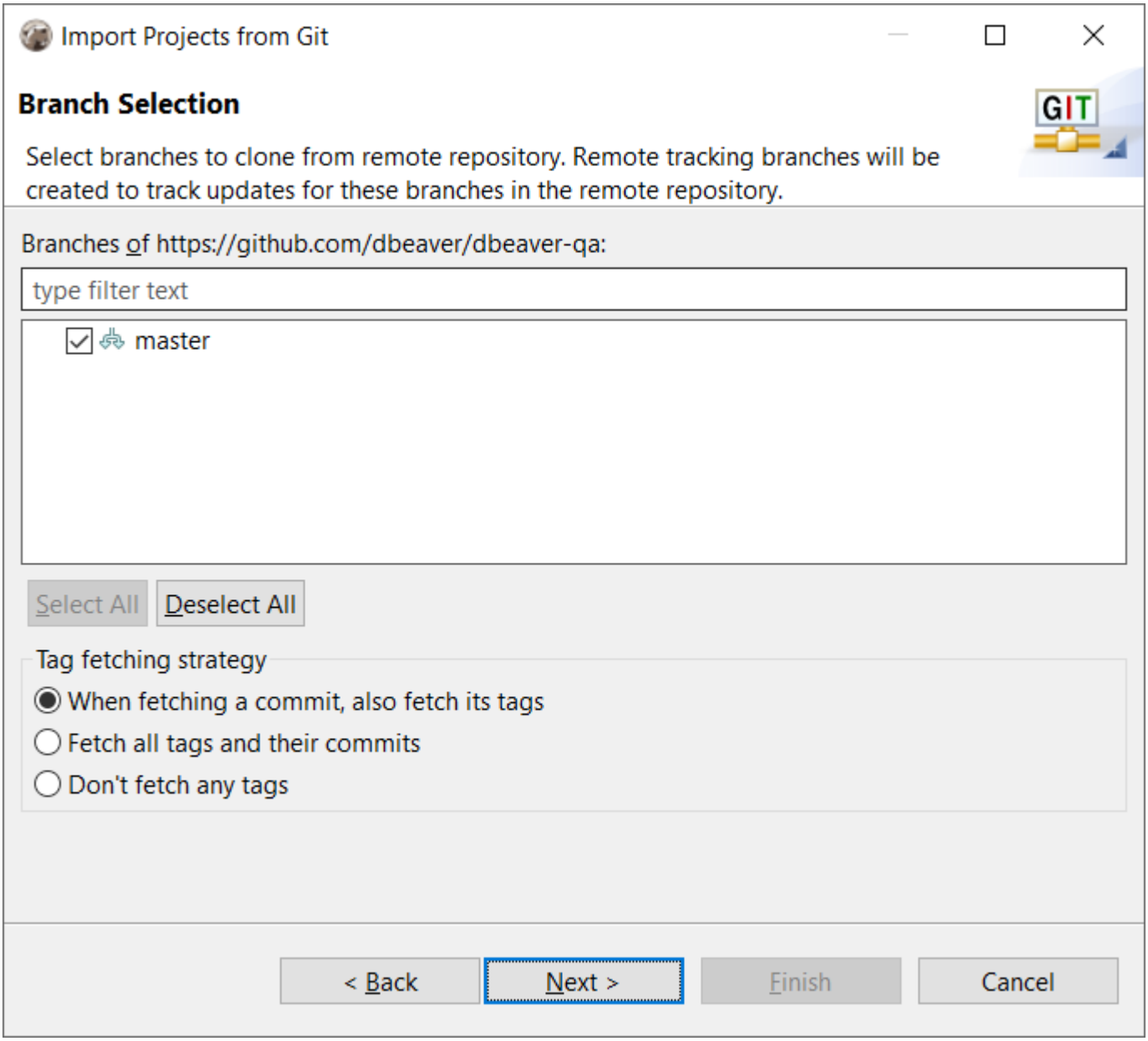
3. The remote repository's location comes from the repository's page at GitHub. Click on the **Code** button and copy the URL:



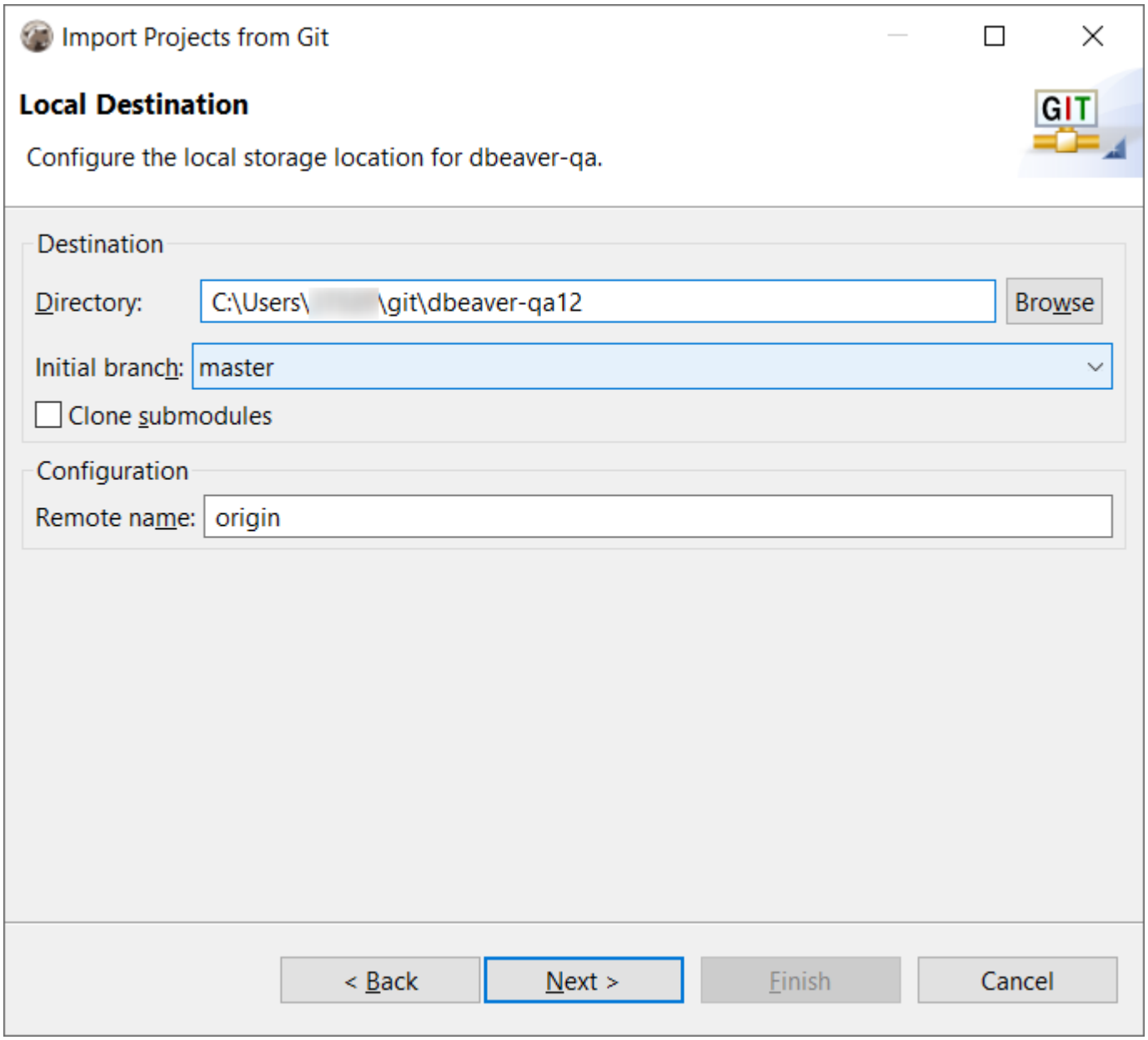
4. The GitHub access token is located at `Settings/Developer Settings/Personal access tokens`. You need to generate the new one and tick some options. Please, tick all from `'repo'` section (`repo:status`, `repo_deployment`, etc.). You should also tick `'read:org'` in the `admin:org` section. You will see the **only once** token:



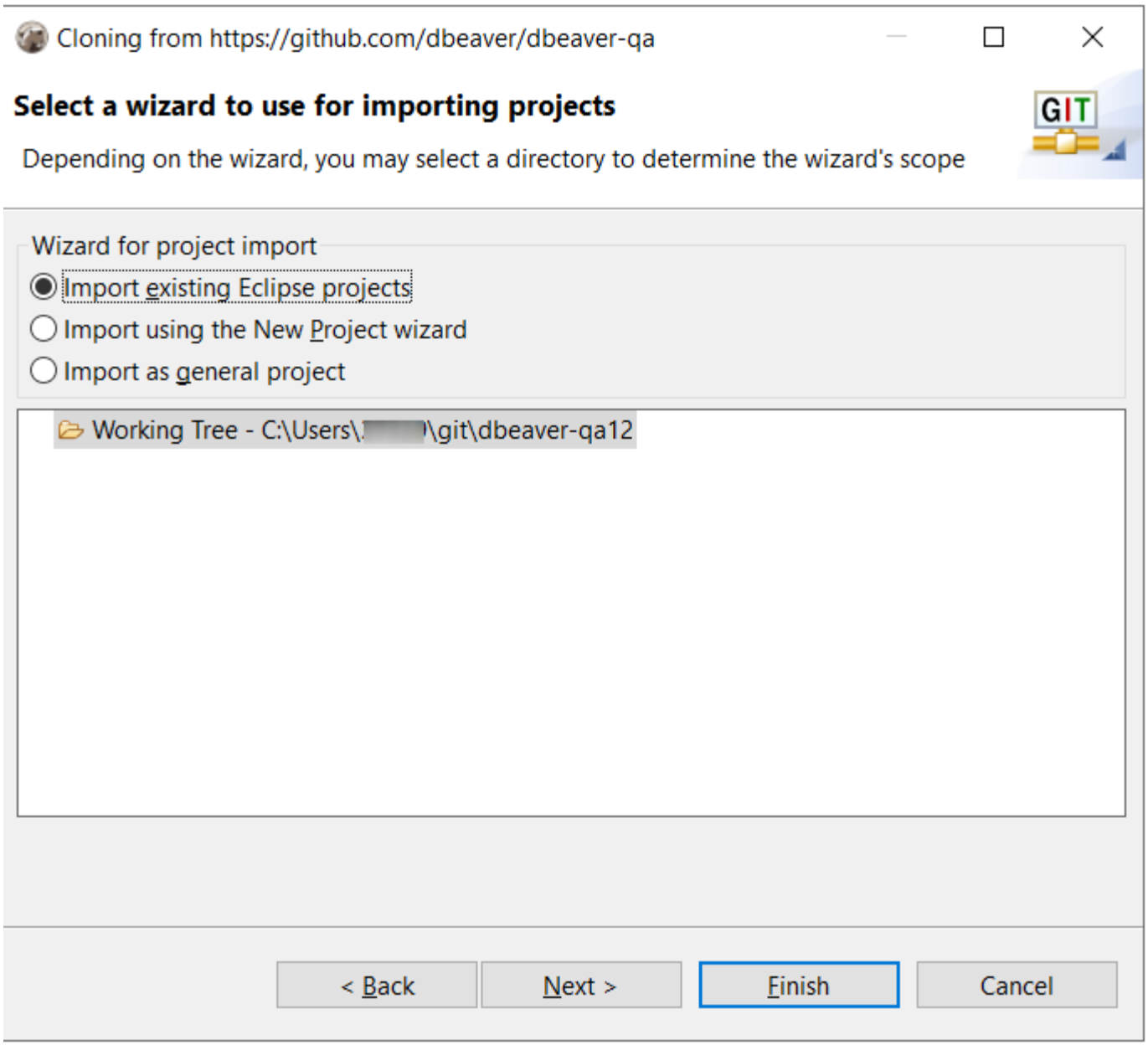
5. On the next page, choose which branches should be cloned from the remote repository:



6. On the next page in DBeaver define where you want to store the repository on the local file system and define some initial settings:



7. Choose the type of wizard you want to use and then finish the repository cloning. The project should now appear in the Navigator/Project Explorer:

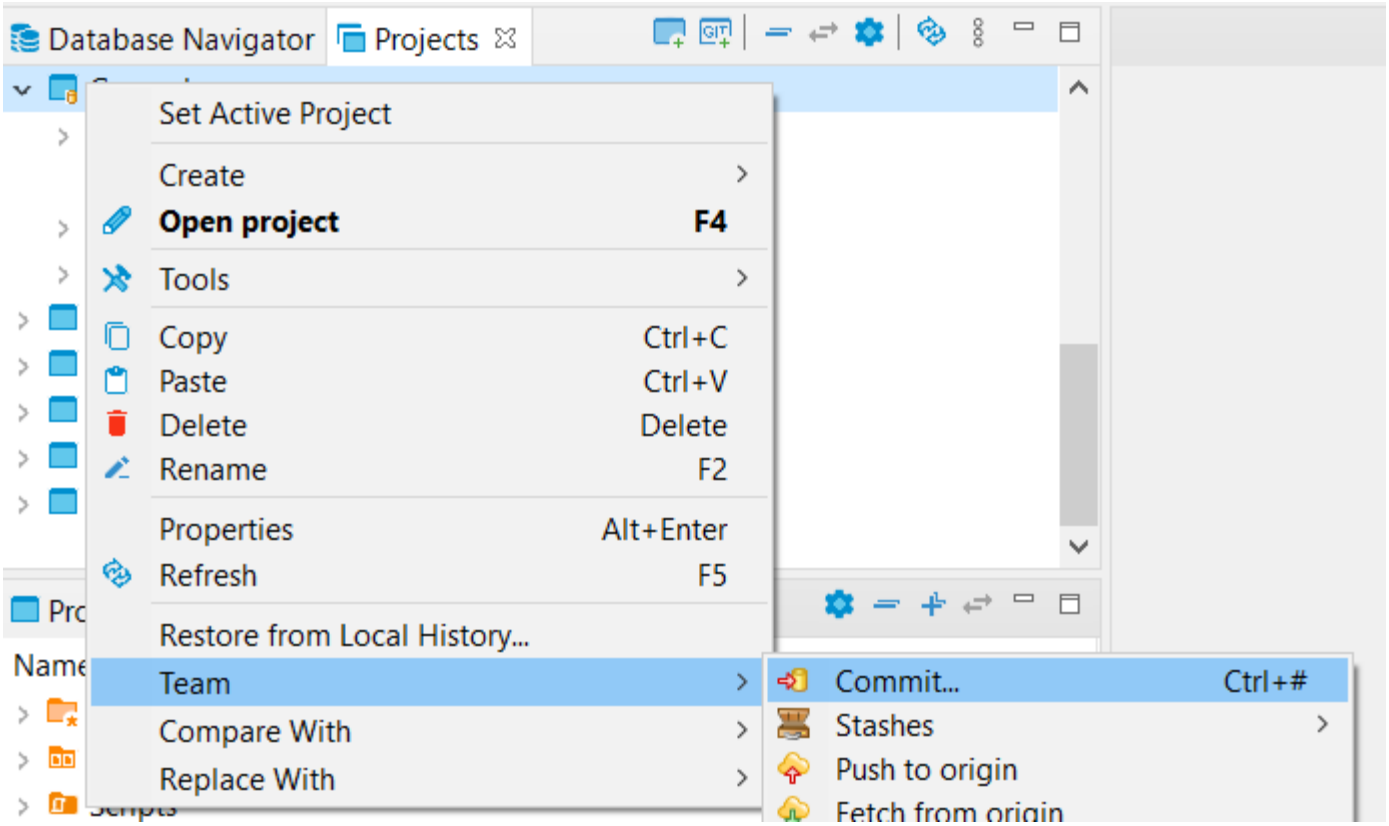


More information can be found here

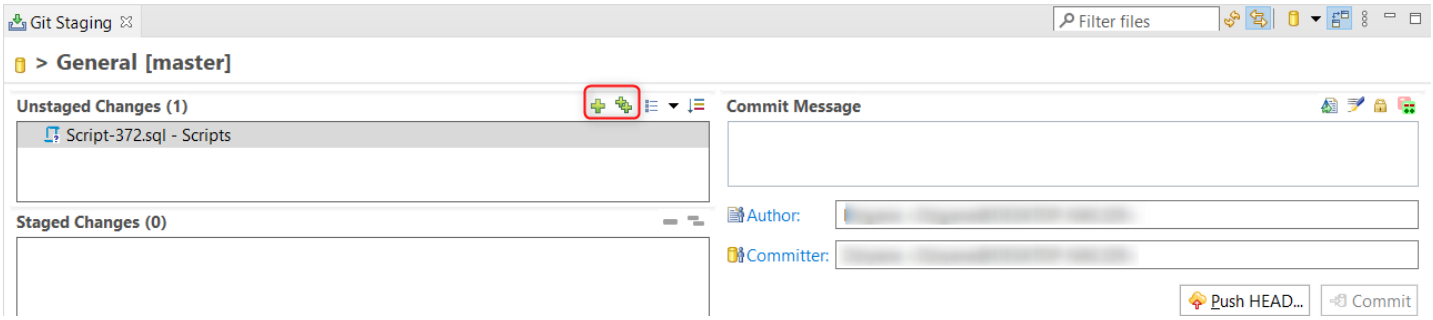
Work with Project (Connections, Scripts, etc.)

If some changes were made, e.g., Connection, Script or ERD was created, deleted or changed and you want to update your remote repository, you should:

1. Go to the Context menu of the project (**Projects** tab) and choose `Team - Commit`:



2. **Git Staging** tab is opened at the bottom of the screen. **Unstaged Changes** are shown here. Add needed changes to the index using **Add Selected files to the index** or **Add all files to the index** button (highlighted at the picture), input commit message and then Commit changes (**Commit** button or `Ctrl+Enter`). Commit appears on the History tab (Main menu: `Window - Show view - Other - Version Control(Team) - History`):



3. The `Commit and Push` button can also be used. In case you need to Push changes, it can be done from the Project context menu (`Team - Remote - Push`) or from the **History** tab if all commits do not need to be pushed (right click at needed commit - Push commit). The changes will appear in your git repository.

More information can be found here

Pulling New Changes from Upstream Branch

Right-click on a project in the **Project** Tab and select `Team - Pull` or right-click on a repository in the Git Repositories view and select **Pull** to pull new changes from the upstream branch that your local branch is tracking.

If no local change was made or you want to discard your local changes, use `Team - Reset`

Security in PRO products

The level of security is one of the key questions for enterprises, and the DBeaver team pays a lot of attention to it. One of the best reasons to use PRO versions is to take advantage of its security tools and features, such as password protection, SSO authentication, teams and roles in Team Edition.

This article briefly describes the most important security options available in DBeaver PRO.

- Changing database password
- Password protection for Projects
- Secure authentication
- Predefined connections
- User roles and permissions
- Centralized updates
- License management

Changing database password

Users can change the current database password directly in DBeaver in the following databases: Cockroach, Exasol, Greenplum, Netezza, Oracle, PostgreSQL, Redshift, Snowflake, SQL Server, and Vertica.

Oracle, PostgreSQL, and Netezza allow changing an expired password in DBeaver as well.

- How to change the user password

Password protection for Projects

Master password for all your Projects

You can protect all Projects in your local workspace with a master password. This feature is only available in DBeaver PRO versions.

You can set this password and store it in DBeaver password provider or use a generated password from your local password provider (for instance, OS X Keystore Integration or Windows integration provider).

Learn more about master pasword

Password for one project

You can specify a password for any project to protect all the project's configurations. This feature is only available in DBeaver PRO versions.

Learn more about project pasword

Secure authentication

In DBeaver PRO versions, you can connect to databases using secure authentication via Kerberos or GCP, AWS, and Azure cloud services.

Kerberos support

Kerberos is an authentication protocol, the default authentication technology used in Microsoft Windows.

You can connect via Kerberos using keytab, kinit, or a password. Open the connection settings, choose one of the supported databases and select Kerberos as the authentication method.

Learn more about authentication via Kerberos

SSO authentication

Users can connect to all company services using only one login and password. This is possible if you use SSO - Single Sign-On authentication service.

You do not need to manage, store, and transfer user credentials. When a user connects to the database, DBeaver opens a web browser with SSO authentication.

DBeaver supports the following SSO authentication services:

- AWS SSO
- GCP SSO
- Azure

Predefined connections

Connections import

You can describe all available database connections in configuration files (in JSON format) or import from CSV or XML DBeaver.

Read-only connections

If you want to restrict users from editing connection parameters, you can protect them with passwords.

Users roles and permissions

Configuring preferences

You can customize users preferences before they run DBeaver. For example, you can set the default simple mode for all connections (to show only schemas and tables and hide all system and service objects).

How to manage preferences

Roles in Team Edition

The best way to manage user access, restrictions, and permissions is to use Team Edition.

Team Edition allows you to create users and assign them appropriate roles with predefined capabilities.

You can add Viewers and Editors to work with prepared data, Managers to prepare data for them, Developers to work with scripts and connections, and administrators to manage everything.

Learn more about Team Edition

Centralized automatic updates

If your team works on Microsoft Windows, you can organize DBevaer mass updates in silent mode, without user input, using the Windows Installer command line options.

Learn more about silent install

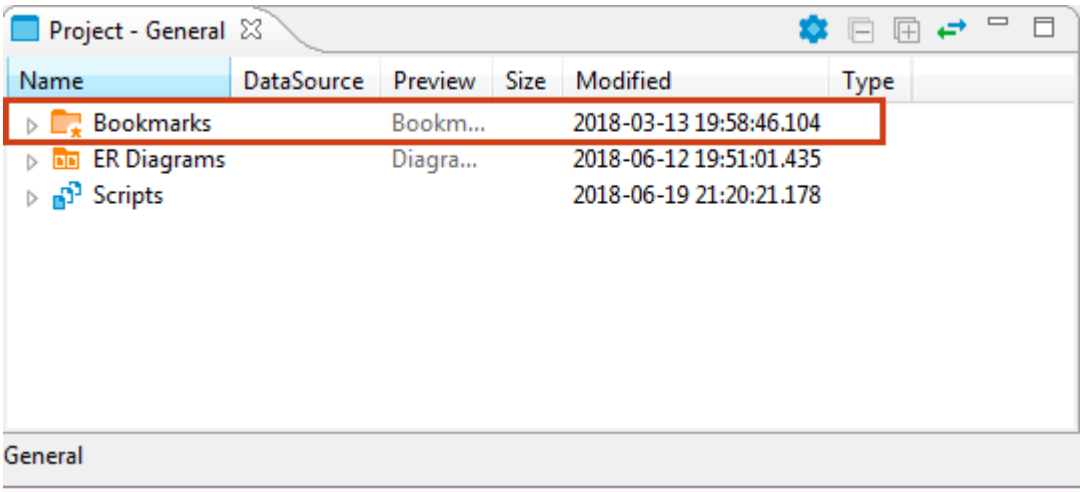
License management

You can place the license file in the user's workspace or store it elsewhere, and specify the license path on the command line or in the DBeaver configuration file.

- Learn more about lisense management automation
- How to specify license path in config

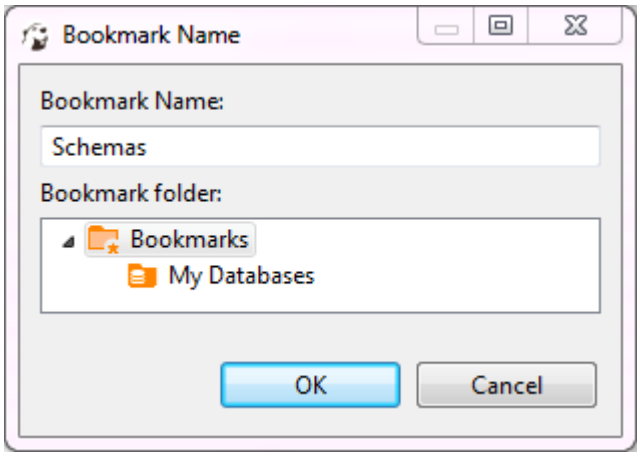
Bookmarks

Bookmarks are quick access links to objects of a database. They appear in the project tree inside the Projects or Project Explorer views.



To create a bookmark:

1. In the Database Navigator or under **Connections** node of the Projects view, click the database object of interest to focus on it.
2. Press `CTRL+ALT+SHIFT+D` . The **Bookmark Name** dialog box appears.
3. In the **Bookmark Name** field, enter the bookmark name, then in the **Bookmark folder** field, click the folder, and then click **OK**:



The bookmark appears in the selected folder of the related project.

To open an object using its bookmark, double-click the bookmark or right-click it and click **Open Bookmark** on the context menu. You can rename and delete bookmarks using the context menu as well.

Shortcuts

Here is a complete list of default hotkeys in DBeaver UI for Windows, Linux, and macOS users. It will help you work in DBeaver faster and more efficiently. Remember that you can always change the keyboard shortcut in the DBeaver settings. Go through: *Window - Preferences - User Interface - Keys*. Select command and add a keyboard shortcut to the *Binding row.*

If you mostly use keyboard, to get more shortcuts you can switch default key scheme to *DBeaver Keyboard Only* key scheme in *Window - Preferences - User Interface - Keys* settings.

Connection

Shortcut for Windows /Linux	Shortcut for macOS	Action
F4	F4	Open object editor
Ctrl+Shift+D	^⇧D	Open database meta-object
Ctrl+Alt+Enter	^⌥↩	Open a new SQL console. No script file will be created.
Ctrl+]	^]	Create a new SQL script(***)
F3	F3	Open existing SQL script (***)
Ctrl+Enter	^↩	Open most recent SQL script(***) for an active connection
Ctrl+Alt+Shift+K	^4	Commit changes in current session
Alt+`	^V	Shows context menu with database tools
Ctrl+Alt+Shift+U	^⌥⇧U	Generates UUID/GUID and inserts into an active text editor or data editor
Ctrl+Alt+Shift+B	^⌥⇧B	Open database dashboard
Ctrl+Alt+Shift+R	^8	Rollback changes in current session
Ctrl+0	^0	Select active schema for current database
Ctrl+9	^9	Select connection for current editor

Result Set

Shortcut for Windows/Linux	Shortcut for macOS	Action
Alt+Insert	⌥Insert	Add new row
Ctrl+D	^D	Copy values from row above to current row
Ctrl+Alt+D	^⌥D	Copy values from row below to current row
Alt+Delete	⌥⌦	Delete current row
Ctrl+Alt+Insert	^⌥Insert	Duplicate current row
Enter	↩	Edit cell value with inline editor
Esc	Esc	Reset cell to original value
Ctrl+S	^S	Apply data changes
Ctrl+Alt+Shift+T	^⌥⇧T	Set focus to filter editor/data editor
Ctrl+Shift+7	^⇧7	Switch focus to results viewer/active panel
Ctrl+Alt+Shift+Enter	^⌥⇧↩	Apply cell changes
Shift+Enter	⇧↩	Edit cell value in separate dialog/editor
Ctrl+Shift+=	^⇧=	Fetch all rows
Ctrl+Alt+;	^⌥;	Fetch next page of results
Ctrl+F11	^F11	Filter by unique attribute values
F11	F11	Filter context menu
Ctrl+Alt+Shift+Left	^⌥⇧←	Move to first row
Ctrl+Alt+Shift+Right	^⌥⇧→	Move to last row
Alt+Space	⌥Space	Follow foreign key link
Ctrl+Alt+Right	^⌥→	Move to next row
Ctrl+Alt+Left	^⌥←	Move to previous row
Ctrl+Shift+1	^1	Foreign keys and tables referencing current table

Ctrl+R	^R	Reject data changes
Ctrl+Backspace	^⌫	Set cell to default value
Ctrl+`	^`	Switch results presentation
Tab	Tab	Toggle results Grid/Record view
F7	F7	Toggle extra result panels
Ctrl+2	^2	Toggle results sort order (ascending/descending/default)
Ctrl+Alt+C		Select column(s)
Ctrl+Alt+R		Select row(s)
Shift+F11		Show context menu for column

Result Set (Additional shortcuts for *DBeaver Keyboard Only* key scheme)

Shortcut for Windows/Linux	Shortcut for macOS	Action
Alt+Shift+Right		Move column(s) right
Alt+Shift+Left		Move column(s) left
Alt+Shift+C		Copy column names
Alt+Shift+H		Hide columns
Alt+Shift+T		Show hidden columns
Alt+Shift+F		Open filters settings
Ctrl+Alt+F6		Show/hide results panel (Calc)
Ctrl+Alt+F5		Show/hide results panel (Grouping)
Ctrl+Alt+F4		Show/hide results panel (Metadata)
Ctrl+Alt+F3		Show/hide results panel (References)
Ctrl+Alt+F2		Show/hide results panel (Value)

Data Editor

Shortcut for Windows/Linux	Shortcut for macOS	Action
Ctrl+Space	^Space	Enable autocomplete
Ctrl+F	⌘F	Find and replace text
Ctrl+Shift+Space	^⇧Space	Show Context Information in DIalogs and Windows
Ctrl+Alt+Space	^⌥Space	Show Context Information (SQL Editor Context)
Ctrl+X	⌘X	Cut the selection to the clipboard
Ctrl+C	⌘C	Copy the selection to the clipboard
Ctrl+V	⌘V	Paste from the clipboard
Delete	⌦	Delete the selection
Ctrl+O	⌘O	Export Diagram
Ctrl+K	⌘K	Find next item
Ctrl+Shift+K	⇧⌘K	Find previous item
Ctrl+J	⌘J	Incremental find
Ctrl+Shift+J	⇧⌘J	Incremental find reverse
Ctrl+Shift+Q	^⇧Q	Toggles quick diff information display on the line number ruler
Ctrl+1	⌘1	Suggest possible fixes for a problem
Ctrl+Y	⇧⌘Z	Redo the last operation
Ctrl+Z	⌘Z	Undo the last operation
Shift+Delete	⇧⌦	Remove selected Pictogram Elements
Ctrl+A	⌘A	Select all
Alt+Shift+A	⌥⌘A	Toggle block/column selection in the current text editor
Ctrl+Shift+Insert	⇧⌘Insert	Toggle insert mode
Alt+Shift+Y	⌥⌘Y	Toggle word wrap in the current text editor
Ctrl+F5	⌘F5	Update selected Pictogram Elements
Alt+/	^.	Context insensitive completion

Ctrl+Shift+Space	^⇧Space	Context insensitive completion (SQL Editor Context)

Text Editor

Shortcut for Windows/Linux	Shortcut for macOS	Action
Ctrl+Shift+Y	⇧⌘Y	Changes the selection to lowercase
Ctrl+Shift+X	⇧⌘X	Changes the selection to uppercase
Ctrl+Numpad_Subtract	⌘Numpad_Subtract	Collapses the folded region at the current selection
Ctrl+Shift+Numpad_Divide	⇧⌘Numpad_Divide	Collapses all folded regions
Ctrl+Alt+Down	⌥⌘↓	Duplicates the selected lines and moves the selection to the copy
Ctrl+D	⌘D	Delete a line of text
Ctrl+Delete	⌥⌦	Delete the next word
Ctrl+Backspace	⌥⌫	Delete the previous word
Ctrl+Shift+Delete	⇧⌘⌦	Delete to the end of the line of text
Ctrl+Alt+Up	⌥⌘↑	Duplicates the selected lines and leaves the selection unchanged
Ctrl+Numpad_Add	⌘Numpad_Add	Expands the folded region at the current selection
Ctrl+Numpad_Multiply	⌘Numpad_Multiply	Expands all folded regions
Ctrl+Shift+Enter	⇧⌘↩	Adds a new line above the current line
Shift+Enter	⇧↩	Adds a new line below the current line
Ctrl+Alt+J		Join lines of text (Editing Text)
Ctrl+Shift+J	^⇧J	Join lines of text (SQL Editor Context)
End	⌘→	Go to the end of the line of text
Home	⌘←	Go to the start of the line of text
Alt+Down	^⇧↓	Moves the selected lines down (Editing Text)
Ctrl+Shift+Down	⌥↓	Moves the selected lines down (SQL Editor Context)
Alt+Up	^⇧↑	Moves the selected lines up (Editing Text)

Ctrl+Shift+Up	⌥↑	Moves the selected lines up (SQL Editor Context)
Ctrl+Right	⌥→	Go to the next word
Ctrl+Left	⌥←	Go to the previous word
Ctrl+Shift+Numpad_Multiply	⇧⌘Numpad_Multiply	Resets the folding structure
Ctrl+Down		Scroll down one line of text
Ctrl+Up		Scroll up one line of text
Shift+End	⇧⌘→	Select to the end of the line of text
Shift+Home	⇧⌘←	Select to the beginning of the line of text
Ctrl+Shift+Right	⌥⇧→	Select the next word
Ctrl+Shift+Left	⌥⇧←	Select the previous word
	⇧↘	Select to the end of the text
	⇧↖	Select to the beginning of the text
F2	F2	Displays information for the current caret location in a focused hover
Ctrl+End	↘	Go to the end of the text
Ctrl+Home	↖	Go to the beginning of the text
Ctrl+Numpad_Divide	⌘Numpad_Divide	Toggles folding in the current editor
Insert	Insert	Toggle overwrite mode
Ctrl++	⌘+	Zoom in text, increase default font size for text editors
Ctrl+=	⌘=	Zoom in text, increase default font size for text editors
Ctrl+-	⌘-	Zoom out text, decrease default font size for text editors

SQL Editor

Shortcut for Windows /Linux	Shortcut for macOS	Action
Alt+X	⌥X	Execute SQL script(**)

Alt+N	⌥N	Execute SQL script natively
Ctrl+Enter	^↩	Execute SQL statement(*)
Ctrl+	^	Execute SQL statement in a new tab
Ctrl+Alt+Shift+X	^⌥⇧X	Execute script's statements in separate results tabs
Ctrl+/	^/	Add or remove single line comment
Ctrl+Shift+/	^⇧/	Add or remove multi line comment
CTRL+Alt+6	^⌥6	Switch active SQL editor panel
Ctrl+6	^6	Show/hide results panel
Ctrl+Shift+\	^⇧\	Close results tab
Ctrl+Shift+F	^⇧F	Format text
Ctrl+Alt+'	^⌥'	Select value of the selected SQL expression
Ctrl+Shift+E	^⇧E	Explain execution plan
Ctrl+Shift+[	^⇧[	Position cursor on the matching bracket
Ctrl+Alt+Shift+O	^⌥⇧O	Load SQL script from file system
Ctrl+Shift+6	^⇧6	Maximize/normalize results panel
Alt+Down	⌥↓	Switch to the next query
Alt+Up	⌥↑	Switch to the previous query
Ctrl+F2	^F2	Rename current SQL script
F4	F4	Open editor of current (highlighted) database object
Ctrl+Alt+Shift+A	^⌥⇧A	Select and show all rows (no fetch size limit)
Ctrl+Alt+Shift+C	^⌥⇧C	Select row count for query under cursor
Ctrl+Shift+.	^⇧.	Set active connection from database navigator selection
Ctrl+Shift+O	^⇧O	Show server output console
Ctrl+Alt+Shift+W	^⌥⇧W	Toggles text editor soft word wrap

ER Diagrams

See ER Diagrams documentation for a list of shortcuts.

Visual Query Builder

Shortcut for Windows/Linux	Shortcut for macOS	Action
Ctrl+B	^B	Open Visual Query Builder
Ctrl+Shift+B	^⇧B	Show/hide generated SQL query text

Search

Shortcut for Windows/Linux	Shortcut for macOS	Action
Ctrl+H	^H	Open the Search dialog
Ctrl+Alt+G	⌥⌘G	Searches the files in the workspace for specific text
Ctrl+Alt+Shift+F	⌥⇧⌘L	Quick search in Windows

Data viewer

Shortcut for Windows/Linux	Shortcut for macOS	Action
Tab	Tab	Cycle through hand tool and previously used tool
Ctrl+=	⌘=	Zoom In
Ctrl+-	⌘-	Zoom Out
Alt+Shift+Q, Q	⌥⌘Q Q	Shows a particular view
Alt+Shift+Q, B	⌥⌘Q B	Shows a particular view (Breakpoints)
Alt+Shift+Q, C	⌥⌘Q C	Shows a particular view (Console)
Alt+Shift+Q, L	⌥⌘Q L	Shows a particular view (Error Log)

Alt+Shift+Q, Z	⌥⌘Q Z	Shows a particular view (History)
Alt+Shift+Q, O	⌥⌘Q O	Shows a particular view (Outline)
Alt+Shift+Q, X	⌥⌘Q X	Shows a particular view (Problems)
Alt+Shift+Q, S	⌥⌘Q S	Shows a particular view (Search)
Alt+Shift+Q, Y	⌥⌘Q Y	Shows a particular view (Synchronize)
Alt+Shift+Q, V	⌥⌘Q V	Shows a particular view (Variables)

Window

Shortcut for Windows /Linux	Shortcut for macOS	Action
F12	⌘F12	Activate Editor
	⌘W	Closes the active Dialog
Ctrl+3	⌘3	Find actions. Quickly access UI elements
Ctrl+M	^M	Maximize/restore a state of active view or editor
Ctrl+F6	⌘F6	Switch to the next editor
Ctrl+F8	⌘F8	Switch to the next perspective
Ctrl+F7	⌘F7	Switch to the next view
Ctrl+Shift+F6	⇧⌘F6	Switch to the previous editor
Ctrl+Shift+F8	⇧⌘F8	Switch to the previous perspective
Ctrl+Shift+F7	⇧⌘F7	Switch to the previous view
Ctrl+E	⌘E	Open the editor drop-down list
	⇧F10	Show the context menu
Alt+Shift+F3	⌥⇧F3	Shows contribution information for the currently selected element
Ctrl+Shift+L	⇧⌘L	Show the key assist dialog
Ctrl+F10	⌘F10	Show the context menu for the ruler

Alt+-	⇧⌘F10	Show the system menu
Ctrl+F10	⌘F10	Show the view menu
Ctrl+Shift+E	⇧⌘E	Switch to an editor
Alt+F11	^⌘F	Toggles the window between full screen and normal
Ctrl+_	⇧⌘-	Split or join the currently active editor (Horizontal)
Ctrl+_	⇧⌘[	Split or join the currently active editor (Vertical)

Database Navigator

Shortcut for Windows/Linux	Shortcut for macOS	Action
Ctrl+Shift+,	^⇧,	Link with editor
Ctrl+Alt+Shift+D	^⌥⇧D	Add Bookmark
Enter	↩	Opens task configuration
Ctrl+Shift+A	^⇧A	Set as default object

Properties Entity Editor

Shortcut for Windows/Linux	Shortcut for macOS	Action
Alt+Shift+Up	Alt+Shift+Up	Switch to the next entity property left-side tab
Alt+Shift+Down	Alt+Shift+Down	Switch to the previous entity property left-side tab

Navigation tree

Shortcut for Windows/Linux	Shortcut for macOS	Action
Alt+Left	⌘[	Move backward in the editor navigation history
Ctrl+Shift+Numpad_Divide	⇧⌘Numpad_Divide	Collapse the current tree
Ctrl+Shift+Numpad_Multiply	⇧⌘Numpad_Multiply	Expand the current tree

Alt+Right	⌘]	Move forward in the editor navigation history
Ctrl+G	^G	Go to a specified line of text in Windows
Ctrl+L	⌘L	Go to a specified line of text (Editing Text)
Ctrl+.	⌘.	Navigate to the next item
Ctrl+Alt+Right	^⌥→	Next edit location
Alt+F7	⌥F7	Switch to the next page
Alt+PageDown	⌥⇟	Switch to the next sub-tab
Ctrl+PageDown	^⇟	Switch to the next tab
Ctrl+Shift+R	⇧⌘R	Open an editor on a particular item
Ctrl+,	⇧⌘.	Navigate to the previous item
Ctrl+Alt+Left	^Q	Previous edit location
Alt+Shift+F7	⌥⇧F7	Switch to the previous page
Alt+PageUp	⌥⇞	Switch to the previous sub-tab
Ctrl+PageUp	^⇞	Switch to the previous tab
Alt+Shift+W	⌥⌘W	Open the Show In menu

File

Shortcut for Windows/Linux	Shortcut for macOS	Action
F2	F2	Rename the selected item
F5	F5	Refresh the selected items
Ctrl+S	⌘S	Save the changes in current file
Ctrl+Shift+S	⇧⌘S	Save changes in all open files
Ctrl+F4	⌘W	Close the active editor
Ctrl+Shift+F4	⇧⌘W	Close all editors
Ctrl+N	⌘N	Open the New item wizard

Alt+Shift+N	⌥⌘N	Open the New menu
Ctrl+O	^O	Open a file
Ctrl+P	⌘P	Print
Alt+Enter	⌘I	Display the properties of the selected item

Run/Debug

Shortcut for Windows/Linux	Shortcut for macOS	Action
Ctrl+Alt+M	⌥⌘M	Add memory block
Ctrl+W	⌘W	Close the selected rendering
F11	⌘F11	Launch in debug mode
Ctrl+Z	^D	Send end of file
Ctrl+G	⌘G	Go to Address
Ctrl+N	⌘N	Add a new rendering
Ctrl+Alt+N	⌥⌘N	Show renderings from next memory monitor.
Ctrl+Shift+.	⇧⌘.	Load next page of memory
Ctrl+Shift+,	⇧⌘,	Load previous page of memory
Ctrl+F11	⇧⌘F11	Launch in run mode
Ctrl+R	⌘R	Resume and break when execution reaches the current line
Ctrl+Alt+B	^⌥B	Sets whether or not any breakpoint should suspend execution
F5	F5	Step into
F6	F6	Step over
F7	F7	Step return
F8	F8	Resume
Ctrl+F2	⌘F2	Terminate
Ctrl+Shift+B	⇧⌘B	Creates or removes a breakpoint

Ctrl+T	⌘T	Toggle visibility of the Memory Monitors Pane
Shift+F5	⇧F5	Toggles enablement of debug step filters

Git

Shortcut for Windows/Linux	Shortcut for macOS	Action
Ctrl+Left	⌘←	Collapse Working Tree
Ctrl+#	⌥⌘3	Commit
Ctrl+C	⌘C	Copy Path to Clipboard
Ctrl+O	⌘O	Show the quick outline for a unified diff
F2	F2	Rename Branch
Ctrl+Shift+K	^⇧K	Commit changes to Git
Ctrl+Shift+U	^⇧U	Update changes from Git

Project

Shortcut for Windows/Linux	Shortcut for macOS	Action
Ctrl+B	⌘B	Build all projects

Oracle

Shortcut for Windows/Linux	Shortcut for macOS	Action
Ctrl+F9	^F9	Compile

Utility

	Shortcut for	

Shortcut for Windows&Linux	macOS	Action
Ctrl+Shift+C	^⇧C	Advanced Copy
Ctrl+Shift+V	^⇧V	Paste with extra settings
CTRL+ALT+PAGE_UP CTRL+ALT+PAGE_DOWN		Switch between tabs in database object Properties view
Alt+Insert	⌘+N	Create Object It is available both in the **Database Navigator** and the **Properties Editor**.

Help

Shortcut for Windows&Linux	Shortcut for macOS	Action
F1	F1	Open the documentation

References

`*` - Current query is the query under cursor or the selected text. Query is separated from other script queries by delimiter (; by default) or by empty lines.

`**` - Current script is a set of all queries in the current SQL file. If there is a text selection then only queries in this selection are processed. Queries are separated from each other with a delimiter (; by default).

`***` - Current connection detected from active window and selection. If active (focused) window is SQL editor or database object editor then the current connection is the same as in this editor. If the active window is the database navigator then the active connection is the "owner" connection of the currently selected element. In other cases there is no current connection and DBeaver will ask you to choose the connection explicitly.

Accessibility

DBeaver offers you the ability to use most of its features with a keyboard and screen readers. Here you will find a guidelines on how to use DBeaver without a mouse in some common cases.

Note: DBeaver is tested mainly for use with NVDA screen reader and Windows keyboard shortcuts.

Table of contents:

- General
  - Structure and navigation
  - Shortcuts
  - Colors and fonts
- Views and Wizards
  - Database Navigator
  - New Connection
  - Object Properties
  - SQL Editor
  - Result Set
  - ER Diagrams

General

Structure and navigation

DBeaver is an application consisting of many parts, so-called Views, such as Database Navigator, SQL Editor, Data Editor, etc. A Toolbar with many buttons inside each View and horizontal and vertical tabs allows you to switch between **Views** and inside them.

Main navigation shortcuts

- You can switch Views by using `Ctrl + F7 | #F7`.
- You can switch between buttons on the toolbars and menus using **Tab** and the **arrows**.

How to switch tabs

- You can switch horizontal tabs by using `Ctrl + PageUp / PageDown`.
- Sub-tabs can be switched by using `Alt + PageUp / PageDown`.
- If there are vertical tabs inside the view, you can switch them by using `Ctrl + Alt + PageUP / PageDown`.

When you focus on any interface object, such as a table, column, tab, or cell, you can open the **Context menu** to see all the actions available for that object.

To open the Context menu, use `Shift + F10 | #F10`.

Shortcuts

Most of DBeaver's functions are accessible through special shortcuts. You can find them in **Window -> Preferences –> User Interface –> Keys**. If you mostly use the keyboard, you can switch to the **DBeaver Keyboard Only** scheme, which has additional shortcuts.

Learn more DBeaver shortcuts

Colors and fonts

Themes: DBeaver supports theme customization to enhance your user experience. Among the available options, DBeaver provides integration for the High-Contrast theme native to Windows systems. For more details on this and other themes, see our comprehensive theme guide.

Fonts: To change fonts in DBeaver, navigate to **Window -> Preferences –> User Interface —> Appearance —> Color and fonts**, and open DBeaver fonts. The **Main font** allows you to change the font family, size, and color in most elements of DBeaver's interface, including the **Database Navigator** and the **Data Grid**. The Monospace font changes the font in the **SQL Editor** and other **Views** with monospaced fonts.

Data Colorization: You can color rows in the **Data Grid** by following some rules or even by selecting colors for particular data types.

DBeaver Views and Wizards

Let's see some comments on using the most popular **Views** in DBeaver with the keyboard and screen reader.

Database Navigator

In the **Database Navigator**, you can view and open all the database connections and objects inside them. You can move between database objects and open nodes using arrows.

- To view object properties, use `F4`.
- To open **SQL Editor** associated with the selected object, use `F3`.

See all **Database Navigator** shortcuts

New Connection

To create a new connection, open the **Wizard window** using `Ctrl+N` | `#N`, then type **connect** and select **Database connection**.

See **Connection** shortcuts

Object Properties

The Properties Editor is used to see all the information about database objects. When you focus on the first row of the table, NVDA reads the contents of the row with the corresponding column names.

- You can open object properties with `F4`.
- To switch between left-side tabs, use `Alt + Shift + Up | Alt + Shift + Down`.

See **Properties Editor** shortcuts

SQL Editor

The SQL Editor allows you to create and execute SQL scripts associated with a database connection. To open the **SQL Editor**, focus on the connection or table in the **Database Navigator** and press `F3`. To execute the SQL script under the cursor, press `Ctrl + Enter`.

- See **SQL Editor** shortcuts

Result Set

When you place your focus on the first cell in the data table, the NVDA screen reader says the table name and the name of a column consisting of this cell. If you move from one cell to another, the screen reader announces the column names when you change focus.

If you need to work with columns using the keyboard, you can select an entire column, copy the column name, and resize the column using special keyboard shortcuts. Some of them are only available in the **DBeaver Keyboard Only** scheme.

See Result Set shortcuts

ER Diagrams

ER Diagrams are fully accessible with the keyboard and the NVDA screen reader. You can move inside the table, move or resize it, and listen to the screen reader pronounce the table name and column names inside that table.

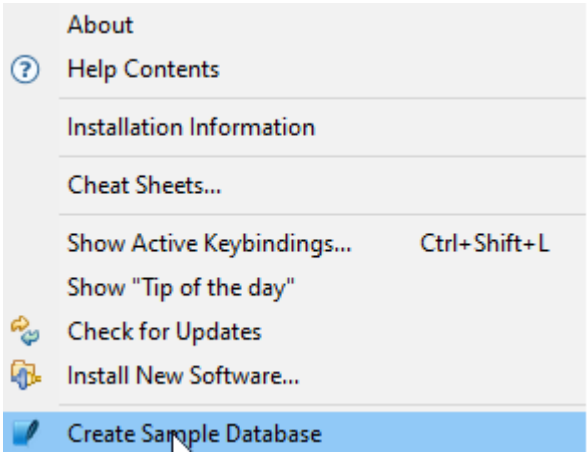
Learn more about ER Diagram shortcuts

Sample Database

Introduction

If you want to test **DBeaver** features, you can create a demo database. There is a specialized database that can be created. You can create a sample *SQLite* database at the first launch. The database can be added at any time by the following sequence. Chinook is used as a reference database

> Help -> Create Sample Base



Database Connections

To be able to manage your database in DBeaver, you need to create a connection to this database – see Creating Connections. A connection includes a driver and a number of configuration parameters including the location of the database and credentials to access it. You need to create a separate connection to every database you want to manage. Every database type requires its own set of connection parameters.

Connections reside in the Database Navigator and in the Projects views. In these views, you can:

- Edit connections, see Editing Connections
- Rename and delete connections - via corresponding context menu items, see Database Navigator
- Connect to and disconnect from databases using connections, see Connect to Database and Disconnect from Database.

Database connections might have the following states:

 - not connected

 - has network settings specified (such as SSH tunnel, etc.)

 - connected

 - connection error

Also see:

- Disable multiple connections opening for MFA authentication

Create Connection

DBeaver provides a wizard that guides you through the steps to create a connection. To create a connection, do one of the following:

Use the New Connection Wizard

- Click the **New Connection Wizard** button  in the Toolbar or in the Database Navigator view toolbar.

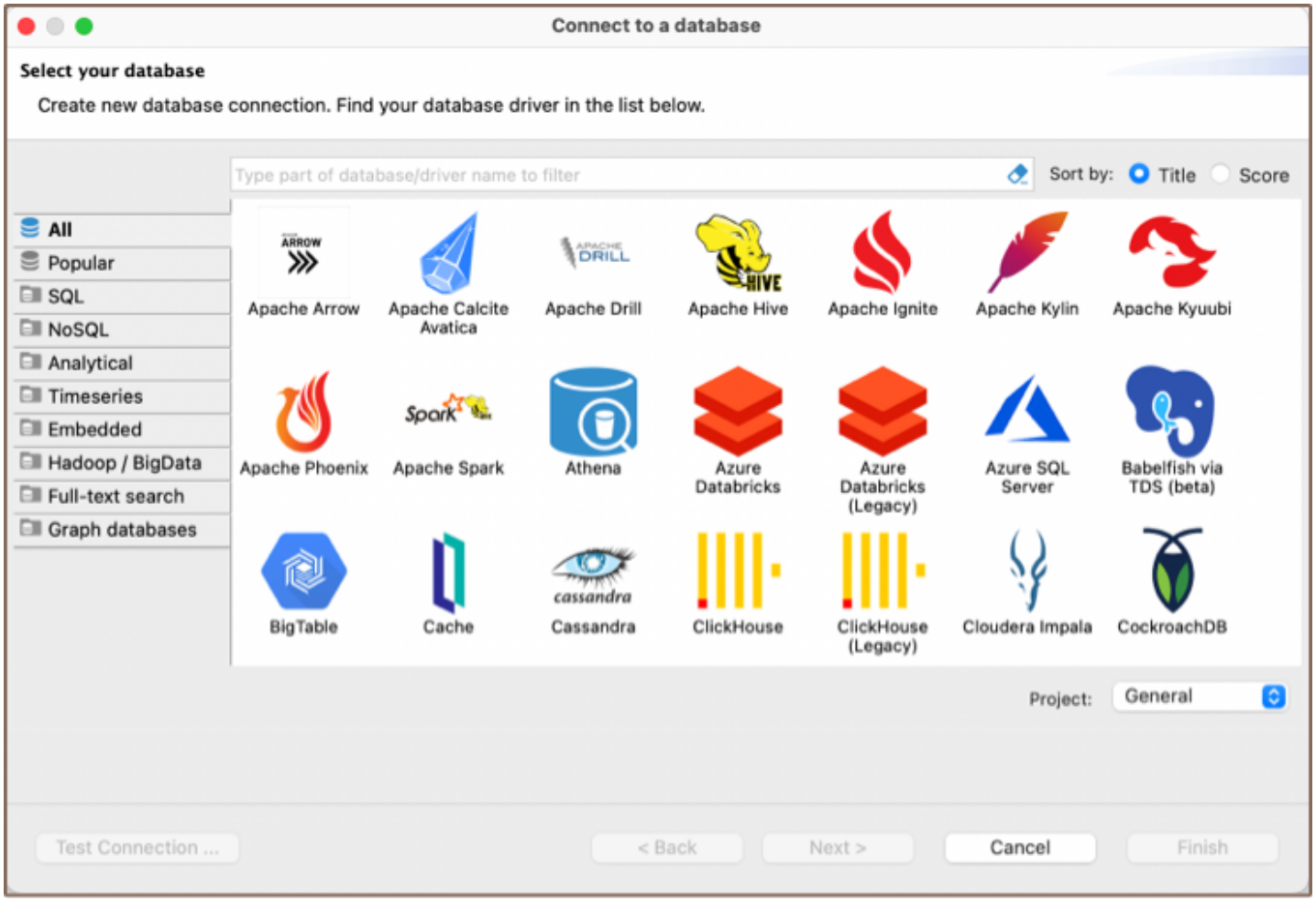
Use the Menu Bar

- Click **Database** -> **New Database Connection** in the menu bar.
- Press `Ctrl+N` -> **DBeaver**. Then, in the wizard, click **Database connection** and then click **Next**.
- click **File** -> **New** -> **DBeaver**and then, in the wizard, click **Database connection** and then click **Next**.

Use the Database Navigator

- Right-click in the **Database Navigator** window area and select **Create** -> **Connection**.

Using the Create New Connection Wizard



To quickly find the needed driver for your database connection, you can:

- Type a hint in the text field above the list of drivers.
- Browse the database sections to find the required database.
- Search for a section using the **Search** field.
- Sort the options by **Title/Score**.

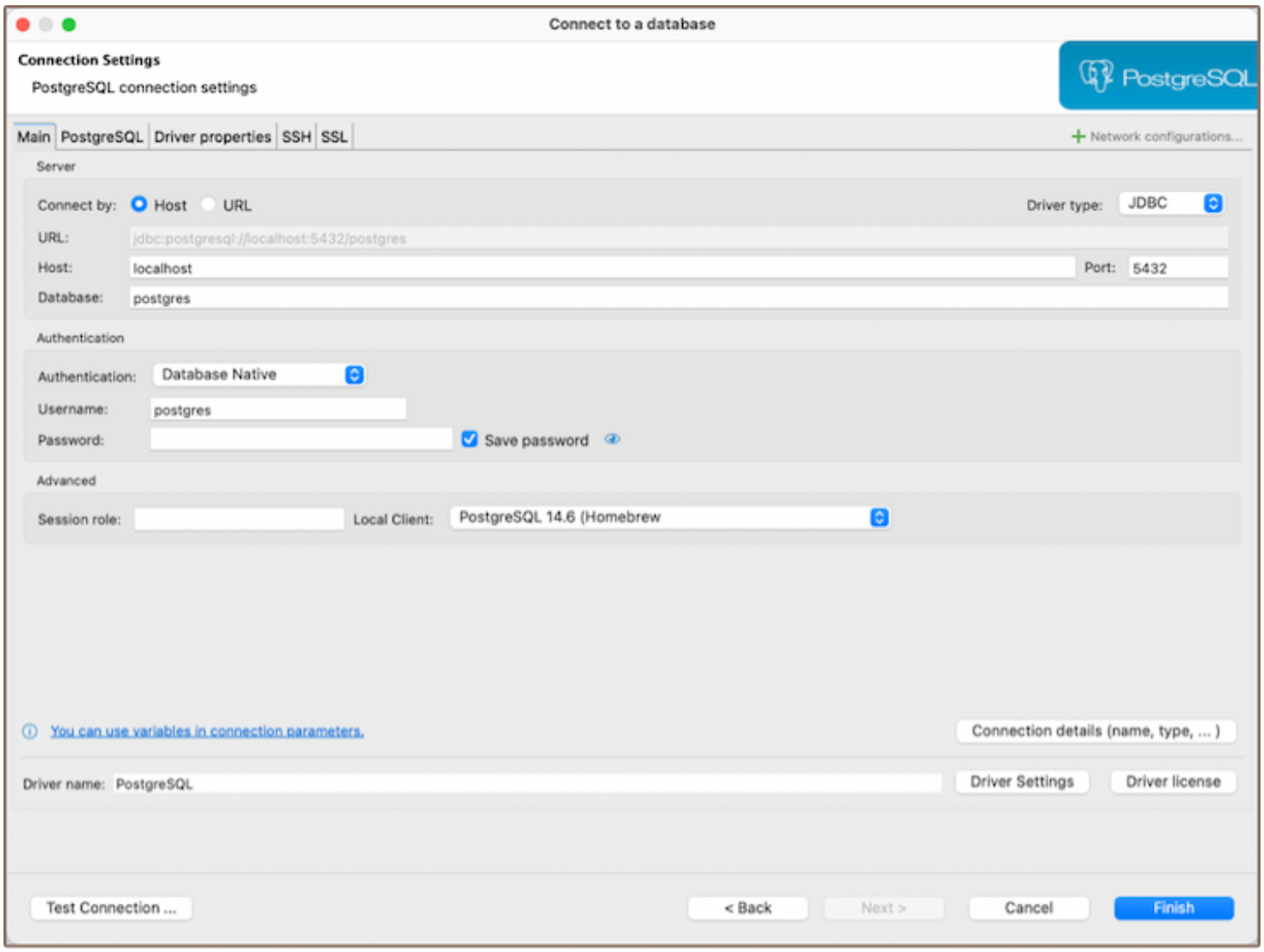
After selecting the required driver, click **Next**.

Tips

- If you cannot find a driver for your database, then probably there is no suitable driver, and you need to create one. Please see the Database Drivers article.
- Within the **New Connection Wizard**, you can select the project where the database will be saved.

Configuring Connection Settings

In the Connection Settings screen, on the **Main** tab, you need to set all primary connection settings. Here's what you typically need to configure:



Setting	Description
Host	The address of the database server.
Port	The port number for the connection.
Database Name	The name of the specific database you want to connect to.
Username and Password	The credentials required for authentication. This falls under the Authentication section, where different databases may use various types of authorizations. Common types include Database Native and DBeaver Profile.

However, the number and type of connection properties are very dependent on the driver. For example, embedded drivers (such as SQLite, Derby Embedded, HSQLDB, H2 Embedded), unlike remote ones, require only the path to the database. For more detailed information about the settings specific to each driver, you can refer to the driver documentation articles available in the DBeaver documentation.

Additional Settings

If necessary, you can specify advanced settings. See the Advanced Settings section below for more details.

Testing and Finalizing the Connection

- **Test the Connection**: Click **Test Connection** to verify if the connection works.
- **Finish**: Click **Finish**. The connection appears in the connections tree in the **Database Navigator**.

By following these instructions, you can configure and establish a connection to your desired database within DBeaver.

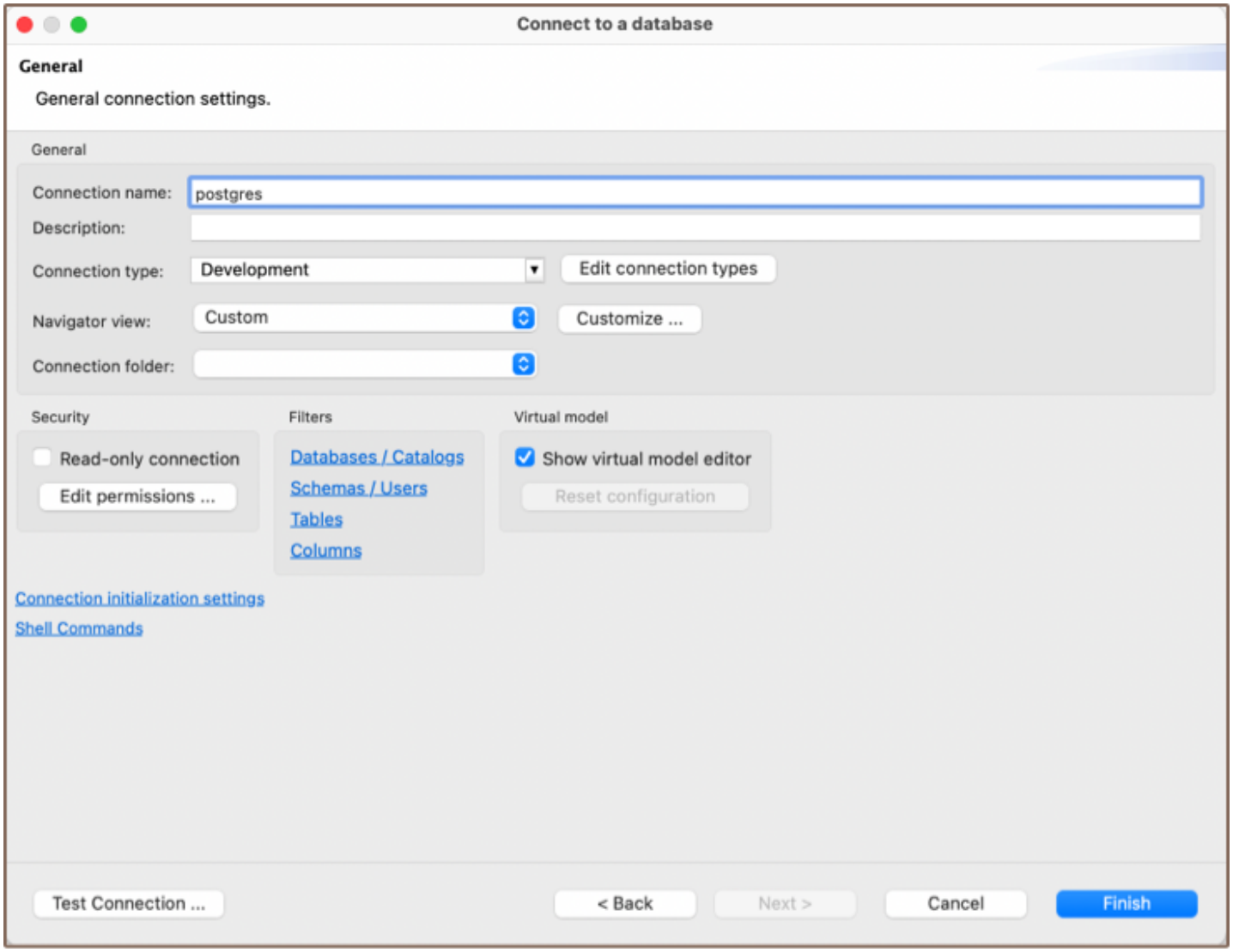
Advanced Settings

The Advanced Settings tab in the connection configuration process provides additional options to fine-tune your connection. Here's what you can configure:

Network Settings

Setting	Description
SSH Tunnel	Configure an SSH tunnel if your database requires a secure connection.
SOCKS Proxy	Set up a SOCKS proxy if you need to route the connection through a specific proxy server.
SSL Configuration	Enable and configure SSL if your connection requires encryption.
Kubernetes	Configure Kubernetes settings if your database is hosted within a Kubernetes cluster.

Connection Details (name, type, etc.)



Setting	Description
Connection Name	Assign a specific name to the connection for easier identification. You can use variables in all connection parameters and in the driver properties. Variables are system environment variables or one of the following list: `${host}` : Host name `${port}` : Port number `${database}` : Database name `${server}` : Server name `${url}` : Connection URL `${user}` : User name `${connection.type}` : Connection type (e.g., production, development) `${project.name}` : Project name `${date}` : Current date `${host_or_database}` : Host name or database name

Connection Type	Choose the type of connection, such as production, development, or test.
Descriptions	Add descriptions for your connection for better understanding.
Navigator View	Customize the navigator view settings.
Connection Folder	Organize your connections by placing them in specific folders.

Note: `${password}` property is deprecated due to security concerns since 23.2.0.

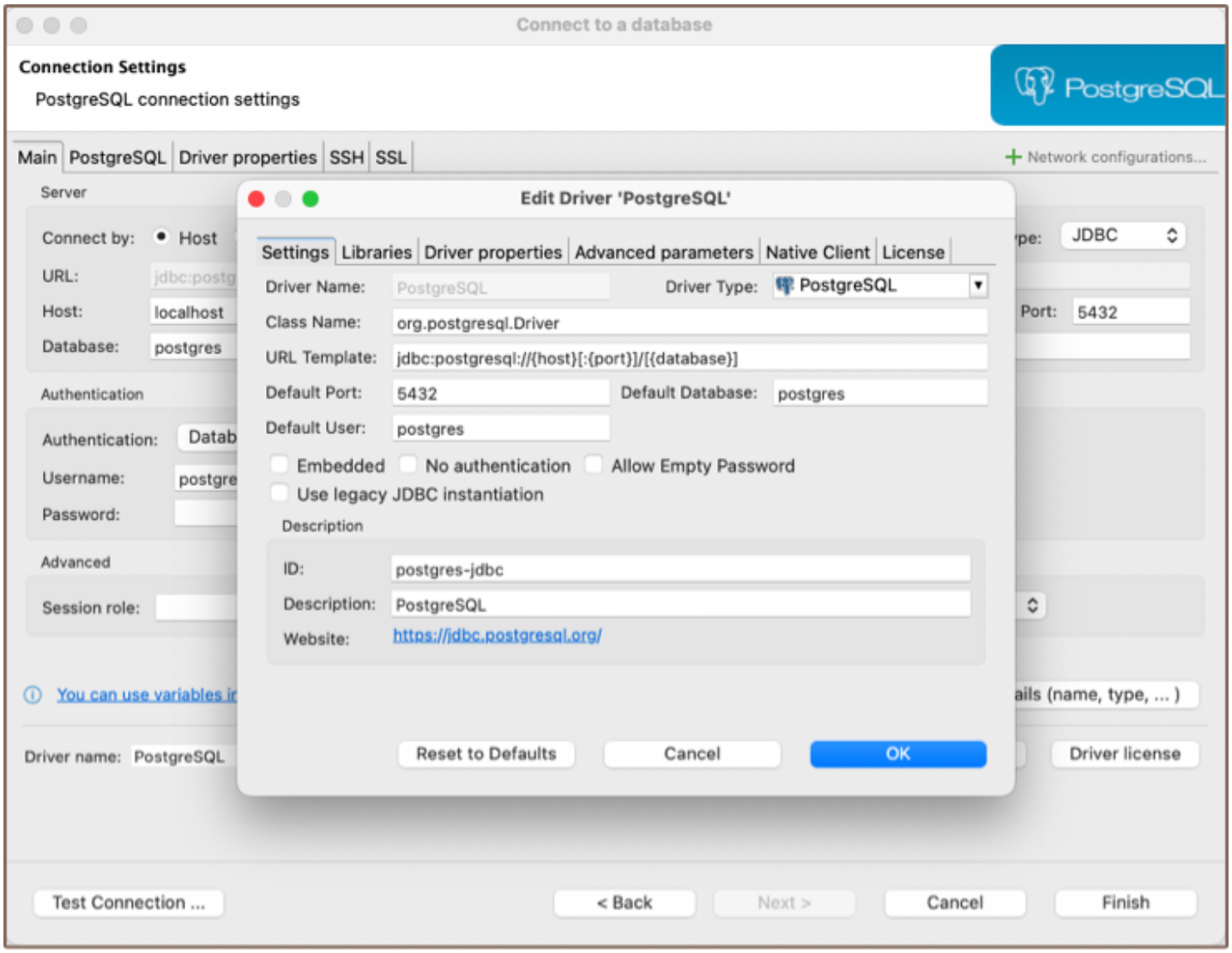
Other Options

Setting	Description
Security	Enable the Read-only connection checkbox and select **Permissions** by clicking the **Edit Permissions** button.
Filters	Configure filters for your connection. See the Filters article for more details.
Virtual Model Editor	Reset virtual model settings, including: - All virtual constraints and foreign keys. - All column transformers. - All table row colorings.
Connection Initialization Settings	Customize the initialization settings for your connection.
Shell Commands	Configure shell commands for your connection. See the Shell Commands article for more details.

These advanced settings allow you to tailor the connection to fit specific requirements and preferences. Be sure to consult your database documentation or system administrator if you need guidance on the appropriate settings for your particular database and use case.

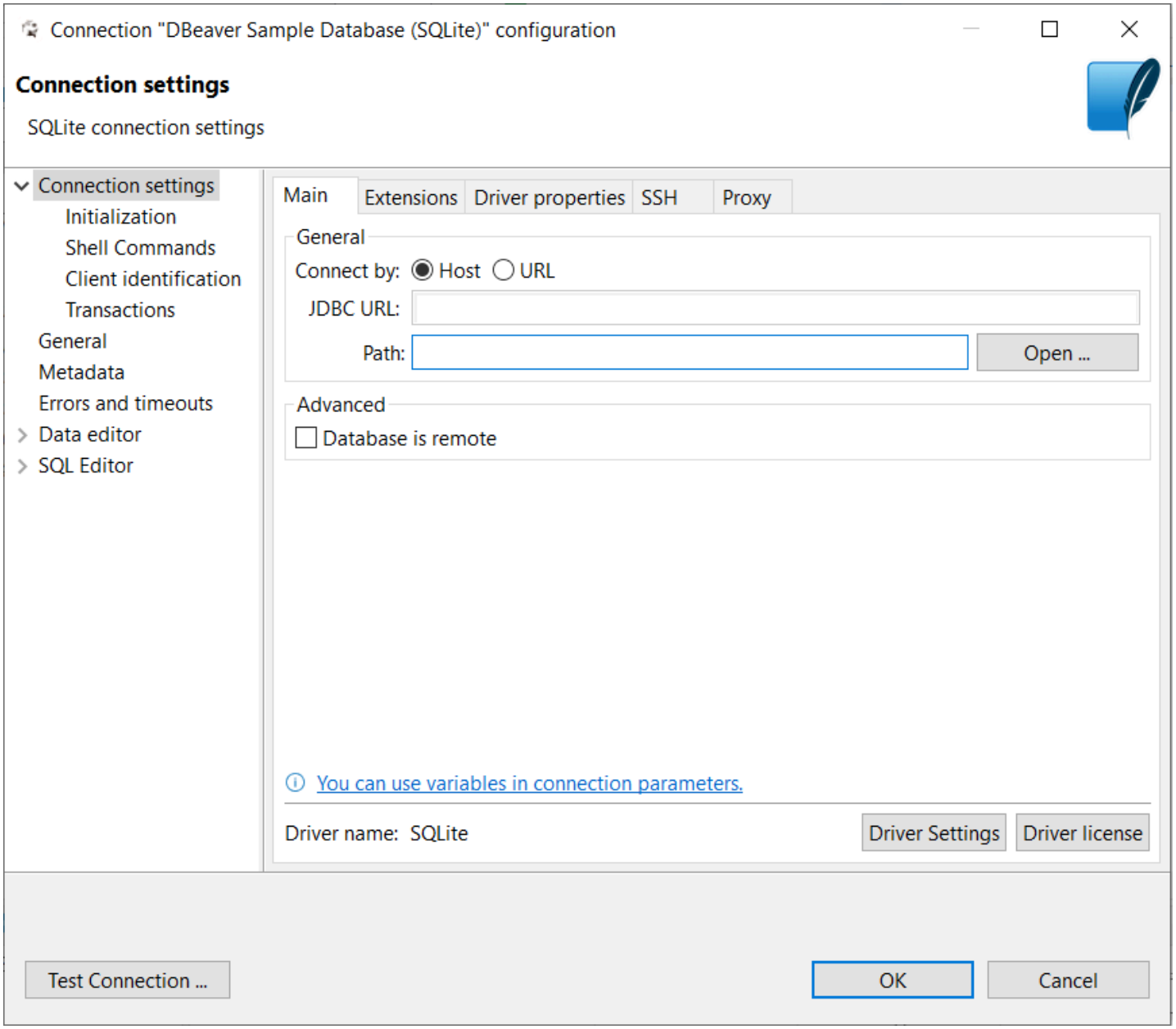
Driver Properties

Each driver has its own set of additional properties. Refer to the driver documentation to get information about available properties and their values. Understanding and configuring these properties can help you optimize the connection according to the specific requirements and behavior of the database driver you are using.



Edit Connection

To edit the configuration settings of a database connection, in the Database Navigator or in the Projects view, right-click the connection and click **Edit Connection** on the context menu. The Connection configuration window will open:



The navigation pane on the left displays the configuration sections, most of which are the same as those in the Create new connection wizard, see Connect to Database. There are additional configuration sections as well, such as **Result Sets** and **SQL Editor**. Click the section name to open the configuration settings for editing.

You can test if your connection works with modified settings - click **Test Connection**. When you finish editing your connection, click **OK** to save the changes or **Cancel** to discard them.

Driver settings

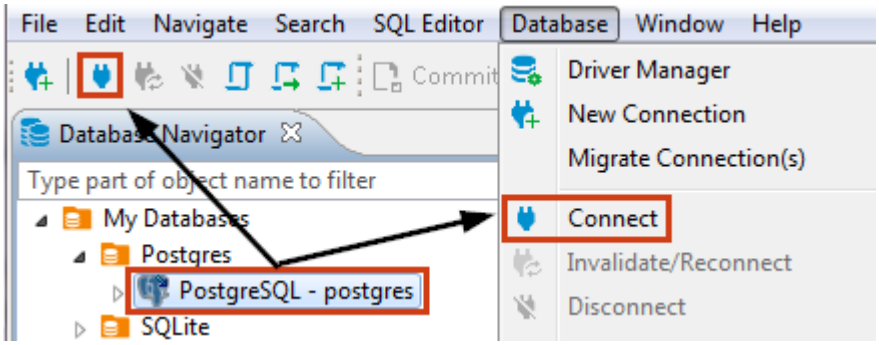
- In development

JDBC Time Zones

Connect to Database

To be able to work with the content and structure of a database, you need to connect to it. When you create a new connection to a database, DBeaver automatically connects to the new database, see Create Connection.

To connect to a database using an existing connection, in the Database Navigator or Projects view, click the connection and then click the Connect button in the toolbar or click **Database -> Connect** on the main menu:



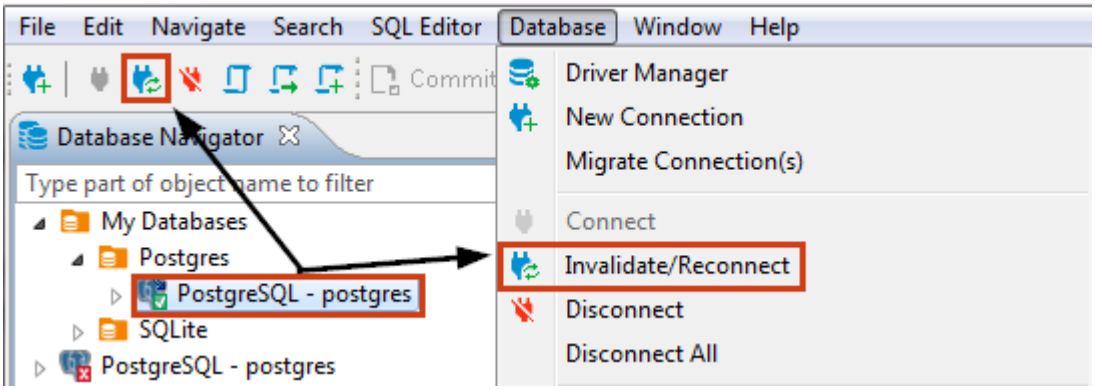
You can also right-click the connection and click **Connect** on the context menu.

If a database connection exists but DBeaver is not connected to the database, the connection appears with its original icon (for example,  for PostgreSQL database). When DBeaver connects to the database, the icon changes to signal the connected status:  .

If DBeaver cannot connect to a database, the connection will appear with an error sign:  . If you attempt to connect to such a database, DBeaver displays an error message describing the cause for the error.

Invalidate/Reconnect to Database

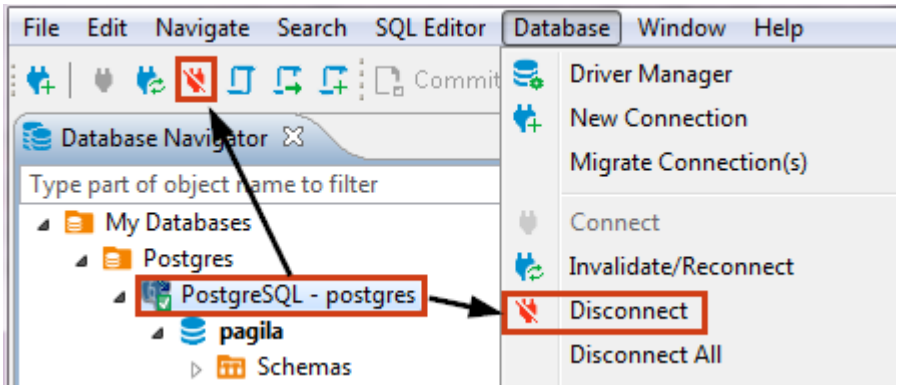
You might need to invalidate a database connection and then reconnect to it again in cases such as where the connection to the server is temporarily lost, etc. To invalidate a database connection and then reconnect to the database, click the database connection in the Database Navigator or Projects view, and then click the **Invalidate /Reconnect** button in the toolbar or **Database -> Invalidate/Reconnect** on the main menu:



You can also right-click the connection and click **Invalidate/Reconnect** on the context menu.

Disconnect from Database

You might need to disconnect from a database to free up resources or close transactions. To disconnect from a database, click the connection in the Database Navigator or Projects view, and then click the Disconnect button in the toolbar or click **Database -> Disconnect** on the main menu:



You can also right-click the connection and click **Disconnect** on the context menu.

NOTE: The Disconnect button and menu items are available only for those connections that are activated, that is, marked with the connected sign: .

When DBeaver disconnects from a database, its icon changes to its original state (not connected), for example, for PostgreSQL database.

To disconnect from all active connections, click **Database -> Disconnect All** on the main menu.

Change current user password

DBeaver can change credentials for the current database user.

Databases that support this feature:

- PostgreSQL
- Greenplum
- Cockroach
- Redshift ★
- Netezza ★
- SQL Server
- Oracle
- Exasol
- Snowflake ★
- Vertica

Changing expired user password

Different databases perform password management for users in different ways. For some of them, you can change the user password after the expiration date. It works for Oracle, PostgreSQL, and Netezza databases.

For the rest, database users must change the password before the expiration. Otherwise, it will be impossible to do it in DBeaver.

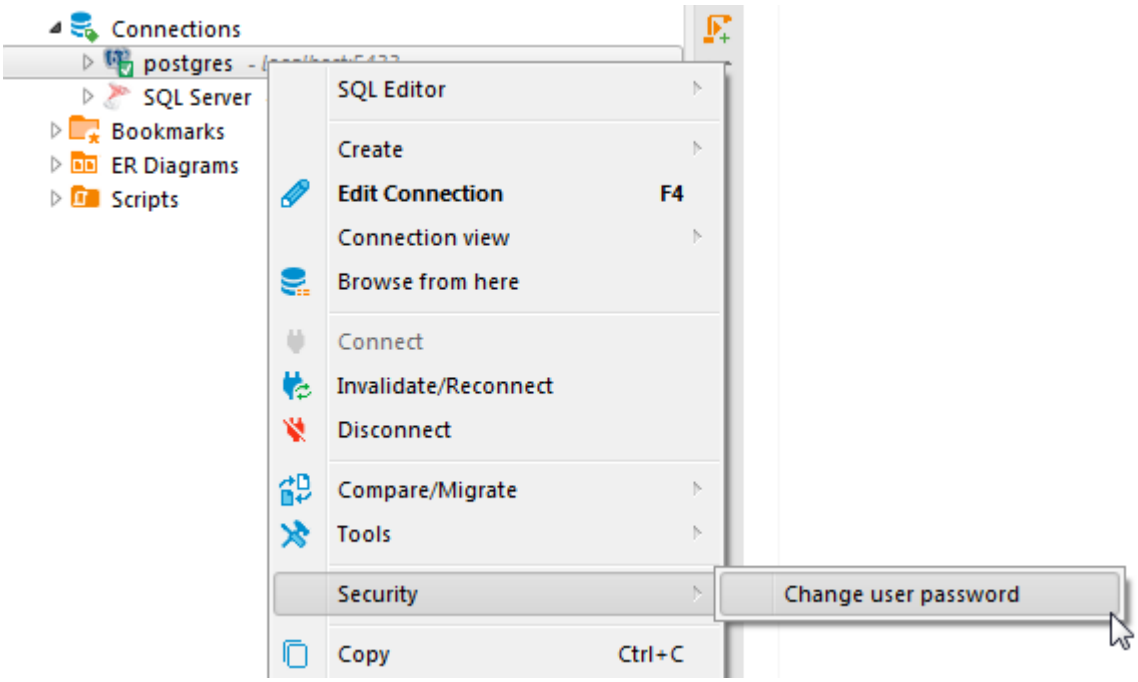
How to change the user password

You can change the current user password in the Navigation Tree according to the instructions below:

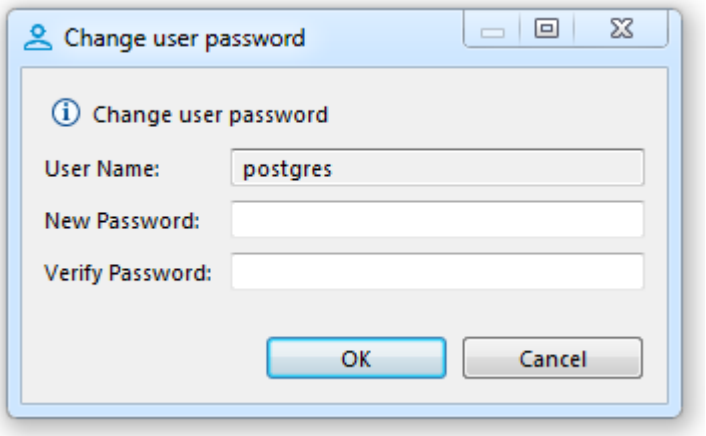
1. Connect to the database.

2. Open the context menu by right-clicking on the connection in the Navigation tree.

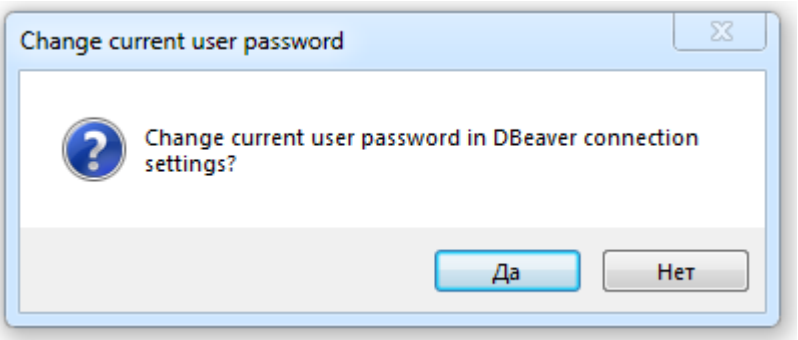
3. Select a "Security" point and click on a "Change user password" point in a sub-menu.



4. When a new password input dialog opens, enter and confirm the new password.



5. Confirm the password changes. (This dialog will not appear if the entered password is incorrect).



The password has been changed.

SSH Configuration

Sometimes the database cannot be accessed directly, in such cases you can use the SSH tunnel. The SSH or Secure Shell is a network communication protocol that allows two computers to communicate and encrypts the shared data.

DBeaver supports the following SSH authentication methods:

- Username/password
- Public key authentication
- Agent authentication

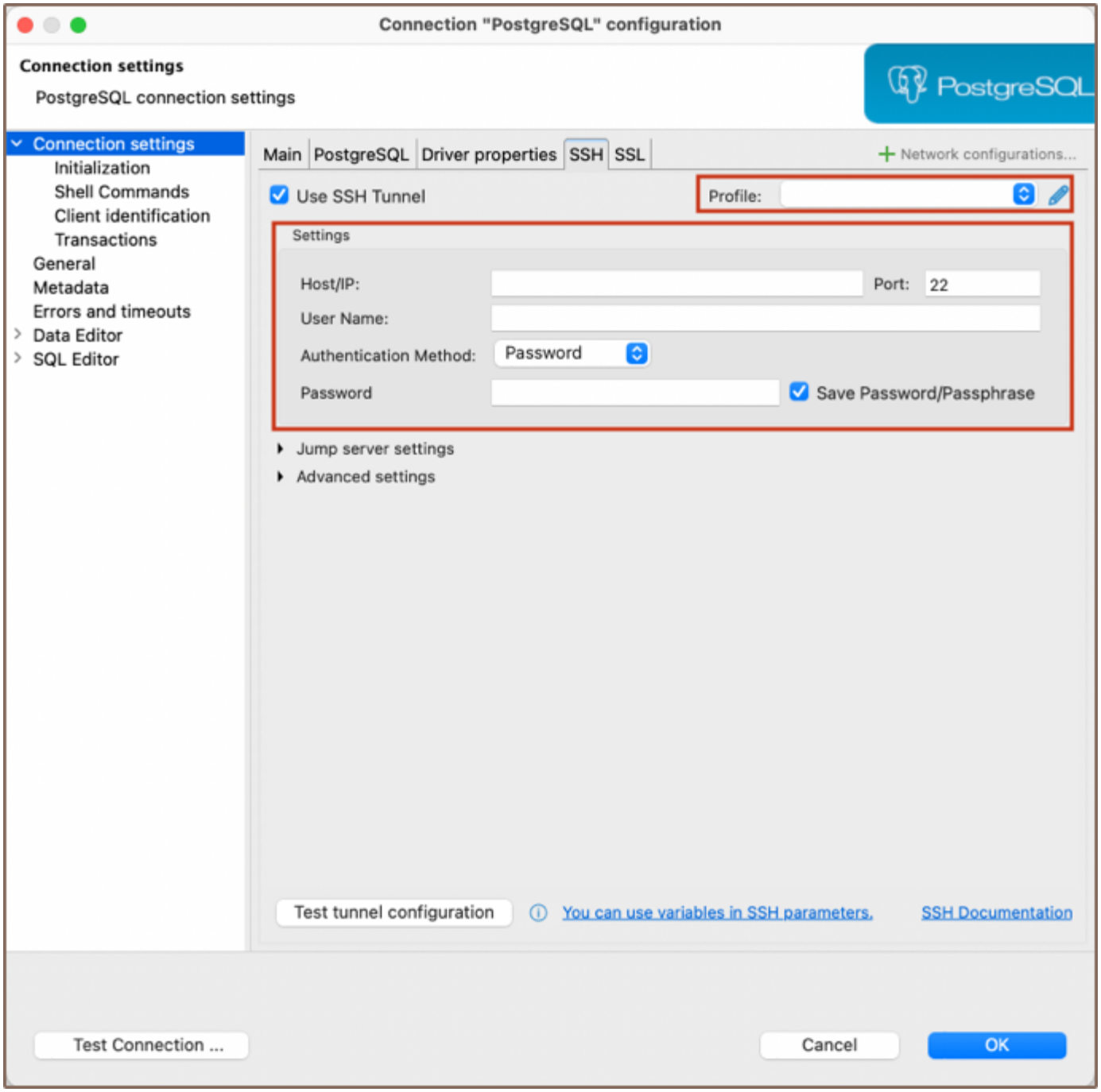
Available agent authentication implementations:

- pageant *(Windows only)*
- ssh-agent

Note that you have to run the agent first before connecting to the database in DBeaver.

Configuring SSH

SSH can either be configured individually for each connection in **Connection settings** -> **SSH** or as a network profile which can be selected from the drop-down menu on the same page mentioned earlier:



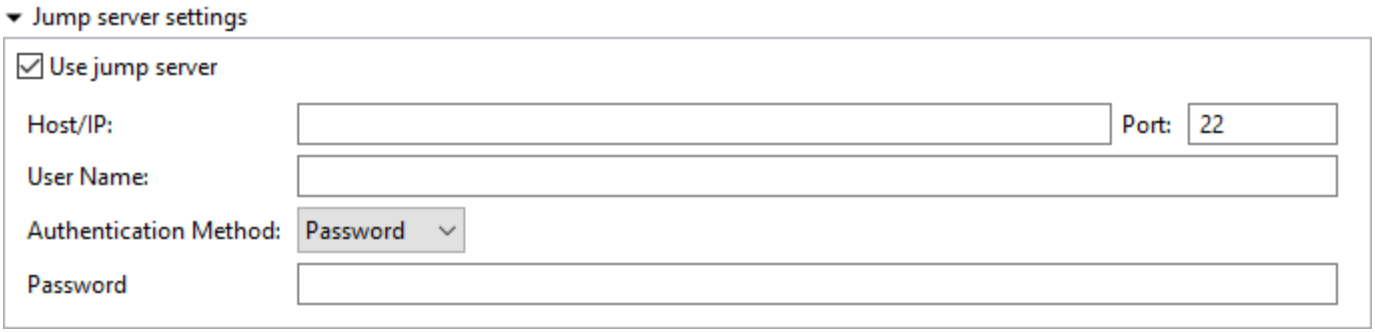
In order to use **SSH**, you have to enable it first. After that, configure it for your needs, then click on `Test tunnel configuration` to test whether all parameters are valid or not.

It is often required to set the hostname to `localhost` in **Connection settings** -> **Main**. **SSH** establishes a connection between two machines, authenticating each side to the other, and passing commands and output back and forth. After connecting to remote machine, all commands you execute are executed on that remote machine, so by having the host set to `localhost` you are opening a connection to the database on this remote machine, not on your local machine, and just redirect I/O to the latter.

Configuring Gateway Host

Gateway host, sometimes known as jump host, is used in situations when you cannot access a particular machine directly from your local machine, but it is possible to use a gateway server.

Just like SSH configuration, you have to enable it first. Gateway has the same configuration as for regular SSH tunnel:



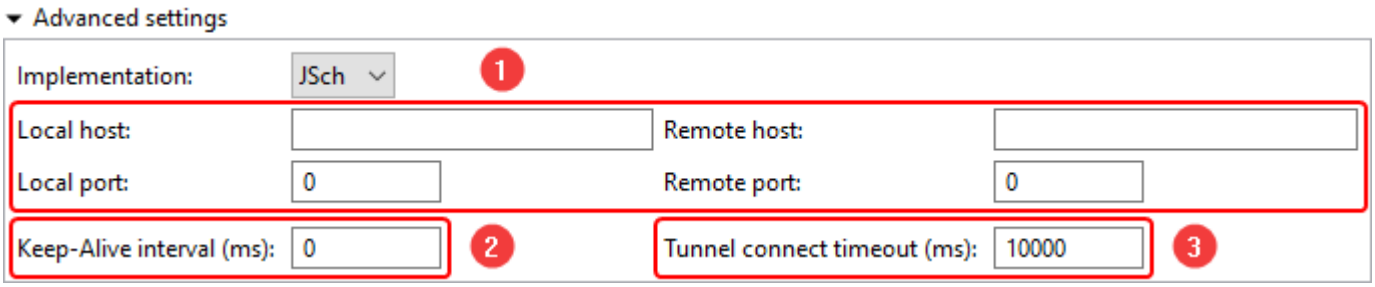
Practical use

Imagine you have the following situation: your database is located on remote machine with IP address `100.100.100.100` and which is accessible through SSH. You cannot access this IP directly from your machine because of your network settings or firewall, but you can access other machines with the IP address `200.200.200.200` through SSH and which **is able** to access the desired remote machine. You will need to specify `100.100.100.100` as a host in regular SSH configuration and `200.200.200.200` as a host in a gateway configuration to achieve such a "connection order": `localhost` → `200.200.200.200` → `100.100.100.100`.

Availability note

Jump servers are only available with Jsch implementation. See information below.

Advanced options



1. Optional port forwarding configuration, see information at below.
2. Keep-alive interval. Can be used to "ping" a remote machine to make sure that the SSH connection is still alive.
3. Connection timeout. If the destination SSH machine is quite slow and the connection takes ages to complete, you can adjust the value of this parameter.

Port Forwarding

Port forwarding is used to expose some socket sitting on a remote port to your local port, allowing you to communicate with it. After the SSH connection to the remote machine is established, you now need to connect to the database on that machine. Now you need to pass the port on which the database is listening to from that remote machine to your local machine. This is when port forwarding becomes handy.

By default, it randomly chooses the local port and forwards it to the destination port set in `Connection setting ⇒ Main`

It is not usually required to change this configuration, but it can be useful e.g. when you want to use a particular local port.

SSH + URL-based connections

By defaul DBeaver opens random local port and forwards it to a target database port. It then implicitly replaces port configured in the connection with this random port.
So technically DBeaver will connect to something like `localhost:45678` (radnom port on localhost) instead of `database.server.domain:5432` (default PostgreSQL server port).
But if you use URL instead of manual connection configuration then you connect to `jdbc:postgresql://database.server.domain/postgresql`. DBeaver cannot replace port number automatically because you can use any type of sophisticated URLs. DBeaver just don't know how to chnge this URL.

If you use URL connection then you must configure port forwarding manually. In the Advanced Settings set both Local Port and Remote Port to 5432 (database server port, 5432 is the default value for PostgreSQL). It will work in most cases.
But it will fail if you have PostgreSQL server on your local machine because port 5432 is already occupied. In this

case you can use any random port number for Loca Port (e.g. 15432). Use the same 5432 for Remote Port because it is real port number for remote PostgreSQL server.

Proxy configuration

External resources access

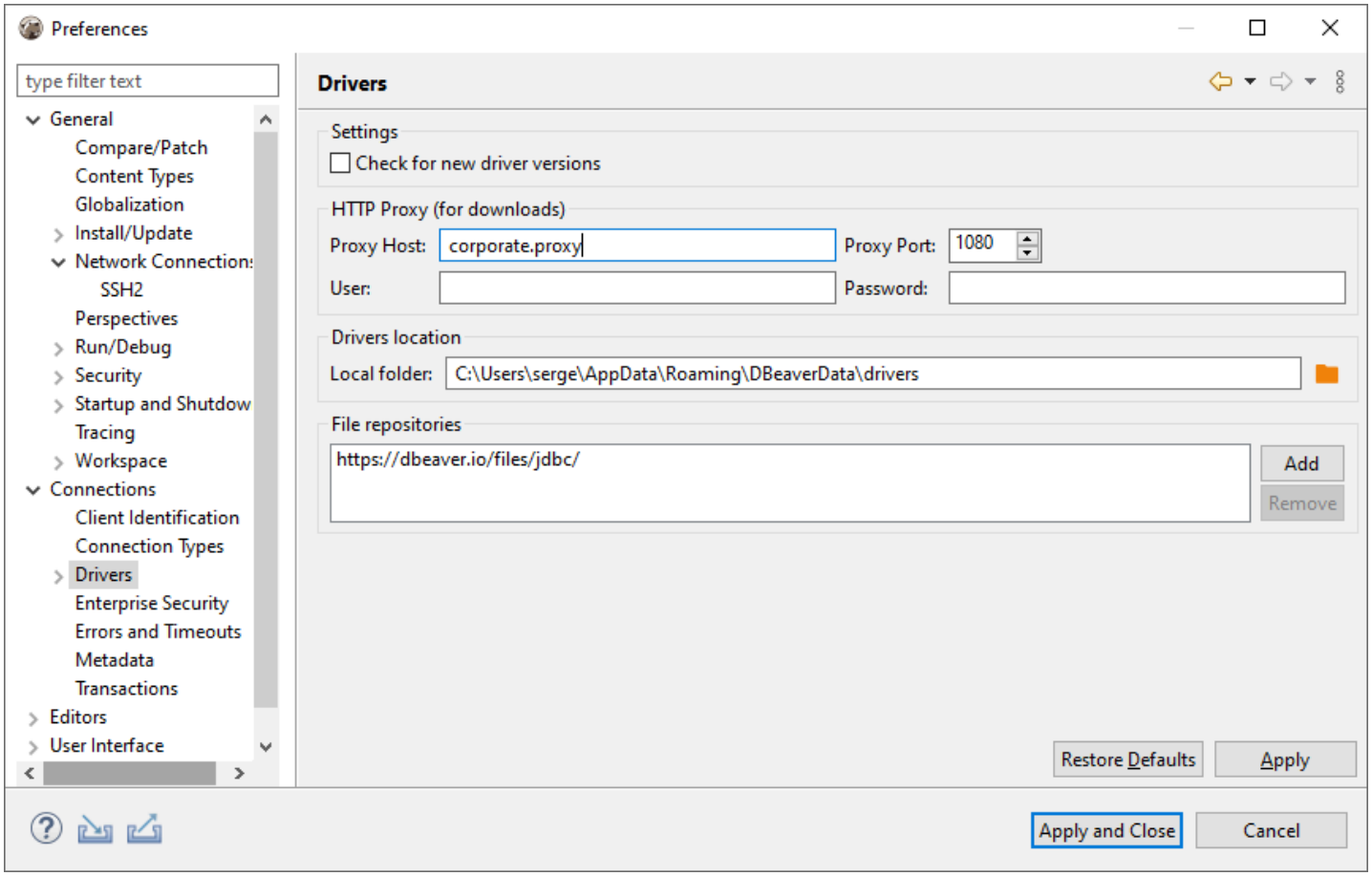
Sometimes DBeaver needs to access external internet resources for tasks such as:

- 3rd party JDBC drivers download
- Information about a new DBeaver version
- Connect to remote databases outside of your corporate network
- Subscription license activation (commercial version)
- License information update (commercial version)

If you are behind some corporate firewall which restricts access to external internet resources then it may become a real problem.
Sometimes corporate firewalls allow access to external resources using a web browser but restricts this for all other applications.

How to configure a proxy for drivers download

You need to ask your network administrator about proxy parameters.
Then go to Preferences->Connections->Drivers.



You can enter a proxy host/port and (optionally) user/password here. It will be used for drivers download only.

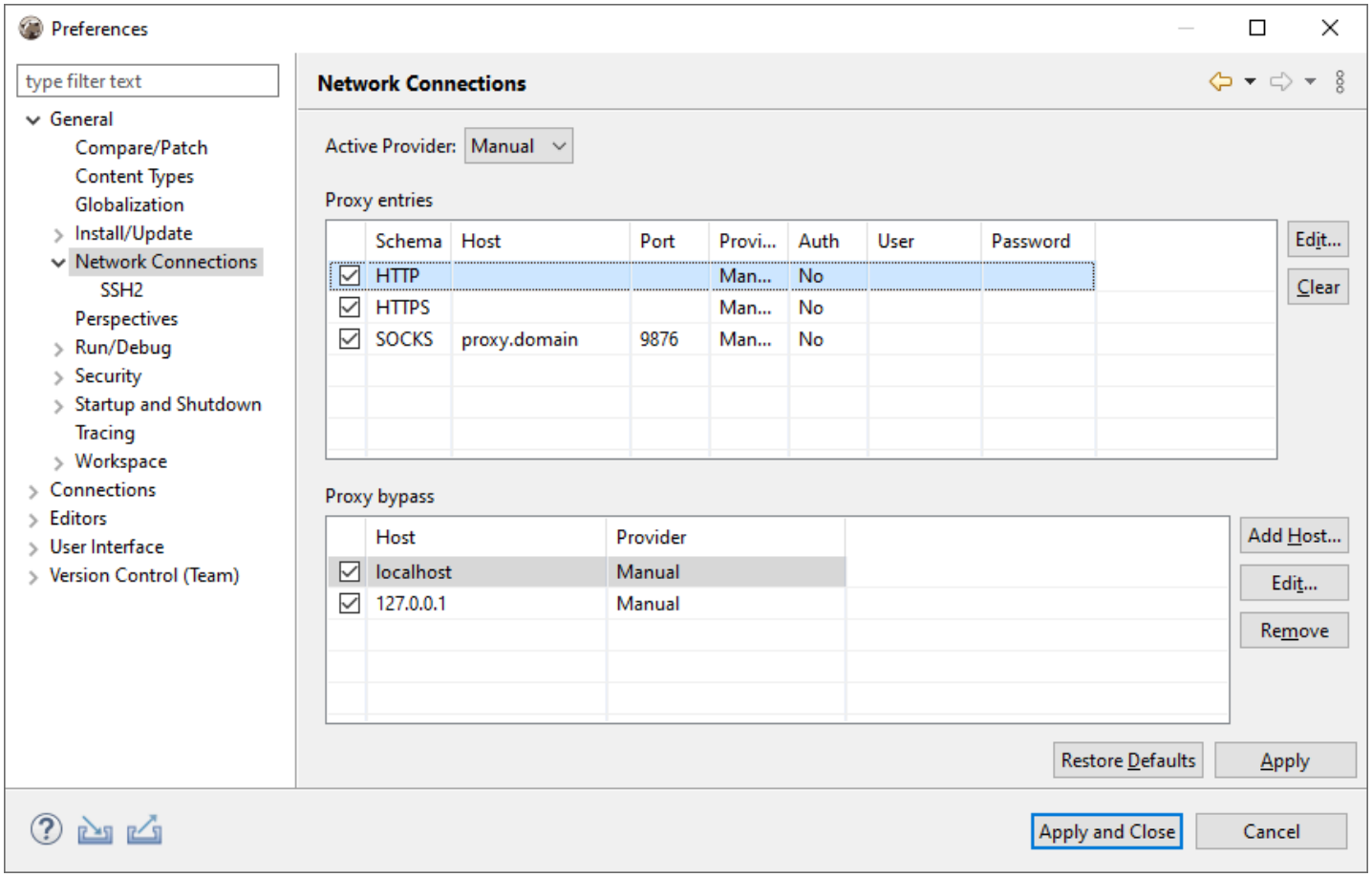
Drivers are usually downloaded from maven.org web site. You may also ask your network admin to add `maven.org` to the list of allowed external domains.

How to configure network for license activation

You need to configure a global proxy server.

If you cannot activate your subscription license then you first need to use a trial version to start DBeaver and configure a proxy.

Go to Preferences->General->Network Connections:



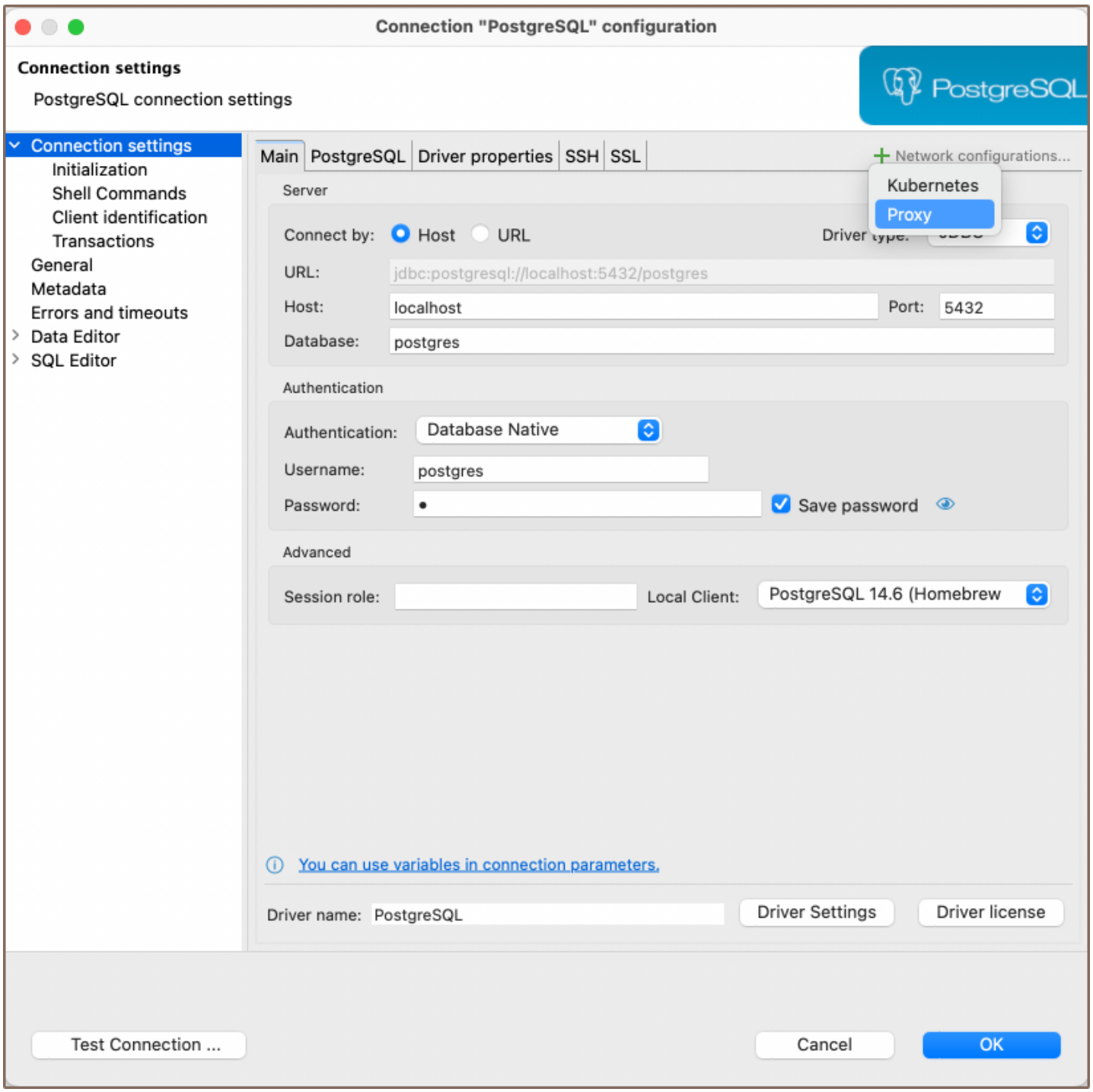
Switch to Manual or Native proxy (native proxy settings will use an active web browser proxy configuration).

Note: in order to activate/update a license DBeaver only needs to access the website `dbeaver.com`. You may ask your network administrator to add `dbeaver.com` to the white list.

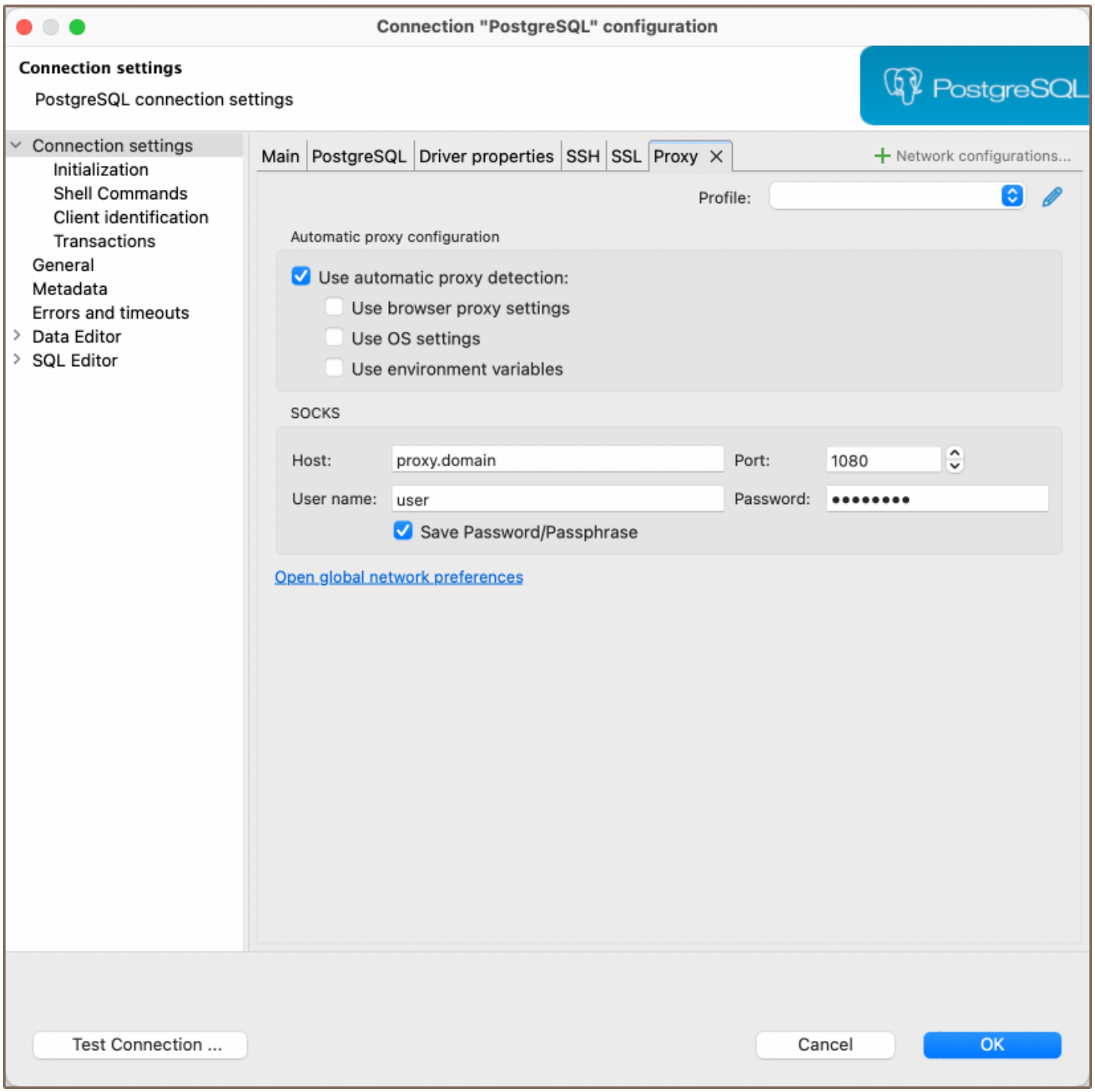
How to configure a proxy for external databases access

You can configure proxy settings for an individual connection.

1) **Add a Network Handler - Proxy**: To do this, click on the **Network configurations...** button and select **Proxy**.



2) **Set the Proxy Settings**: After selecting **Proxy**, a **Proxy** tab will appear in the menu at the top. You may set the proxy settings manually or use the active OS/web browser settings from this tab.



Note: Automatic proxy configuration is available in Lite, Enterprise, and Ultimate editions only.

Kubernetes configuration

Note: This feature is available in Lite, Enterprise, Ultimate and Team editions only.

In some scenarios, you may need to interact with databases that are part of a Kubernetes cluster. DBeaver offers a way to configure Kubernetes settings to facilitate this. Before proceeding, ensure that you meet the following prerequisites:

- You must have the kubectl tool installed on your local machine.
- You need a Kubernetes configuration file. Use the local one or consult your administrator for access.

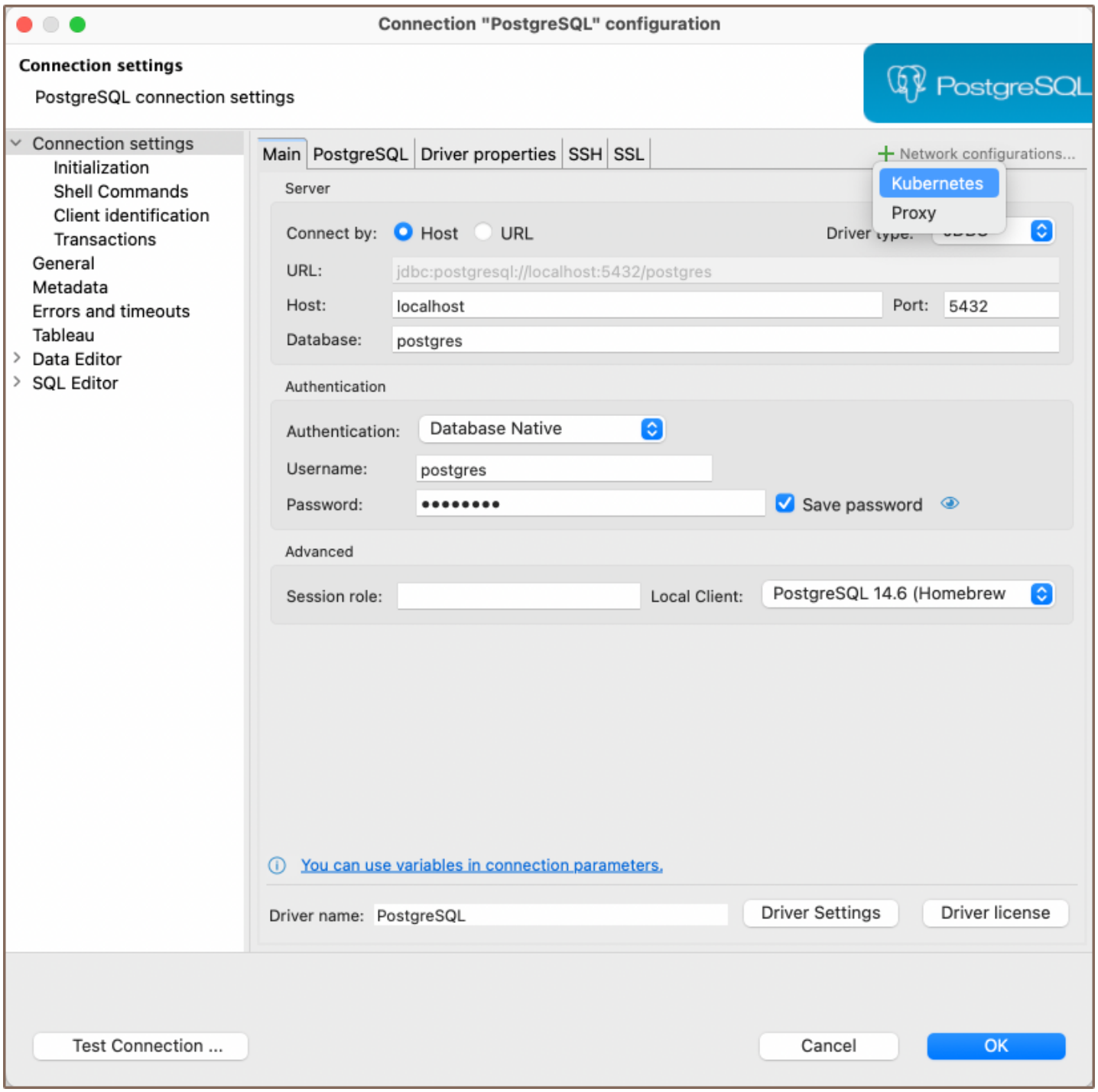
Table of contents

- Setting up Kubernetes
  - Kubectl settings
  - Testing port forwarding

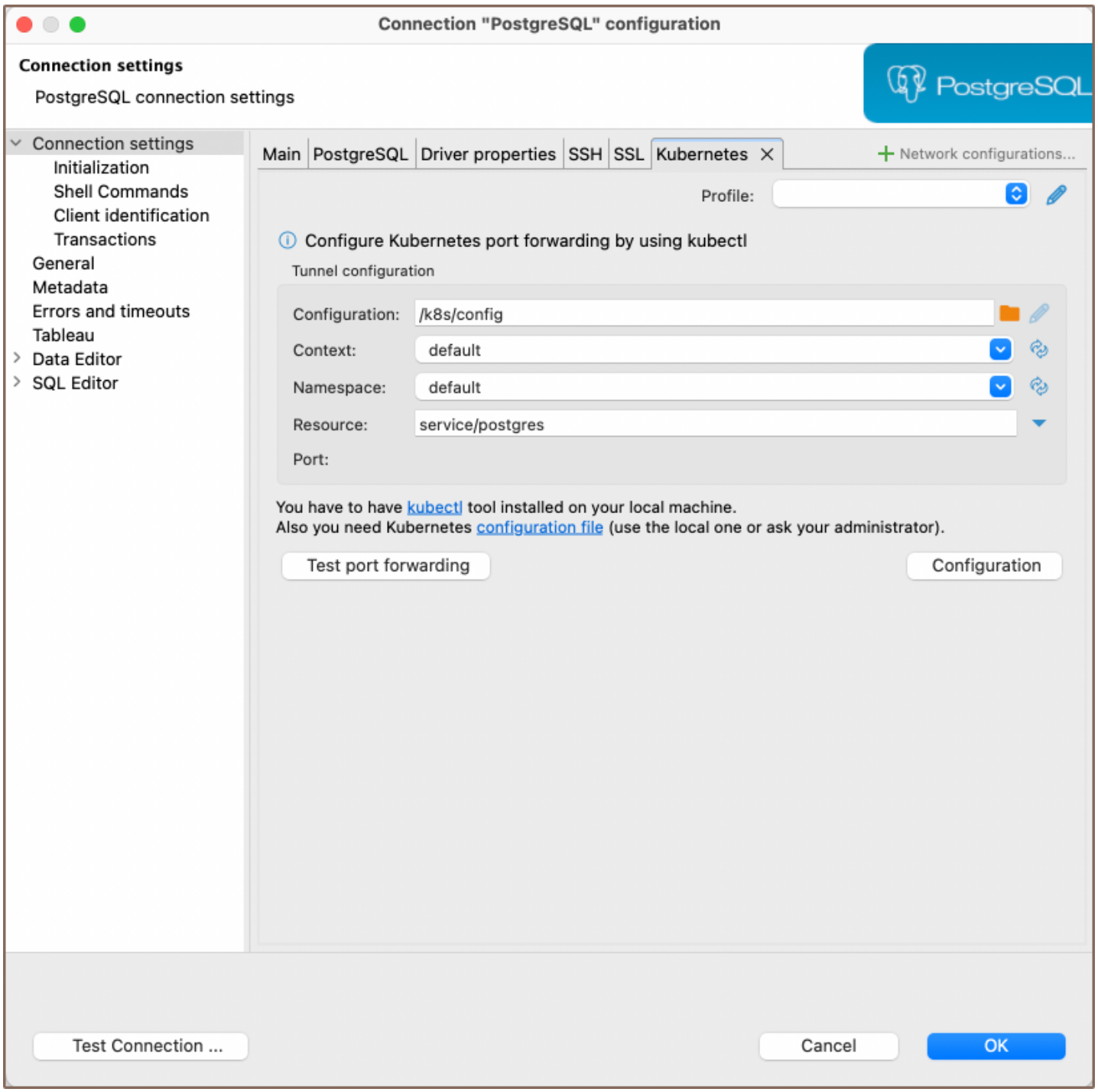
Setting up Kubernetes

To set up a tunnel to a Kubernetes cluster in DBeaver, you need to configure the following settings:

1) First, you need to add a Network Handler for Kubernetes. To do this, click on the **Network configurations...** button and select **Kubernetes**.



2) After making your selection, a new tab will appear in the menu at the top, where you can proceed to configure the various settings.



Here you need to fill out the following fields:

Field	Description
Configuration	Specify the path to your Kubernetes configuration file.
Context	Manually select a **kube-context** if your configuration offers multiple contexts for different clusters. The default setting is `default`.
Namespace	The default value is `default`, but this may differ in managed Kubernetes environments. Obtain the correct namespace from your `kubectl` config.

Resource	By clicking the **Find Resource** button next to the field, a list of available services to connect to will open. From this list, you can select the service you wish to connect to.
Port	Configure Kubernetes port forwarding by using `kubectl`.

Tip: When clicking the dropdown menus in **Context** and **Namespace**, you can select settings directly from your `kubectl` config.

Kubectl settings

The Kubernetes Command-Line Tool, `kubectl`, is essential for interacting with Kubernetes clusters and is integral to the functioning of DBeaver's Kubernetes features. Learn more in the official Kubernetes documentation.

To configure `kubectl` settings, you have two options:

1) Click on **Configuration** button on the Kubernetes connection settings page to access the `kubectl` settings.

2) Alternatively, navigate to **Window** -> **Preferences** -> **Connections** -> **Command Line Tools** -> **Kubernetes** to specify the path to your `kubectl` executable.

Testing port forwarding

The **Test port forwarding** button utilizes the `kubectl port-forward` feature to enable secure access to applications running inside Pods from your local machine. This provides a secure way to interact with the application without exposing it to the outside world. Learn more in the official Kubernetes documentation.

To test port forwarding:

1) Fill out the necessary Kubernetes settings as described in the previous section.

2) Click on the **Test port forwarding** button to initiate the port forwarding process.

If the test is successful, this confirms that you can securely access the targeted Kubernetes resource from your local machine using the configured port settings.

Kerberos authentication

Note: This feature is available in Lite, Enterprise, Ultimate and Team editions only.

Databases that support this feature:

- Cockroach
- EnterpiseDB (EDB)
- Fujitsu
- Greenplum
- HANA
- MySQL
- MariaDB
- Oracle
- PostgreSQL
- PrestoDB
- PrestoSQL (Kerberos can be used with SSL from JKS)
- Redshift
- Teradata
- TimescaleDB
- Yellowbrick
- YugabyteDB

There are a lot of ways to use Kerberos in your database authentication. On this page, you will learn a few of them. This page describes only the part that you need to do on the client machine (the one on which DBeaver is installed), and it is implied that Kerberos KDC and the necessary configuration are already done on the server.

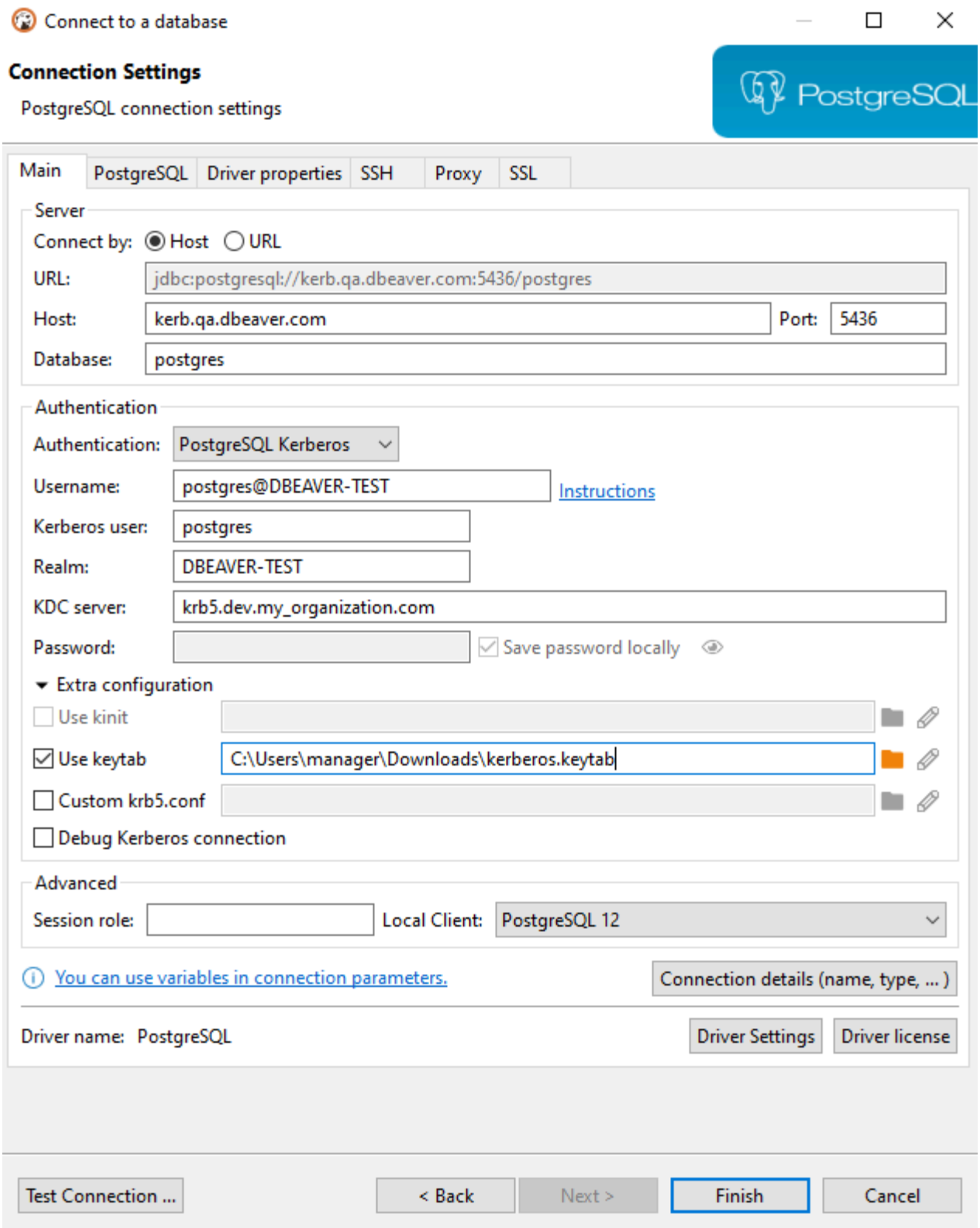
Possible settings

Setting	Description
Authentication	
Username	Name of the user/role in the database
Kerberos User	A Kerberos user represents a unique identity in a Kerberos system to which Kerberos can assign tickets to access the Kerberos-aware services.
Realm	A Kerberos realm is the domain over which a Kerberos authentication server has the authority to authenticate a user, host, or service. A realm name is often, but not always the upper case version of the name of the DNS domain over which it presides.
KDC Server	The hostname of your KDC server. The Kerberos Key Distribution Center (KDC) is a network service that supplies session tickets and temporary session keys to users and computers within an Active Directory domain. The KDC runs on each domain controller as part of the Active Directory Domain Services (AD DS). So the KDC hostname is the hostname of your DC.
Password	The password of your Kerberos user.
Extra configuration	
Use keytab	Select this checkbox if you want to use a keytab file on your machine instead of entering a password.
Use kinit	Select this checkbox if you use the kinit tool on your machine. kinit is a tool that obtains and caches an initial ticket-granting ticket for the user. If this checkbox is selected, you will only need to fill in your Kerberos username in most cases.
Custom krb5.conf	Path to your local Kerberos configuration file.
Debug Kerberos Connection	Select this checkbox if you want to see full Kerberos connection information in your log files.

Important Note Some settings can be saved in a session cache. If you have changed the settings and it does not help to connect - restart DBeaver.

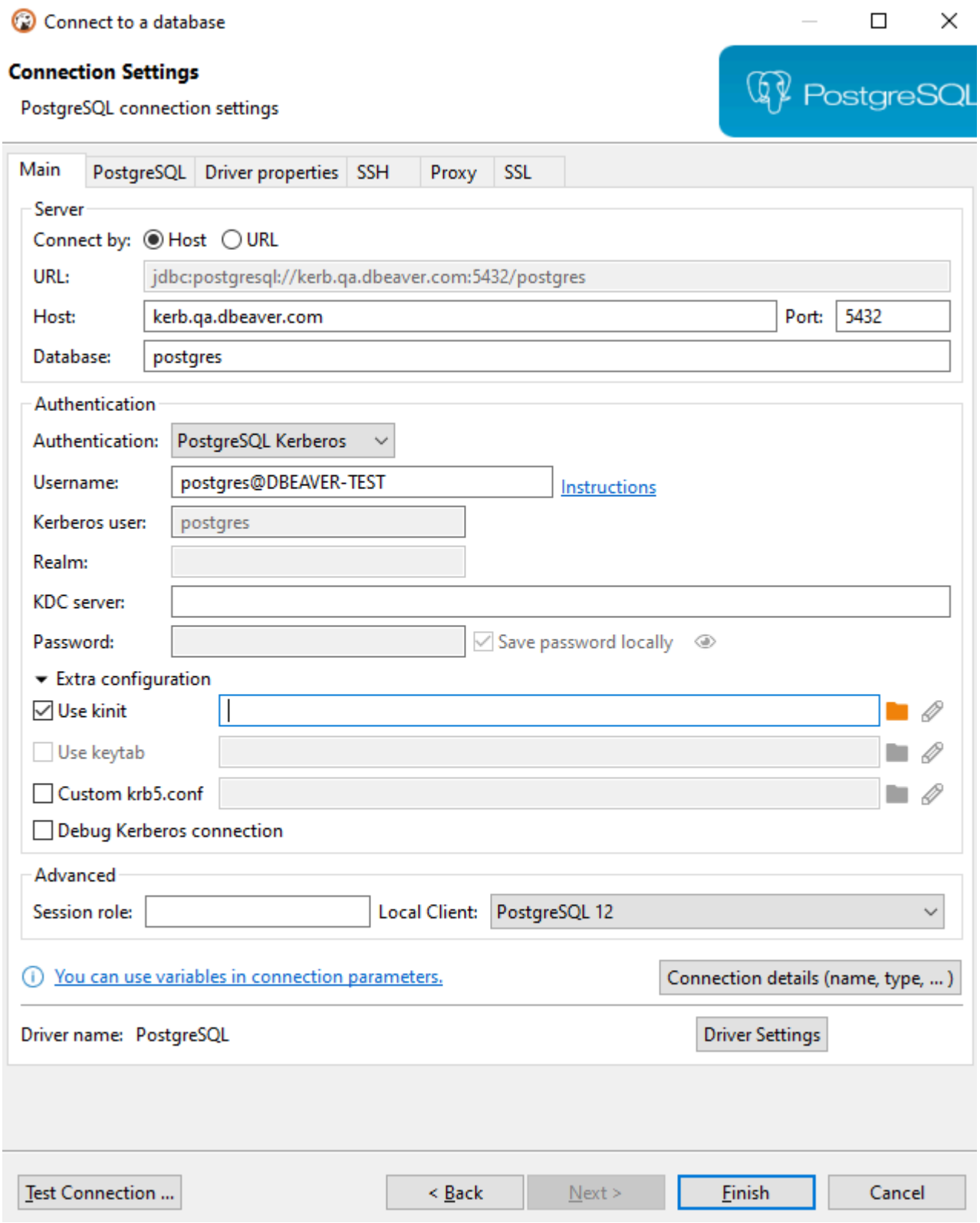
Using the keytab file

A keytab is a file containing pairs of Kerberos principals and encrypted keys (which are derived from the Kerberos password). You can use a keytab file to authenticate various remote systems using Kerberos without entering a password. However, when you change your Kerberos password, you will need to recreate all your keytabs. Keytab files are commonly used to allow scripts to authenticate automatically using Kerberos without requiring human interaction or access to a password stored in a plain-text file. The script can then use the acquired credentials to access files stored on a remote system. For DBeaver, it means that when you use a keytab file, you still need to provide all of the credentials other than a password. You can find this setting in the "Extra configuration" section.



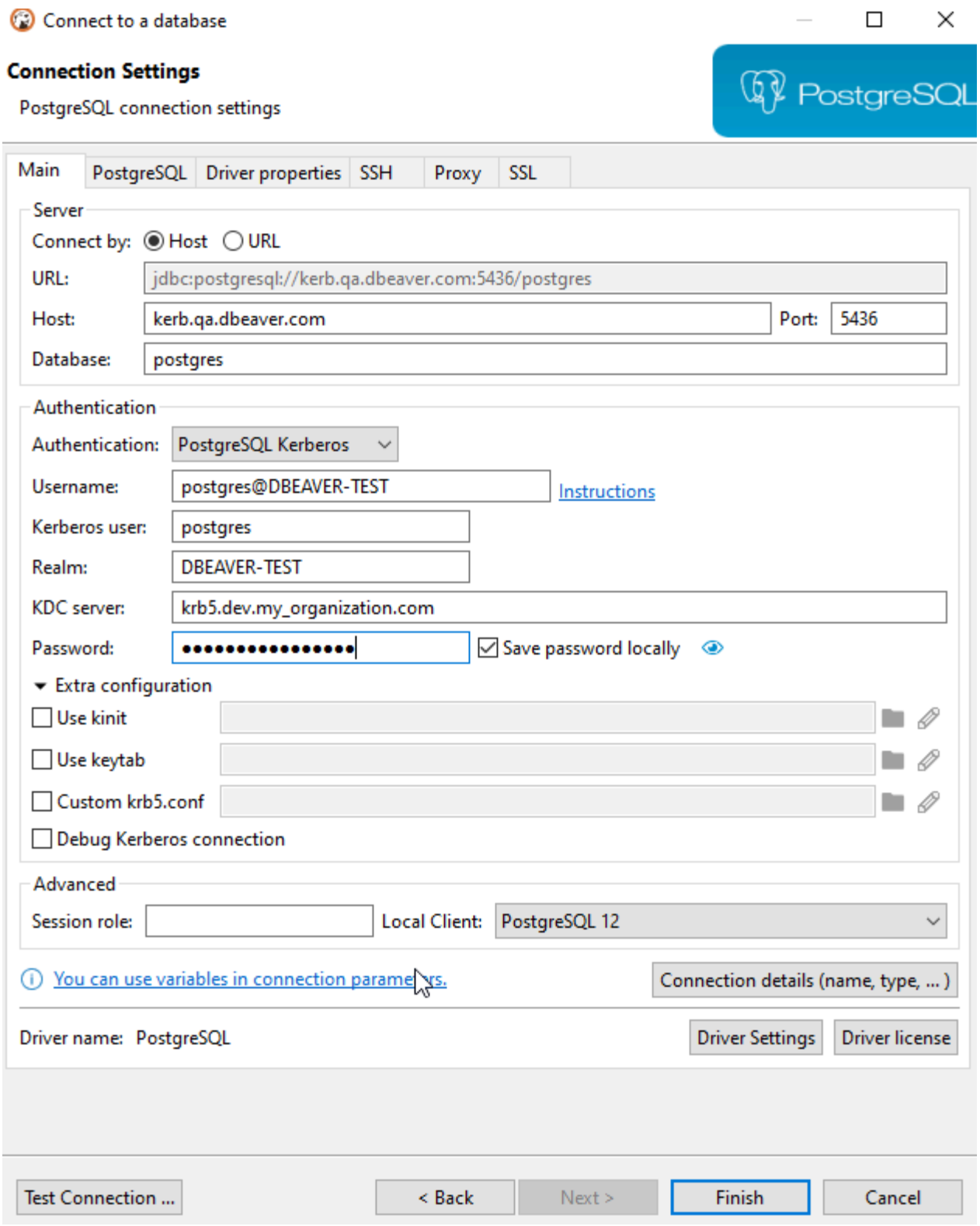
Using kinit

kinit is a command-line/terminal tool that obtains and caches an initial ticket-granting ticket for the Kerberos user. All of the credentials are provided either in a configuration file or in a command-line interface. This means that to authenticate with kinit in DBeaver, you only need to provide a kerberos username and select this checkbox to use kinit. You can find this setting in the "Extra configuration" section.



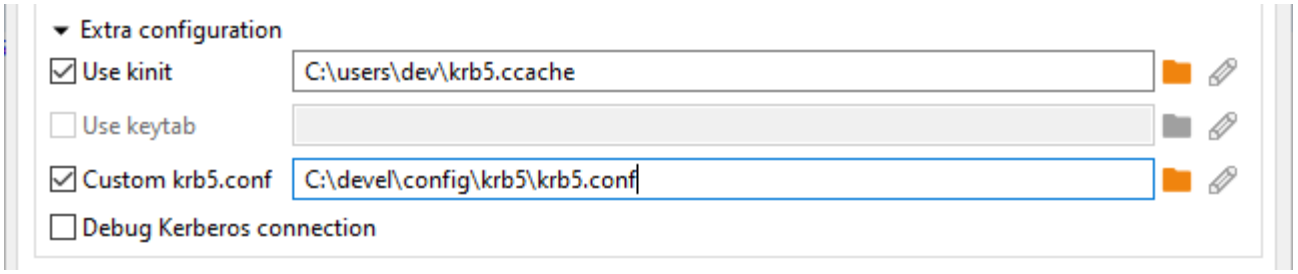
Using a password

This method is almost the same as using a keytab file, but instead of providing an encrypted file, you must manually enter a password.

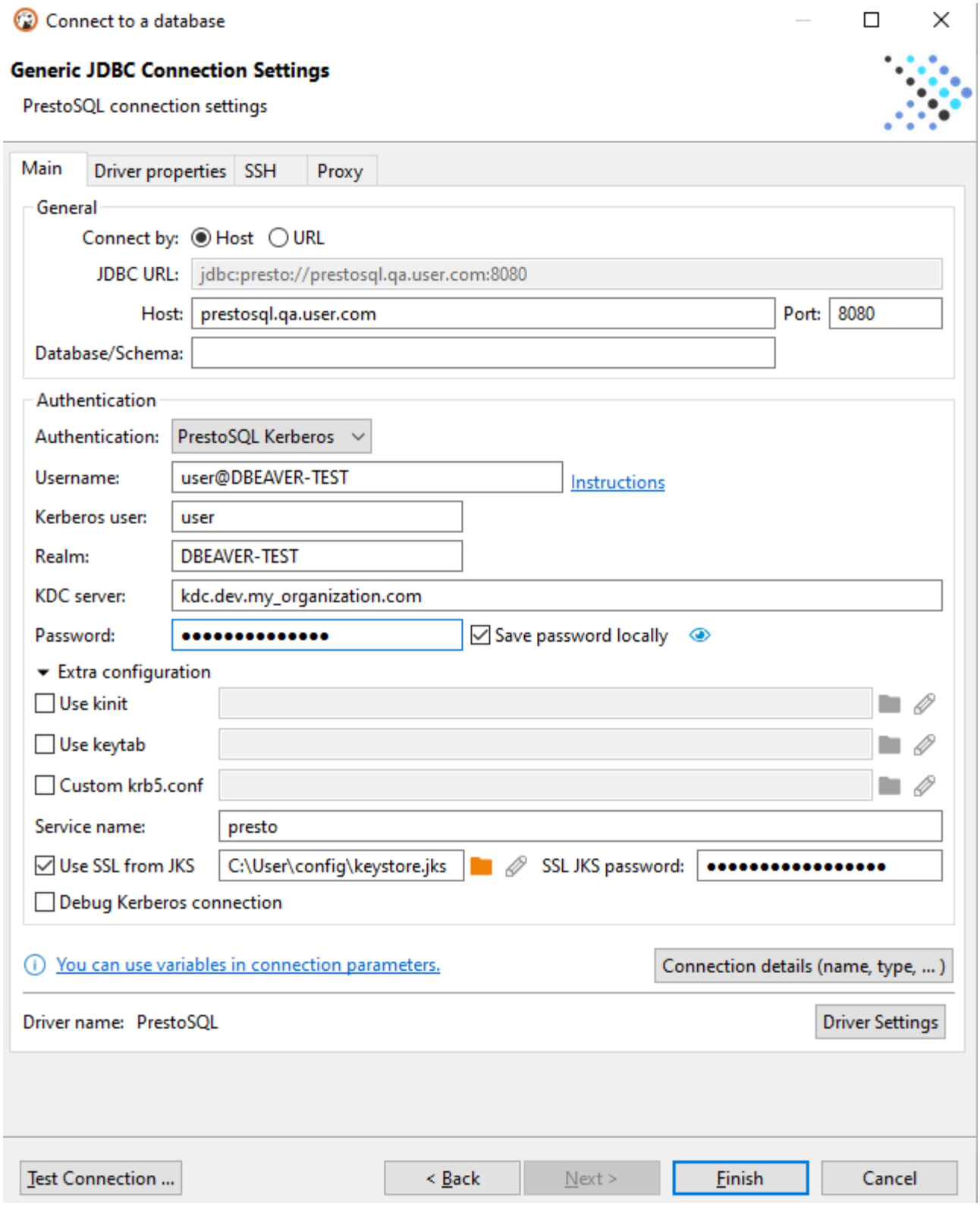


Extra options

Sometimes you may need to specify the path to the Kerberos configuration file. You can do it in the "Extra configuration" settings section. You can also specify the path to Kerberos credential cache file near the "Use kinit" checkbox.



If your database Kerberos authentication requires the remote coordinator Kerberos service name, add it to the "Service" field. If SSL on the server is not trusted - you can check the "Use SSL from JKS" setting and manually add a path to your .jks file. A Java keystore (JKS) file is a secure file format used to hold certificate information for Java applications. SSL JKS password can be added in the "SSL JKS" field. For now, these settings are only available for the PrestoSQL connection.



Oracle

Oracle JDBC driver 21 has broken Kerberos authentication, at least for most of the old configurations.

Use an older driver (12.x or 19.x) in order to use Kerberos authentication in Oracle.

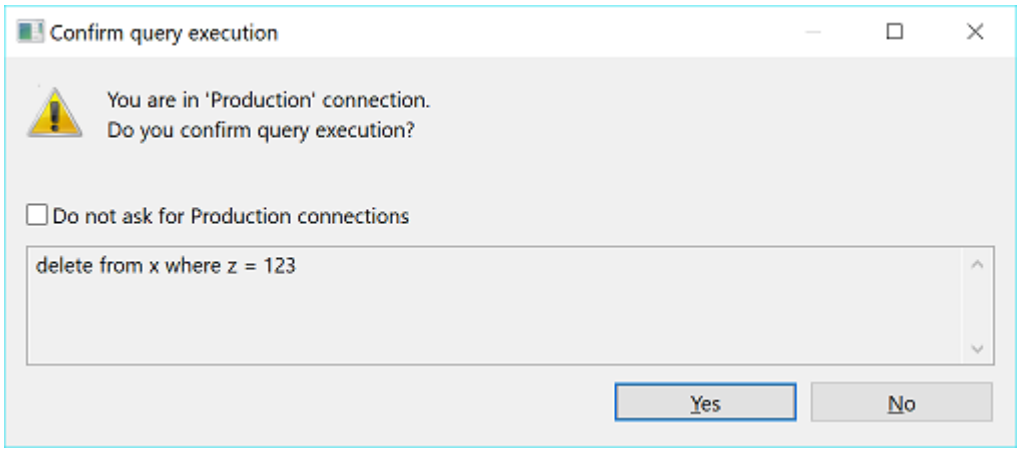
Connection Types

Connection types define how DBeaver behaves regarding:

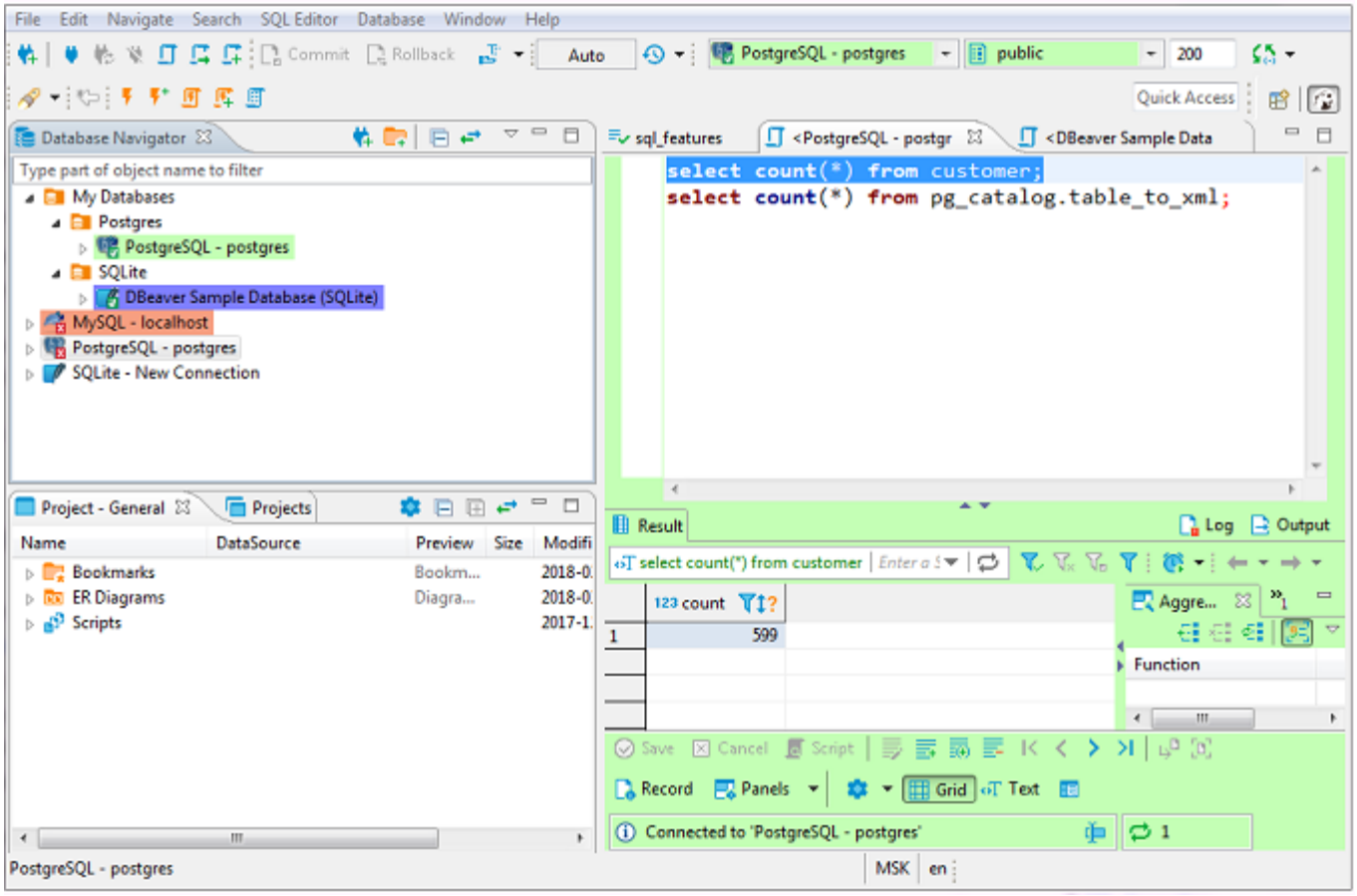
- Default transactions commit mode - with or without automatic commit of changes to the database.

  NOTE: You can override the default commit behavior during your work with connections by changing the commit mode, see Auto and Manual Commit Modes.

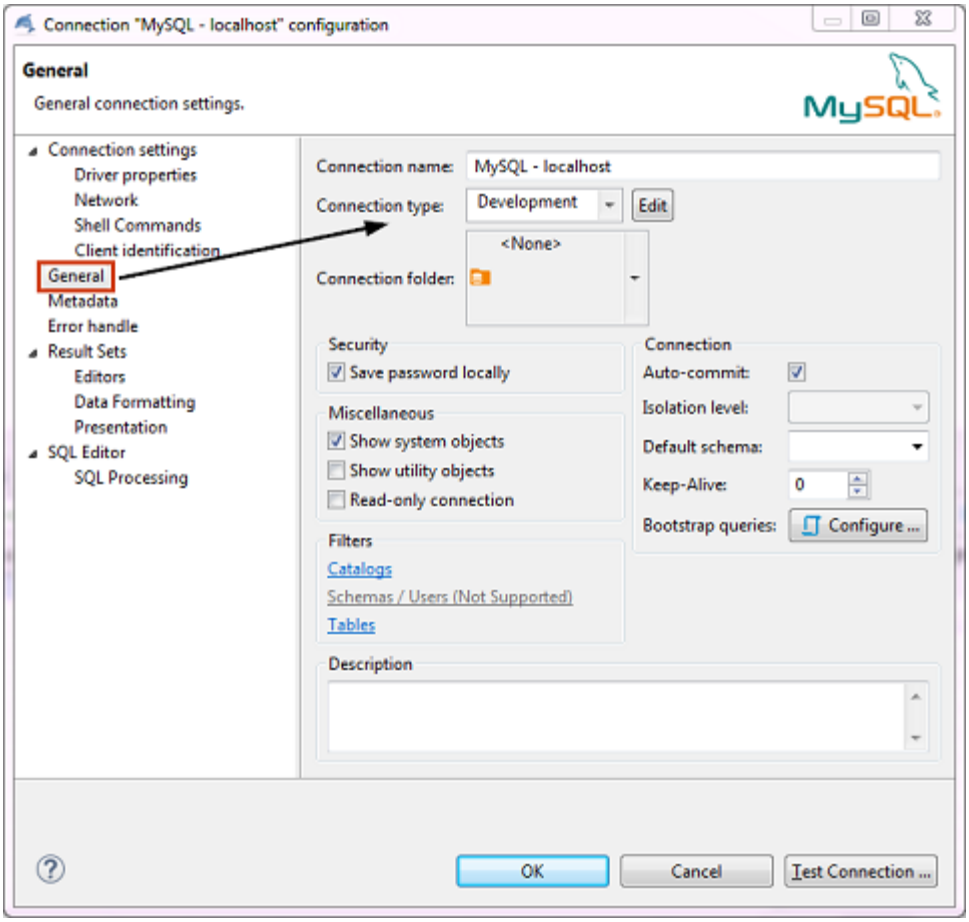
- SQL statements execution (with or without user confirmation): If set to the required user confirmation for SQL execution, DBeaver shows a confirmation message every time you attempt to execute a 'transaction' type of query (INSERT/DELETE/UPDATE, etc.):



For your convenience, DBeaver supports color-coding of connection types so that you know at once which behavior to expect when you use a certain connection. The screenshot below visualizes how the color coordinated database connections are used in Database Navigator and Projects views as well as editors related to these connections:



To manage connection types for a database connection, in the Database Navigator or Projects view, click the connection to set the focus on it and then press F4 to open the connection properties window. Then, in the properties window, in the navigation pane on the left, click **General** to see the general settings. You can see **Connection Type** field among the settings:



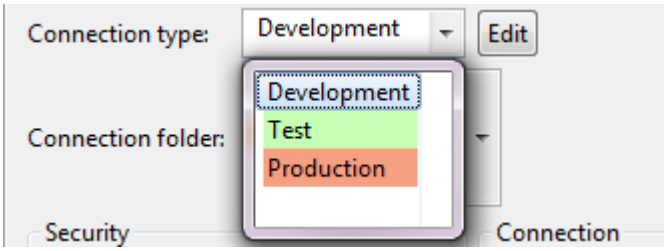
There are three default connection types – **Development**, **Test**, and **Production**. You can change the connection type for your database connection as well as you can create a new connection type, edit or delete an existing one.

Change Connection Type

By default, the **Development** connection is preset for all database connections. You can change the connection type to one of the default connection types or to a custom type, if there are any.

To change the connection type:

1. In the connection properties window, on the **General** page, click the **Connection type** field and then click the connection type in the dropdown list:

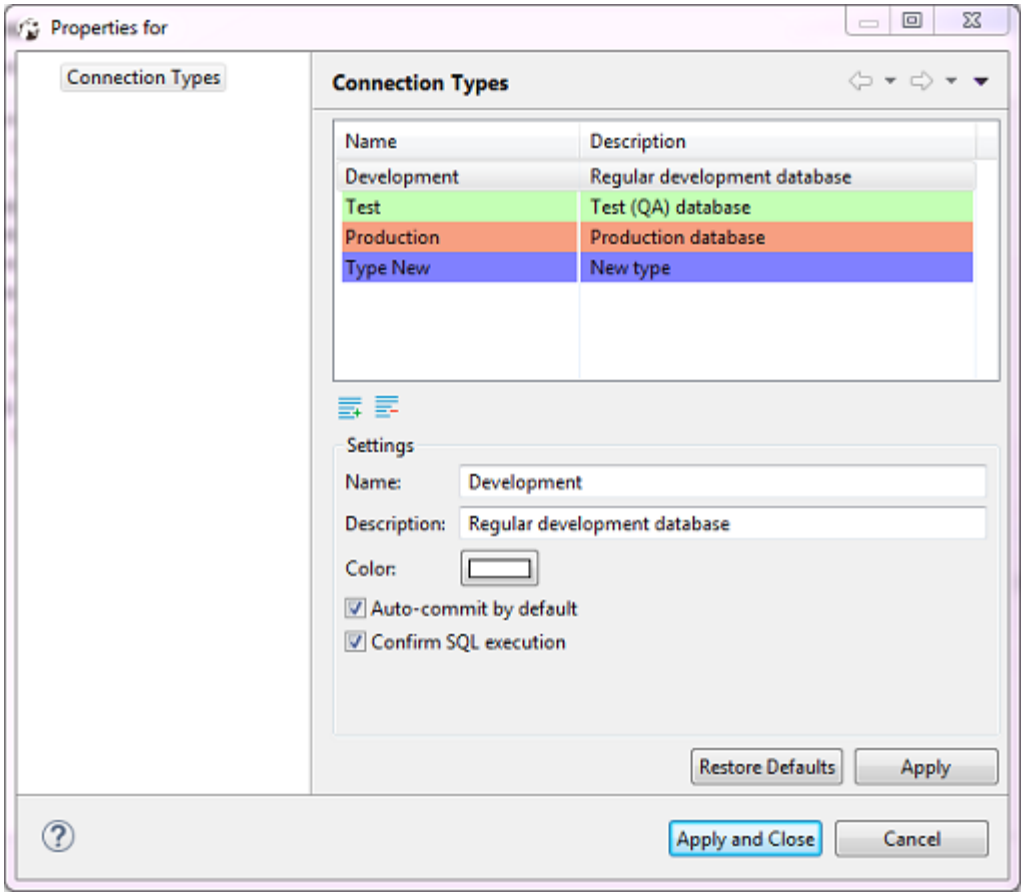


2. To test the connection, click **Test Connection**. To confirm the change, click **OK**.

Create Connection Type

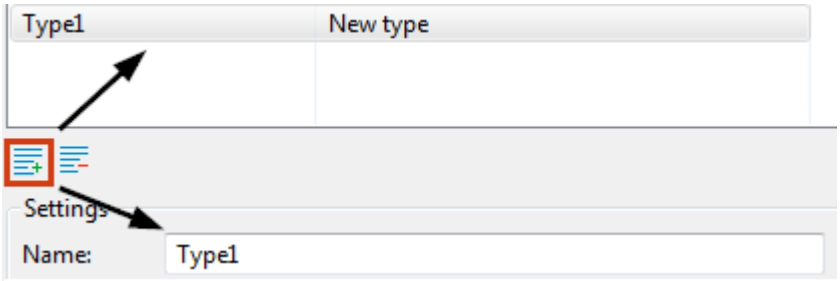
To create a connection type:

1. In the connection properties window (F4 on a connection), on the **General** page, click **Edit** next to the **Connection type** field. The Properties for connection types window opens:



   The window displays existing connection types and their settings.

2. Click the new connection type button. A new connection type appears in the list:



3. Now you can specify the settings for the new connection type:
   - Enter the connection type`s name into the **Name** field.

- Enter a description into the **Description** field, if needed.
- Click the **Color** box and select the color for the new connection type.
- To set DBeaver to automatically commit changes to the database when connections use this connection type, select the **Auto-commit by default** checkbox. Otherwise, leave it empty.
- To set DBeaver to ask for your confirmation at each execution of SQL statement of 'transaction' type, select the **Confirm SQL execution** checkbox. Otherwise, leave it empty.

4. Click **Apply** to apply the changes and keep the window open or click **Apply and Close** to apply the changes and close the window. To discard all changes and return to the previous state, click **Restore Defaults**.

Edit Connection Type

To edit a connection type:

1. In the connection properties window (F4 on a connection), on the **General** page, click **Edit** next to the **Connection type** field.
2. Specify the settings for the new connection type the same way as when you create a connection type, see 'Create Connection Types' section above.
3. When you finish editing the connection types, click **Apply** to apply the changes and keep the window open or click **Apply and Close** to apply the changes and close the window. To discard all changes and return to the previous state, click **Restore Defaults**.

Delete Connection Type

To delete a connection type:

1. In the connection properties window (F4 on a connection), on the **General** page, click **Edit** next to the **Connection type** field. The Properties for connection types window opens.
2. In the Properties window, in the list of connection types, click the connection type to set the focus on it and then click the delete button under the list: 

3. Click **Yes** in the confirmation dialog box to confirm the deletion. Otherwise, click **No**.

4. Click **Apply** to apply the changes and keep the window open or click **Apply and Close** to apply the changes and close the window.

Tableau integration

Note: This feature is available in Enterprise, Ultimate and Team editions only.

Overview

Tableau is a leading data visualization and analytics platform that empowers you to connect, visualize, and share data in a way that makes it easier to understand. With its intuitive drag-and-drop interface, Tableau enables users to create interactive and shareable dashboards, providing insights into complex data sets.

Having a registered Tableau account is a prerequisite for those looking to leverage Tableau's capabilities. This article explores how DBeaver integrates with Tableau, enhancing the data manipulation process.

- Powering Tableau with DBeaver
  - Easy connection to existing databases
  - Working with custom SQL queries
- Filling profile for Tableau connection
  - Adding a Tableau Profile in DBeaver
- Creating a database connection
  - Associate DBeaver and Tableau connection
- Editing Custom SQL Queries

Powering Tableau with DBeaver

Easy connection to existing databases

DBeaver's integration with Tableau allows you to connect effortlessly to databases already present in Tableau. This seamless connection enables you to access and manage data without complex configurations or additional tools. Learn more about creating a connection.

Working with custom SQL queries

A standout feature of DBeaver's integration with Tableau is the ability to edit custom SQL queries stored in Tableau. You can modify these queries directly within DBeaver, providing a streamlined workflow for advanced data analysis. Learn more about editing custom SQL queries.

Filling profile for Tableau connection

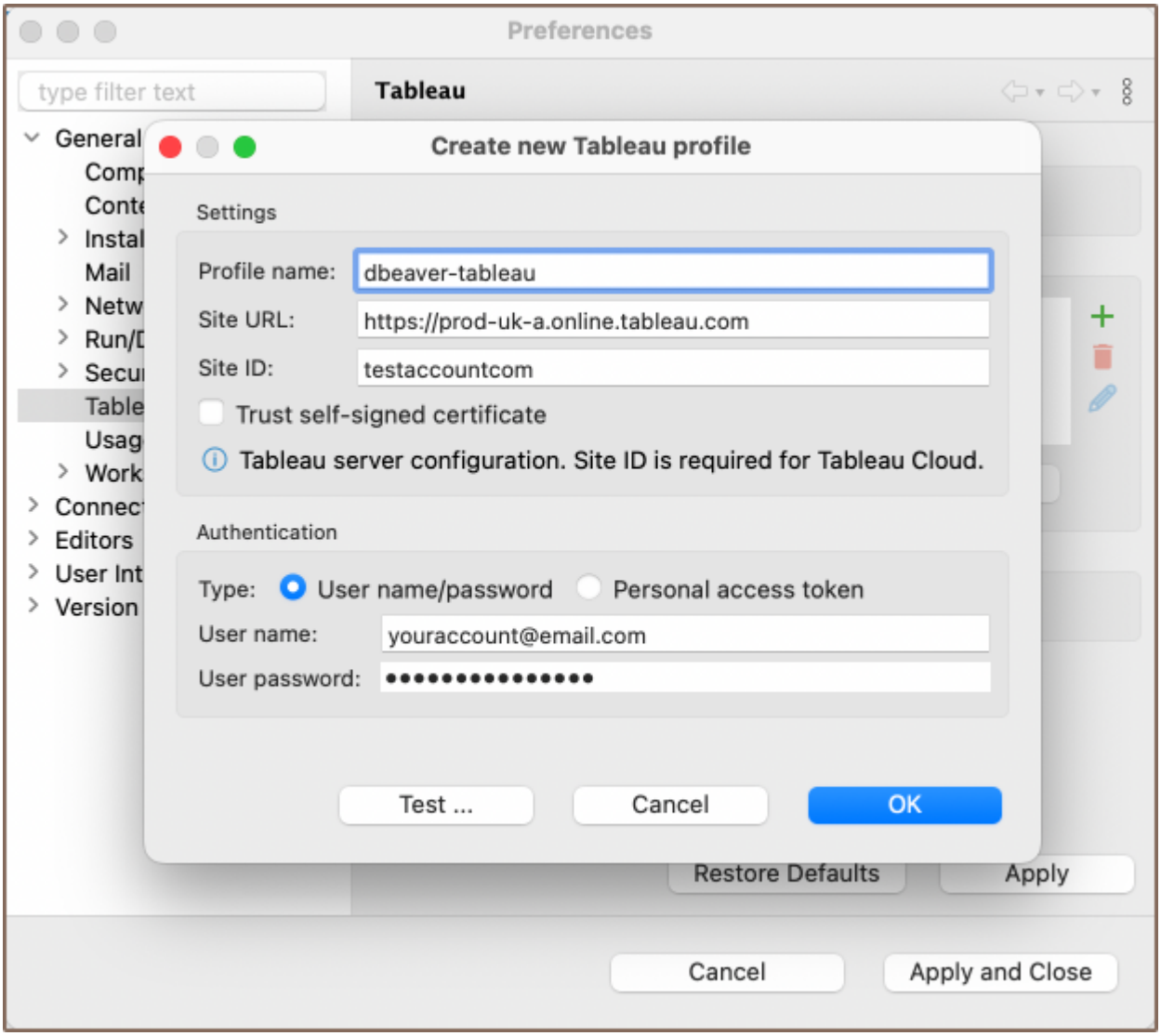
To configure Tableau credentials in DBeaver, follow these steps:

1. Navigate to **Window** -> **Preferences** -> **General** -> **Tableau**.
2. In the Tableau settings, you will find the following options:

Option	Description
Enable Tableau Integration	Enables Tableau in connections and in UI.
Tableau Profile	Manage Tableau profiles. User profiles are displayed here, and you can work with them using the buttons: + - Add profile - Remove profile - Edit profile
Test Connection	Test the connection between DBeaver and Tableau.
Automatically associate Tableau connections	Automatically associate Tableau connections with DBeaver data sources. Learn more about associate connection.

Adding a Tableau Profile in DBeaver

When you click on **Add Profile**, a **Create new Tableau profile** window will appear. This is how you fill out the profile:



Field	Description
Profile name	The name of the profile in DBeaver (choose a convenient name for yourself).
Site URL	The URL should look like this: `https://prod-uk-a.online.tableau.com`.
SITE ID (required for Tableau Cloud)	Enter the correct `SITE ID` used during Tableau registration (e.g., copy `testaccountcom` from your Tableau home page URL like `https://prod-uk-a.online.tableau.com/#/site/testaccountcom/home`).
Trust self-signed certificate	Allows not to validate the server certificate. This option is useful if you are using a self-signed certificate and need it to pass validation. For more information on managing certificates, you can refer to our article about trust stores.
Authentication	Use the credentials of the account you use (login/password or token name/value).

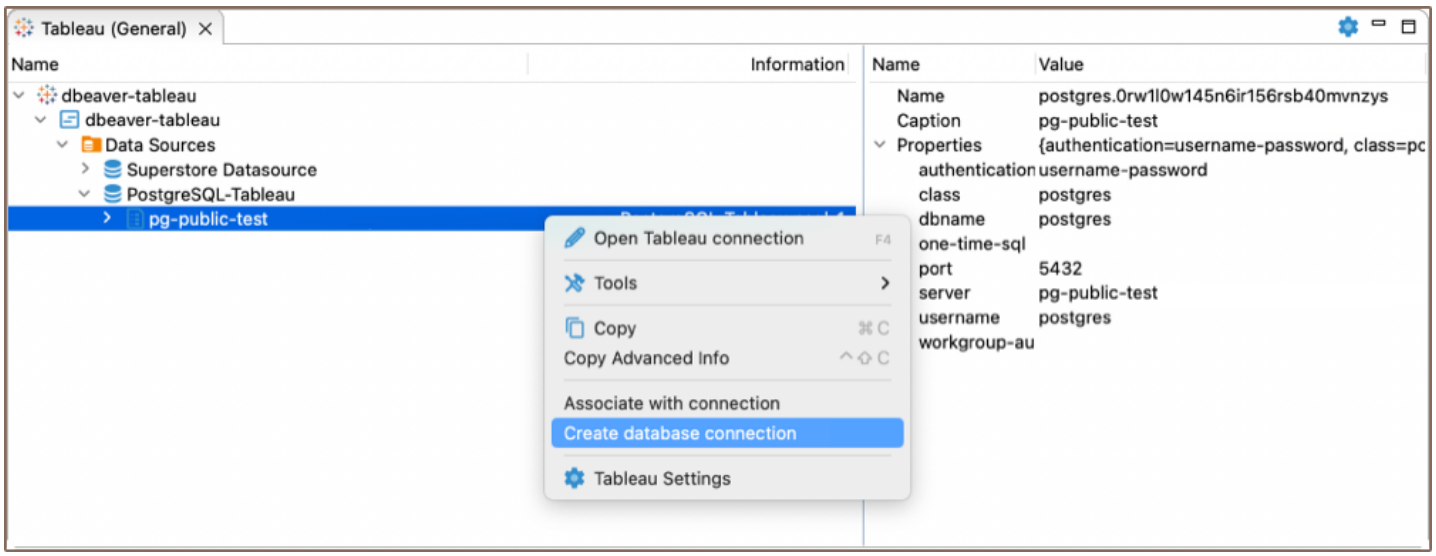
After filling out the profile details, you can test the connection to Tableau to ensure that everything is set up correctly. This step verifies that the provided information is accurate and that DBeaver can successfully communicate with Tableau.

Creating a database connection

Important: Before creating a connection in DBeaver, a connection must already be configured in Tableau. More details can be found in the official Tableau documentation.

Creating a connection between DBeaver and Tableau is a straightforward process:

1. In the DBeaver toolbar, click on the **Show View (Tableau)** button . The Tableau view will open in the lower part of DBeaver, displaying your profiles.
2. In your profile, select the desired database, right-click, and choose **Create Database Connection**.
3. DBeaver will prompt you to name the connection, and it will appear in the Database Navigator.

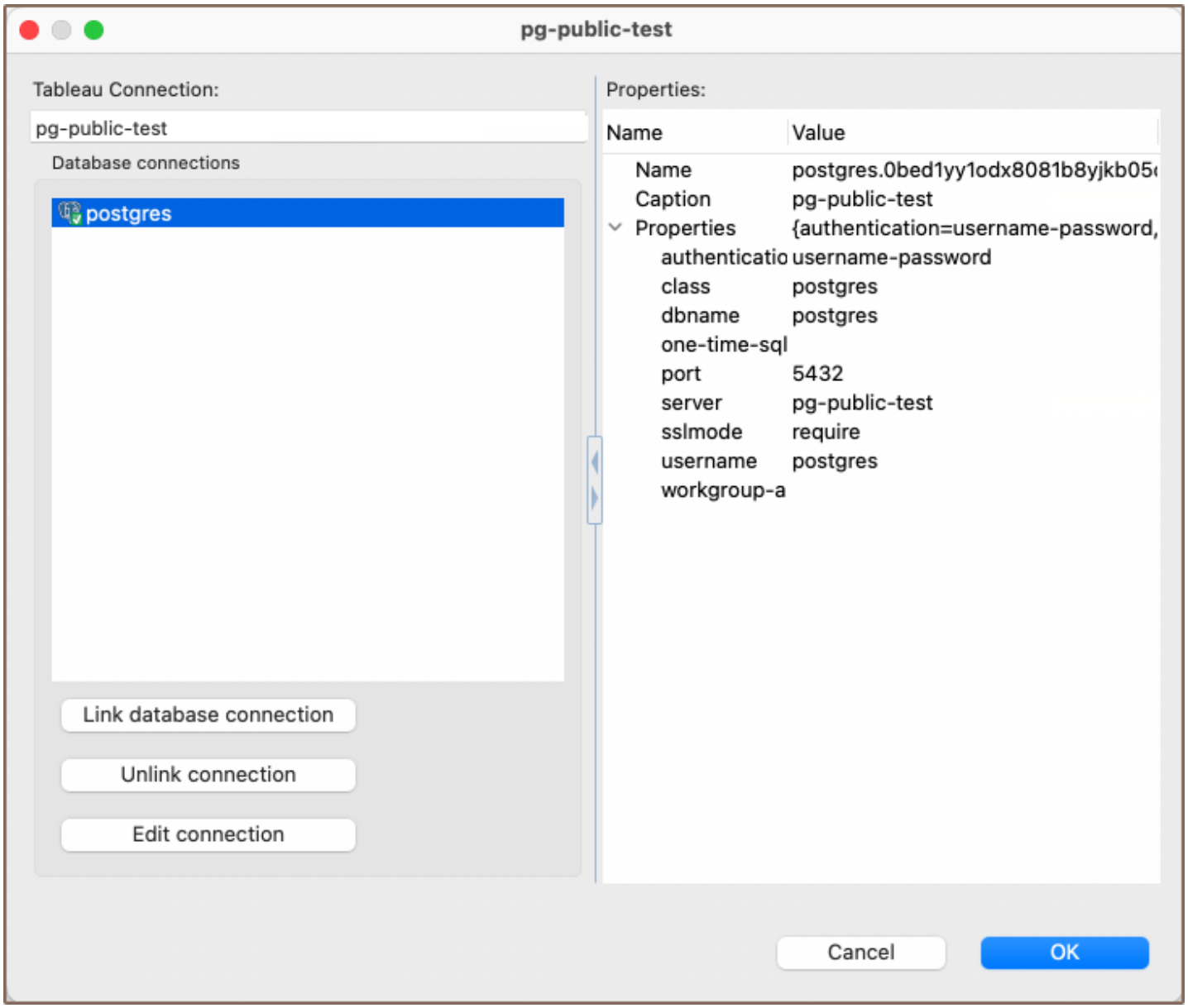


Associate DBeaver and Tableau connection

The **Associate with connection** button allows you to view a list of databases that can be associated with Tableau in DBeaver. It can automatically find the database based on the hosts.

Tip: If the **Automatically associate Tableau connections** option is selected in the preferences, clicking on a table will automatically open it in DBeaver.

To configure the database that is associated with Tableau, you can use the **Open Tableau connections** option or double-click the database in Tableau view. A window with connections that are associated with Tableau will appear. From this window, you can choose to **Link database connection**, **Unlink connection**, or **Edit connection** as needed.



Editing Custom SQL Queries

Working with custom SQL queries in Tableau can be challenging, especially when it comes to complex data manipulation. DBeaver's integration with Tableau offers a solution, allowing you to edit and manage custom SQL

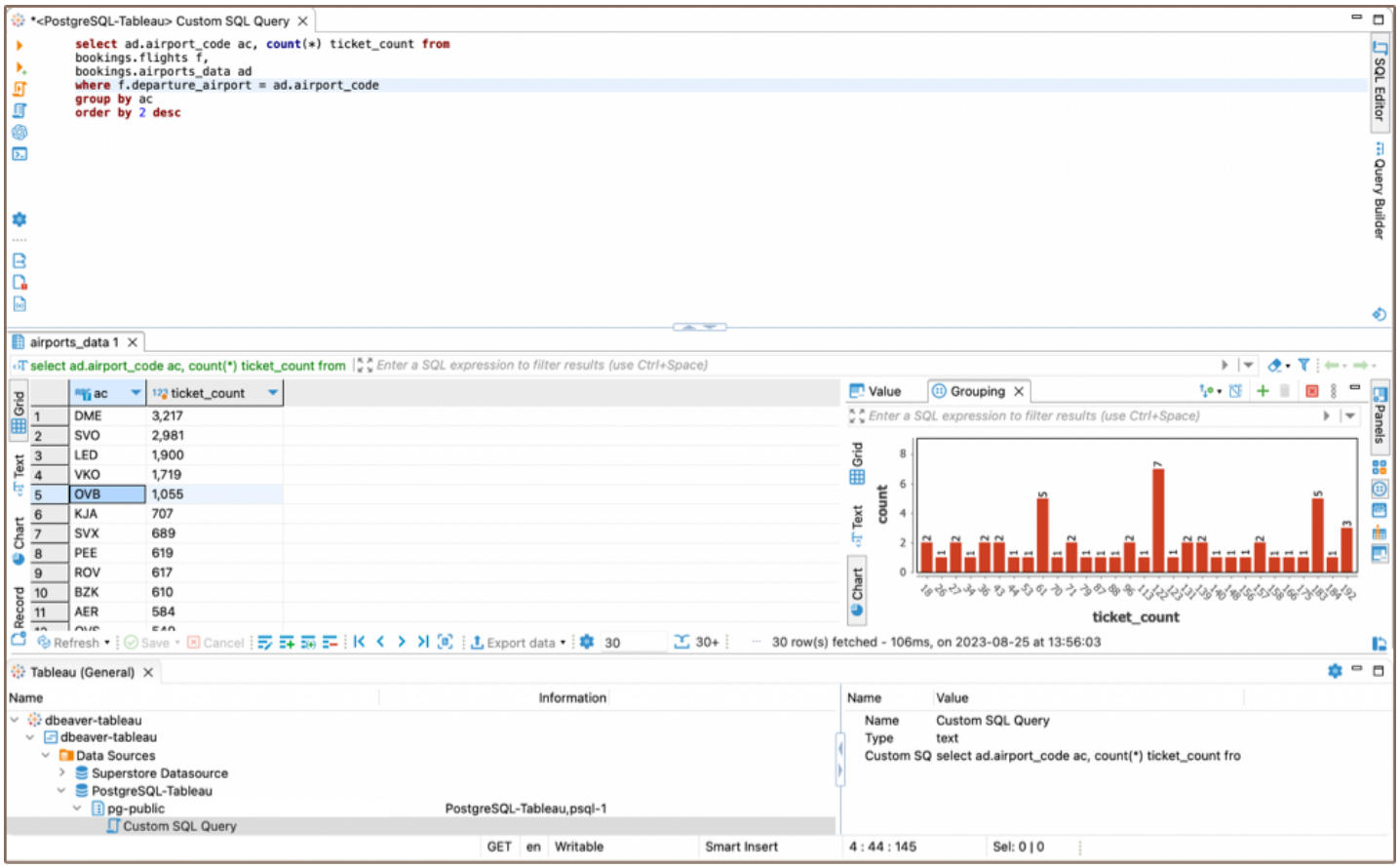
queries with ease. By connecting Tableau to DBeaver, you can leverage DBeaver's robust SQL editing tools to enhance your data analysis workflow in Tableau.

Important: The custom SQL query must already be configured in Tableau. More details can be found in the official Tableau documentation.

This is how you can edit custom SQL queries using DBeaver:

1. In the Tableau view in DBeaver, select the Custom SQL query in your profile. You can double-click it, right-click and choose **Open Custom SQL**, or press the shortcut F4 .
2. The SQL Editor will open, where you can work with the query just like any other query in DBeaver. You can make necessary changes, utilize grouping, filters, and other functionalities, and save and execute the query to see the changes reflected in Tableau.

Note: It is essential to save the query in DBeaver to ensure that the changes are also reflected and saved in Tableau.



This integration simplifies the process of working with custom SQL queries in Tableau, making it more efficient and user-friendly.

Auto and Manual Commit Modes

DBeaver supports two modes for committing changes to the database:

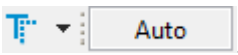
- **Auto-commit** transfers all changes that you make immediately to the database.
- **Manual commit** requires your confirmation before committing a change to the database or rolling it back.

Though available in many cases, the two modes are actionable only in SQL Editor. See the next sections for details of using the modes.

To switch between the modes, use the mode selection button that appears in one of the two views:  or  .

Auto-Commit Mode

Auto-commit mode is the default one for the Development and Test connection types, see Connection Types. Auto-commit mode is on if you can see the auto-commit view of the mode selection button () in the application toolbar. If you see the manual commit view (), then in order to switch to auto-commit mode, click the mode selection button – it changes to auto-commit. At the same time, this disables the two manual commit buttons in the toolbar: **Commit** and **Rollback** – these are available only in manual commit mode.

The statistics field next to the mode selection button always shows **Auto** in auto-commit mode: 

.

Clicking the statistics field opens the Transaction Log.

Manual Commit Mode

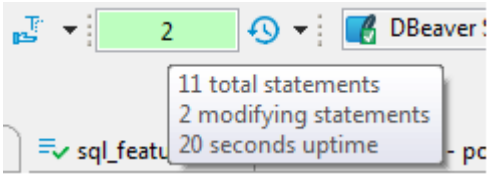
Manual commit is intended to protect your database from inadvertent changes and that is why it is the default mode for Production connection type, see Connection Types.

Manual commit mode is on if you can see the Manual commit view of the mode selection button ( ) in the application toolbar. If you see the auto-commit view ( ), then in order to switch to manual commit mode, click the auto-commit button – it changes to manual commit. At the same time, this enables the two manual commit buttons in the toolbar: **Commit** (Commit) and **Rollback** (Rollback).

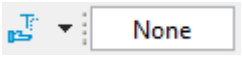
In manual commit mode, when you execute SQL statements (`Ctrl+Enter`), the number of database modifying statements that pend commitment to the database appears in the statistics field next to the mode selection button: 

.

If you hover your mouse over the field, you can see the statistics of your SQL statements:



To commit statements to the database, click the **Commit** button in the toolbar. To discard them, click **Rollback**.

If no modifying statements have been made, the statistics field shows **None**: 

Clicking the statistics field opens the Transaction Log.

Smart Commit Mode

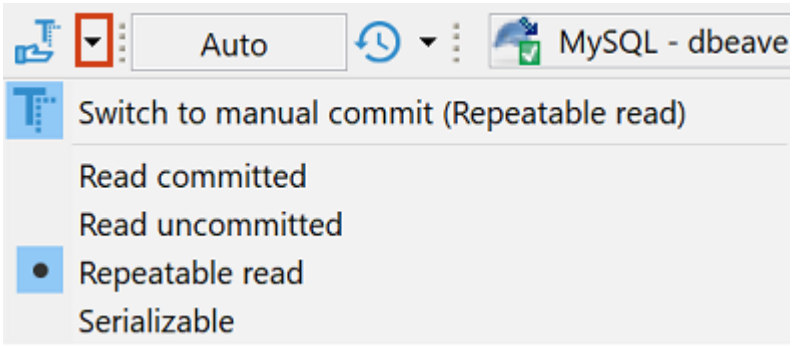
When smart commit is enabled and you are in auto-commit mode, which allows DBeaver to monitor your activity. When you try to execute any data modifying query (UPDATE, INSERT, DELETE, UPSERT, MERGE, etc) DBeaver will switch to manual commit mode before executing your query. Also if you edit any table data and save your changes - DBeaver will also switch to manual mode before the actual data modification.

If the option "Return to auto-commit on transaction end" is on, then DBeaver will switch back to auto-commit mode once you have executed Commit or Rollback command (using the main toolbar or the main menu actions).

Smart commit mode is very useful if you work mostly in the read-only mode. It does not lock tables when you perform SELECT queries. The transaction will only be started when you start to modify your data.

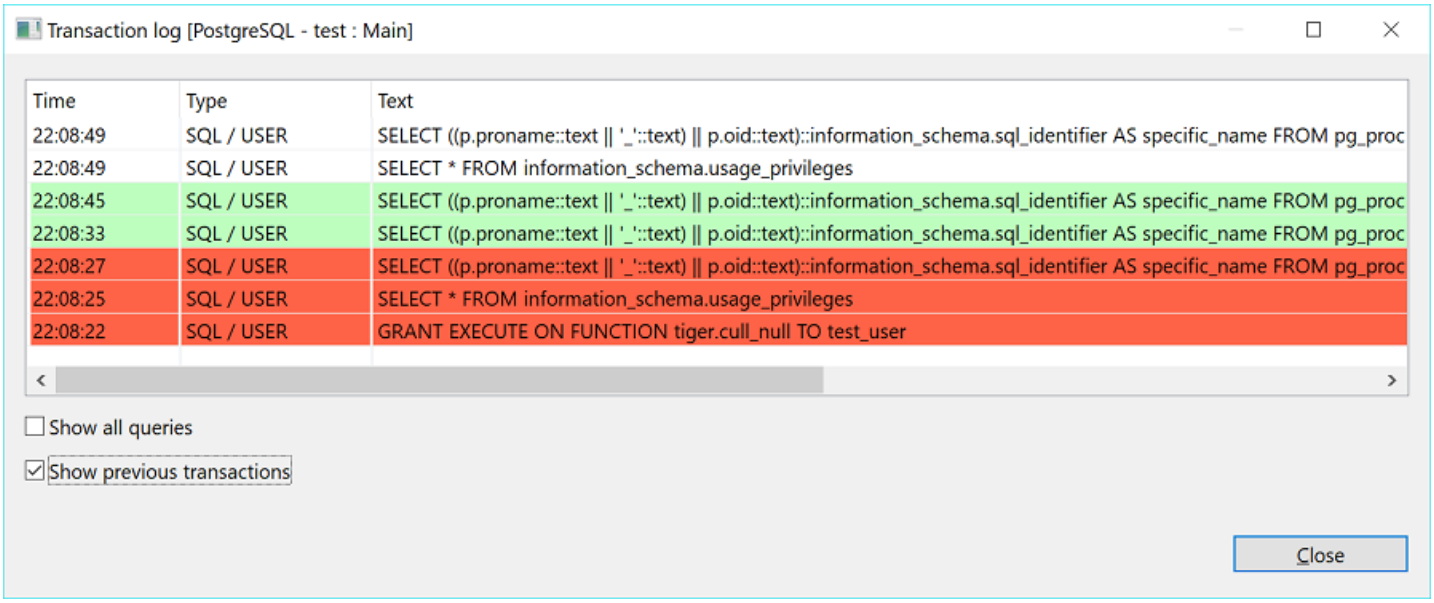
Transaction Isolation Level

For both, Auto and Manual commit modes, you can select the transaction isolation level. To do so, click the arrow next to the mode icon and then click the required option in the dropdown list:



Transaction Log

The Transaction Log shows all transactions (queries of 'transaction' type such as INSERT/DELETE/UPDATE and others) made during the current DBeaver session. To open the Transaction Log, click the **Transaction log** button ( ) in the toolbar or the statistics field to the left of it.



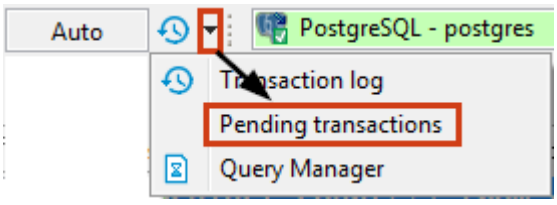
The Transaction Log window shows transactions that are:

- In progress or pending - shown without any special color
- Successfully committed – in green:
- Rolled back – in orange or red:

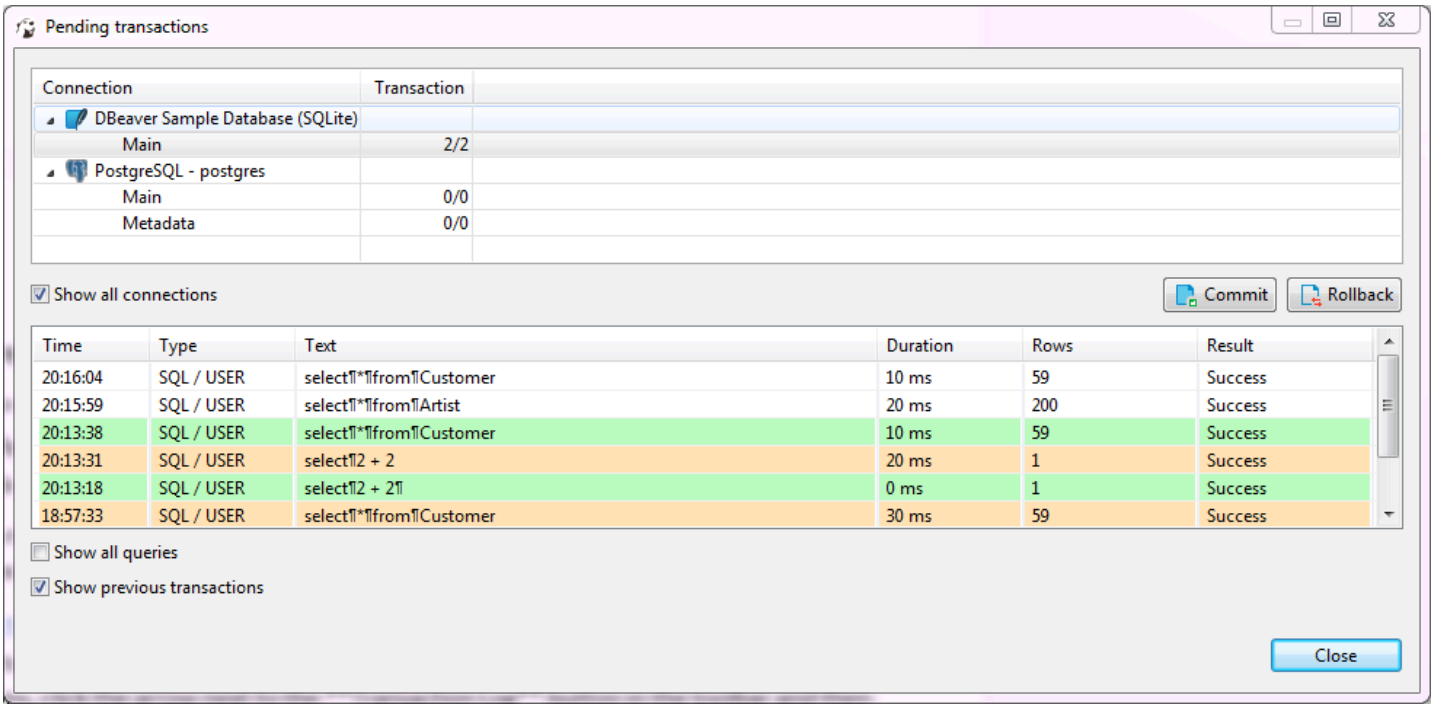
To see all previous transactions during the current session, select the **Show previous transactions** checkbox. To see all queries including non-transactional ones, select the **Show all queries** checkbox.

Pending transactions

It might be useful to check your pending transactions because they might lock your database. To see your pending transactions, click the arrow next to the **Transaction Log** button in the toolbar and then click **Pending Transactions** on the dropdown menu:



The upper table of the Pending transactions window shows currently active connections and the number of their transactions. The bottom table shows query details of the connection that is currently in focus in the upper table:



When the Pending transactions window opens, the upper table shows only those connections that have pending transactions. If no connections have pending transactions, the table is empty. To see all connections that are currently active (connected), select the **Show all connections** checkbox.

You can commit or roll back transactions right from the Pending transactions window: in the upper table, click the row with the required uncommitted transactions and then click the **Commit** or **Rollback** button, depending on your purpose. If a transaction is committed/rolled back successfully, both buttons are disabled. If the operation is unsuccessful, the system displays an error message.

To see all previous transactions made during the current session, select the **Show previous transactions** checkbox. To see all queries including non-transactional ones, select the **Show all queries** checkbox. The green rows are committed transactions, the orange (or red) ones are rolled back, rows without a special color are non-transactional or pending transactions.

Database drivers

You can use a pre-configured database driver or create a new driver.

DBeaver has a lot of pre-configured drivers including SQL, NoSQL, key-value databases, graph databases, search engines, etc. But sometimes you need to connect to a database which was not configured in DBeaver yet.

All you need is a JDBC driver of your database. The rest is easy.

Obtaining JDBC driver

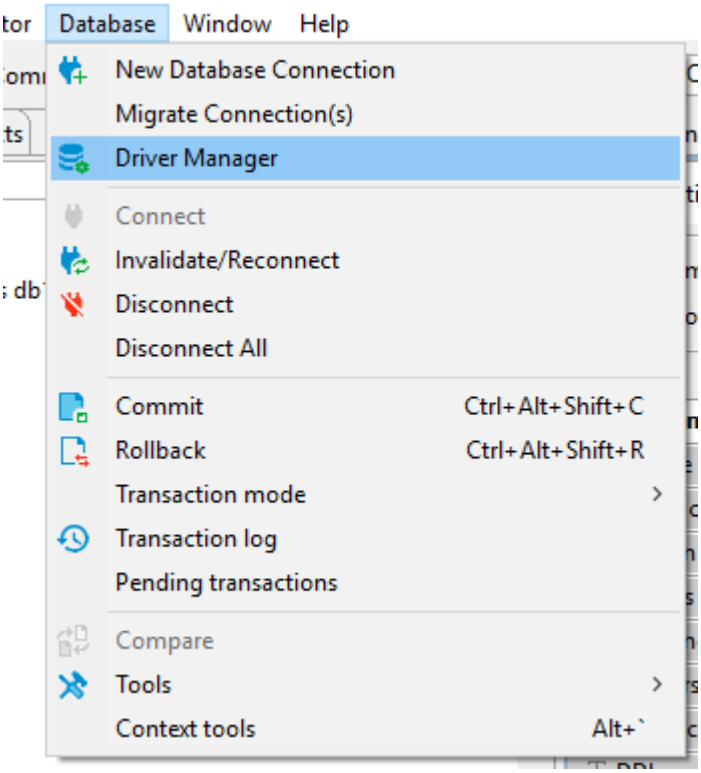
JDBC driver is a program (in Java) which can connect and operate with some local or remote database server. It usually provides all needed functionality to cover 100% of database functionality. The JDBC driver is usually provided by database vendors to allow customers to work with their databases.

The JDBC driver consists of one or multiple `jar` files. The Jar file is a library which contains program code and some other files. You need to download the driver's jar files before adding them to DBeaver. Sometimes the jar files are included in the database server distribution - in that case you need to refer to your database documentation or ask your DBA.

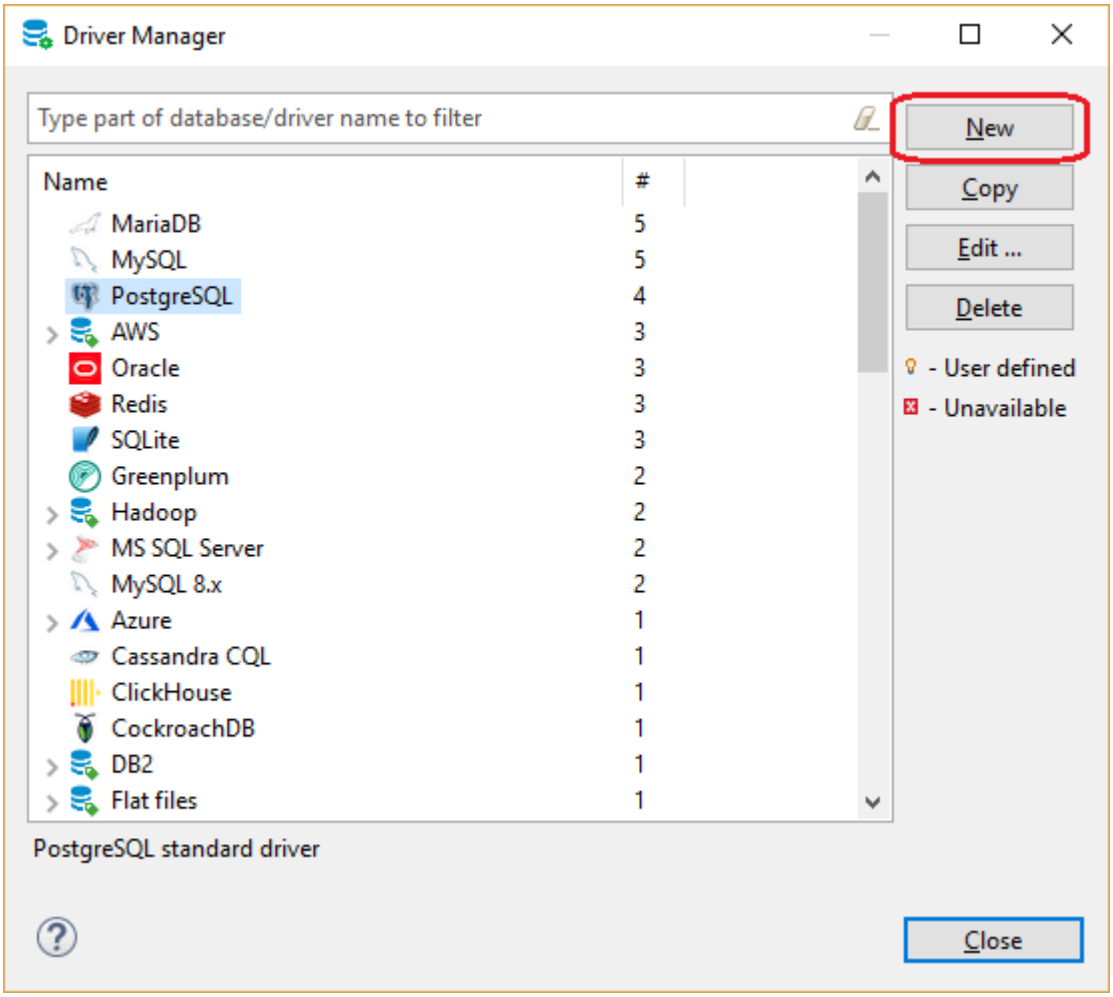
Adding driver configuration in DBeaver

Open driver manager dialog

You can open the driver manager from the main menu:



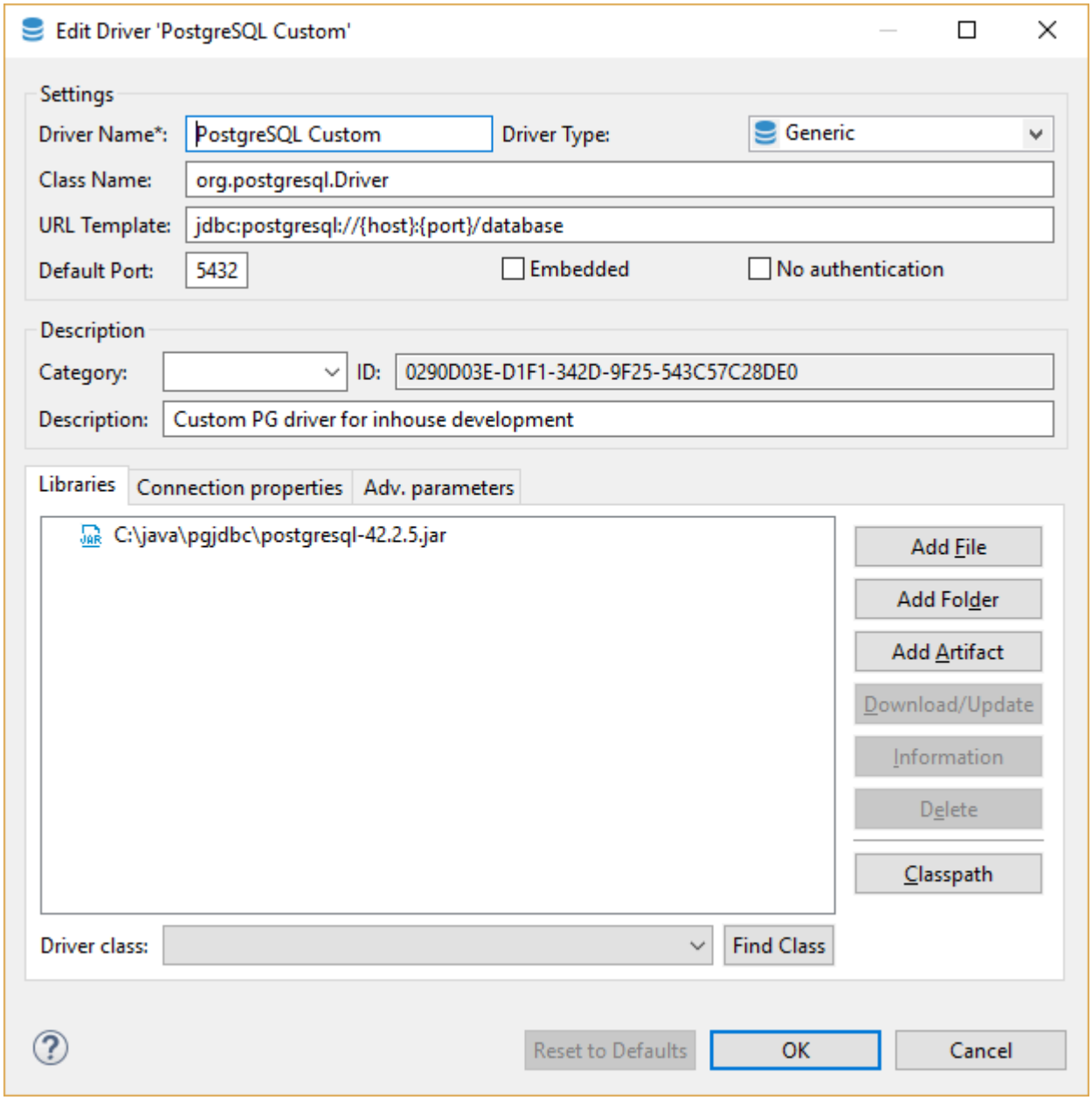
or from Database Navigator drop-down menu.



Add a new driver

Just click the button New and create a new driver. On the driver edit dialog you need to enter all required information:

Main parameters



Parameter	Description
Driver Name	Name of your driver. It can be any name you like
Driver Type	Driver provider. In 99% cases you will need a generic driver (JDBC provider)
	JDBC driver class name. You can get it from the documentation or find it in the jar files (see "Find Class"

Class Name	button description)
URL Template	Template of driver URL. You can leave it empty. But in this case you will be ready to set JDBC URL for each your connection. It is better to define a valid template, which will greatly simplify the connections creation. See "URL Templates" for a detailed description
Default Port	Default database port. You can get it from the documentation or leave it empty
Embedded	Enable it for server-less databases. This flag affects a few config options related to the network/connections management
No Authentication	This means that driver does not require authentication (no user/password fields will be shown)
Category	Driver category, deprecated
ID	Driver unique ID, ignore it
Description	Driver description, it is shown in some dialogs/wizards as a hint

Libraries

This is the list of jar files, binary libraries (dll or so) and any other files required by the driver. In most cases you only need the jar files.
Click "Add File" to add a single jar file, "Add Folder" to add to the folder with Java classes/resources and "Add Artifact" to add the Maven artifact (see below).

After you add the jar files you will be able to find all JDBC driver classes which are found in these jars. Just click on the "Find Class" button and DBeaver will show all of them. In most cases there is just one driver class in the driver. If there are many of them, you need to refer to the driver's documentation.

Maven artifacts

DBeaver can download driver jars directly from the Maven repository (it is a global public repository of Java libraries, usually an open-source). If your database driver is published on some public repository you can use this feature. Maven artifacts are better than plain jar files because you can see all existing driver versions and can change the driver version in runtime without any driver properties reconfiguration.

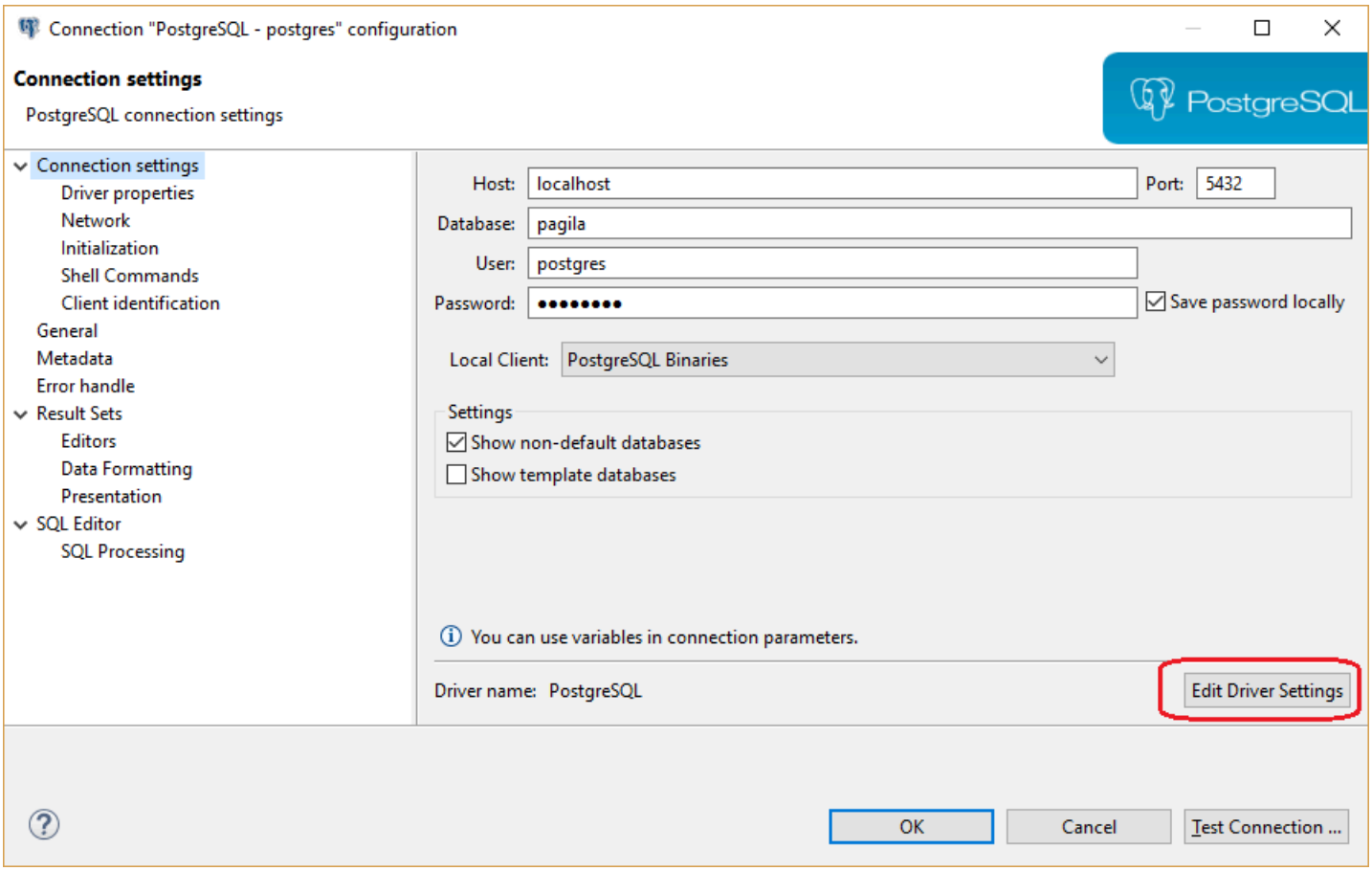
for additional information look How to add additional artifacts to the driver

Saving driver, adding connection

After you have finished configuring your driver, just press the Ok button.

Now you can create connection.

If you need to change some driver properties later you can access them directly from connection properties dialog:



URL Templates

JDBC drivers use URLs to identify remote servers - strings similar to classic web URLs. Usually, URL has form `jdbc:vendor:host:port/database`, for example `jdbc:postgresql:localhost:5432/postgres'. It is not very convenient to edit such a long and an unobvious string. DBeaver can construct this URL from connection parameters (like host, port, etc).

For example above the URL template is: jdbc:postgresql://{host}:{port}/{database}

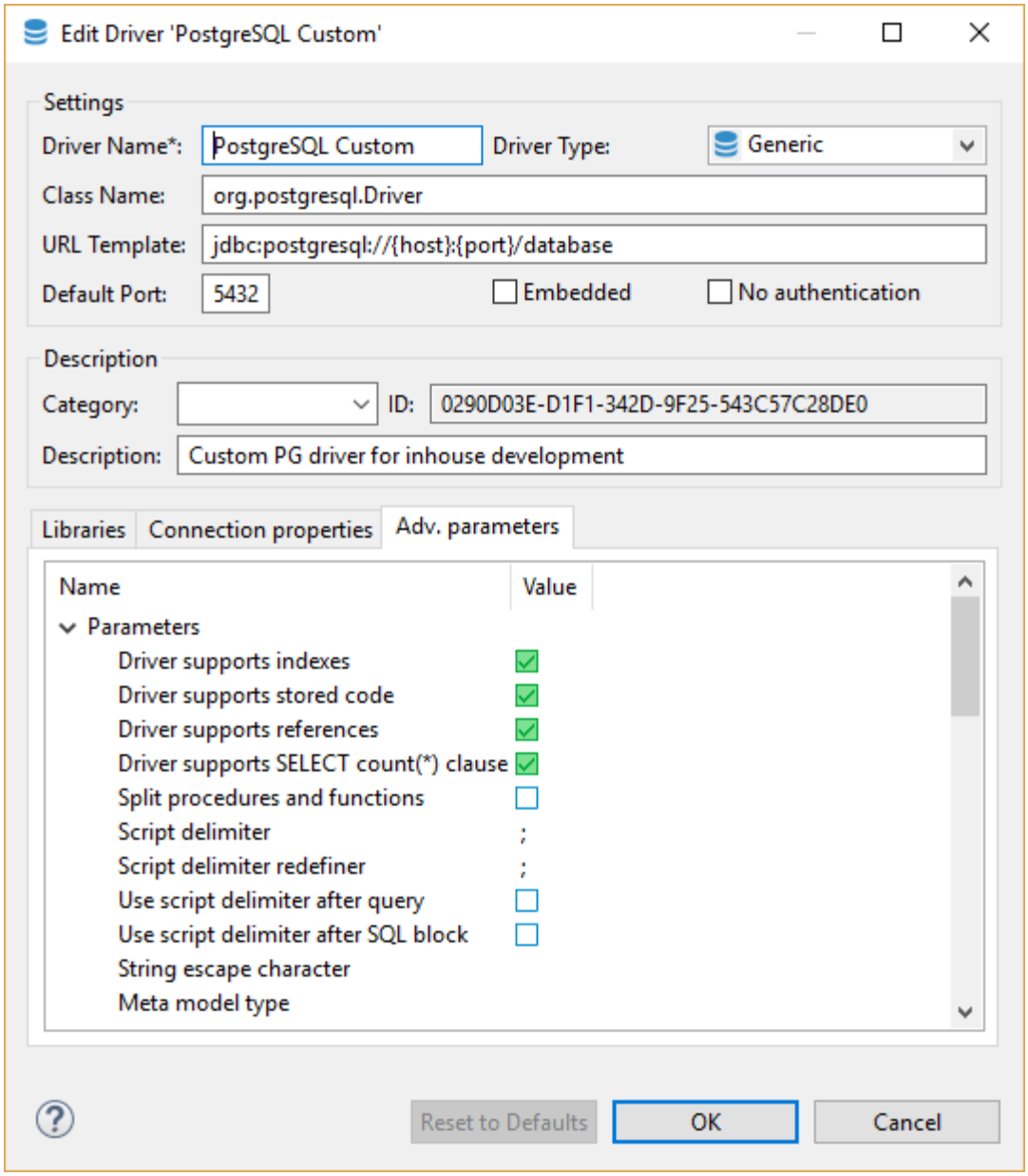
Host, port and database are parameters which you will need to enter on the connection configuration page.

Supported URL variables:

Parameter	Description
{host}	Database server host name
{port}	Database server port number
{database}	Target database name
{server}	Target server name (rarely used)
{folder}	Folder path (on the local file system). Used for embedded drivers
{file}	File path (on the local file system). Used for embedded drivers

Advanced settings

For most drivers you do not need to change any advanced properties. But in some cases you can use this as driver tuning, e.g. for better performance or for structure fixing.



Main parameters

Parameter	Description
Driver supports indexes	Driver supports table indexes
Driver supports stored code	Whether this driver supports stored code (procedures, functions, packages, etc)
Driver supports references	Driver supports table references (foreign keys)
Driver supports SELECT count() *clause* | *Driver supports SELECT count()* clause	
Driver supports views	Driver supports table views
Split procedures and functions	Show procedures and functions in different folders

Script delimiter	Literal for SQL queries separation in scripts
Script delimiter redefiner	SQL clause which redefines script delimiter value
Use script delimiter after query	Keep SQL script delimiter after each SQL query
Use script delimiter after SQL block	Keep SQL script delimiter after SQL script blocks (BEGIN/END)
String escape character	Character used to escape special symbols in strings
Meta model type	Type of metadata reading model - standard or indexed
All Objects Pattern	SQL pattern for all metadata objects
Omit catalog(s)	Do not read and use catalog (aka database) information
Omit single catalog	Hide catalog (database) if there is only one catalog on server
Omit schema(s)	Do not read and use schemas information
Omit single schema	Hide schema if there is only one schema on the server
Use schema filters	Use JDBC schema filters when the database does not support catalogs. Otherwise just read all database schemas and filter on client-side
Omit type cache	Do not use data types provided by driver
Shutdown parameter	Database shutdown URL parameter
Create database parameter	Database create URL parameter
Driver supports multiple results	Driver supports multiple results for a single query
Driver supports result set limit	Driver supports multiple result set limit (max rows)
Driver supports structure cache	Driver supports structure cache reading. Enables schema columns, keys, etc
Driver supports TRUNCATE operation	Driver supports TRUNCATE command. It is much faster than DELETE without criteria

Queries (Custom driver queries)

Parameter	Description
Get active database	Query to obtain active database name
Set active database	Query to change active database

Shutdown database	Query to shutdown active database connection. Used for some embedded databases
PING query	Query to check connection state
Dual table name	Name of dummy 'DUAL' table which is used for expressions evaluation
Active object type	Type of selectable object (schema, catalog)
Driver supports results scrolling	Driver supports resultset scrolling
Quote reserved words	Quote columns/table names if they conflicts with reserved SQL keywords
Escape LIKE masks in search queries	Use to access JDBC metadata API. Enabled by default but should be disabled for some (broken) drivers

DDL (DDL generation options)

Parameter	Description
Drop column short syntax	Use 'ALTER TABLE DROP column-name' instead of standard syntax
Drop column - use brackets	Use 'ALTER TABLE DROP (column-name)' instead of standard syntax
Use legacy SQL dialect for DDL	Use legacy SQL dialect for DDL
Add COLUMN keyword in alter table query	Add COLUMN keyword after keyword ADD and before column name in alter table query

Formatting (SQL values formats)

Parameter	Description
Timestamp format	Format pattern for timestamp columns
Date format	Format pattern for date columns
Time format	Format pattern for time columns

Summary

If you have configured some driver, it works well and you think that it makes sense to have this driver configuration in standard DBeaver, please send your configuration to us. Just create a feature request issue on GitHub and copy/paste driver description to the ticket (in any suitable form).

Thank you.

JDBC-ODBC bridge

Note: This feature is available in Lite, Enterprise, Ultimate and Team editions only.

ODBC allows you to create a connection to almost any database in DBeaver by using native ODBC drivers. You can use it as an alternative to DBeaver drivers or if DBeaver does not have a driver for your database.

As of now, we aim to support any ODBC drivers. However, the following database drivers have been verified for compatibility:

- Amazon Redshift
- BigQuery
- DB2 for LUW
- DB2 for IBM i
- MariaDB
- MySQL
- Oracle
- PostgreSQL
- SQL Server
- SQLite
- Snowflake

Installing ODBC driver manager

DBeaver works with ODBC drivers using ODBC driver manager. A driver manager is a component is installed in your operating system that provides access to drivers written in native programming languages.

- On **Windows**, a driver manager is already installed by default.

  You can access it by pressing `WIN+R` and typing `odbcad32`.

- On **Linux** and **macOS**, you need to install this component separately.

  We recommend the iODBC driver manager as it's confirmed to work flawlessly with DBeaver against Linux and macOS. To install, follow the provided instructions.

  You can access it by typing `iodbcadm-gtk` in the terminal.

Installing ODBC drivers

- On **Windows**, it's done by installing a driver using Windows Installer (`.msi`). The installation is the same as for any other software.

- On **Linux** and **macOS**, a driver can be either installed from a package manager or by building it manually.

  Once a driver is installed, you need to add it to the driver manager. Open the driver manager, go to **ODBC Drivers** tab, press **Add a driver** and select required files.

Configuring ODBC data sources

Note: The process is mostly the same for all operating systems.

1. Open the driver manager
2. Go to **User DSN** or **System DSN** tab
    1. Choose **User DSN** if you want to create a data source accessible only by the current user
    2. Choose **System DSN** if you want to create a data source accessible by all users
3. Click **Add** and choose the appropriate driver
4. Configure data source settings and click **OK**

Creating ODBC connections in DBeaver

Open the **New Database Connection** dialog and select **ODBC** (Not **ODBC - Legacy**).

If you want to use a **configured ODBC data source**, switch to **Data Source** connection type and choose it from a drop-down menu:

Data Source	Data Source (drop-d

If you want to use an **ODBC driver directly** and manually configure connection parameters, switch to **Manual** connection type and choose an appropriate driver from a drop-down menu and fill rest of the fields according to your configuration:

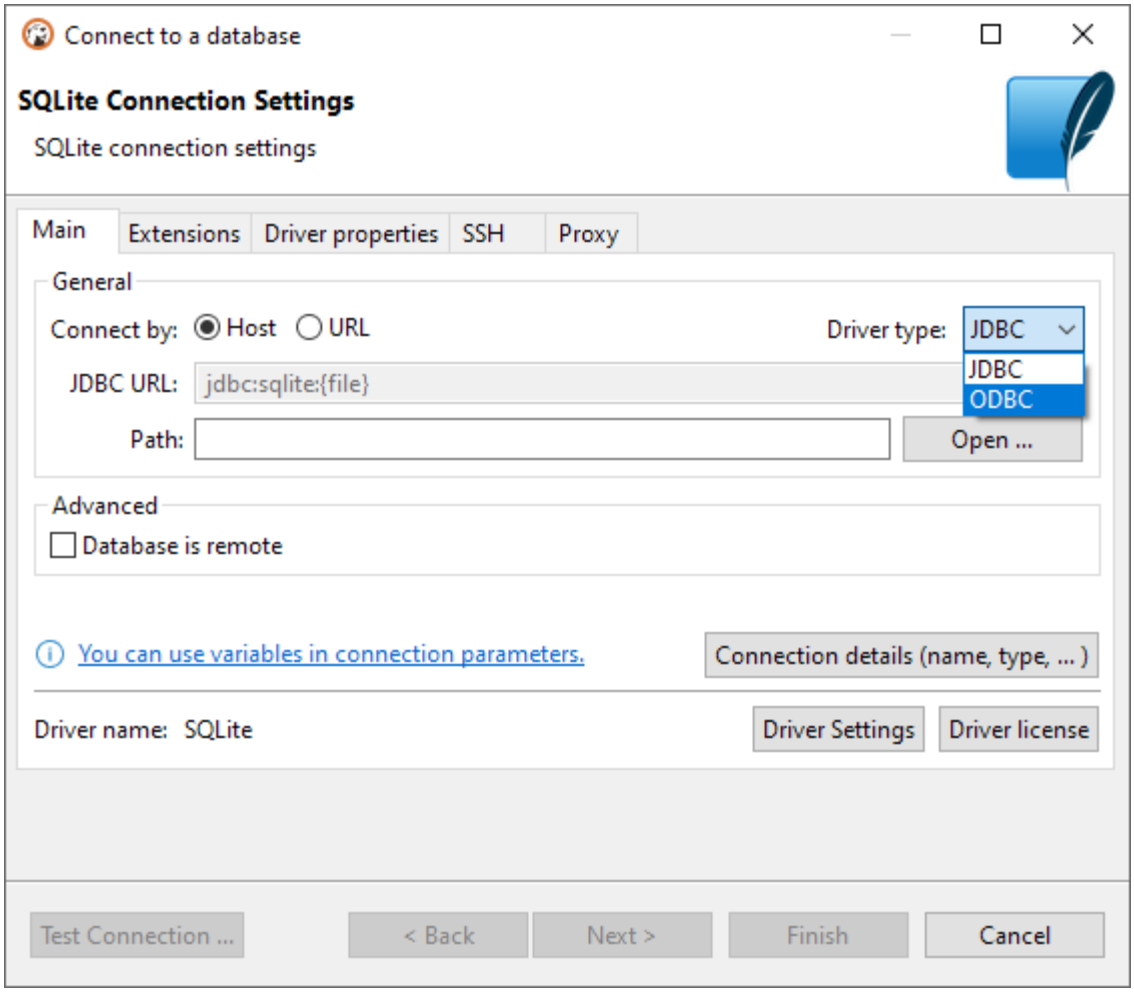
Manual	Manual (drop-down)

Making existing DBeaver connections use ODBC driver

DBeaver allows you to replace the underlying database driver of a connection with our ODBC driver. This is not the same as creating a new ODBC connection as it keeps all database-specific functionality untouched.

Note: In this mode, you can't use database-specific authentication methods and settings.

To do so, open existing connection and choose **ODBC** as the driver type from the drop-down menu:


Connect to a database
SQLite Connection Settings
SQLite connection settings
Main
Extensions
Driver properties
SSH
Proxy
General
Connect by:
Host
URL
Driver type:
JDBC
JDBC
ODBC
JDBC URL:
jdbc:sqlite:{file}
Path:
Open ...
Advanced
Database is remote
You can use variables in connection parameters.
Connection details (name, type, ...)
Driver name: SQLite
Driver Settings
Driver license
Test Connection ...
< Back
Next >
Finish
Cancel

Data export/import

Data transfer is a crucial feature that enables you to export and import data in various formats as well as moving data between tables in the same or to different databases.

- Export data
- Import data
  - Importing from CSV
  - Importing from Excel
  - Importing from XML
  - Importing from database table

Note: The data transfer operation runs in the background, allowing you to continue working with your database during the export or import process. However, try to avoid changing data in the tables that have been selected for export or import until the process is complete.

Export data

You can transfer data from one database to another or export it in different types and formats:

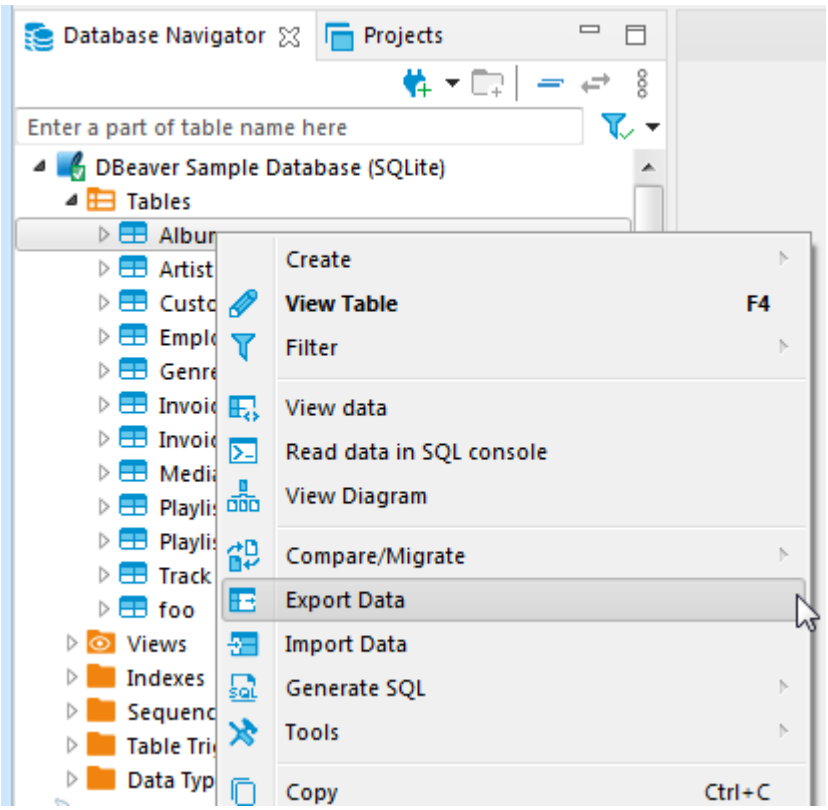
- `CSV`
- `XLSX` (Excel spreadsheet)
- `HTML`
- `XML`
- `TXT`
- `JSON`
- `Markdown` files
- `SQL` statements

- Source code arrays

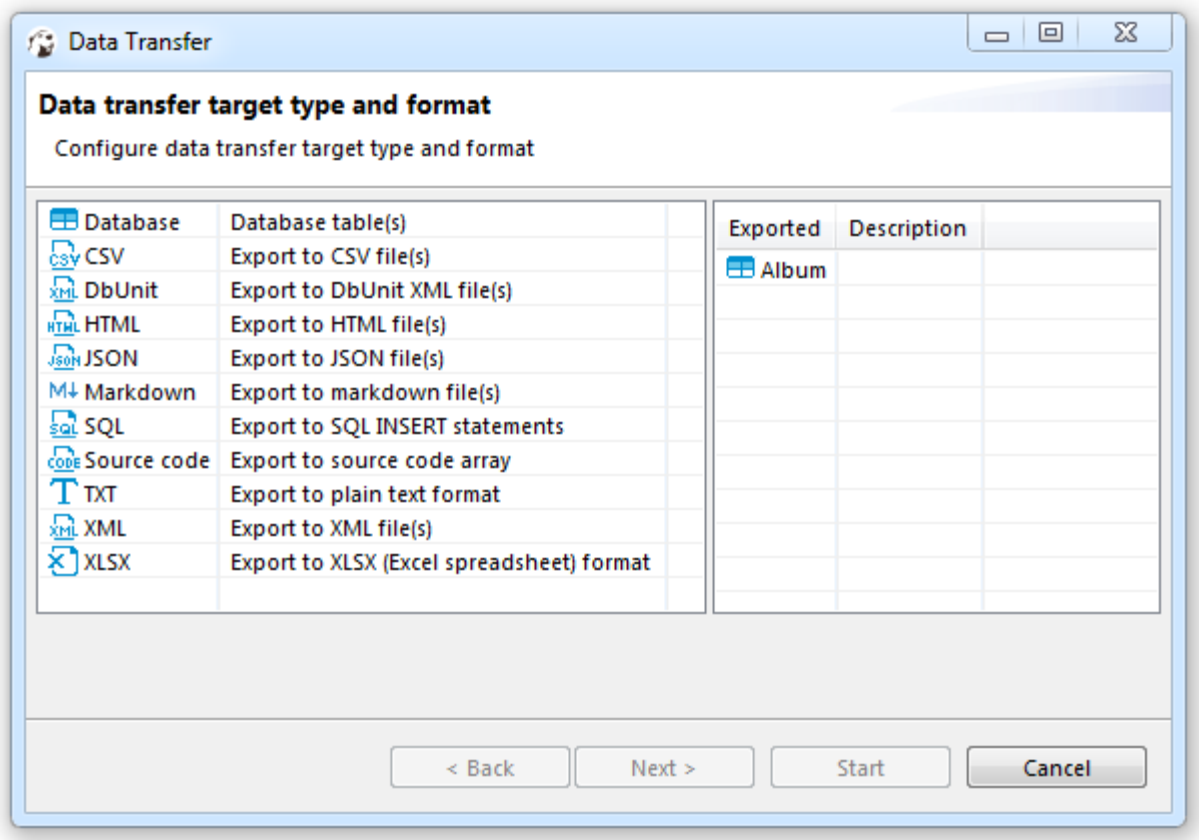
The process of exporting data follows a similar procedure for the supported formats. Therefore, in this guide, we will focus on the data exporting process using the `CSV` format.

1. Select the table or tables you want to export. In the context menu, choose **Export Data**.

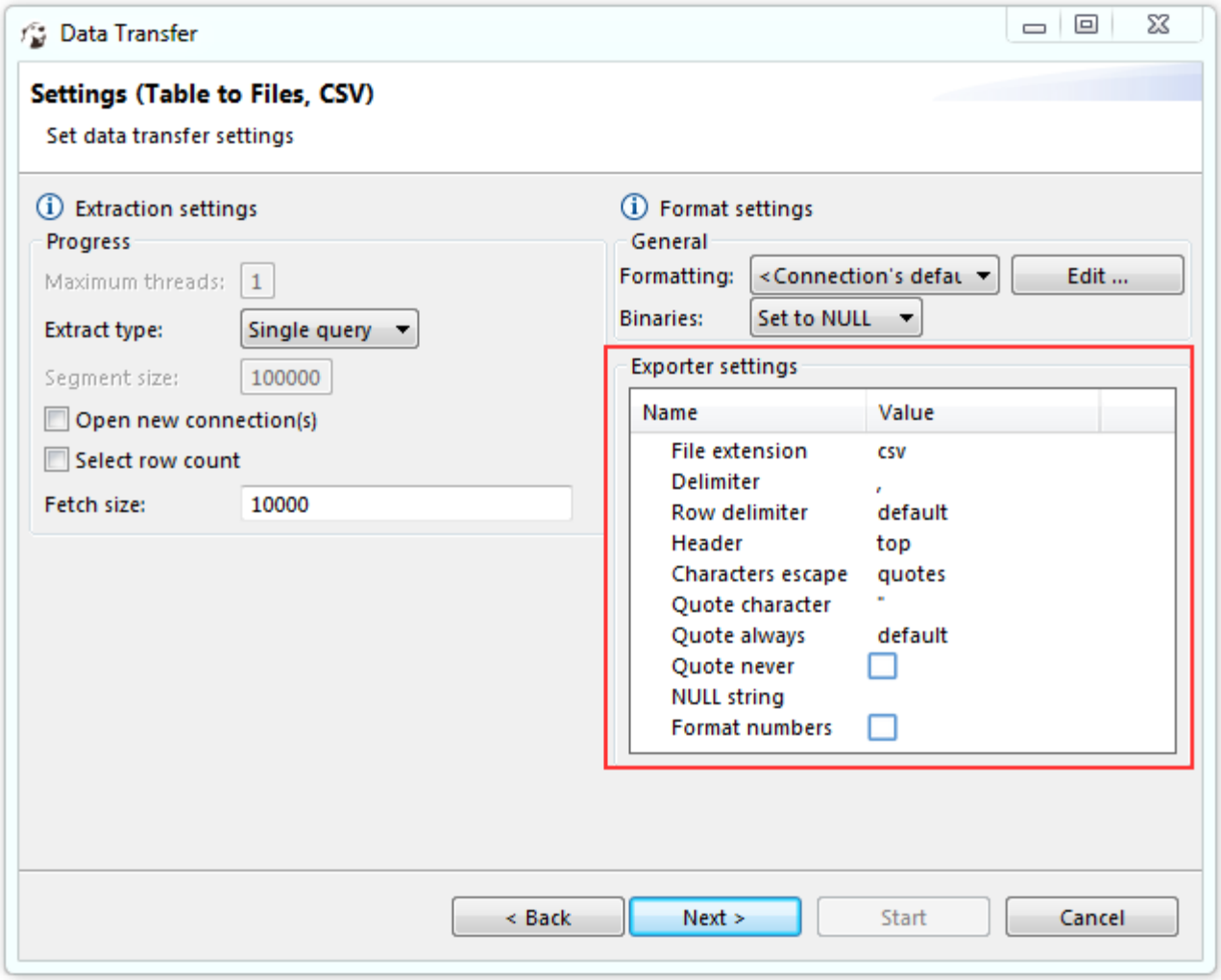
   Note: you can also export data from custom SQL query results. To do that, choose **Export data** in the results context menu.



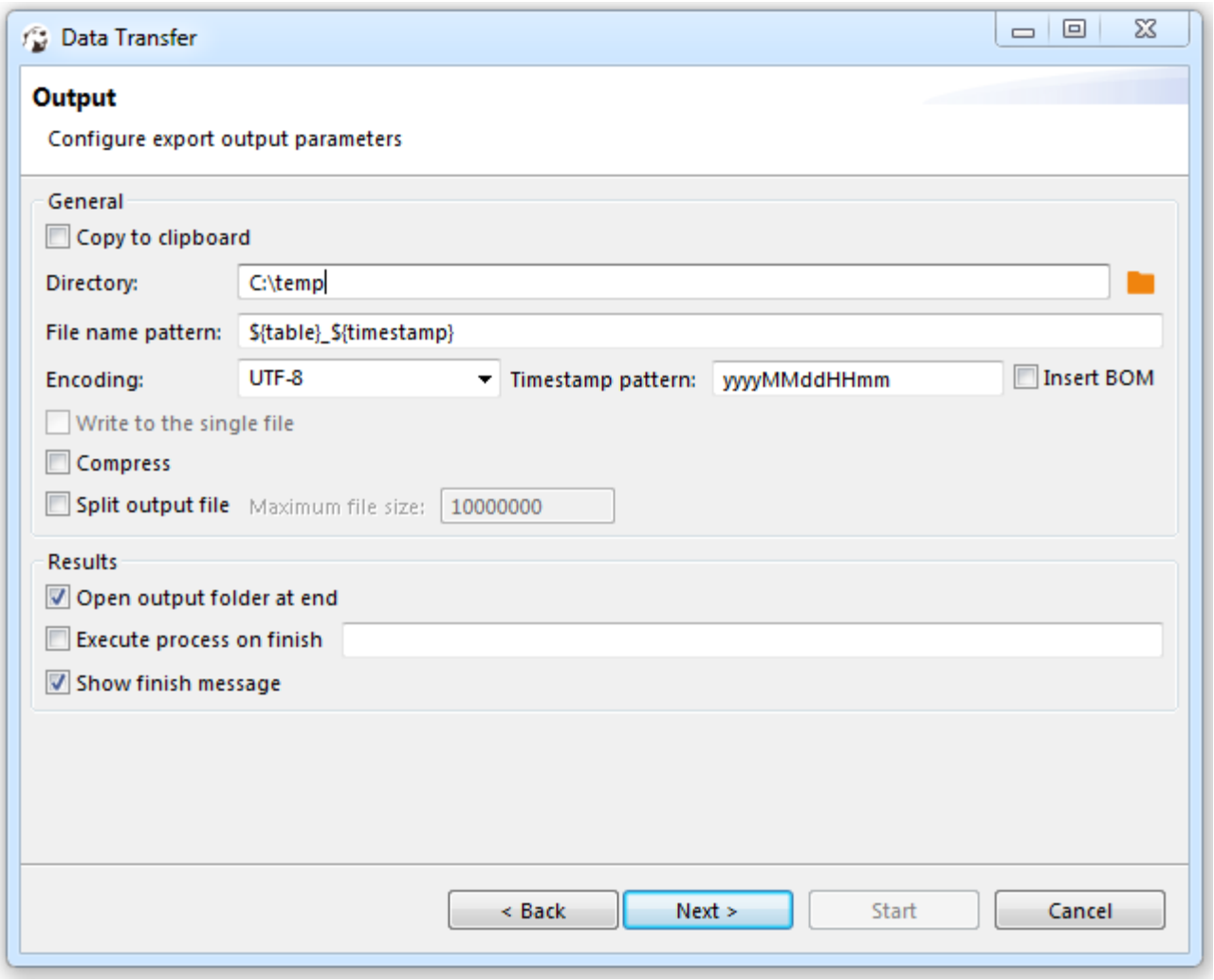
2. In the window that appears, choose `CSV` and click **Next**.



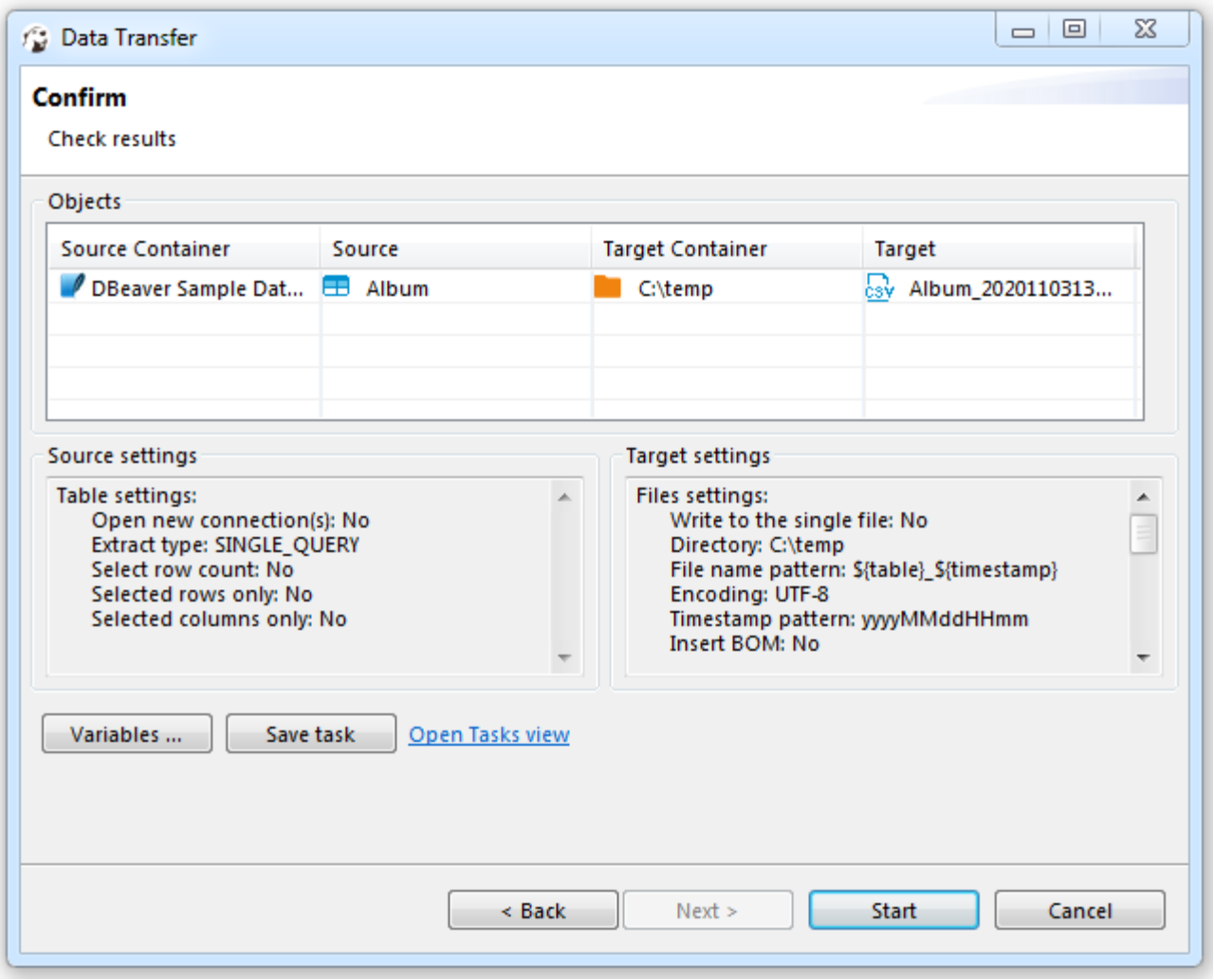
3. Set your data extraction options (how the data will be read from the tables). This may affect the extraction's performance. And set the export format option. They are specific to the data format you chose in step 2:



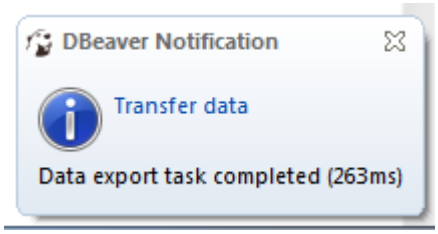
4. Set options for output files or clipboards. **Note**: Timestamp pattern is used here to target the file name pattern:



5. Review what you want to format and which format you will export. You can also save all your settings as a task in this step or change the task variables:



6. Press finish. See extraction progress. You can keep working with your database during the export process as the extraction will be performed in the background.

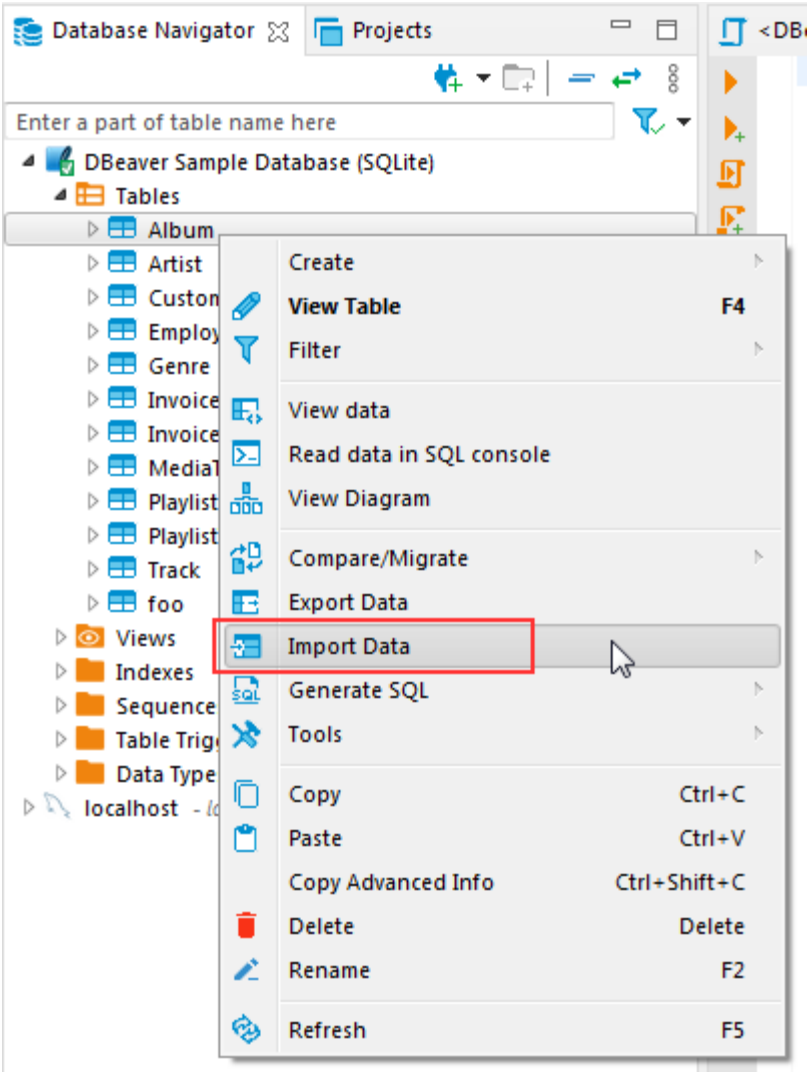


Import data

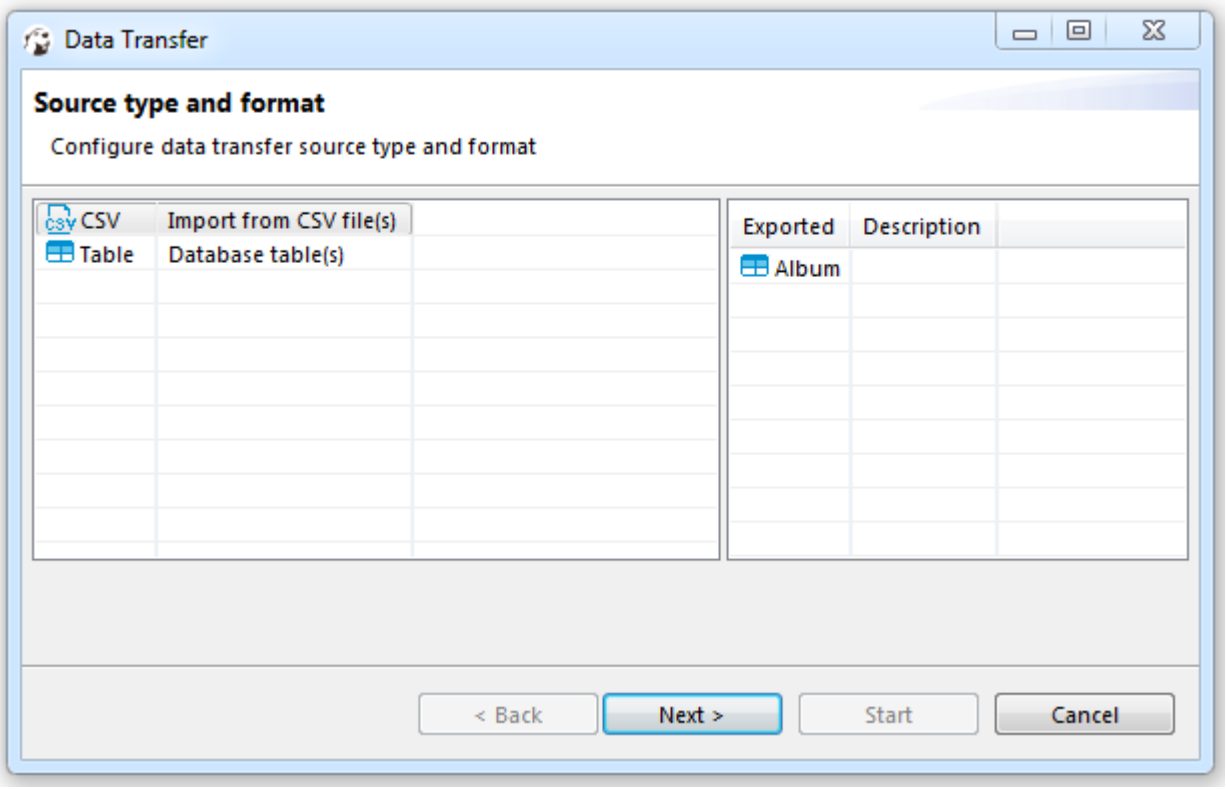
You can import data to your database from `CSV`, `XLSX`, and `XML` files.

Importing data from CSV file

1. Select one or more tables to which you want to import data. In the context menu, choose **Import Data**:

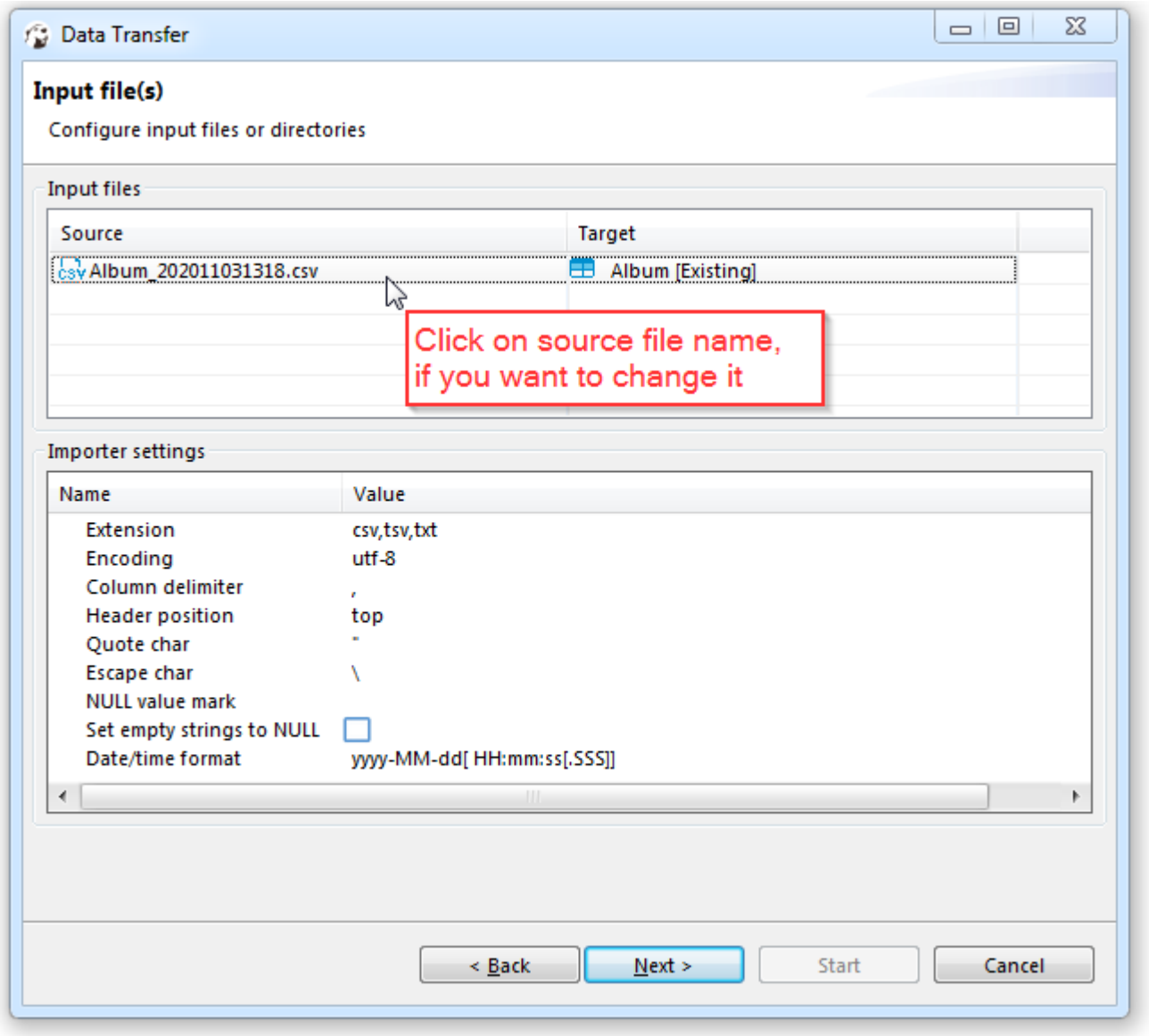


2. In the window that appears, choose `CSV` and click **Next**.

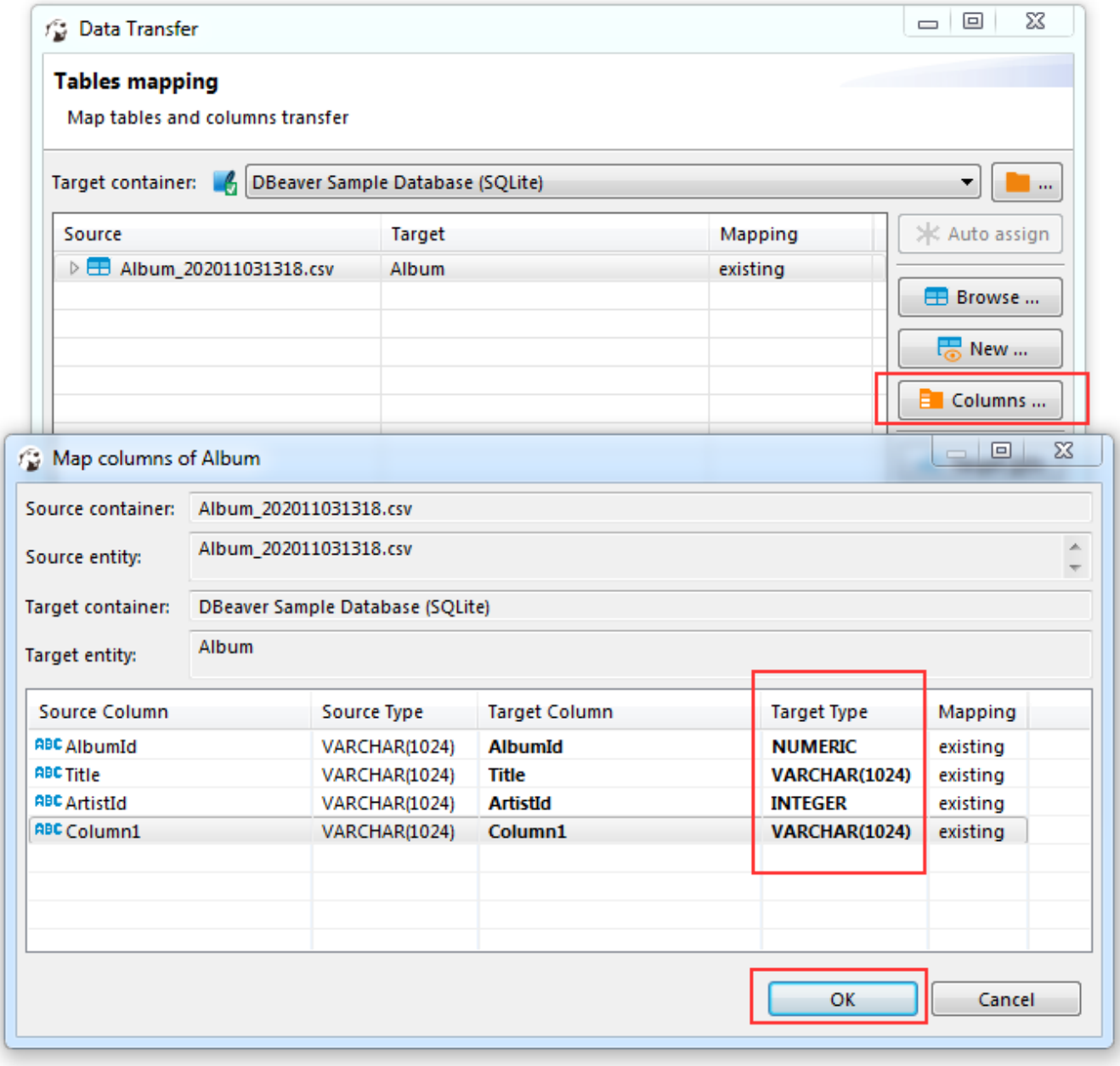


3.

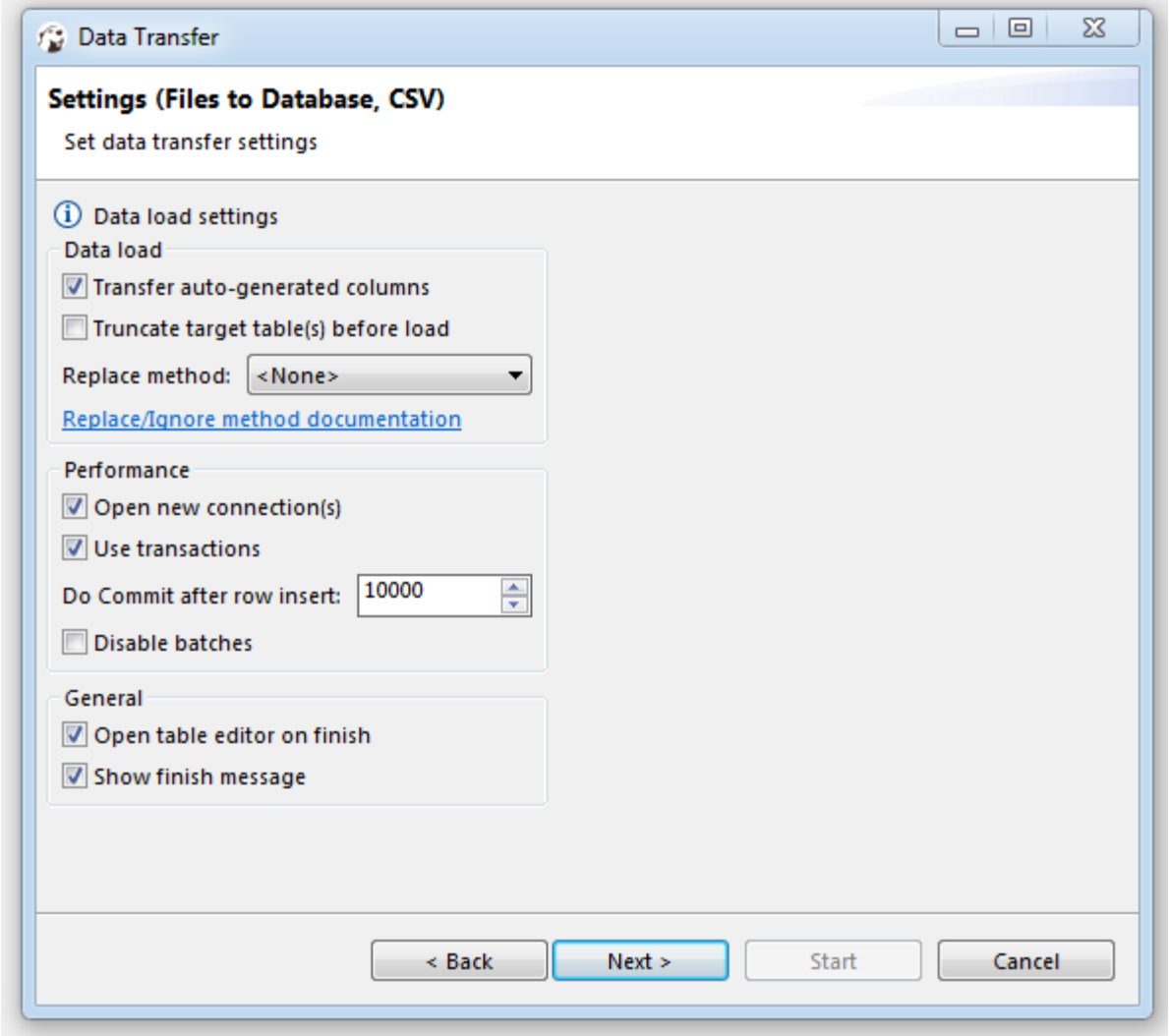
Select the input `CSV` file for each table you want to import. You can change the Importer settings (format specific) during this step:



4. Set CSV-to-table mappings. You must set a column in the `CSV` file for each database table column. You can skip columns (the value will be set to `NULL` in the target table column). You can set constant values for the table column if there is no source column in the `CSV` .

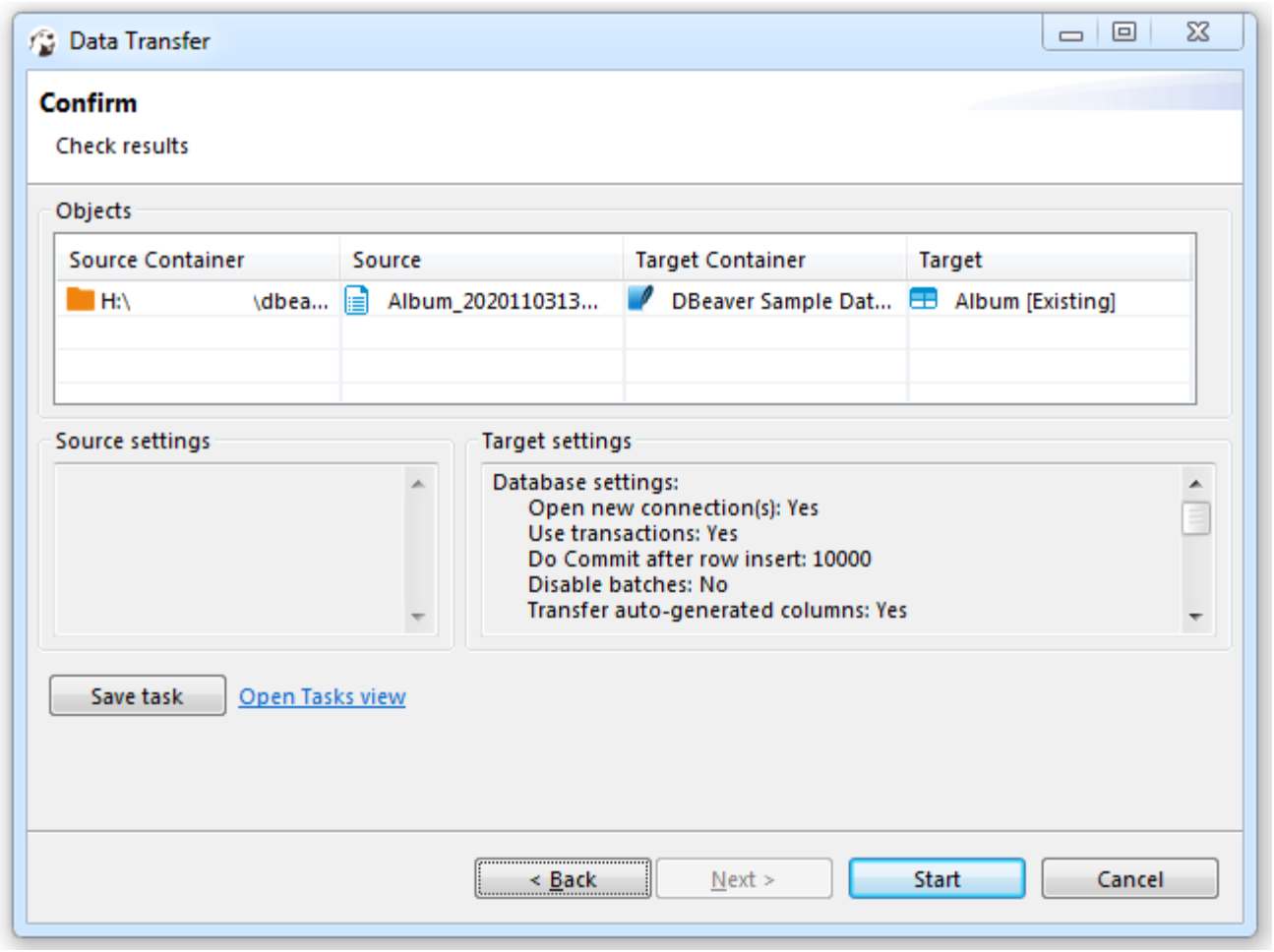


5. Set options for loading data in the database. These options may affect the loading's performance:

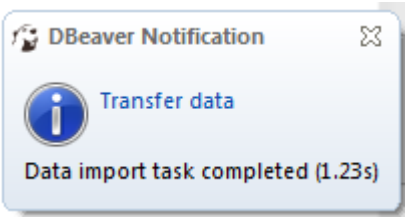


Read our guide on Data Import and Replace to learn more about the replacing method option.

6. Review which file(s) and to which table(s) you will import. You can also save all your settings as a task in this step:

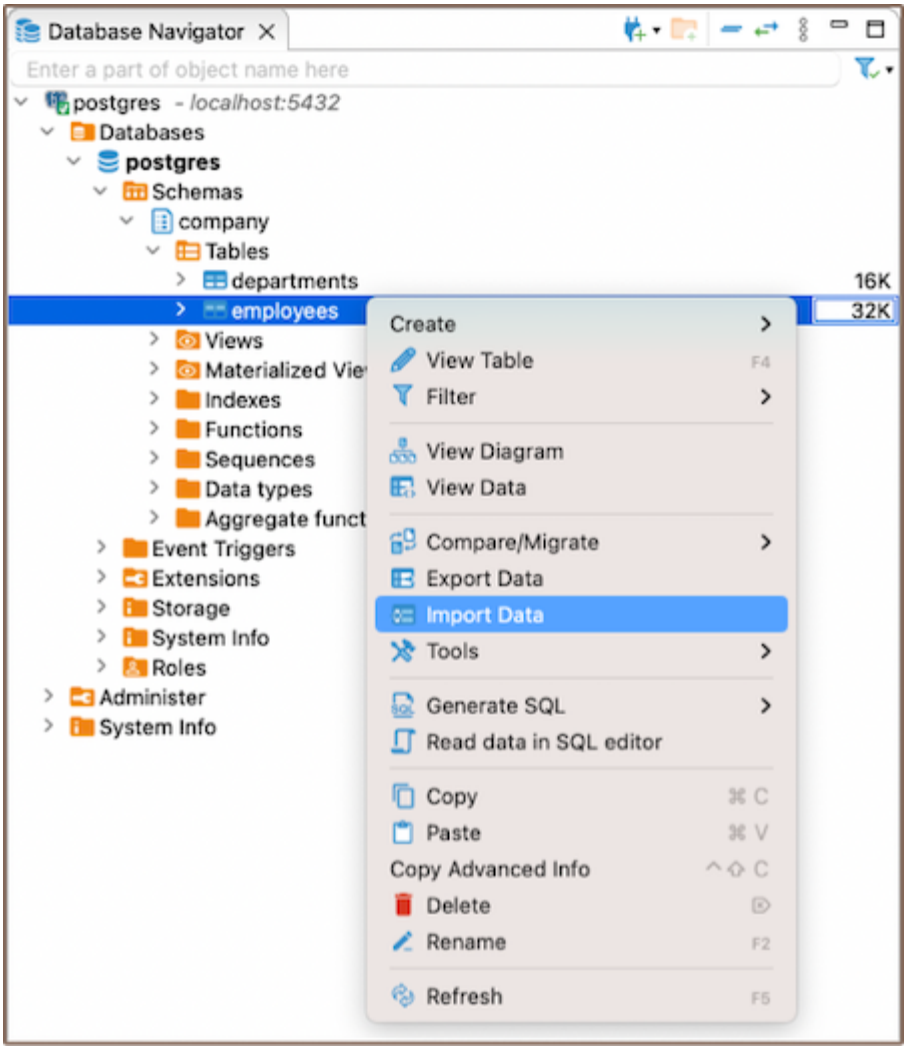


7. Press finish. See extraction progress. You can keep working with your database during the export process as the data loading will be performed in the background. In the end, you will see the following status message:

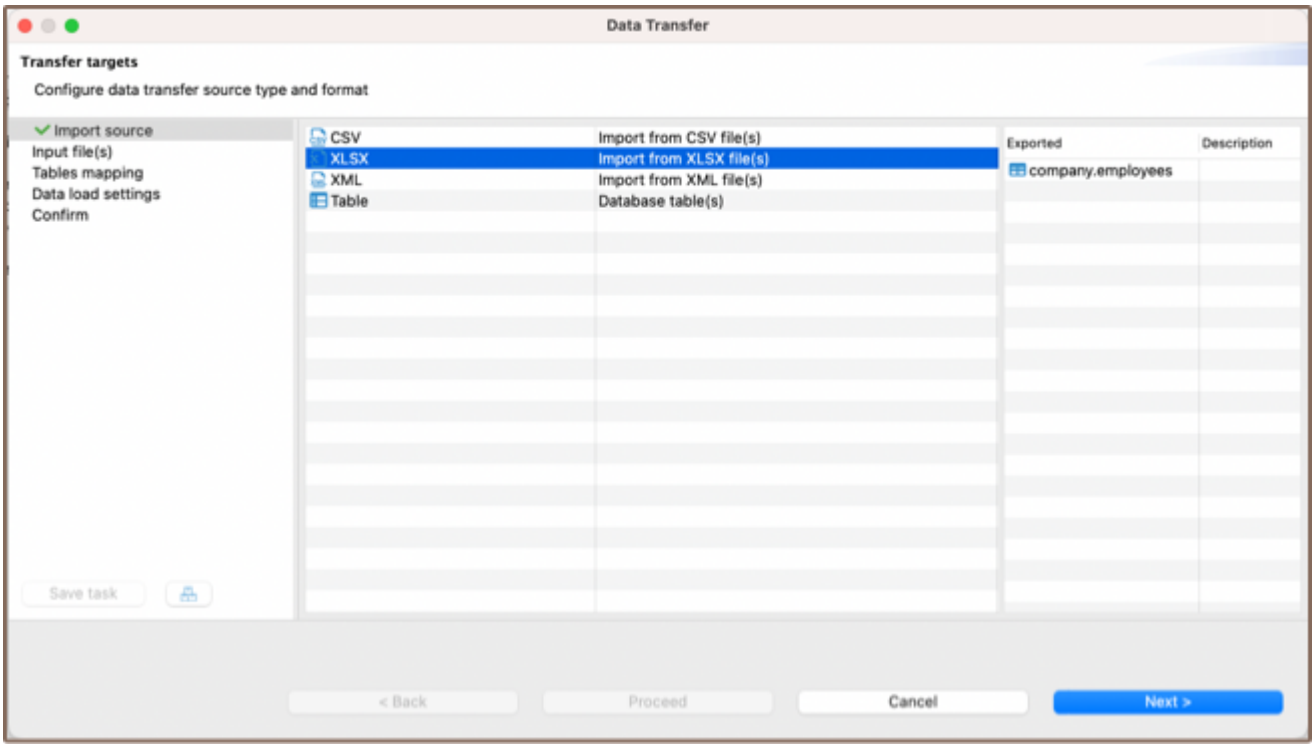


Importing data from Excel file

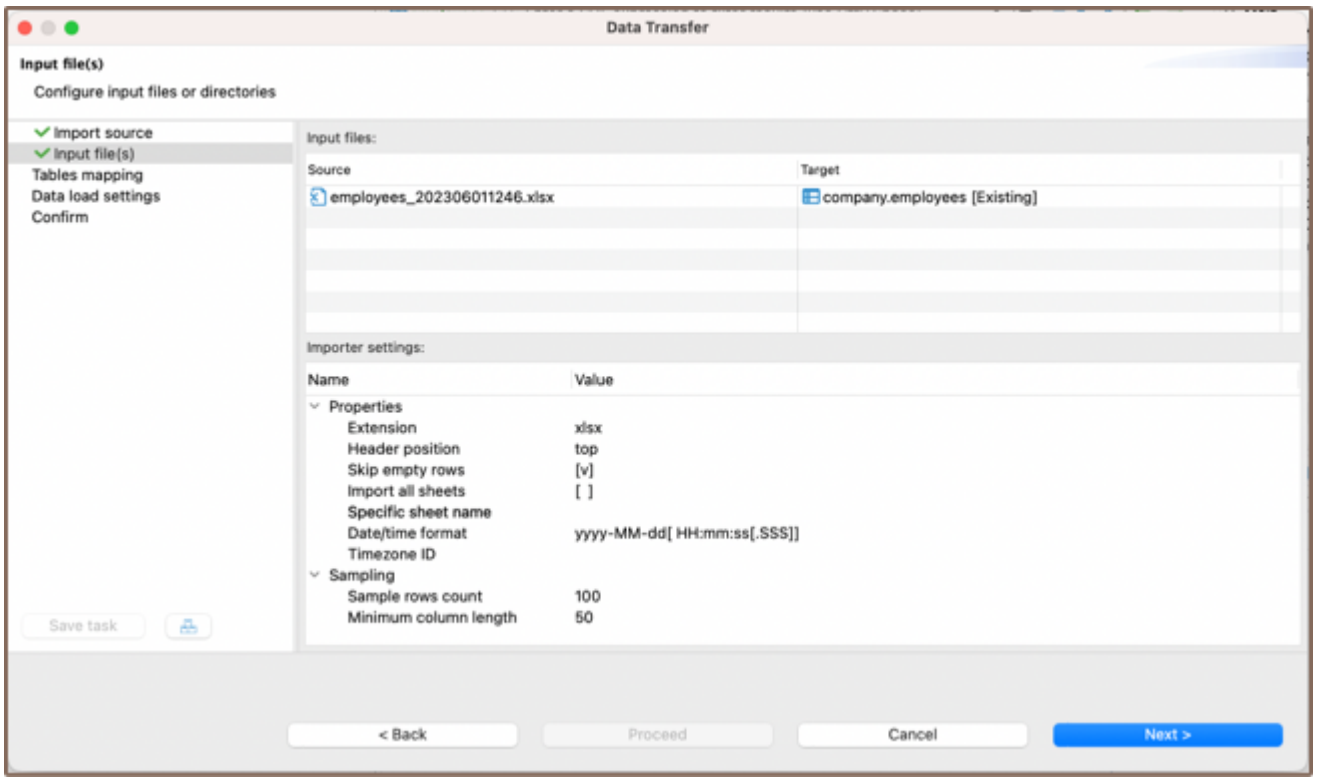
1) Choose the database table (or tables) you want to import data into. Do this by right-clicking on the table name in the **Database Navigator** section and then clicking on **Import Data**.



2) In the window that appears, choose `XLSX` and click **Next**.



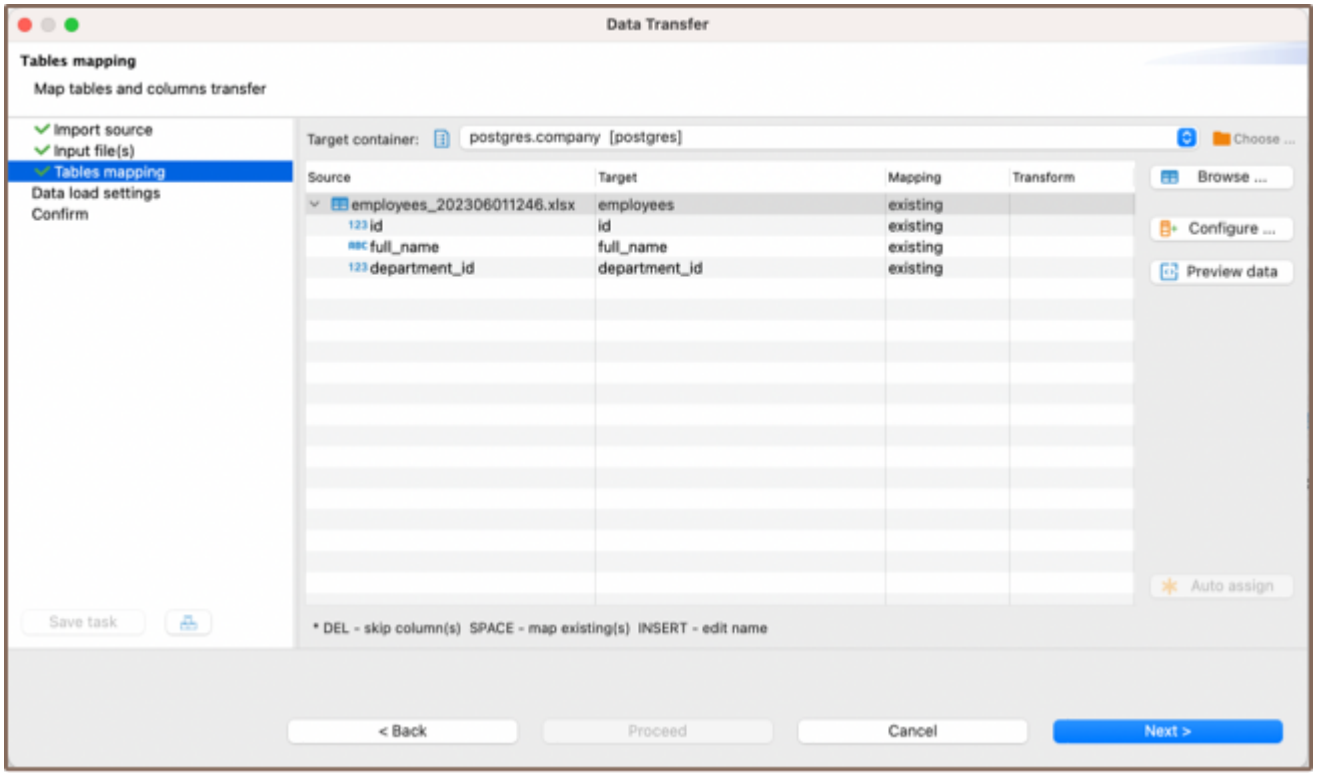
3) In the following window, choose the file that contains the data you wish to import into the table. Select the appropriate settings described below, then click **Next**.



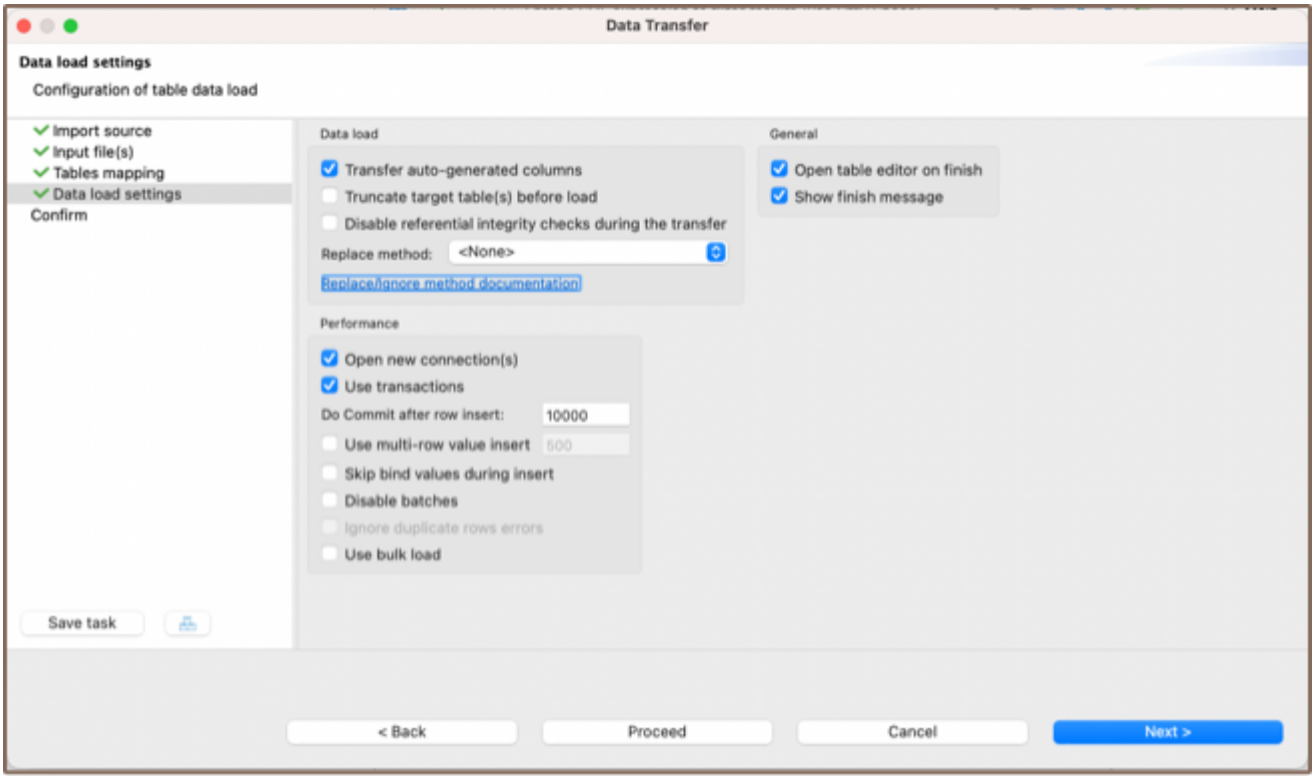
Setting name	Description
Header position	Determines the location of the column names in the Excel table, either at the top or none. This setting specifies whether the column names are located in the first row of the Excel table or if there are no column names present.
Skip empty rows	If this setting is enabled, any open string values encountered during the data processing will be ignored and not inserted into the corresponding cells in the row. If the setting is disabled, all cells in the row will be filled with a `NULL` value if an empty string is encountered.
Import all sheets	Specifies that all sheets in the file should be imported during the data import process.
Specific sheet name	Enables you to choose a particular sheet from the Excel file for importing during the data transfer process.
Date /time format	Use this setting to specify the date format used in the `XLSX` file. This is used to clarify the date format during the import process and does not affect the output data. You can refer to the **java DateTimeFormatter** documentation for details on the format pattern syntax.
	The local machine timezone is used by default. There are three ways to specify the timezone:

Timezone ID	1) Local zone offset: Specify the offset from UTC in the format of either a positive or negative number (e.g., +3, -04:30). 2) Specific zone offset: Specify the offset from GMT or UTC in the format of GMT+/-X or UTC+/-X (e.g., GMT+2, UTC+01:00). 3) Region-based: Specify the timezone using a region-based identifier such as UTC, ECT, PST, etc.
Sample rows count	Determines the number of rows that will be used as a sample to estimate the length and data types of the imported data.
Minimum column length	This value is used when creating a new column and, if necessary, specifying its type. It indicates the minimum number of characters or digits expected in the column. This information helps determine the appropriate data type and size for the column during the creation process.
Save task	Opens the Save Task window to assist in creating a task during the data transfer process. This window provides options and settings for creating and configuring a task related to the data transfer operation.

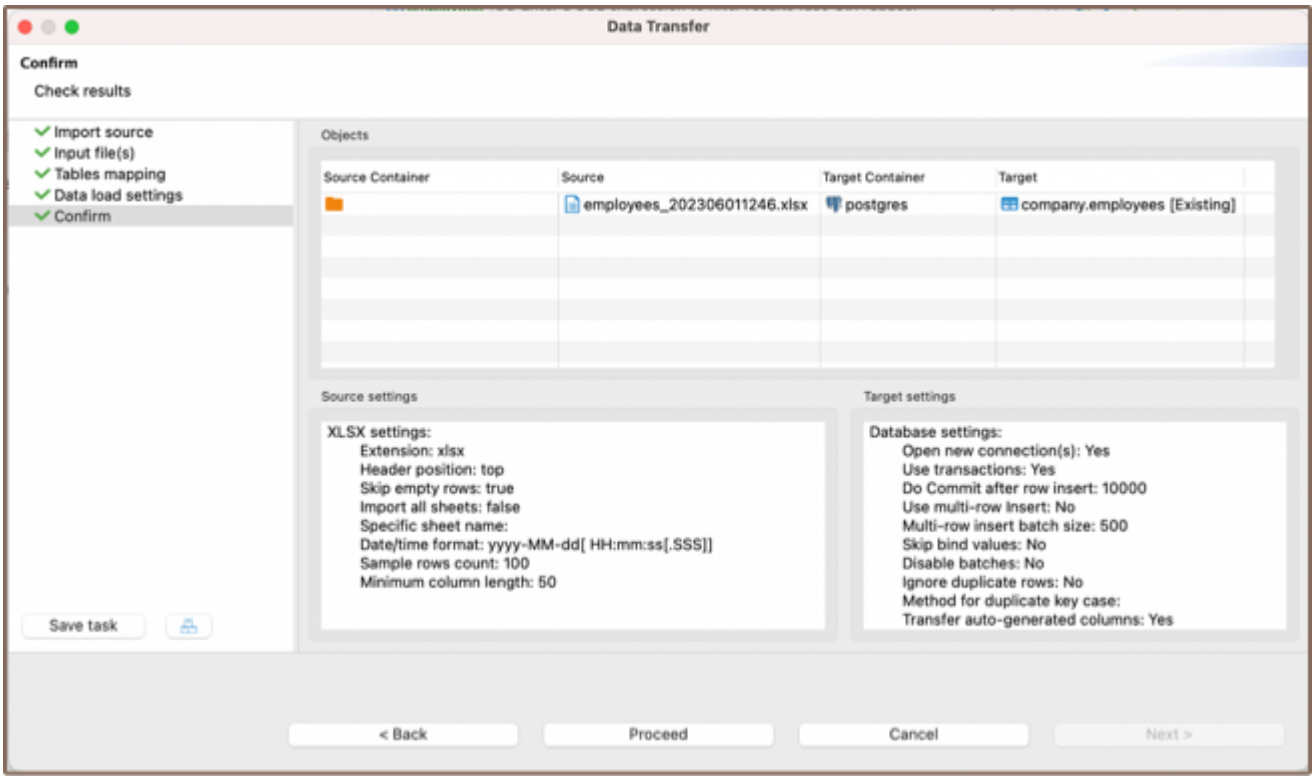
5) In the next window, set XLSX-to-table mappings. Please refer to our mapping process guide for more detailed information.



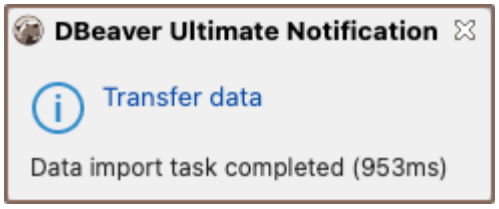
6) Select your data load settings in the subsequent window, and then click **Next**. For more information, please refer to our article's section Data load settings article.



7) In the final window, you can review all the settings you selected earlier. If you missed something, you could go **Back** and fix it. When you are ready, finish the import by clicking **Proceed**.



8) If the `XLSX` file is valid and there are no errors, you will see a notification window with information about the completion of the task. You can keep working with your database during export, as the data loading will be performed in the background.



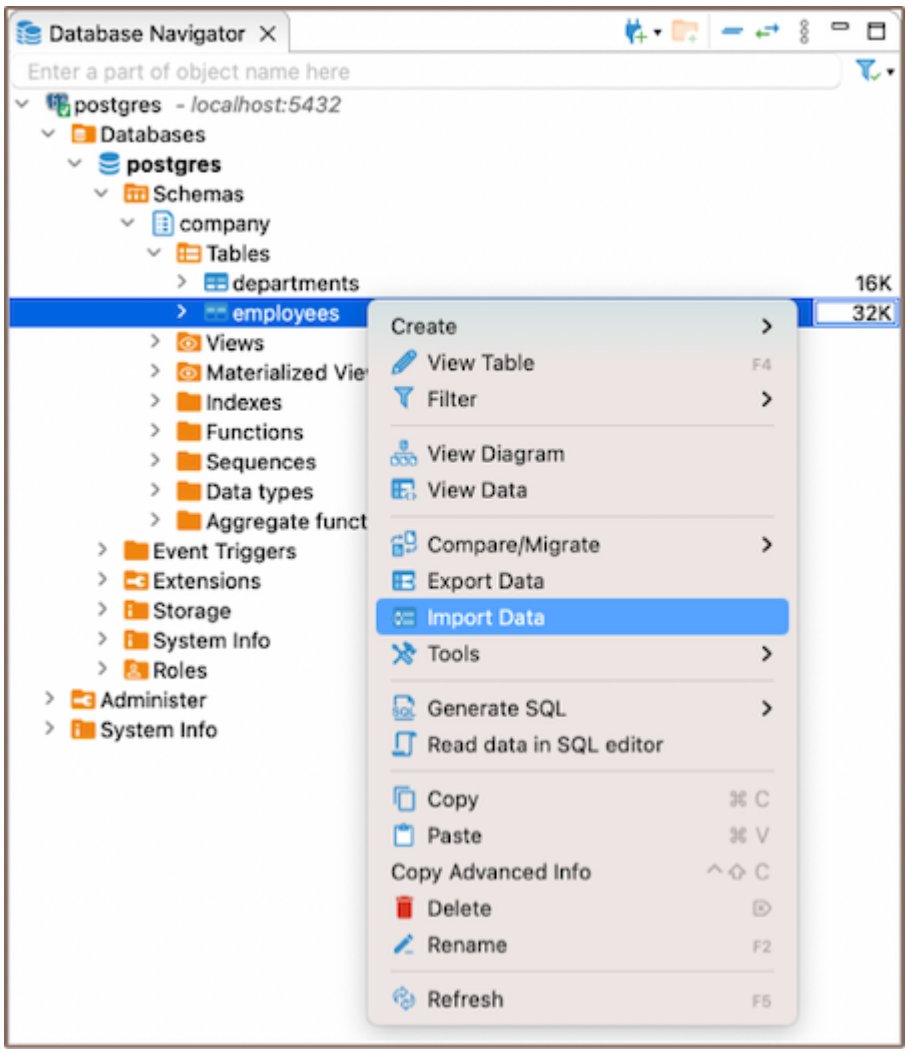
Additional features

When importing from Excel files, DBeaver provides additional features to enhance your data transfer experience:

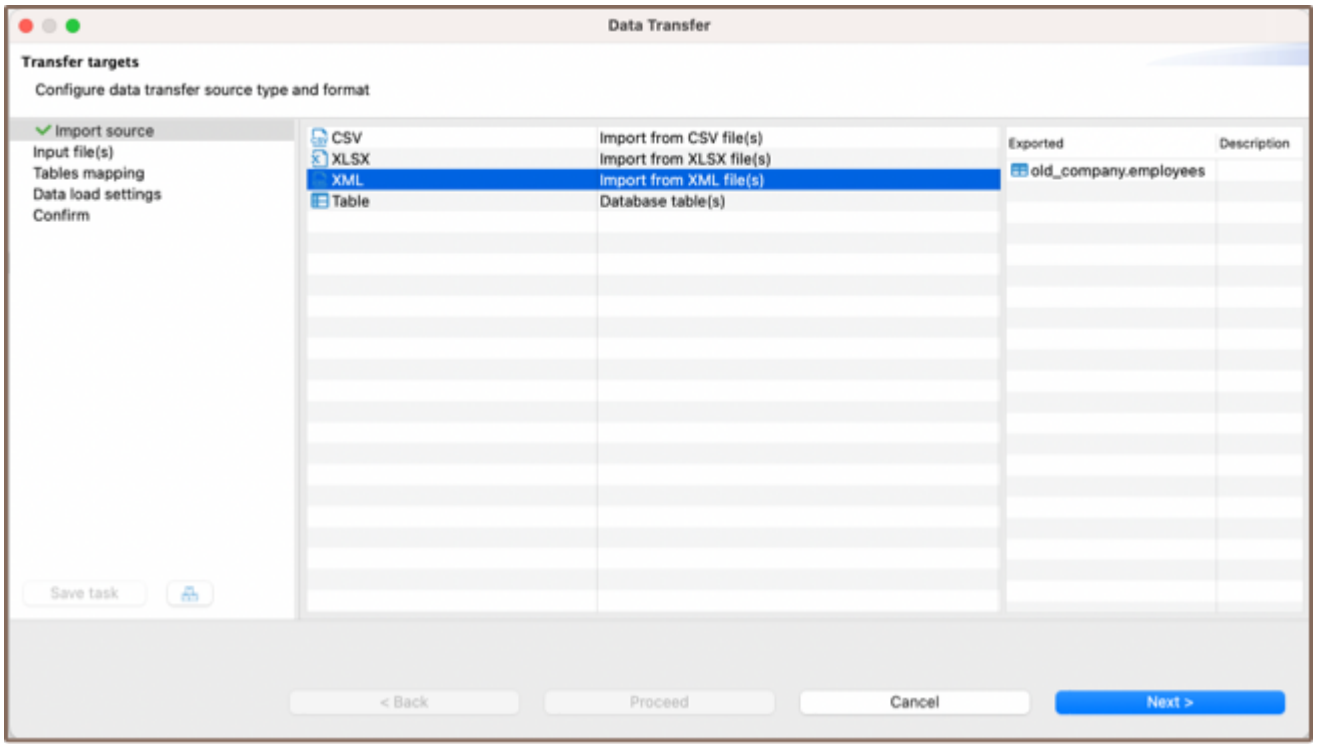
- **Allow importing multiple sheets into different tables**: DBeaver allows you to import multiple sheets from a single `XLSX` file into different tables in your database.

Importing data from XML file

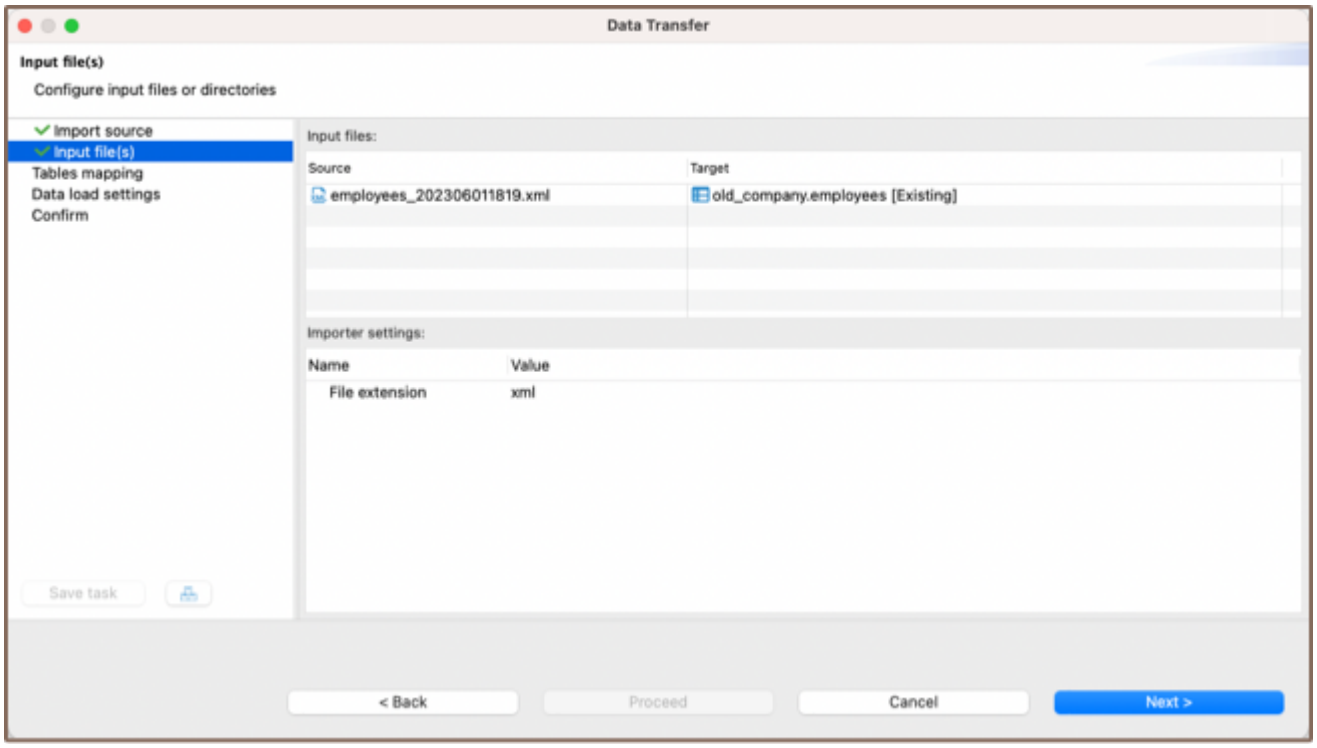
1) Select the database table (or tables) where you want to import data. Do this by right-clicking on the table name in the **Database Navigator** section, then clicking on **Import Data**.



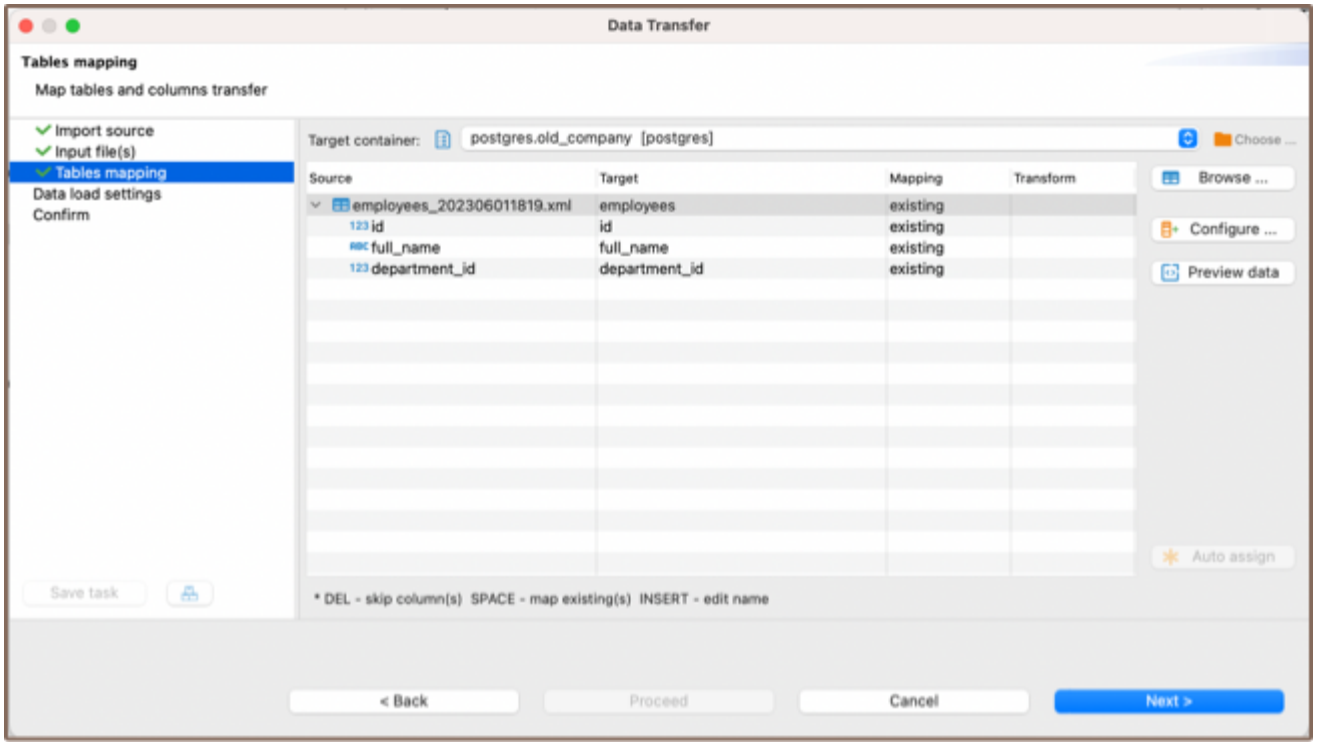
2) In the window that appears, select `XML` and then click **Next**.



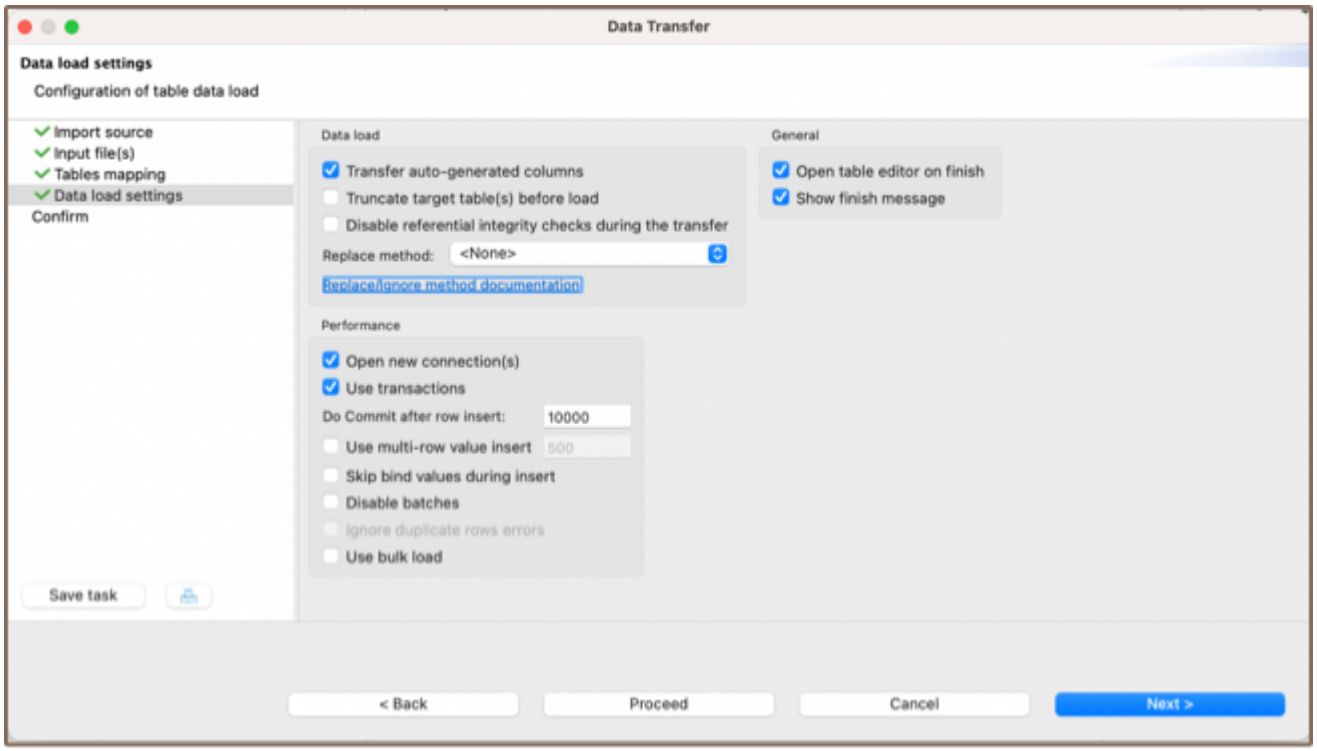
3) In the following window, select the `XML` file that contains the data you want to import, then click **Next**.



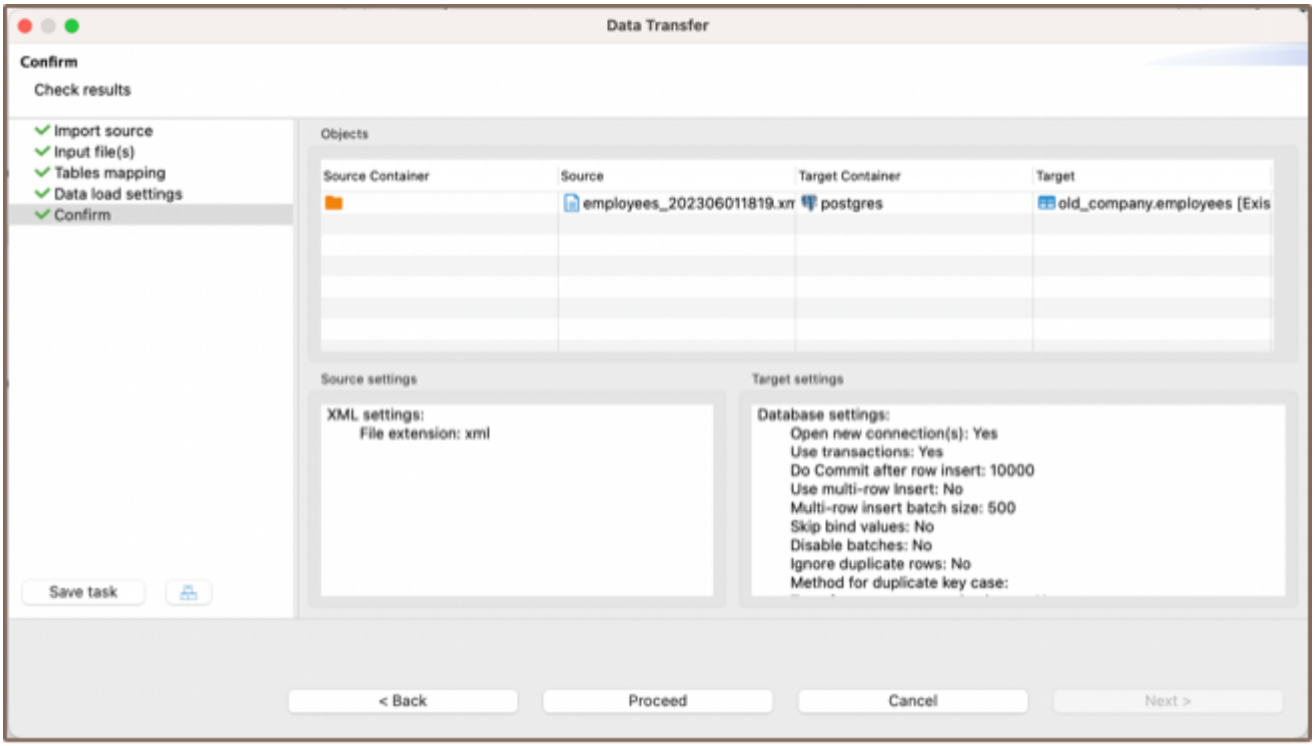
4) In the next window,w set XLSX-to-table mappings. Please refer to our guide for detailed information on the mapping process.



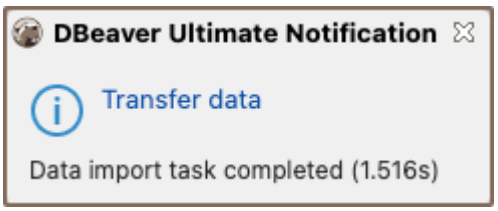
5) select your data load settings in the subsequent window, and then click **Next**. For more information, please refer to our article's section Data load settings.



6) In the final window, you can review all the settings you selected earlier. If you missed something, you can go back and adjust it. Once you are happy with everything, finish the import by clicking **Proceed**.



7) If the file is valid and there are no errors, you will see a notification window with information about the completion of the task. You can keep working with your database during export, as the data loading will be performed in the background.



Importing data from the database table

DBeaver offers seamless data migration capabilities, allowing you to transfer data from one database table to another. For more detailed instructions and insights on data migration, you can refer to our Data Migration guide.

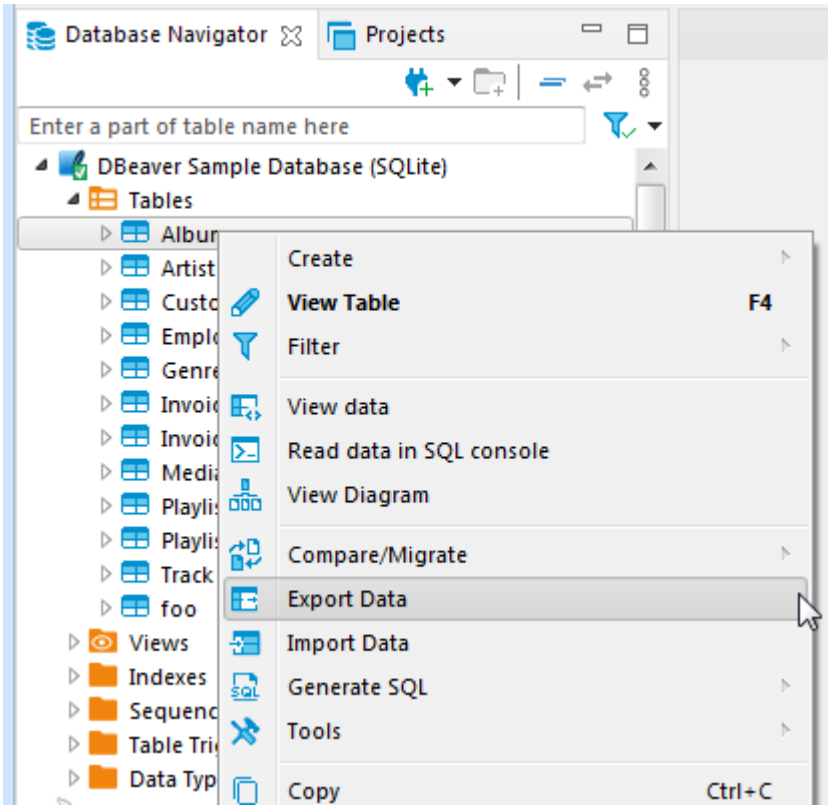
Data migration

DBeaver supports data migration of tables from one database to tables of another one.

To perform a data transfer, please, follow the steps below.

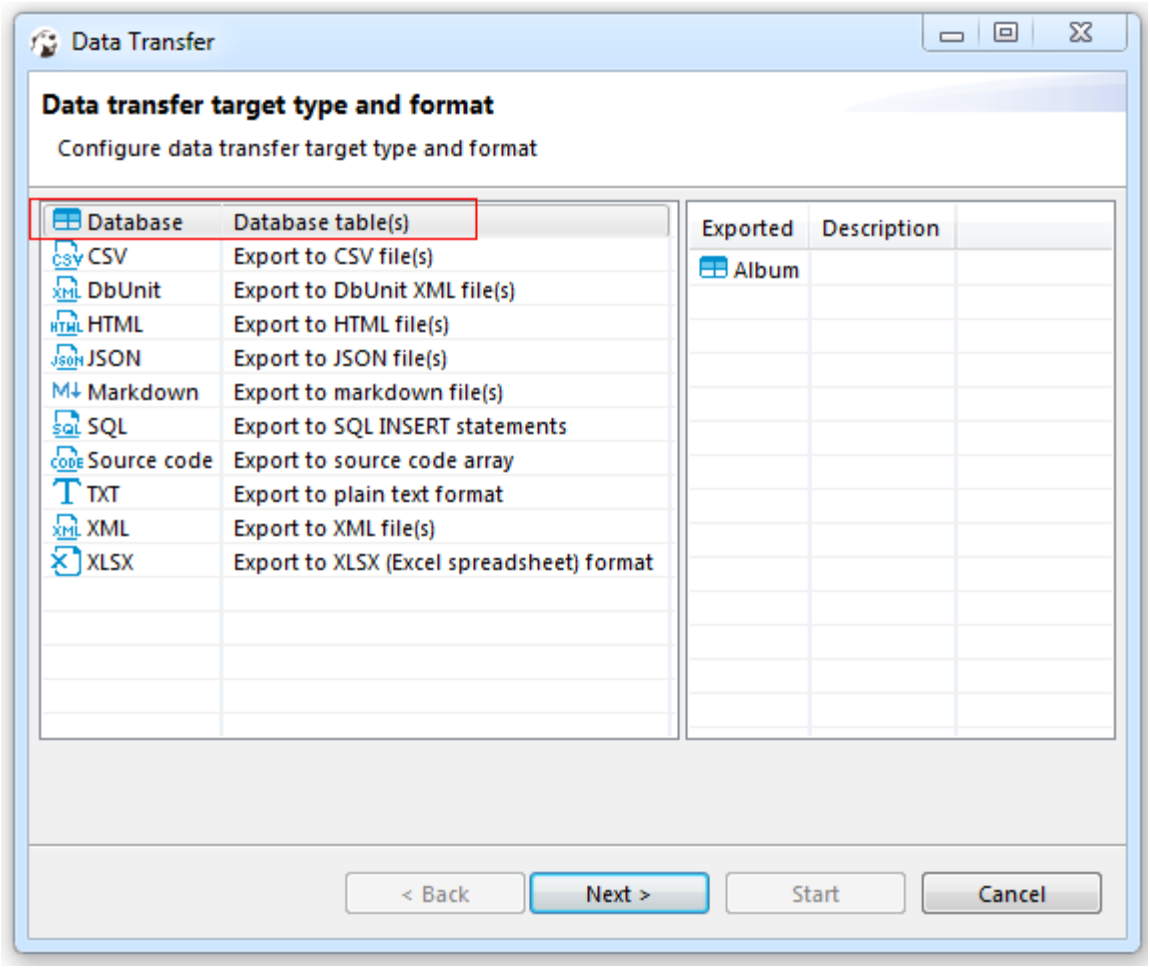
Step 1: Define the data source

In the **Database Navigator** select one or more tables you want to export. In the context menu choose **Export Data**. (Note: you also can export data from the custom SQL query results. For that, choose **Export data** in the results context menu).



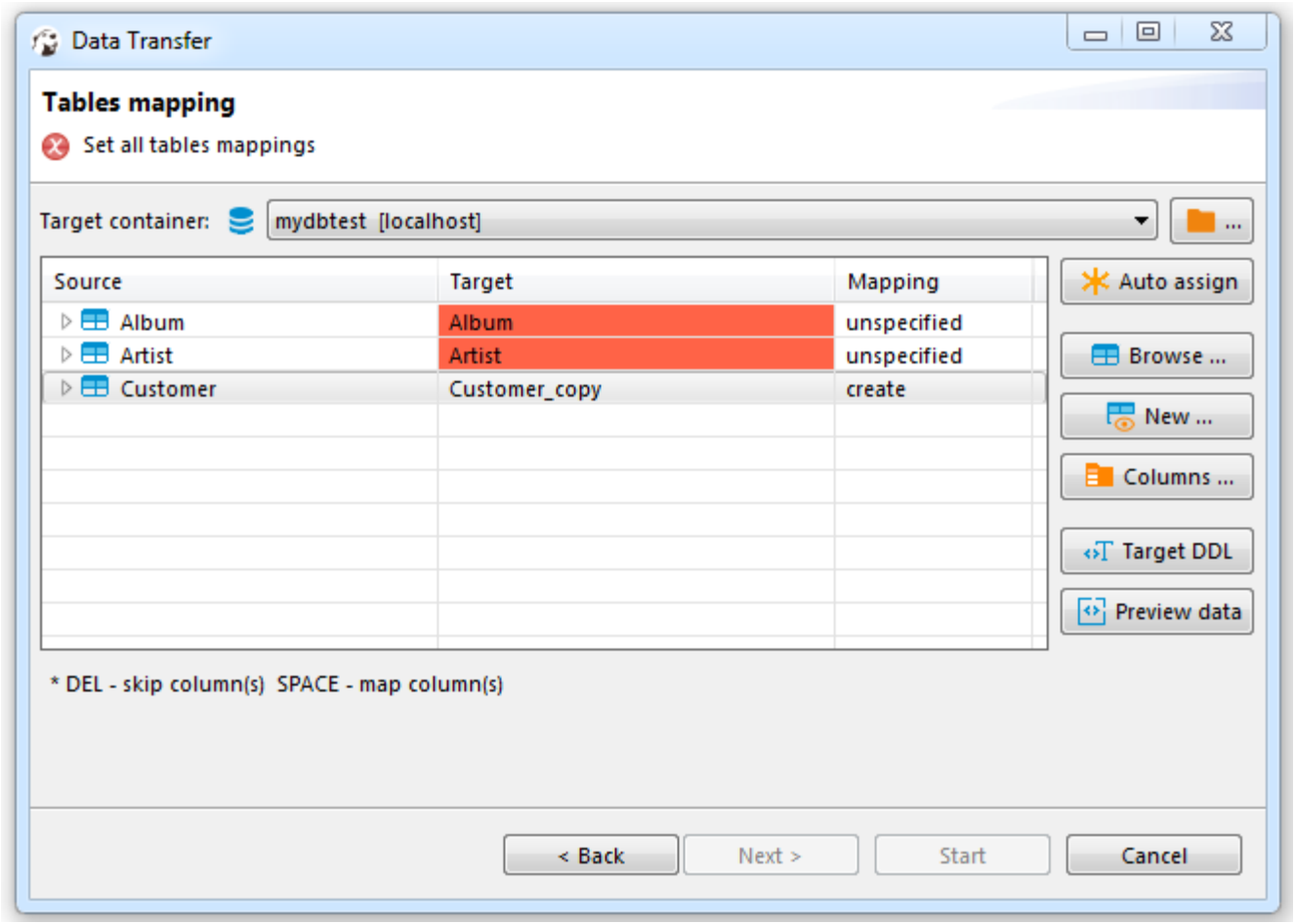
Step 2: Define data transfer target type

In the opened dialog box choose **Database** type as the data transfer target and press **Next**.

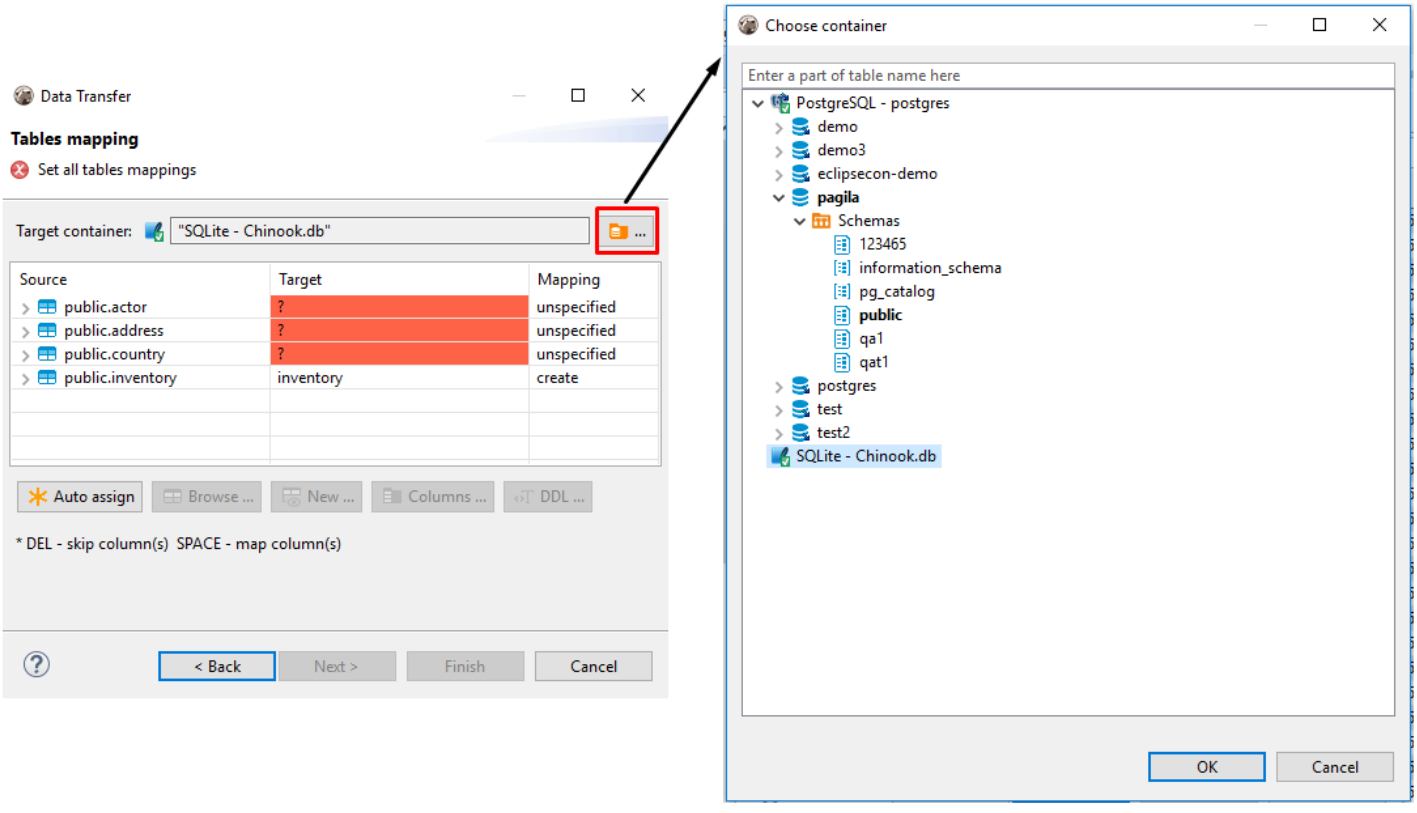


Step 3: Define data mapping

For proper table mapping, the following options are available:



- **Target container** - defines a database or a scheme where the data will be transferred to. Press ... button and choose the container.



- **Source** - contains names of all the tables selected at step 1. You can also see the list of columns existing in the source table by pressing 
- **Target** - contains names of the tables where the data will be transferred to.
- **Mapping** - contains the list of actions to be applied to the source data on data transfer. The following options are available:

  Create- the source data will be populated into a newly created table or column of the target container.

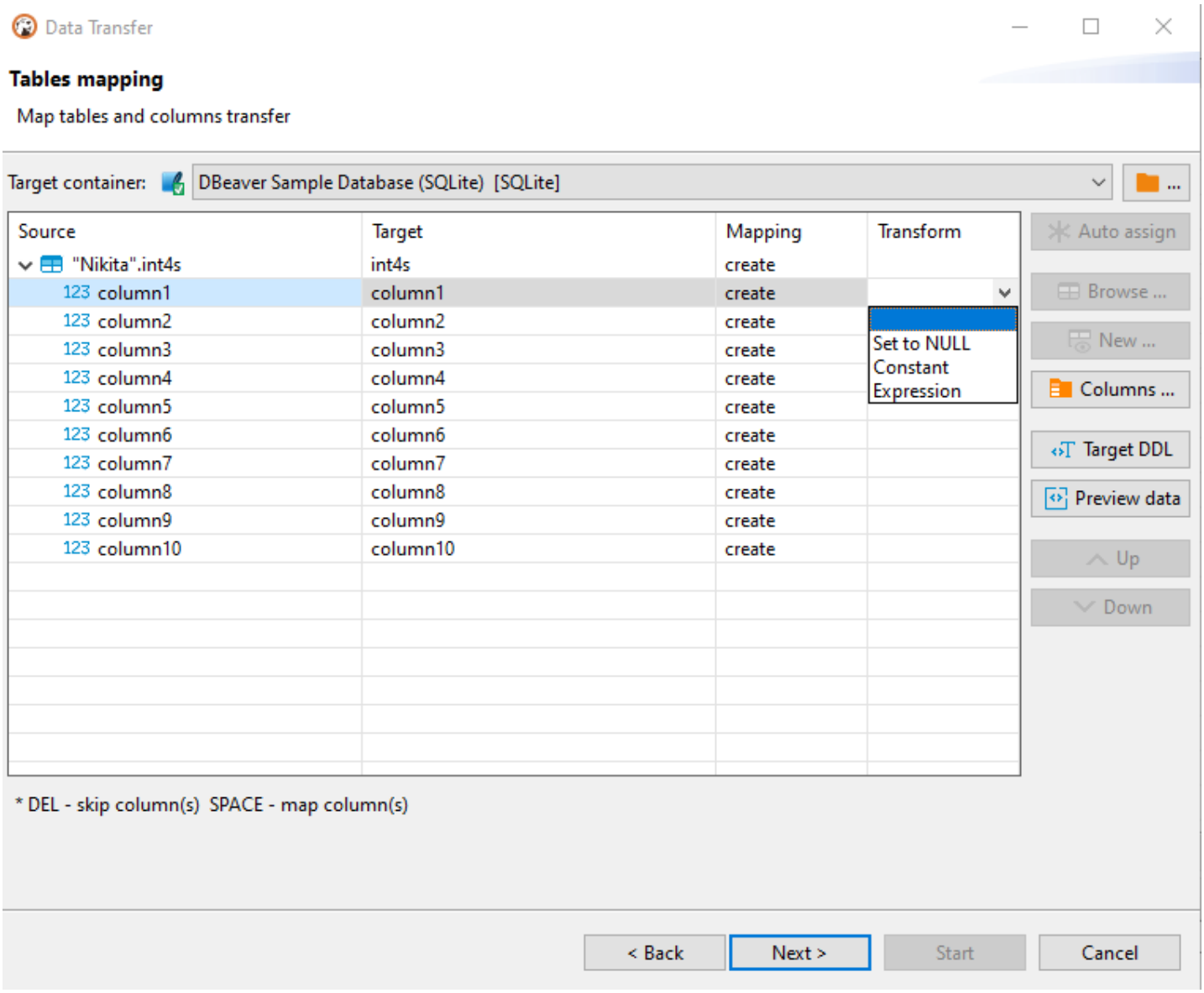
  Skip - the source data will not be transferred to the target container.

  Existing - the source data will be transferred to the table that already exists in the target container.

  Unassigned - this value is set by default when there is no target defined.

If the cells are marked with  , it means that in the target table there are no columns with matching names, otherwise the names will be filled in automatically.

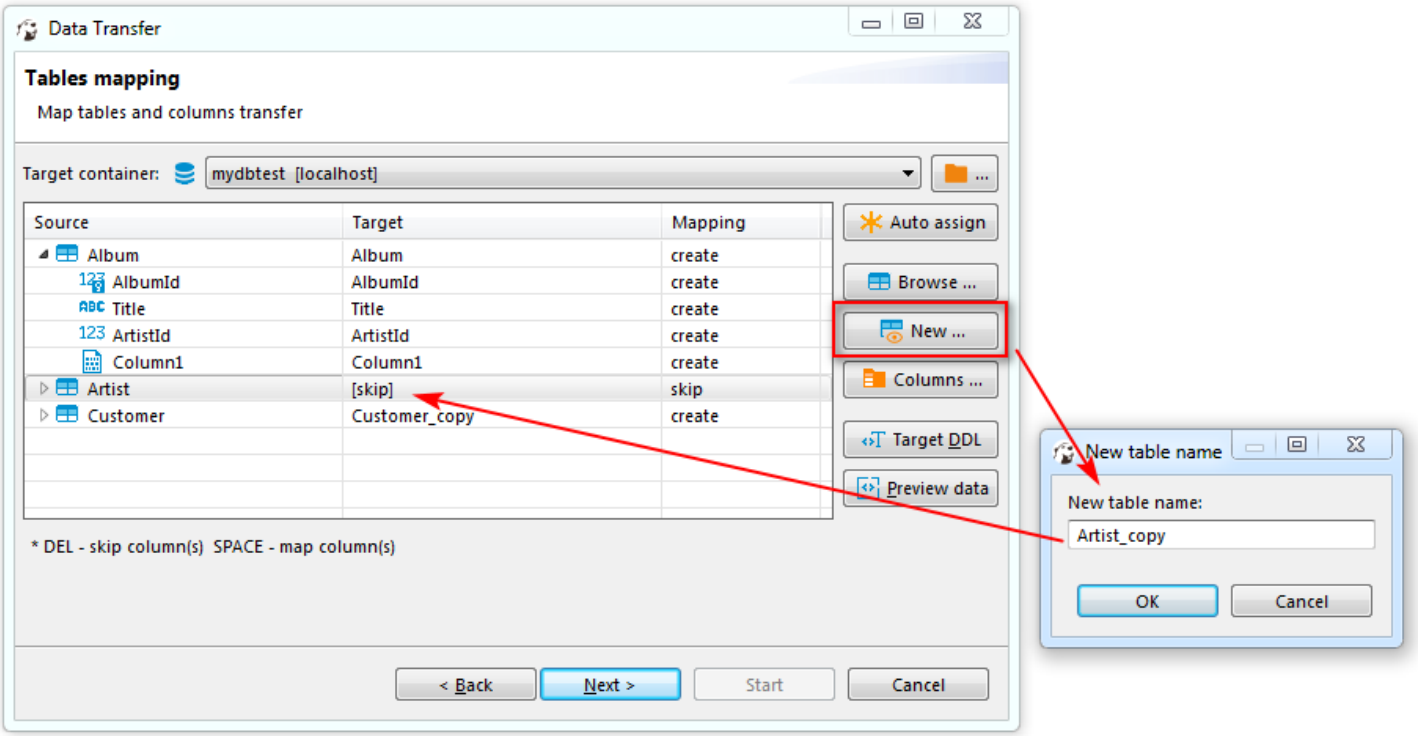
You may also want to transform the values of some columns during the transfer. To do that, define column transformers by clicking on corresponding cells in the *Transform* column. You can choose one of three options:



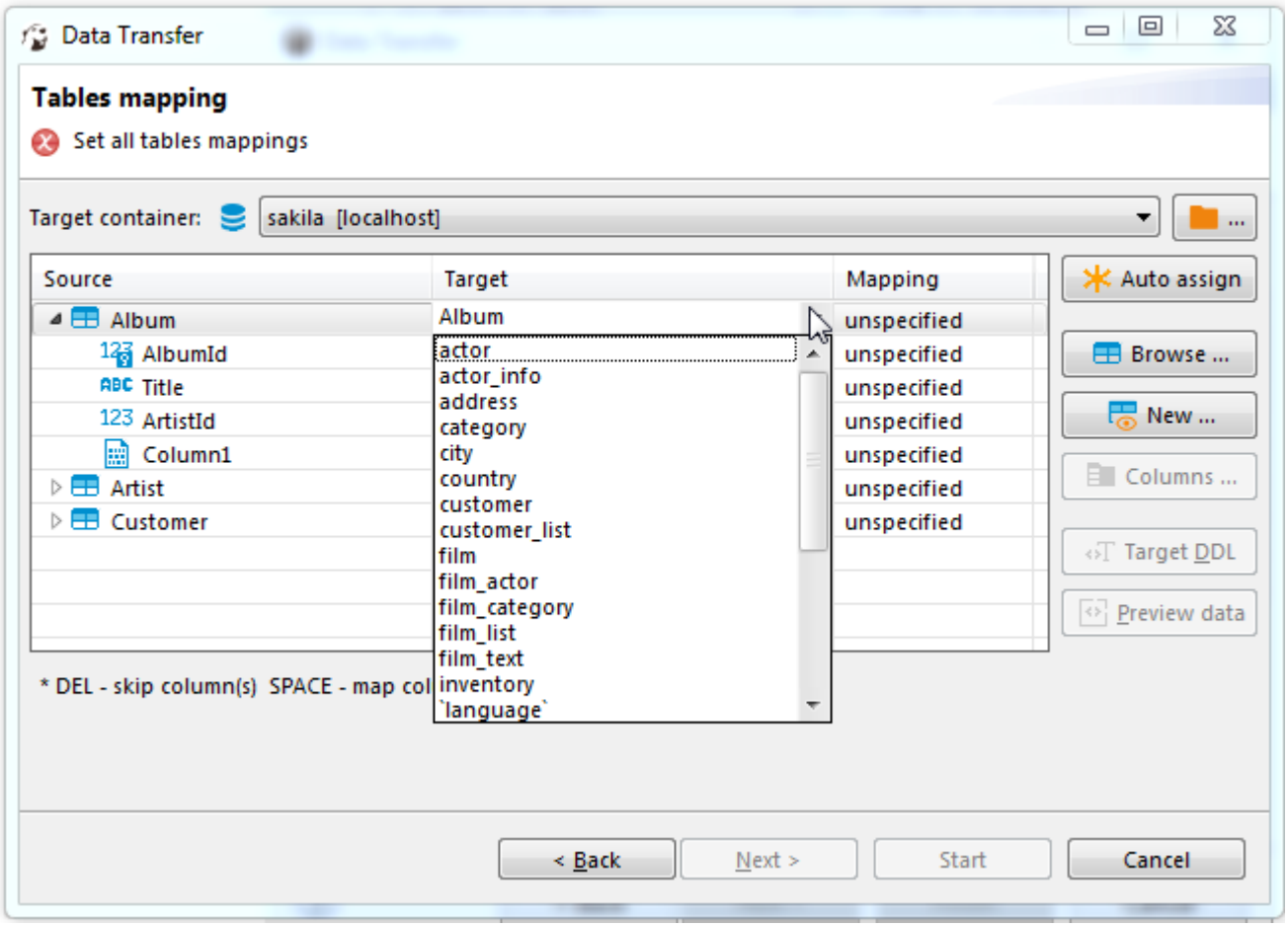
- *Set to NULL*. All values in the corresponding column will be set to null.
- *Constant.* All values in the corresponding column will be set to the specified constant.
- *Expression*. This transformer uses expressions (namely, JEXL expressions) to calculate the cell's value. You can use basic arithmetic operations and column names to construct an expression.

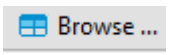
You can define a target table by clicking on a cell in the **Target** column and entering its name, or press the **New** button 

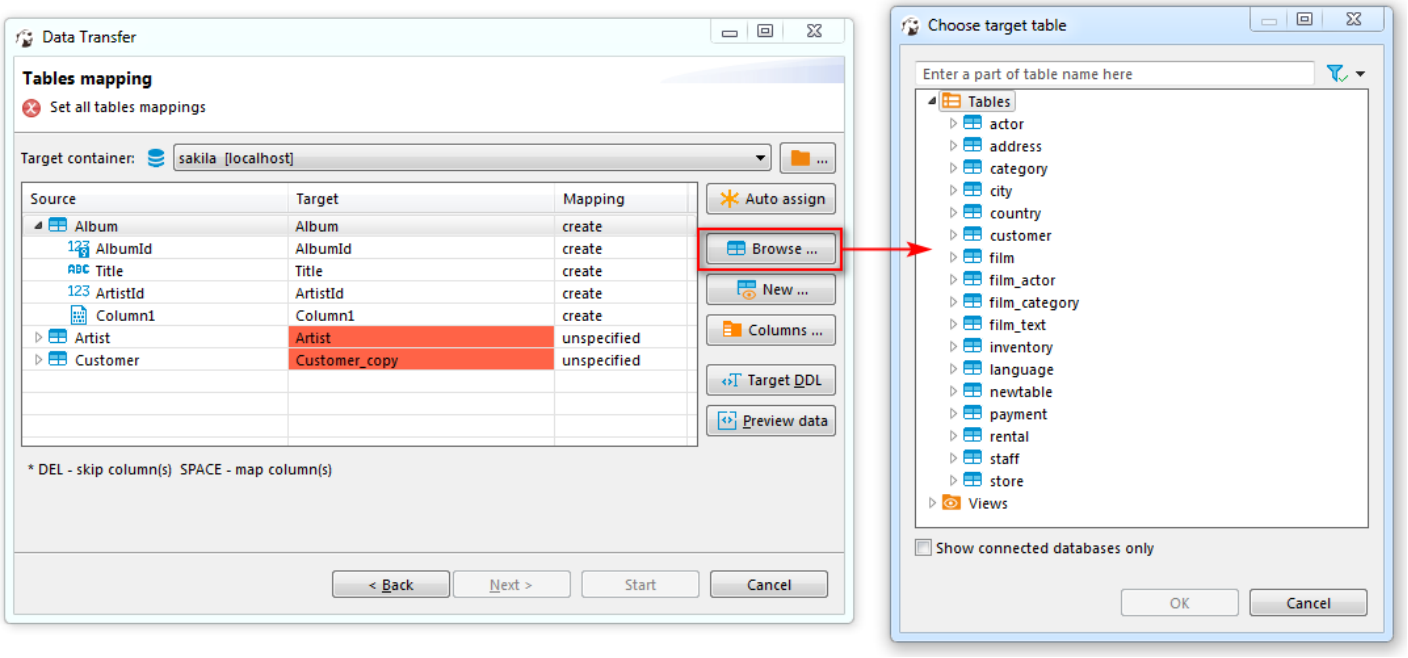
and enter a new name in the opened dialog box.



You can also choose a name for a target table from the drop-down list.



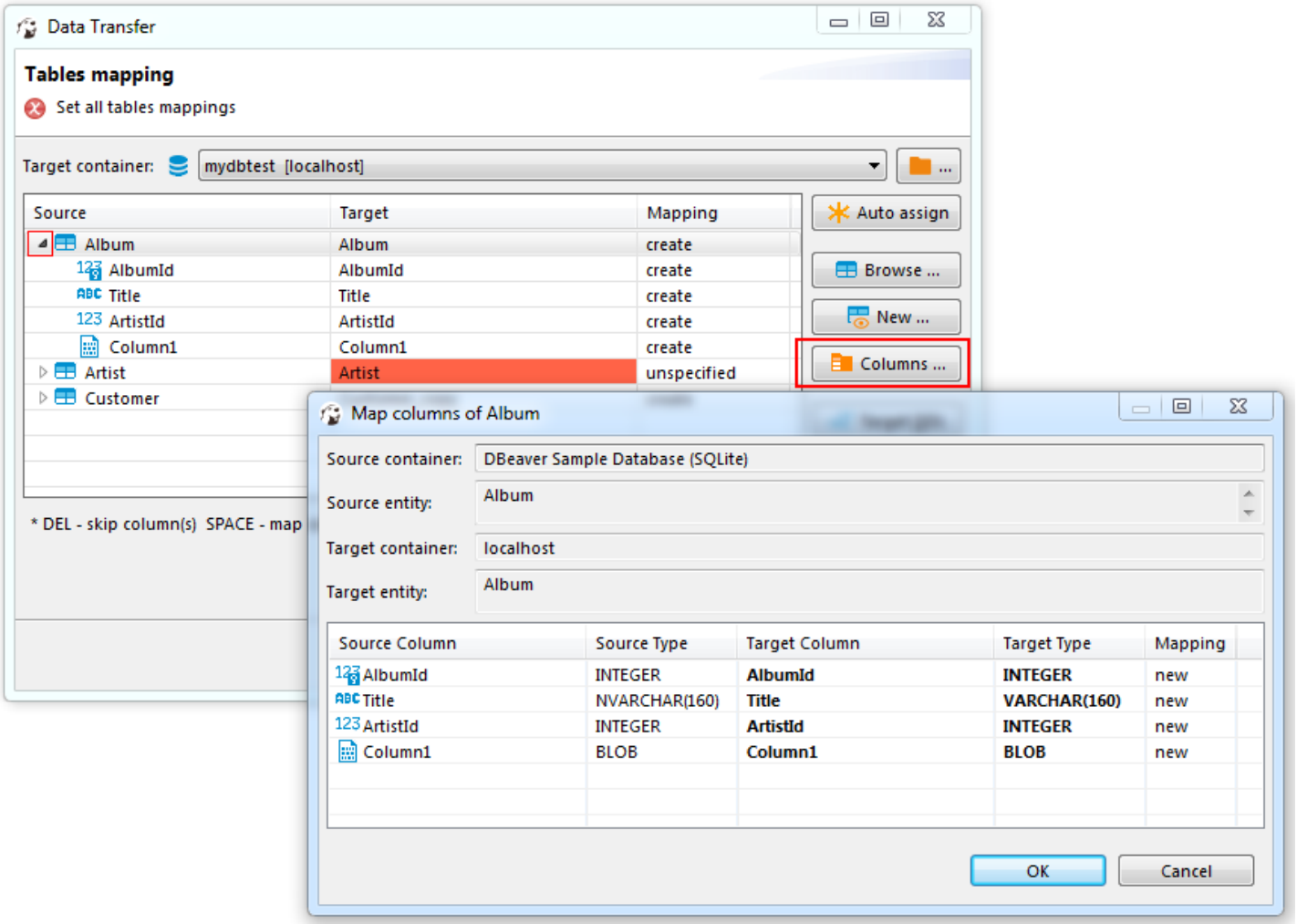
Or select a table from the existing tables in the target container by pressing the **Browse** button 



To define the mapping setting for a column in a target table, release the list of source table columns by pressing next to the table's name in the **Source** column, then click the name of the target column and enter a new one or select one from the dropdown list. To collapse the list, press

If you want tables of the target container to be named like those of source, press the **Auto assign** button Auto assign and the **Target** column will be automatically populated.

You can also define the names of target columns, as well as their data types, by clicking a row with a table name and pressing the **Columns** button Columns ...



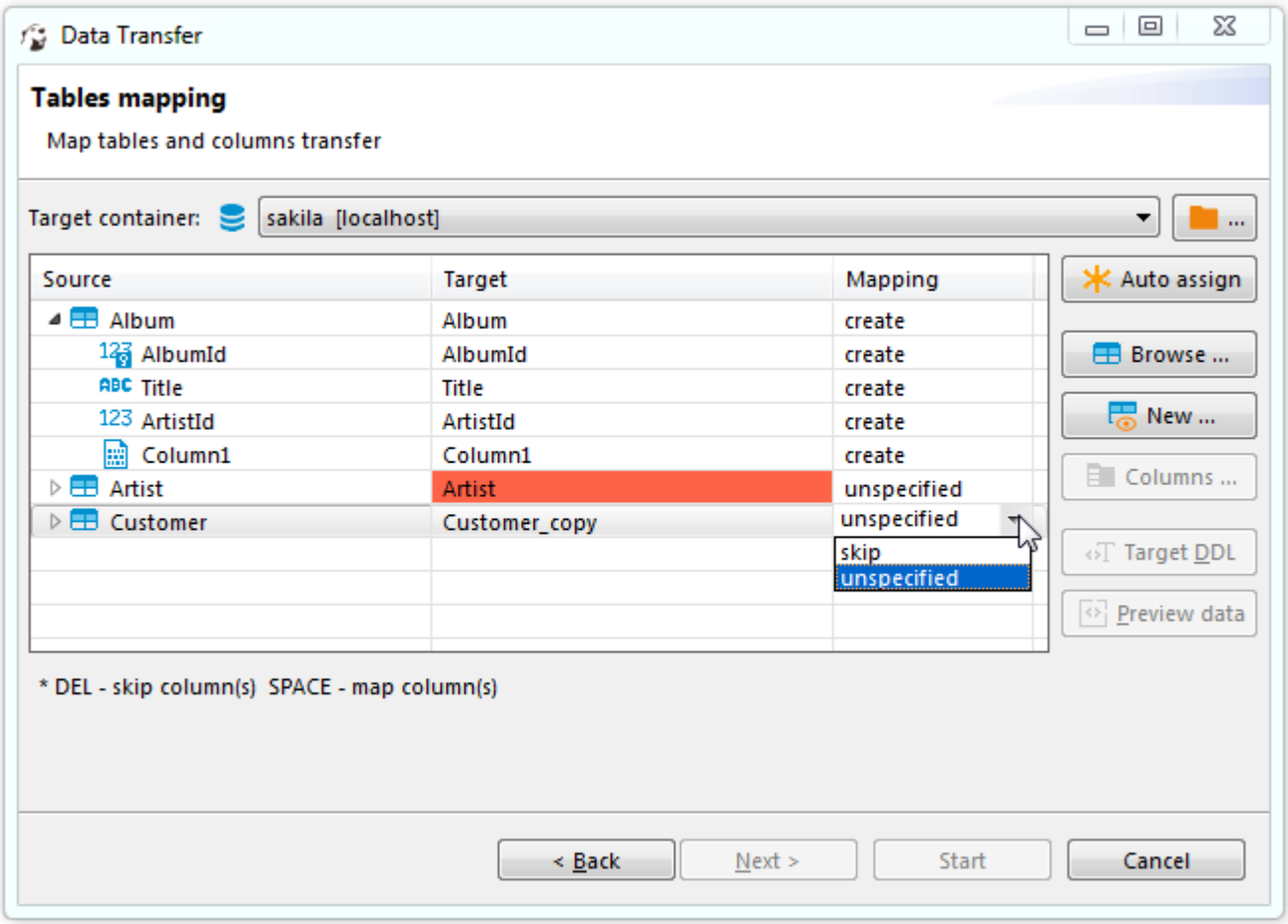
The following elements are available here:

- **Source column** -this column contains names of columns existing in the selected source table;
- **Source type** - this column contains the list of data types assigned to the columns in the selected source table;
- **Target column** - this column contains names of target table columns where the data from the source column will be transferred to. To change the name, click the cell and enter a new name.
- **Target type** - this column contains the list of data types that will be assigned to the columns in the target table.

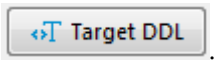
  IMPORTANT: Sometimes data types that are supported on the source database are not supported on the target or vice versa.

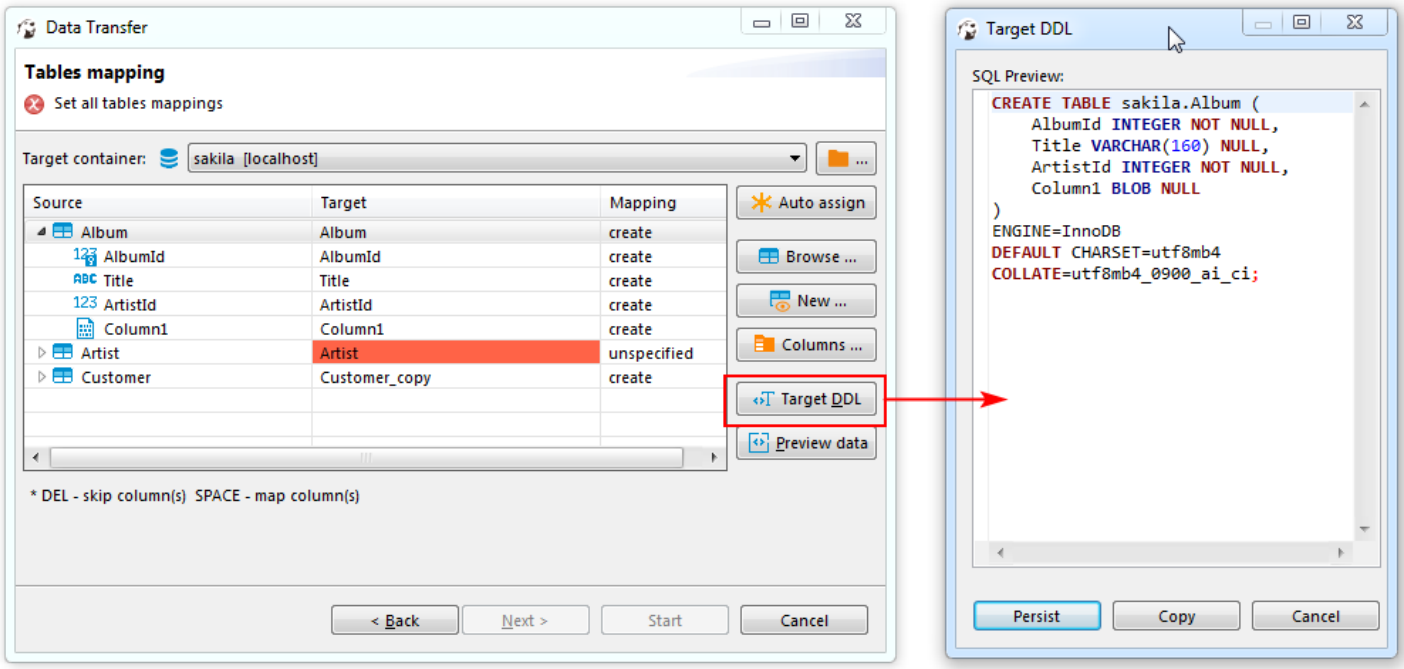
  To set a data type for a column in a target table, click the cell in the **Target Type** column and select one from the dropdown list of data types supported on the target.
- **Mapping** - this column contains the list of actions to be applied to the data on data transfer.

To change the mapping type, click a cell in the **Mapping** column of **Table mapping dialog box** and select the required mapping type.

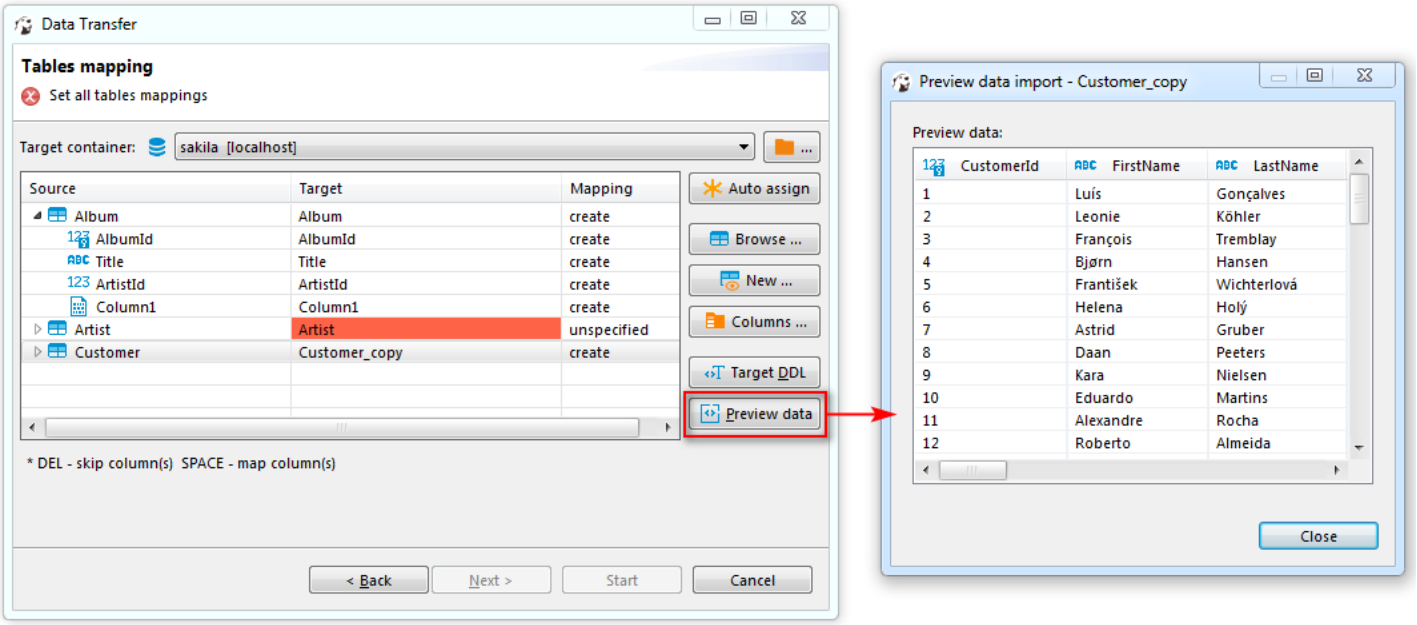


You can also view the SQL script that will be run on data transfer by pressing the **Target DDL** button





If you want to see a preview of the imported data, you can select the **Preview data** button



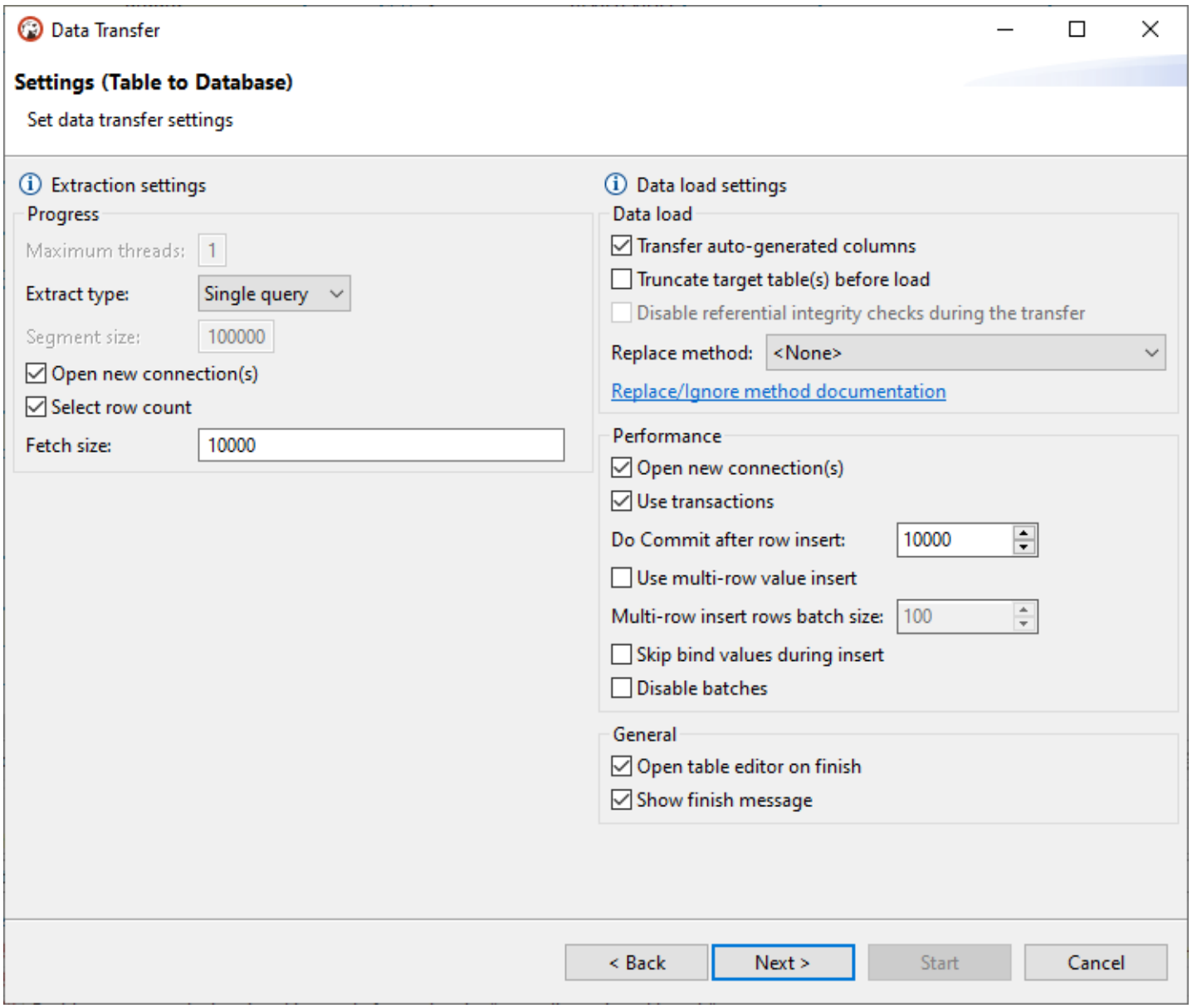
The following keyboard shortcuts for easy navigation within the mapping table area of **Table mapping** screen are supported:

Shortcut	Action
Up	Moves one row up.
Down	Moves one row down.
Right	Releases the list of source table columns.
Left	Swaps the list of source table columns.
Space	Auto-assigns the target.
Del	Sets mapping type to skip.

Configure data mapping and press **Next**.

Step 4: Define export settings

Data export settings are grouped into **Extraction settings** and **Data load settings**.



Extraction Settings

Extraction settings define how the data will be pulled from the source. The following options are available:

Option	Description
Maximum threads	Defines the number of threads to be used for data transfer.
Extract type	Select **Single query** option if your data load is not too big. Select **By segments** option if you need to migrate a solid amount of data. When this option is selected you can set the **Segment size** value, that is to define the number of rows to be transferred in each segment.
Open new connection	If selected, a new connection will be opened and the data transfer will not interfere with other calls to the

(s)	database whose data is being transferred.
Select row count	If selected, a progress bar displaying data migration process will be shown.
Fetch size	Specifies the number of rows to fetch per server round trip. This setting has a significant impact on the extraction performance during data retrieval.

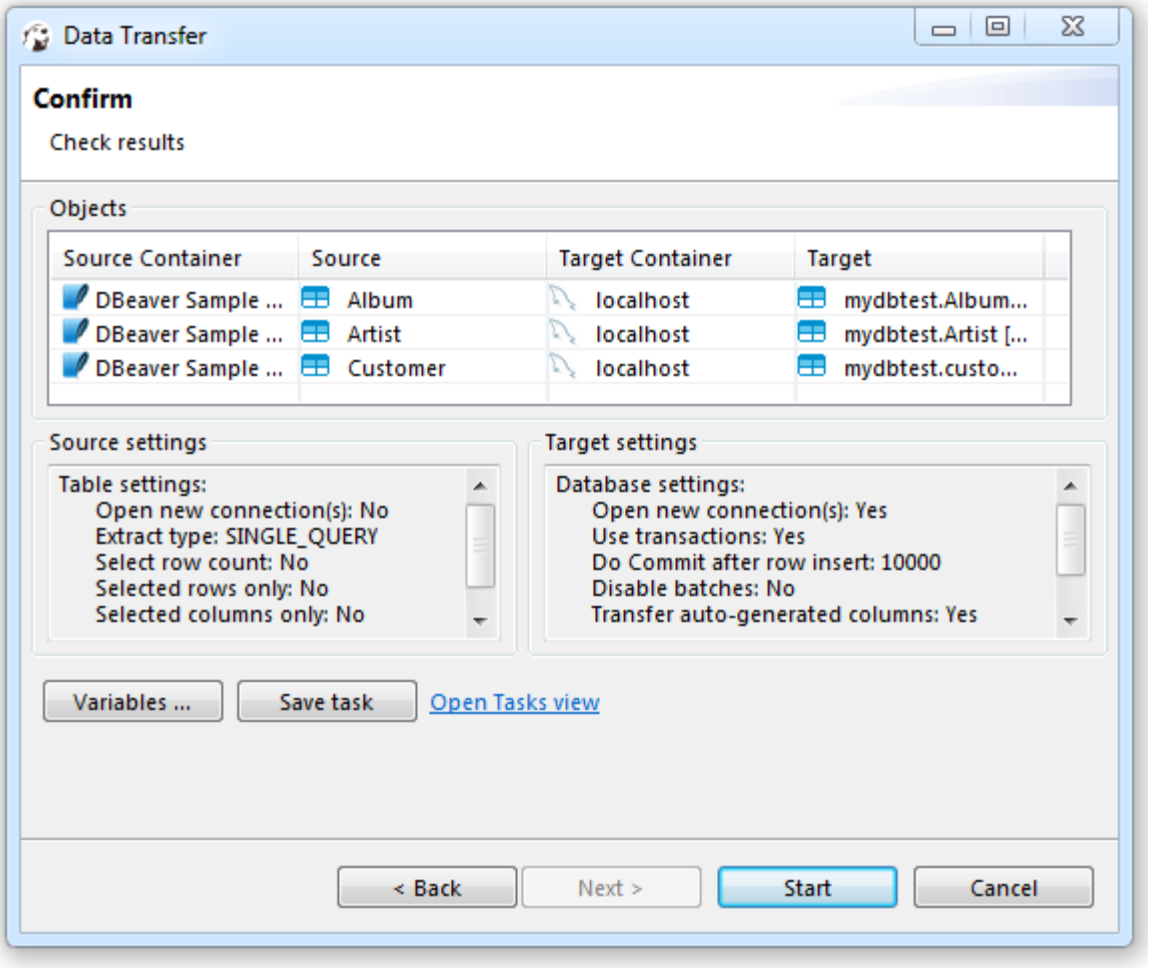
Data load settings

Data load settings define how the extracted data will be pushed to the target. The following options are available.

Option	Description
Transfer auto-generated columns	Fill in or skip columns marked with the "autogenerated" status. Some databases accept values in such columns, while others will throw a syntax error.
Truncate target table(s) before load	Select this checkbox only if you want all the data to be cleared from the target table. Be very careful with this option!
Disable referential integrity checks during the transfer	Disabling constraints in the target table. This setting prevents database errors by temporarily disabling the constraints. However, please note that not all databases support this functionality.
Replace method	Read our guide on Data Import and Replace to learn more about the replacing method option.
Open new connection(s)	Use this option to speed up data transfer. If selected, a new connection will be opened and the data transfer will not interfere with other calls to the database where data is being transferred to.
Use transactions	This option allows you to speed up the data transfer and to define the number of rows for each transaction by setting the **Commit after insert of** parameter.
Do Commit after row insert	Performing a commit after a certain number of inserted rows. This setting specifies that a commit operation should be executed after a specified number of rows have been inserted into the table.
Use multi-row value insert	Use multi-row insert with extended values number for higher performance. Database-specific setting.
Skip bind values during insert	This option can drastically increase performance for some drivers like Redshift by skipping a process of binding values and setting them directly, but it opens up a vulnerability to SQL injections. Not recommended if you are not sure of imported file contents.

Disable batches	Select this checkbox if you want to disable the use of batch imports. The import will be made row by row. Enabling this function will show all import errors, but make the import process slower.
Ignore duplicate rows errors	In the import process, if a database encounters a duplicate key from the import row in the target table, such errors are ignored, and the import operation continues without failure.
Use bulk load	Bypasses transaction settings and loads the entire dataset using the native tool provided by the database.
Open table editor on finish	If selected, the table editor is to be opened when data transfer is finished.
Show finish message	If selected, a notification message will be shown when the transfer is finished.
Save task	Opens the Save Task window to assist in creating a task during the data transfer process. This window provides options and settings for creating and configuring a task related to the data transfer operation.
Send results by E-Mail	Sends data transfer results by E-Mail on finish. Read our guide to using email to transfer data to learn more.

Step 5: Confirm



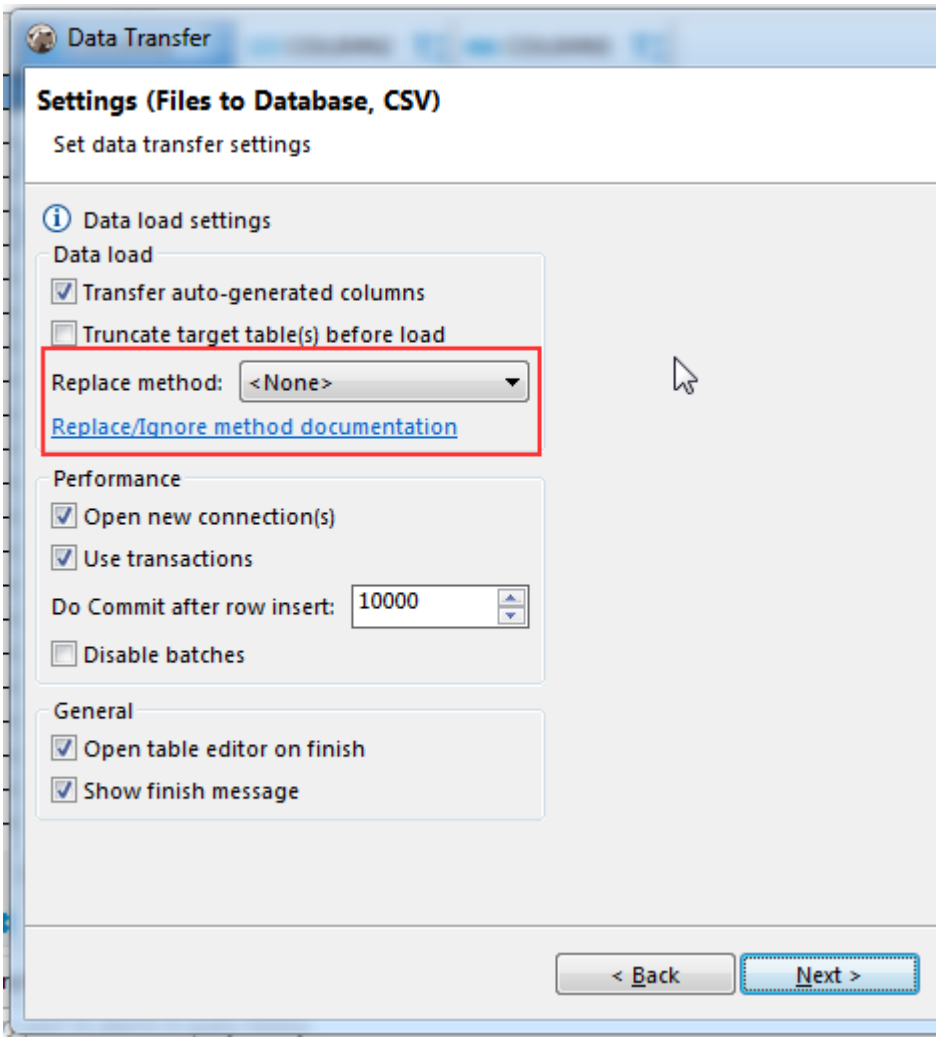
Check out the data transfer settings and press **Start** or save as task.

Data Import and Replace

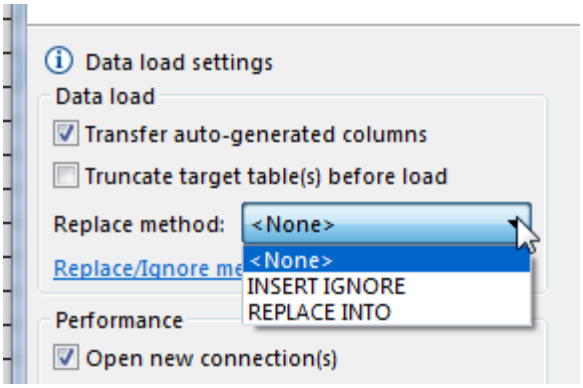
Data Import and Replace.

Sometimes there are situations when you want to ignore the current primary key value when importing into a table. Some databases have syntax constructs in addition to the INSERT INTO that may help.

The choice of the replacement method is in the import settings - in "Data load settings".



By default, the selection is `<None>`, you can select other options from the drop-down list. The options available depend on the target database you are importing to.



The database can only support the replace method or the ignore method. In this case, the list of methods will consist of only one item except `<None` >. If the base does not support the replacement methods, or if we have not added an implementation yet, then the combo with the list will be disabled.

Further, you will find a list of databases supporting these methods and examples of syntax.

Let's take a look at an example of how this works. We use a small, simple, slightly-modified Sakila (MySQL) table - sakila.language

```
CREATE TABLE language_insert (
language_id tinyint unsigned NOT NULL,
name char(20) NOT NULL,
last_update timestamp NOT NULL,
PRIMARY KEY (language_id)
);
```

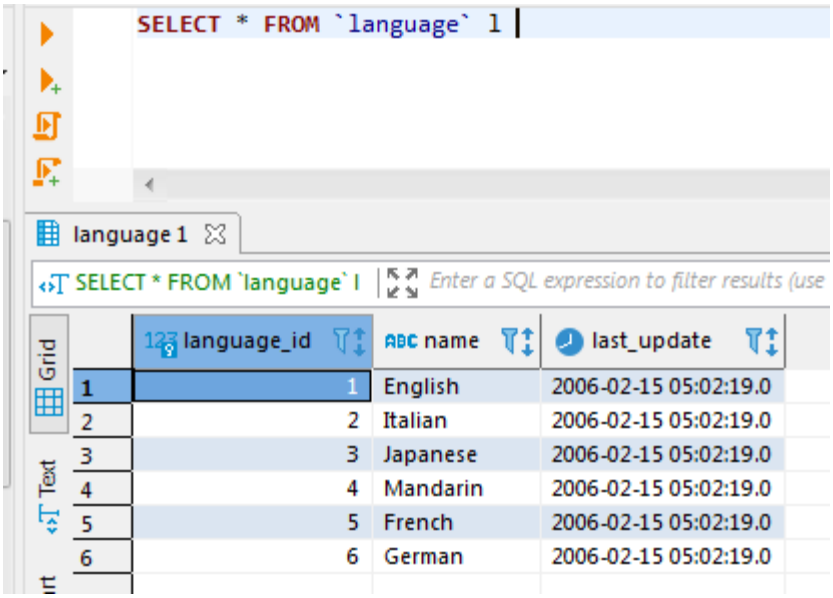
```
INSERT INTO sakila.language_insert (language_id,name,last_update) VALUES
(1,'English','2006-02-15 05:02:19.0'),
(2,'Italian','2006-02-15 05:02:19.0'),
(3,'Japanese','2006-02-15 05:02:19.0');
```

If we try to execute this request twice, we will get the following error: SQL Error [1062] [23000]: Duplicate entry '1' for key 'language_insert.PRIMARY' (This message may look different in other databases).

Let's take a new .csv file with the following content and try to use the replace methods.

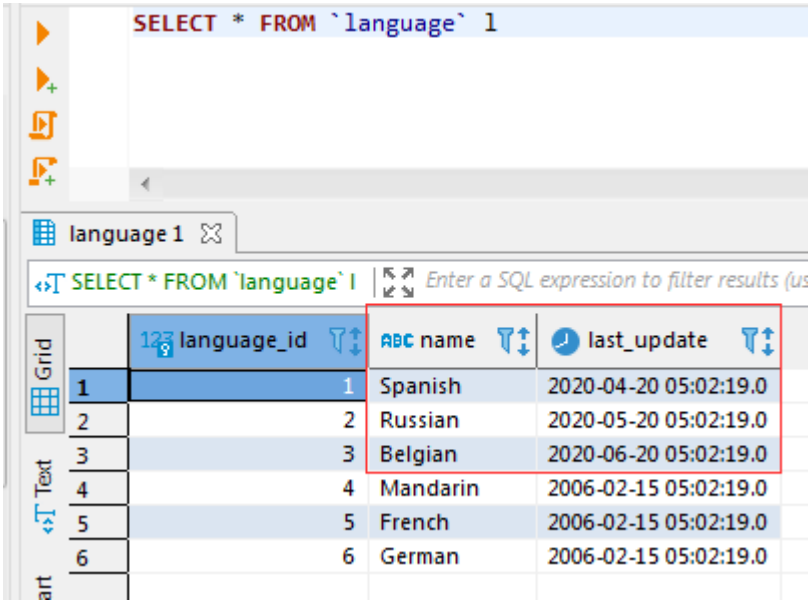
```
"language_id","name","last_update"
1,Spanish,"2020-04-20 05:02:19.0"
2,Russian,"2020-05-20 05:02:19.0"
3,Belgian,"2020-06-20 05:02:19.0"
4,Mandarin,"2006-02-15 05:02:19.0"
5,French,"2006-02-15 05:02:19.0"
6,German,"2006-02-15 05:02:19.0"
```

If we set the "INSERT IGNORE" method in the settings, the result of the insert will look like this:



There will be no insertion errors, the first three lines will not change, and the fourth to sixth lines will be added to the table.

If we set the "REPLACE INTO" method in the settings, the result of the insert will look like this:



There will be no insertion errors, the first three lines will be replaced and the fourth to sixth lines will be added to the table.

Which databases support replace/insert methods?

MySQL

INSERT IGNORE and REPLACE INTO

Insert examples:

"INSERT IGNORE"

```
INSERT IGNORE INTO language_insert(language_id, name, last_update)
VALUES(1, 'English', '2006-02-15 05:02:19.0');
```

"REPLACE INTO"

```
REPLACE INTO language_insert(language_id, name, last_update)
VALUES(1, 'English', '2006-02-15 05:02:19.0');
```

SQLite

Documentation

"INSERT OR IGNORE" and "INSERT OR REPLACE"

Insert examples:

"INSERT OR IGNORE"

```
INSERT OR IGNORE INTO language_insert(language_id, name, last_update)
VALUES(1, 'English', '2006-02-15 05:02:19.0');
```

"INSERT OR REPLACE"

```
INSERT OR REPLACE INTO language_insert(language_id, name, last_update)
VALUES(1, 'English', '2006-02-15 05:02:19.0');
```

PostgreSQL

Available for PostgreSQL version 9.5.

"ON CONFLICT DO NOTHING" and "ON CONFLICT DO UPDATE SET"

Insert examples:

"ON CONFLICT DO NOTHING"

```
INSERT INTO language_insert(language_id, name, last_update)
VALUES(1, 'English', '2006-02-15 05:02:19.0') ON CONFLICT DO NOTHING;
```

"ON CONFLICT DO UPDATE SET"

```
INSERT INTO language_insert(language_id, name, last_update)
VALUES(1, 'English', '2006-02-15 05:02:19.0')

ON CONFLICT (language_id)
DO UPDATE SET (language_id, name, last_update) = (EXCLUDED.language_id, EXCLUDED.name, EXCLUDED.last_updat
```

FireBird

Available for FireBird version 2.1.

"UPDATE OR INSERT INTO"

Insert examples:

"UPDATE OR INSERT INTO"

```
UPDATE OR INSERT INTO language_insert(language_id, name, last_update)
VALUES(1, 'English', '2006-02-15 05:02:19.0');
```

Oracle

Available for Oracle version 11.2.

"INSERT IGNORE ROW INDEX"

Insert examples:

"INSERT IGNORE ROW INDEX"

```
INSERT /*+ IGNORE_ROW_ON_DUPKEY_INDEX(LANGUAGE_INSERT, LANGUAGE_INSERT_PK) */
INTO LANGUAGE_INSERT(LANGUAGE_ID, NAME, LAST_UPDATE) VALUES(1, 'English', TIMESTAMP '2006-02-15 05:02:19.0
```

Database backup/restore

Note: This feature is available in Enterprise, Ultimate and Team editions only.

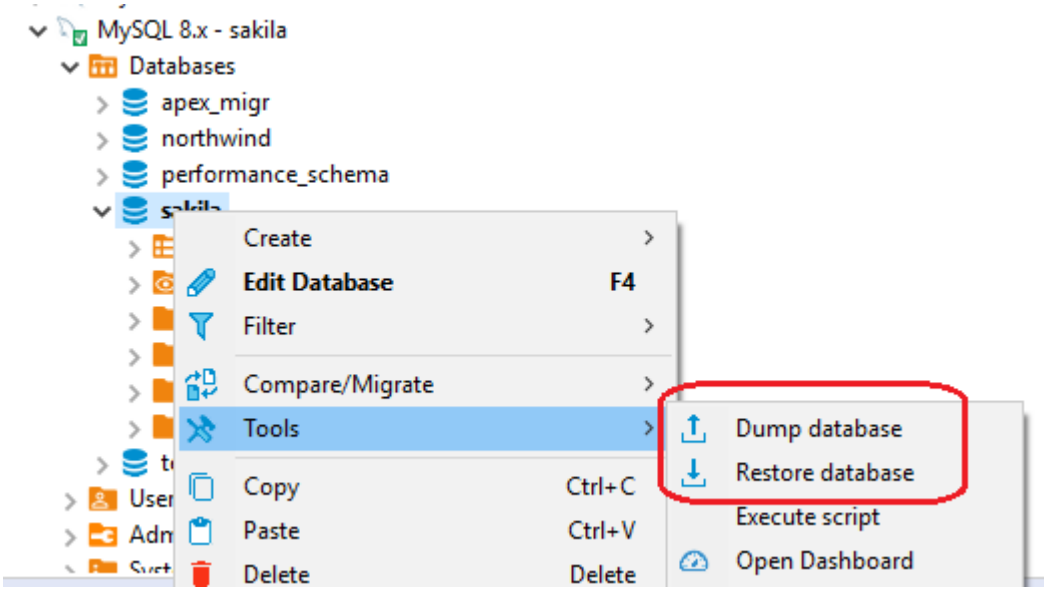
Database Backup/restore

DBeaver supports native database backup/restore functions for the following databases:

- PostgreSQL
- MySQL

The native backup restore differs from the standard DBeaver data transfer feature. It uses database native dump formats and it may work much faster as it uses special utilities for the direct high-performance database access.

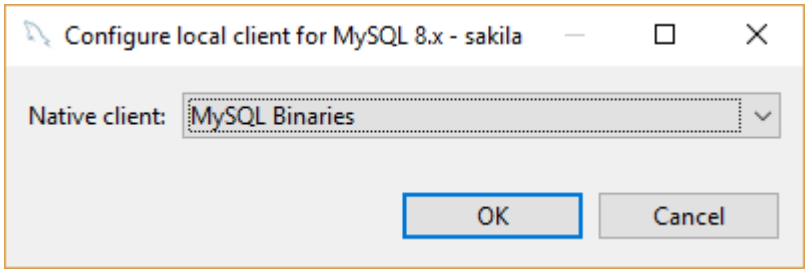
These functions can be accessed from the Context Menu's Tools or the Main Menu's Database->Tools.



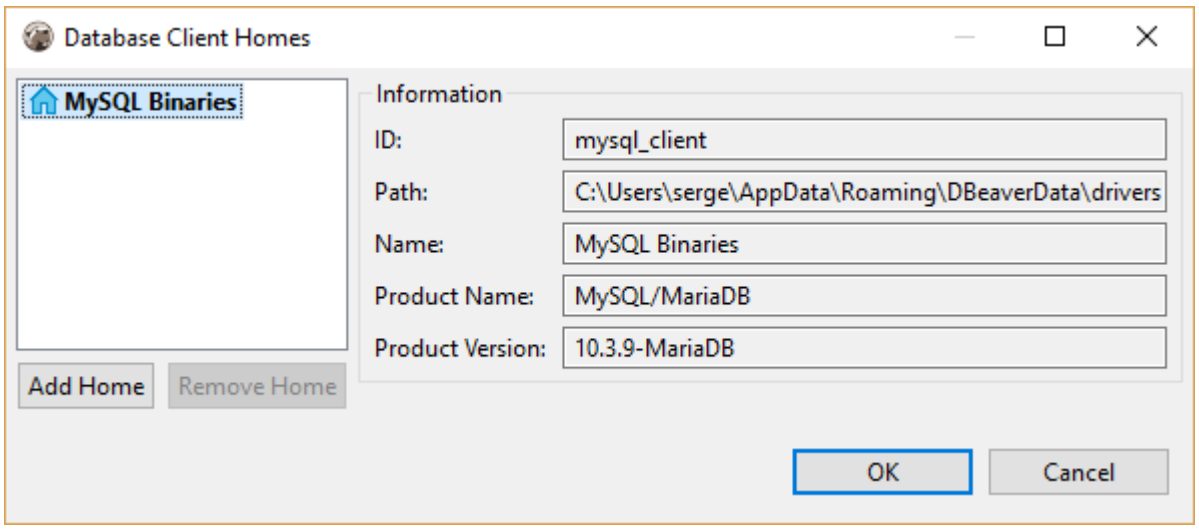
Native client configuration

In order to execute native backup/restore tools you need to configure the database native client. The native client is a set of binaries (different for different OSes) which will be executed by DBeaver to process an actual backup

/restore. The native client configuration can be done in driver editor dialog or directly from the backup/restore wizard. Just click on the `Client ...` button in the button bar:

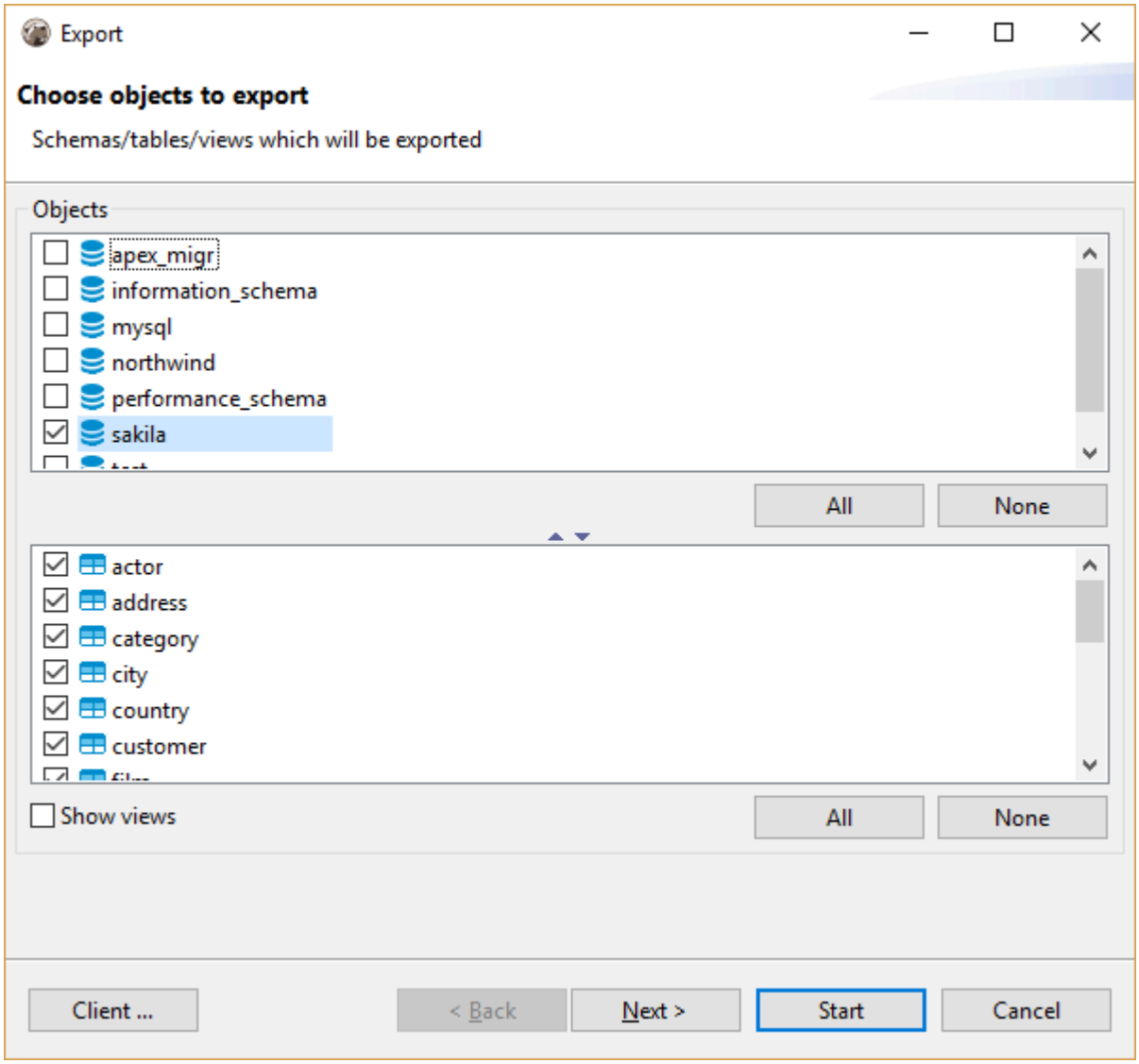


To configure a new client location, choose the `Browse ...` item and add a new client in the following dialog:



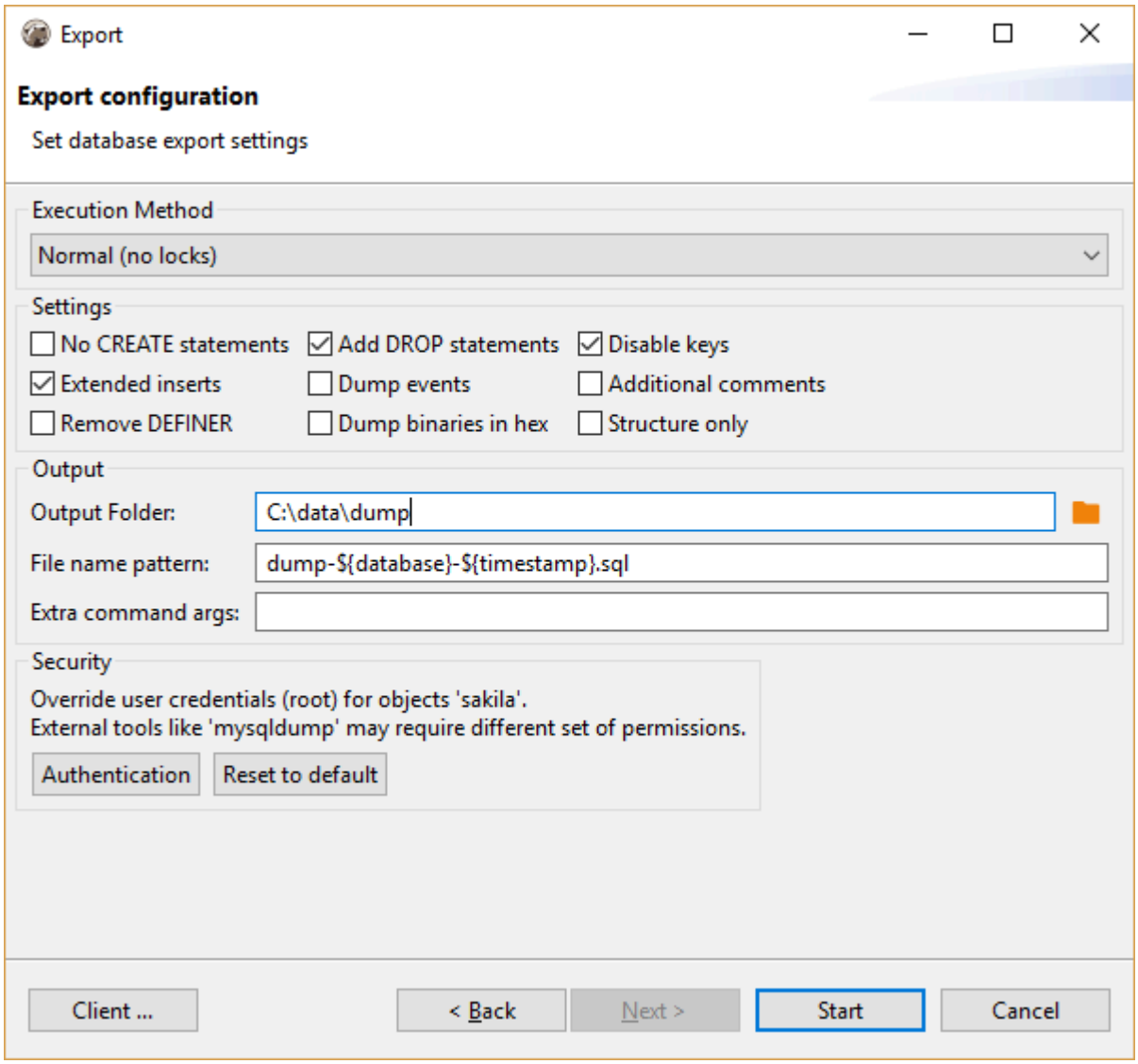
Database dump object selector

You can choose what schemas/tables you want to backup/dump:



Database native tool configuration

You can pass a set of additional dump/restore parameters to the native tool. The particular set of configuration options depends on the database type.



Task management

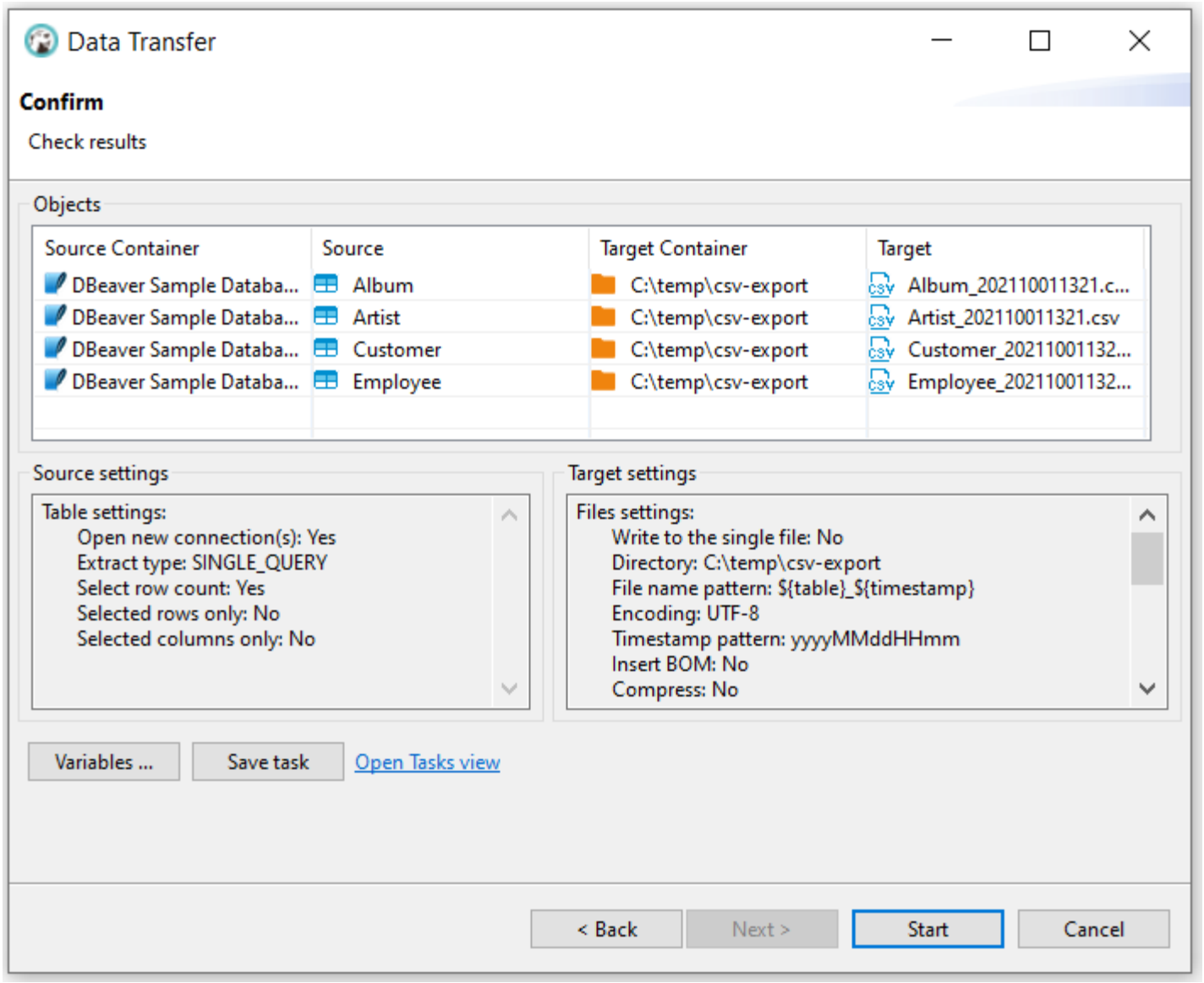
Note: This feature is available in Enterprise, Ultimate and Team editions only.

Creating tasks

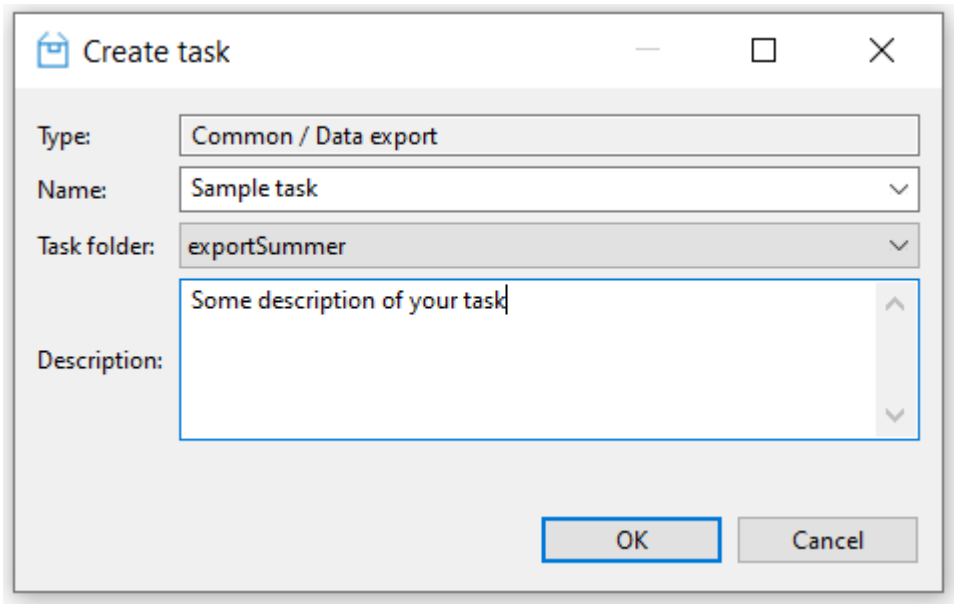
The task is a saved configuration of a database tool. It can be started from the task management view or from the menu by a single click. You can create tasks for frequently used tools. Also, tasks can be scheduled for regular execution.

Create a task from tool configuration

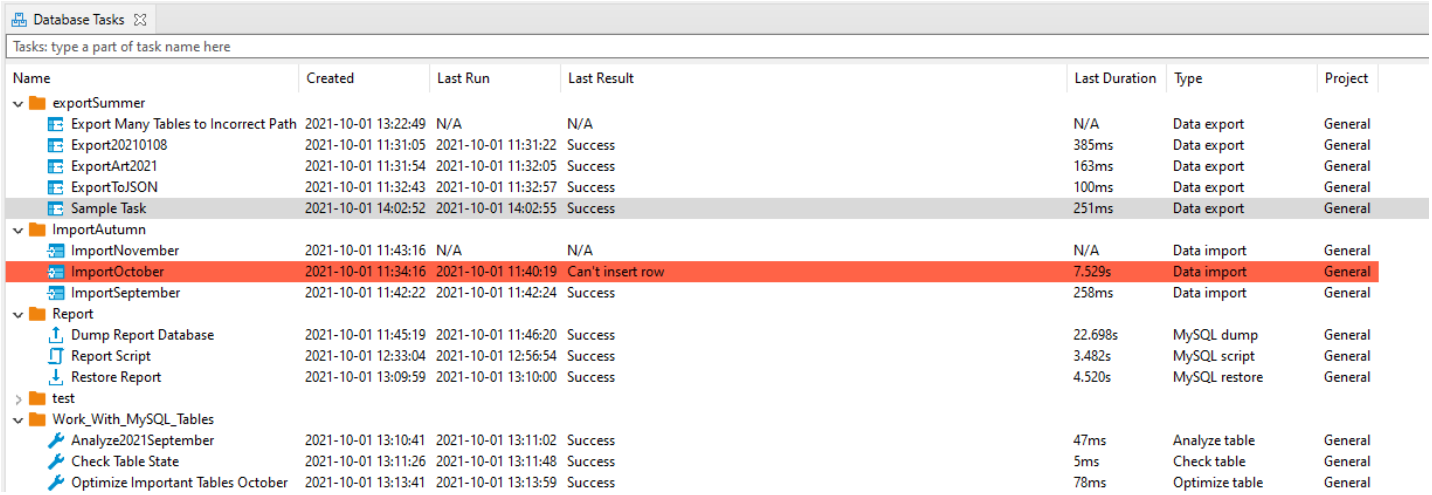
You can save the tool configuration into a task and run your task later with a single click. For example, you can start the Data Transfer wizard and configure the data export from several tables in the SQLite database into CSV files:



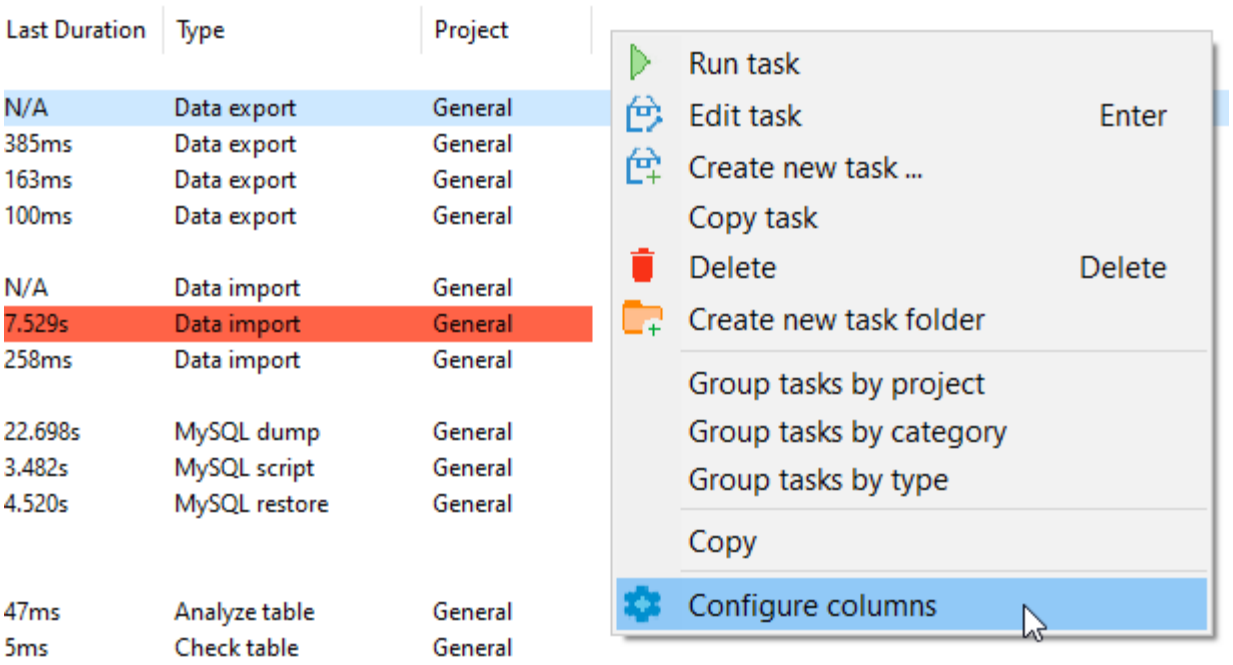
Click on the `Save configuration as task` button and fill the task properties:

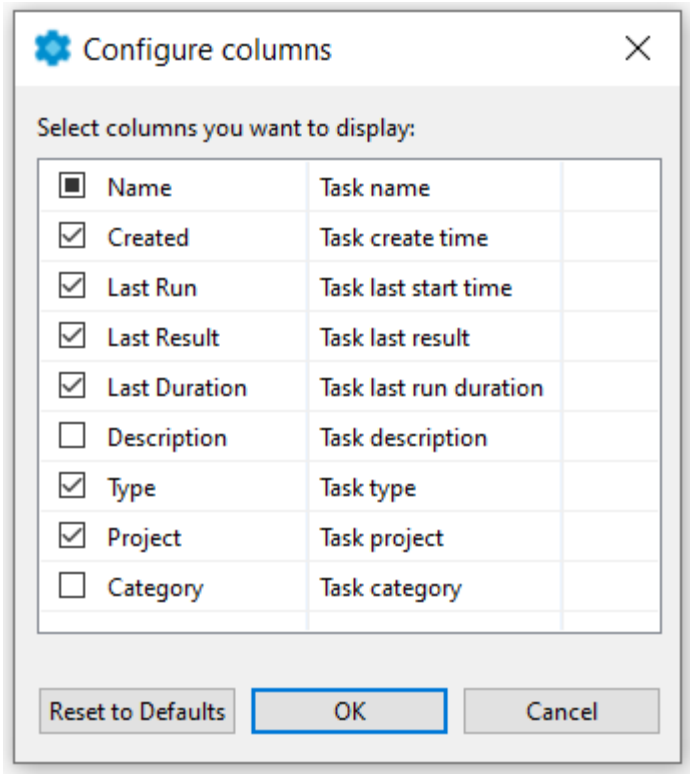


Now click on the `Open Tasks view` link to open the task list:

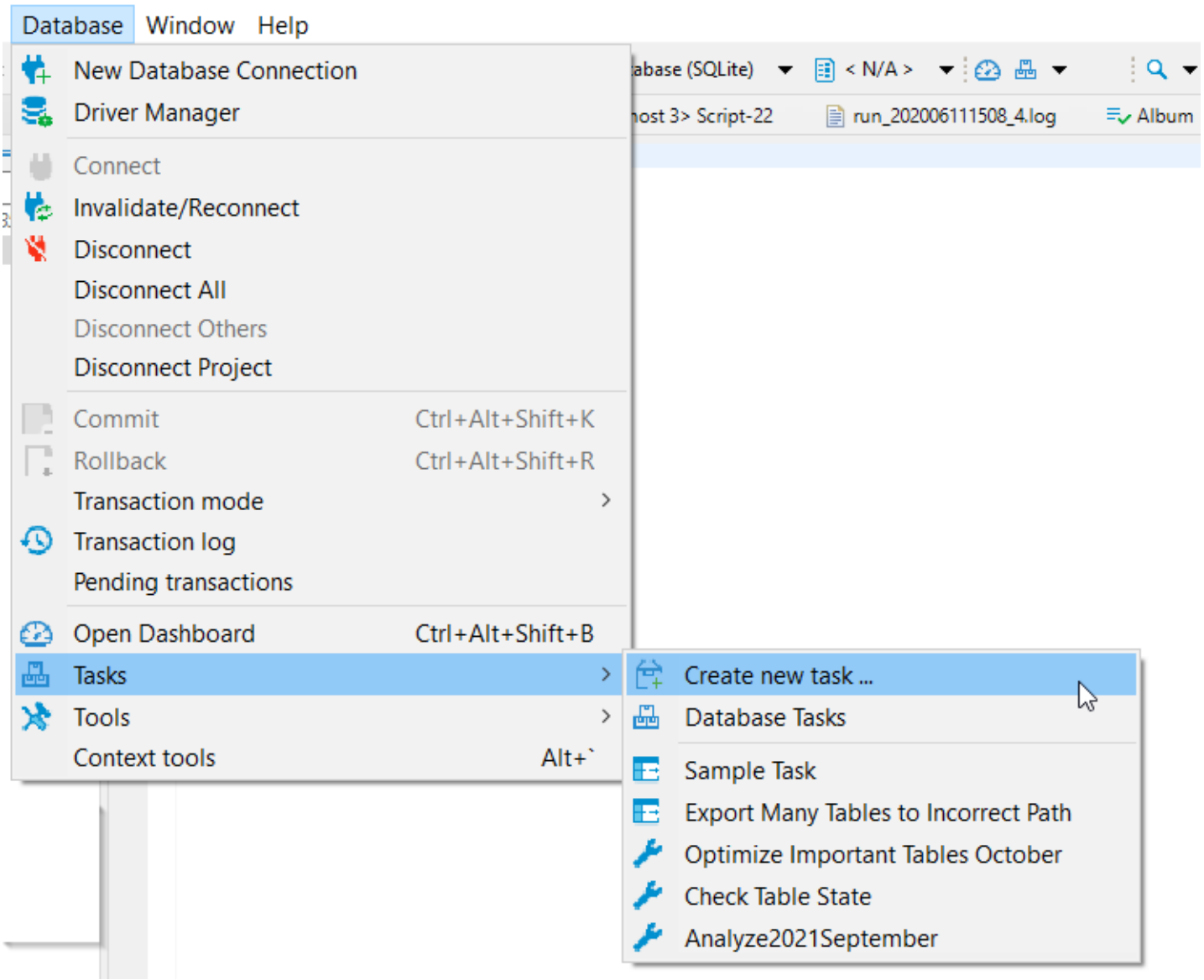


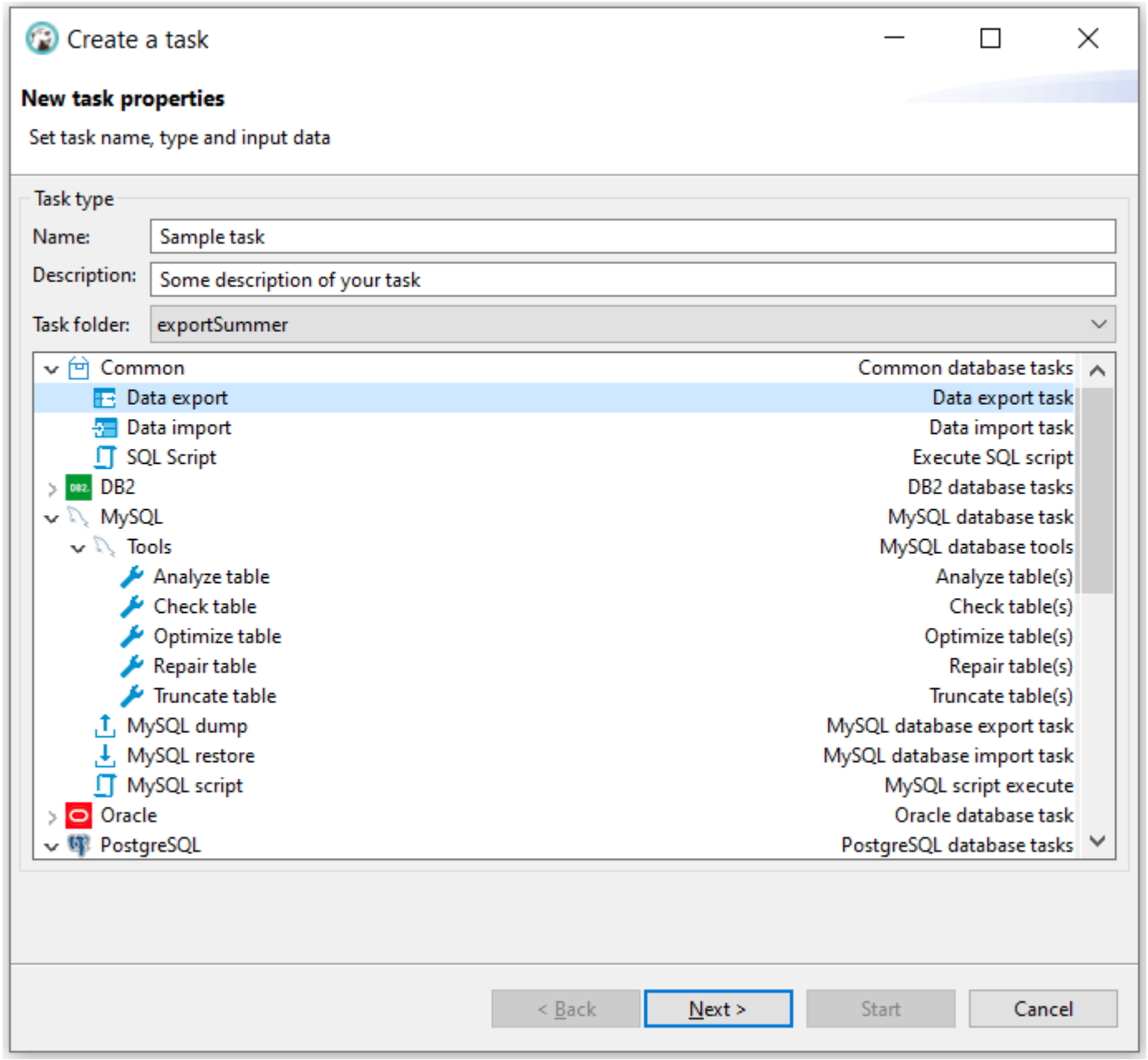
You can configure the Database Tasks View to see more or fewer View columns. Right-click inside the Database Tasks tab and choose the `Configure columns` button from the menu.





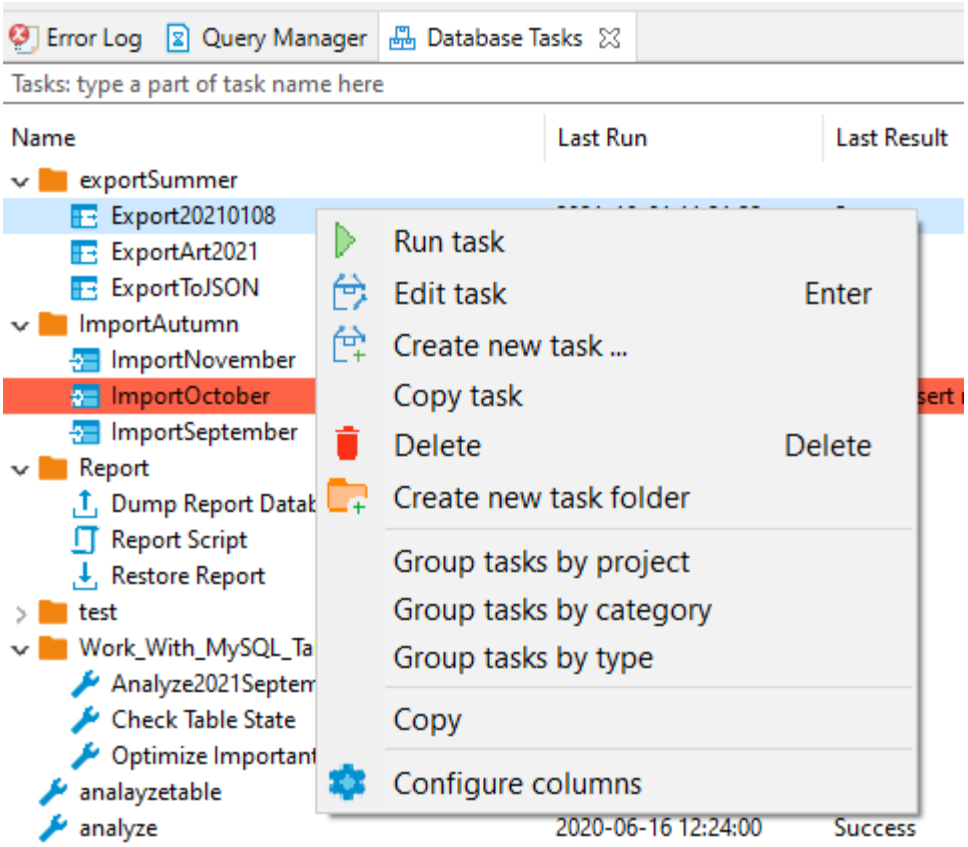
Also, you can create a new task from the main menu `Database -> Tasks -> Create new task...`.



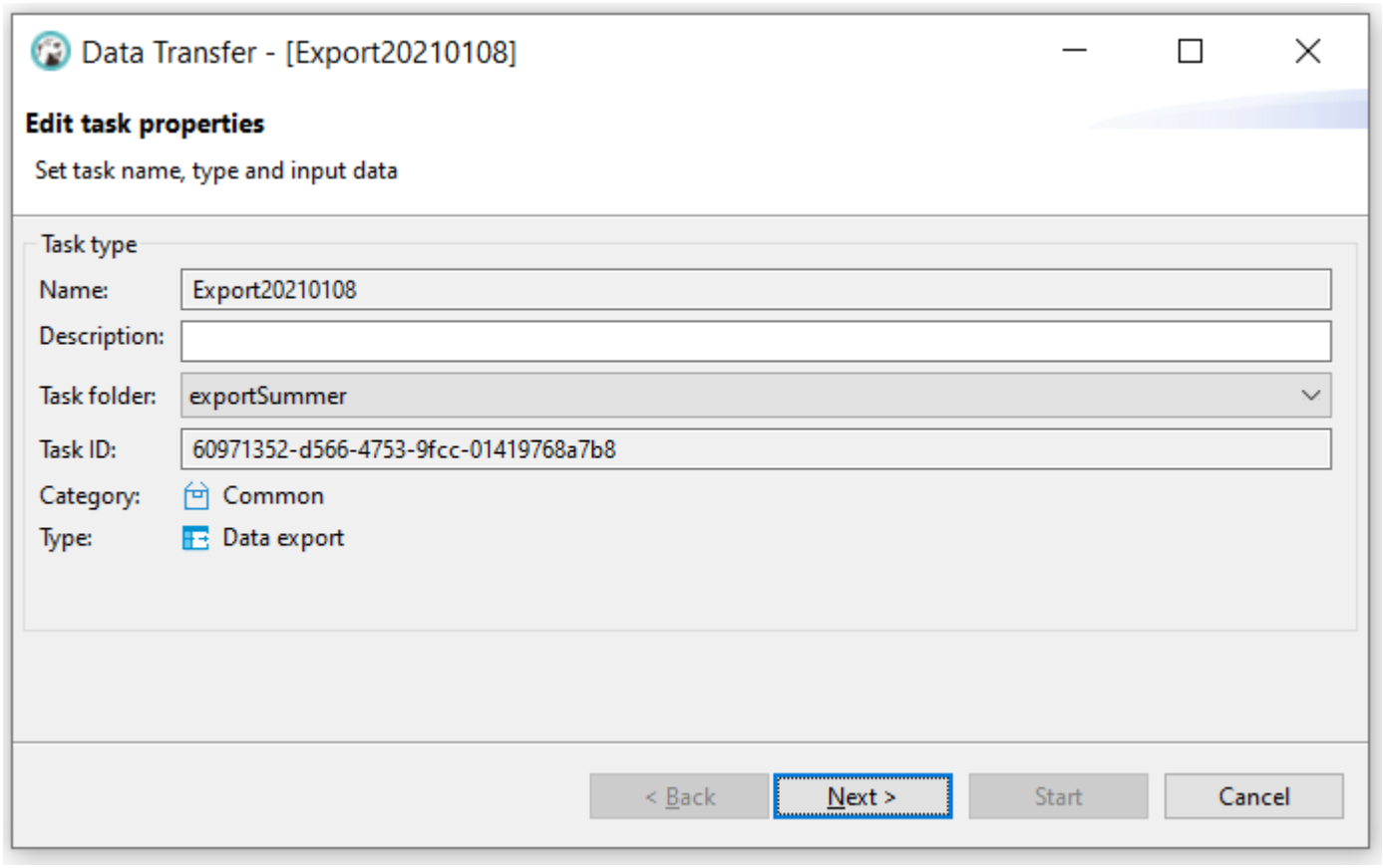


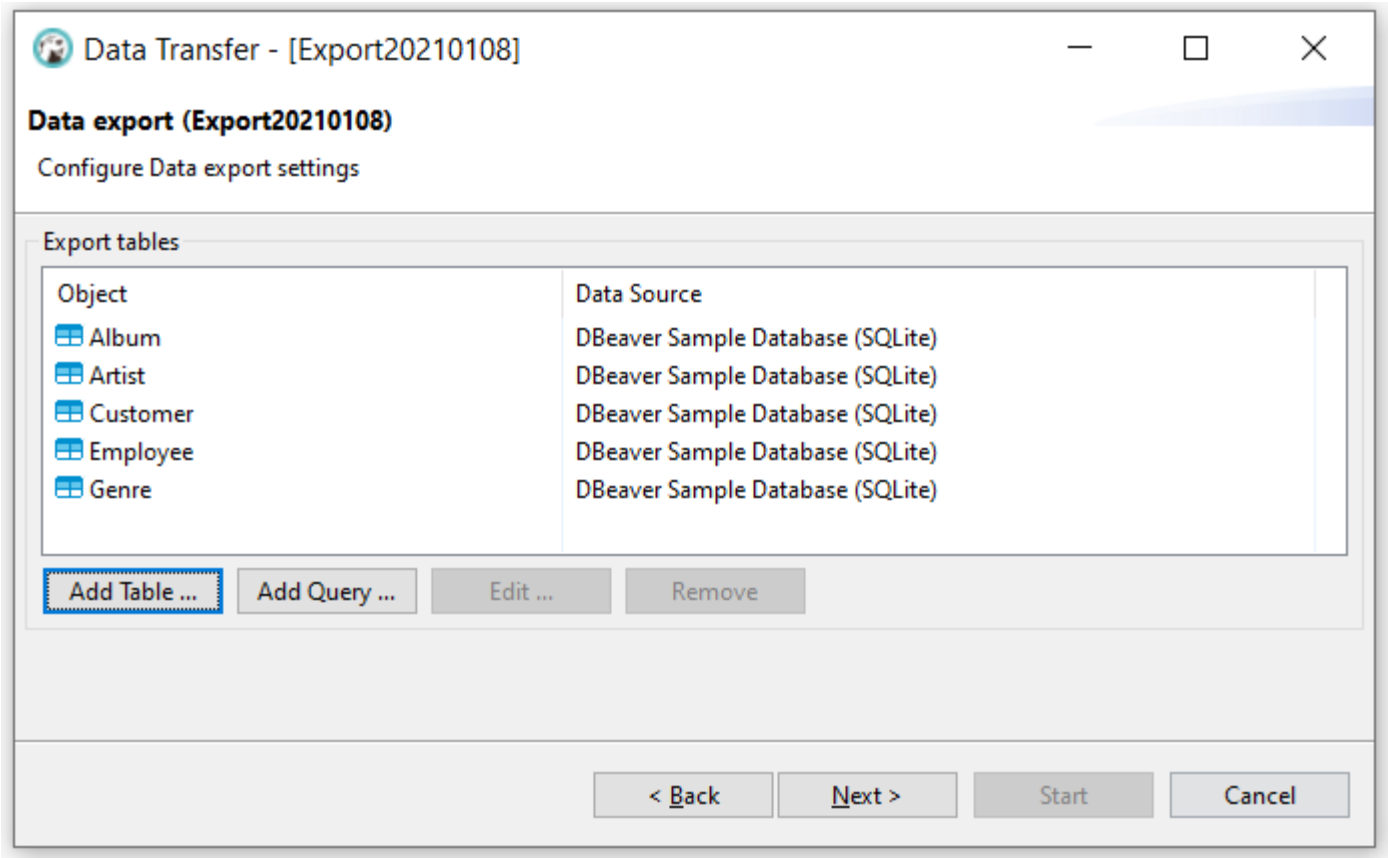
Editing/running tasks

From the task view you can add, edit, remove and execute saved tasks. You can use the context menu or view tools for that:



By clicking on `Edit` or by double-clicking on a task you can open the tasks edit wizard. In this wizard, you can change the task settings as well (use button `Back`) as the actual tool configuration. You can change the set of input objects for data transfer or any export configuration. After changing the task settings, click on the `Save task` button (it is on the last page of the task configuration wizard).



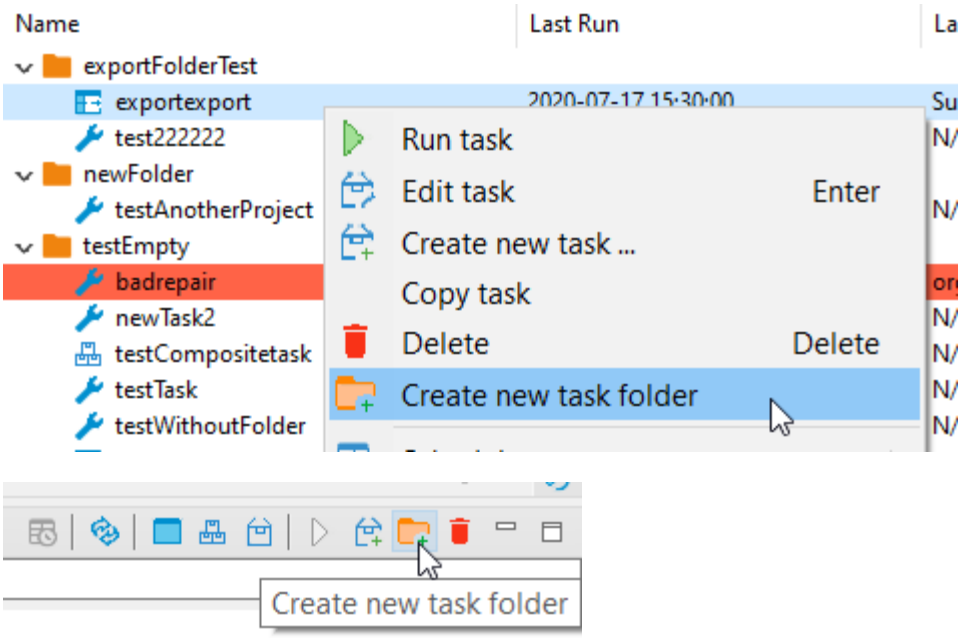


Create a task from task management view

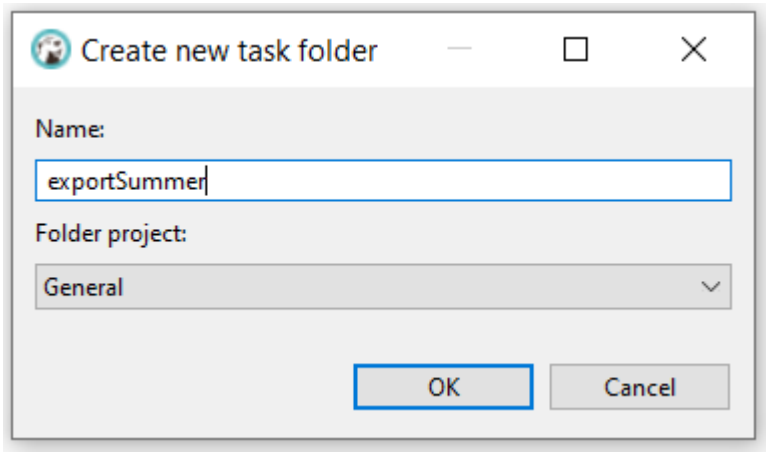
You can create a task from scratch using the tasks view. Open tasks view and click on the `Create new task` button in the View toolbar or in the context menu. In the task wizard, you can choose the task category, task type, task folder, and name. On the next wizard pages, actual tool configuration pages will be shown (they depend on the chosen task type).

Tasks folders

For better structuring, you can store your tasks in the folders. Create folder can also be from the context menu Database Tasks View.



You can choose a project for the new task folder and add a new unique name.



You can change the task folder for an existing task in the task Edit dialog (use button `Back`). Or you can drag your task to another task folder in the View.

The task folder as a task can be deleted with the button `Delete`.

Scheduling tasks

You can schedule tasks for later/regular execution. See the Task Scheduler article.

Task scheduler

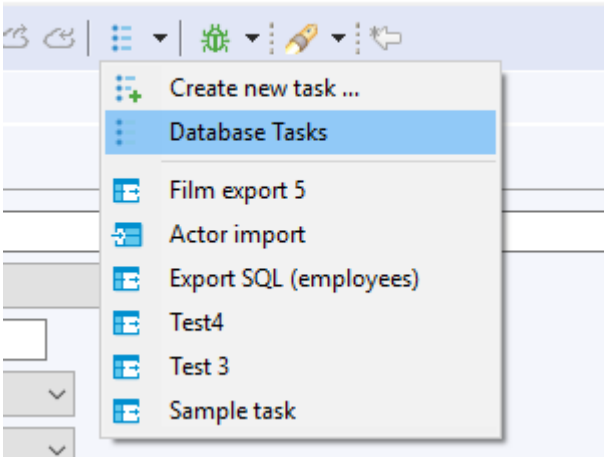
Note: This feature is available in Enterprise, Ultimate and Team editions only.

DBeaver can schedule execution for regular tasks. DBeaver supports `Windows Task Scheduler` on Windows and `cron` on macOS and GNU/Linux. In addition, you can manually configure schedulers using command line.

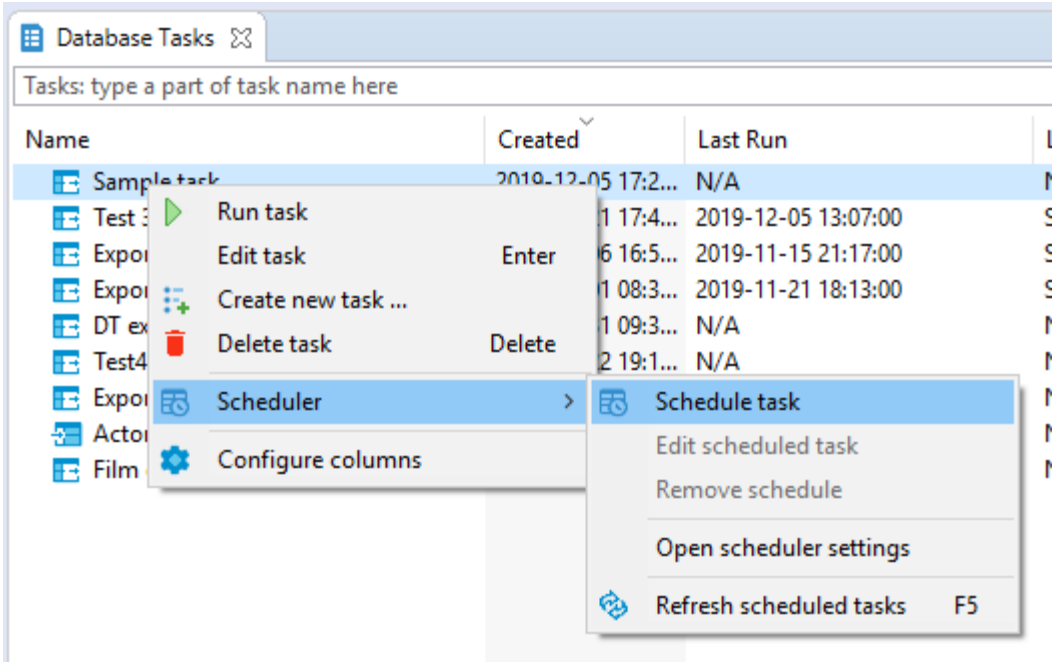
Scheduling tasks from the Tasks view

Windows

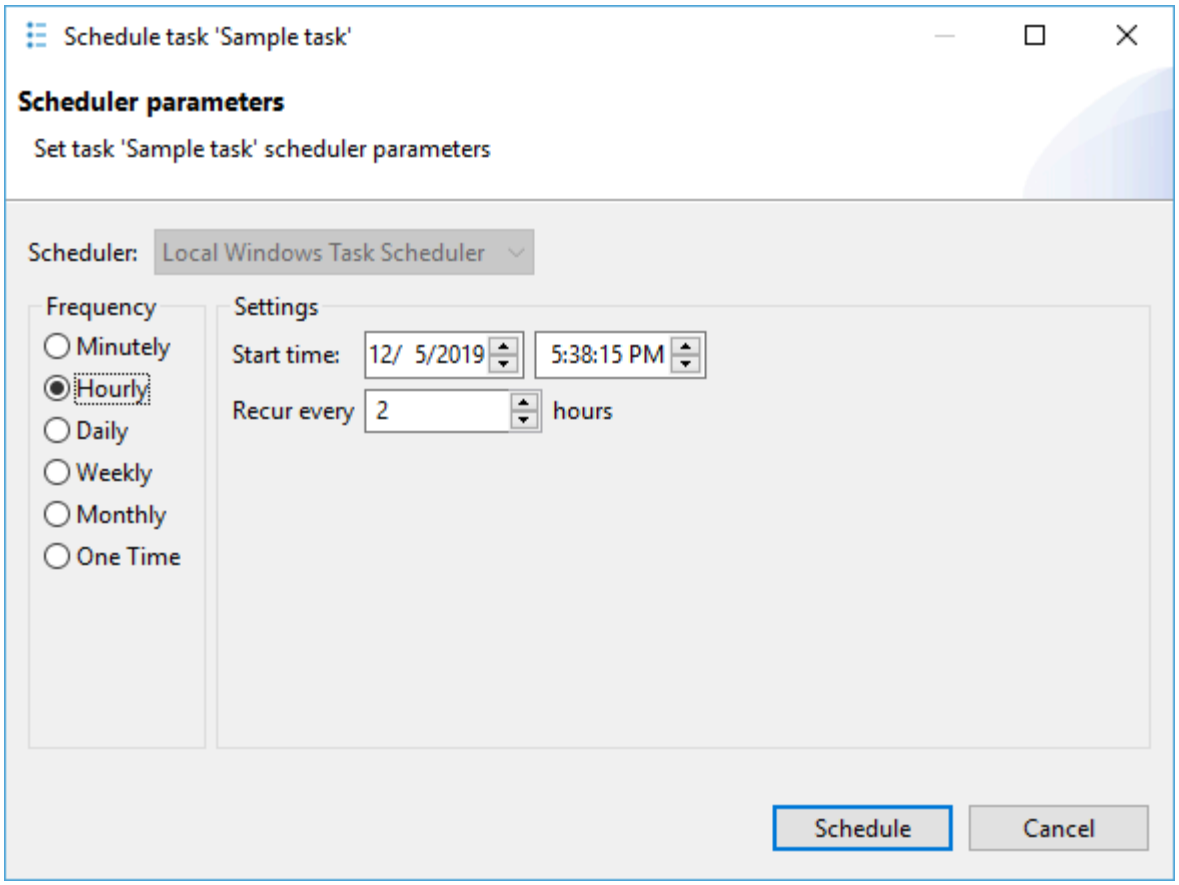
You can open the tasks view from the main toolbar:

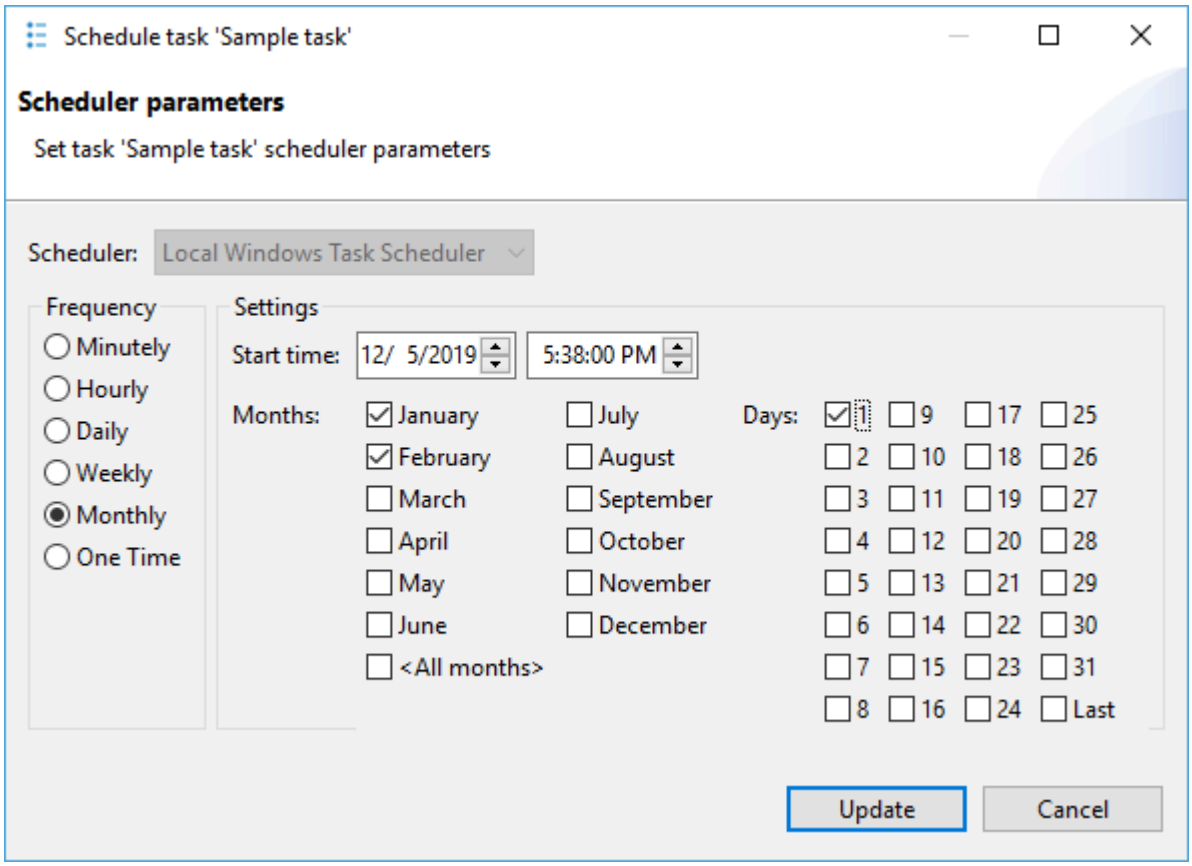


or from the main menu Window. Select a task that you want to schedule in the tasks view and open the context menu:

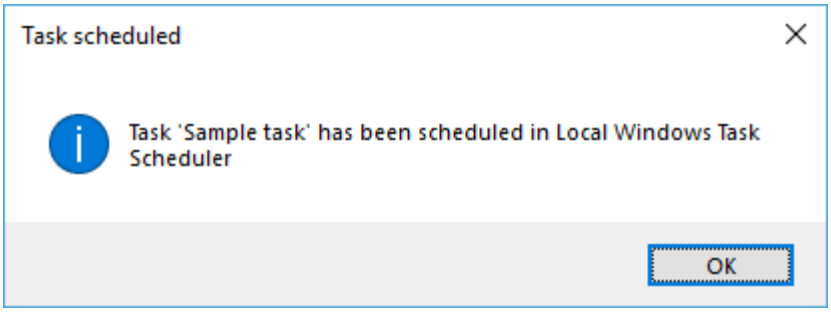


DBeaver will open the scheduler configuration dialog. You can configure task frequency, recurrence period and start time there:





To schedule the task, click on the Schedule button. If everything is OK, you will see the confirmation dialog:



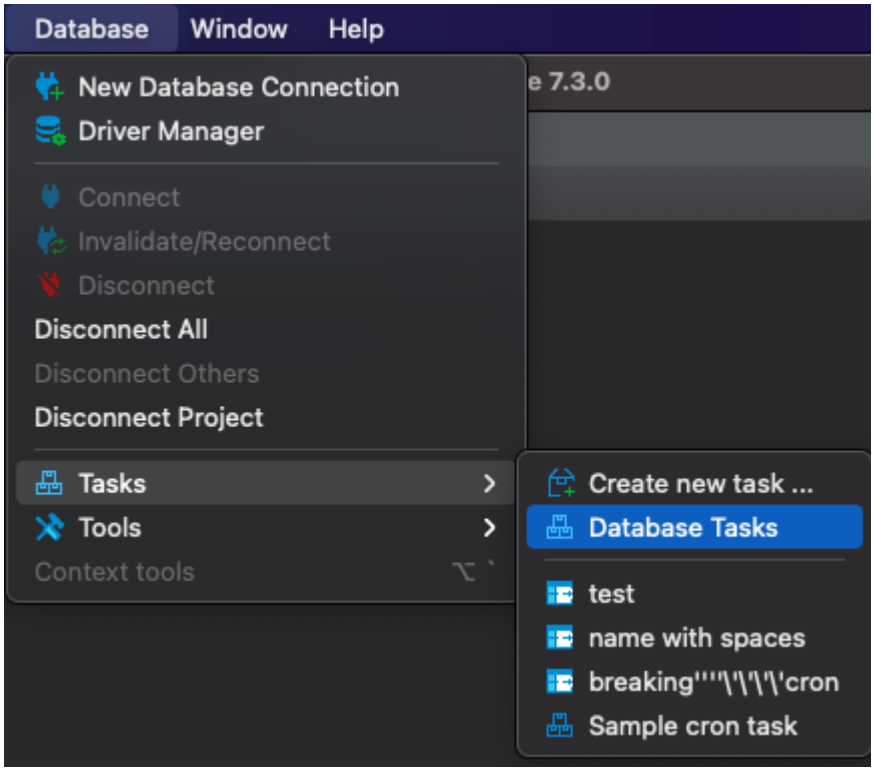
If anything goes wrong, you will see an error message dialog. You can view error details in the Error Log view.

You can change the scheduler settings at any moment by choosing Edit scheduled task command from the context menu. You can also cancel the schedule by clicking on Remove schedule.

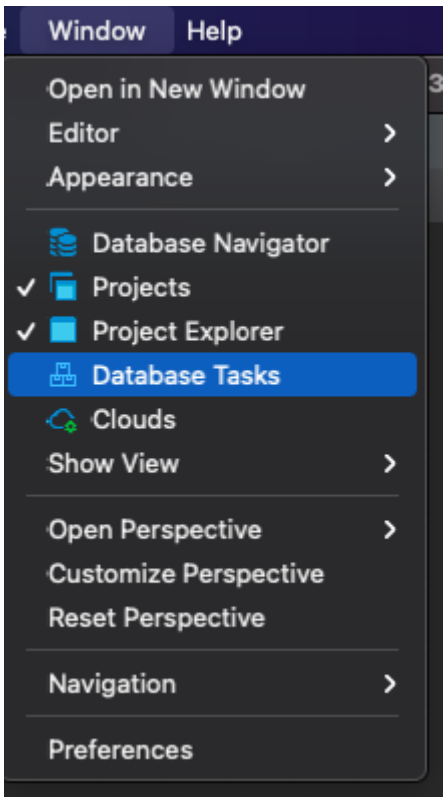
macOS or GNU/Linux

You first need to open the tasks view. There are three ways to do that:

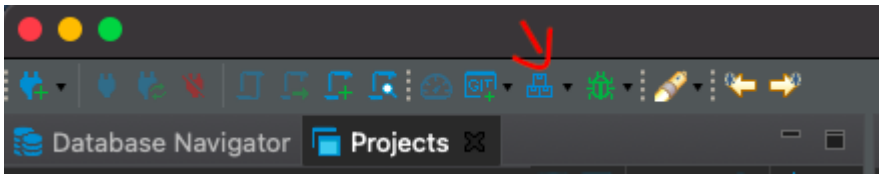
1. Database -> Tasks -> Database Tasks



2. Window -> Database Tasks

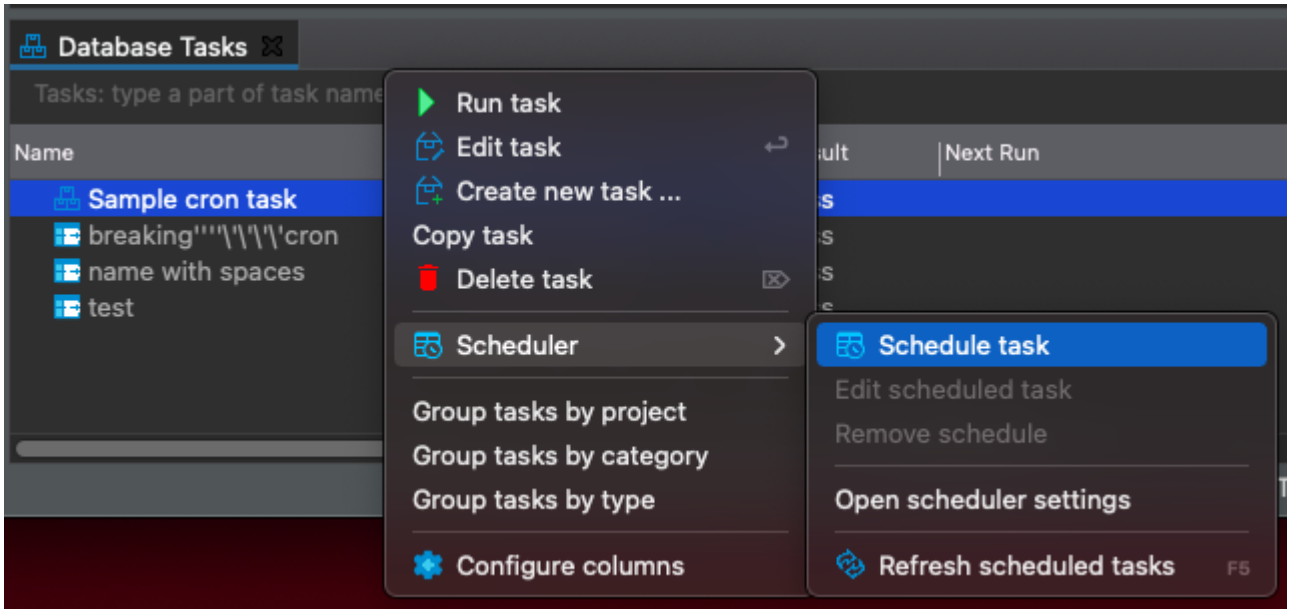


3. Click on 'Show View (Database Tasks)' icon



Select a task you want to schedule in the tasks view. To open the scheduler dialog, either:

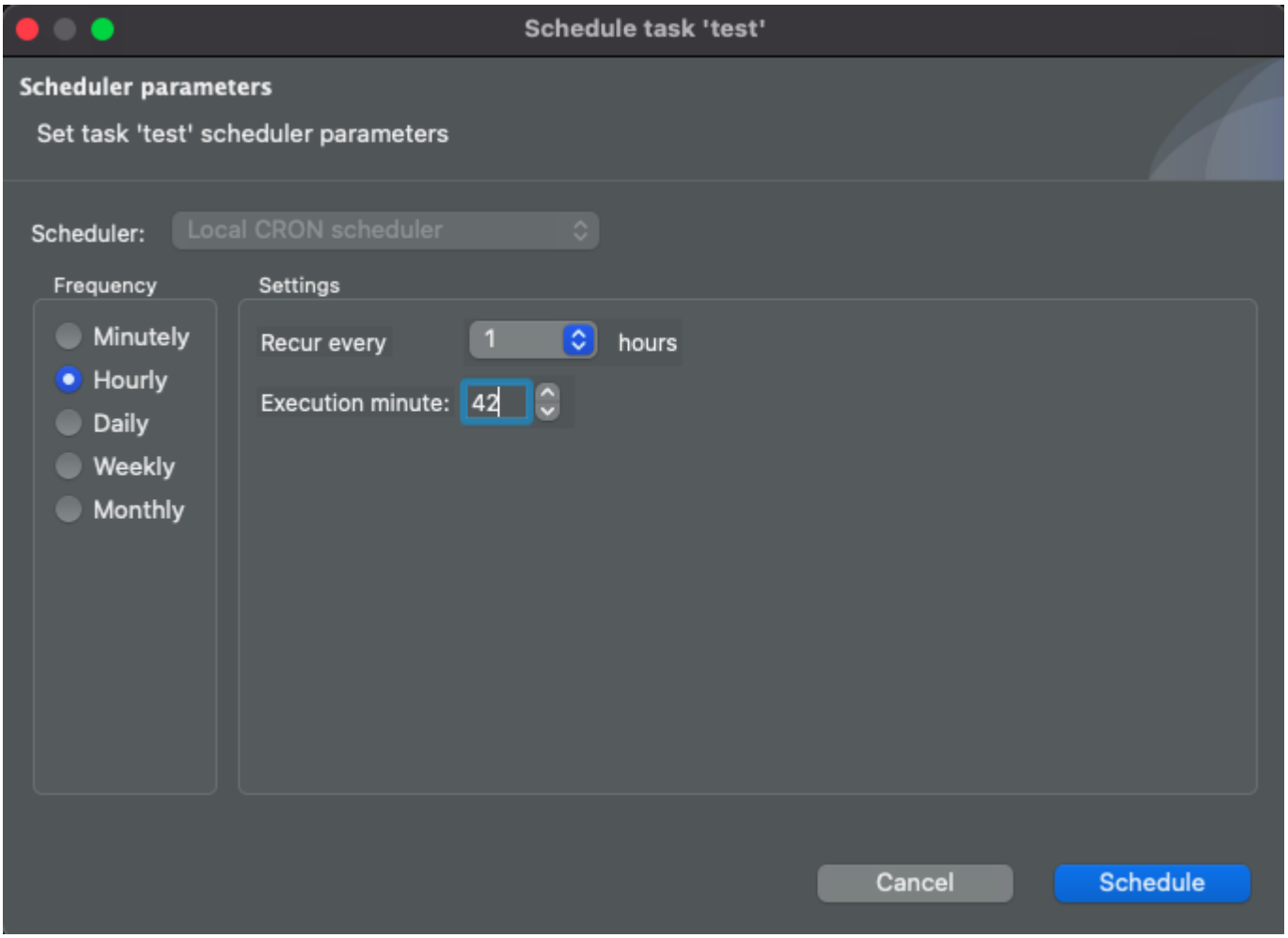
1. Open the context menu with right-click -> Scheduler -> Schedule task



2. or click on the 'Schedule task' icon

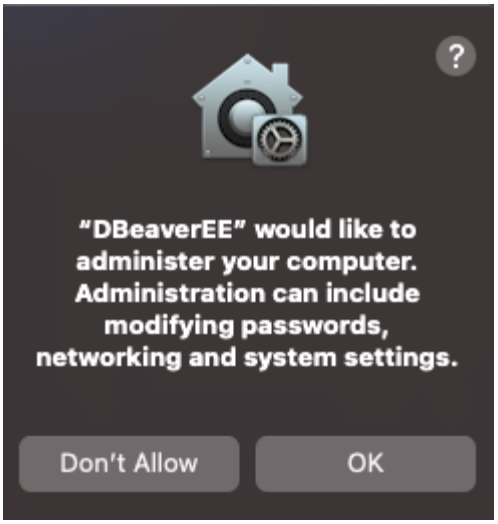


DBeaver will open the scheduler dialog. It has a lot of similarities with the corresponding dialog in Windows, but unfortunately, there are fewer settings on macOS and GNU/Linux due to the limitations of `cron`. For instance, when configuring an hourly task, you can only choose the minute at which the scheduler executes the task. In the example below, the task executes at 1:42 PM, 2:42 PM, 3:42 PM, and so on:



There is also no start date option and, in the case of minutely tasks, no start time either. The scheduler will execute the task at the specified time, but there are no guarantees when the execution starts. It is also worth pointing out that even though you can specify the seconds in the start time selector, the scheduler will ignore them. Even though we try to be compliant with as many cron implementations as possible, most cron implementations do not support this type of granularity.

On macOS 10.15 or newer versions, when scheduling a task for the first time, you will be prompted with something like this:



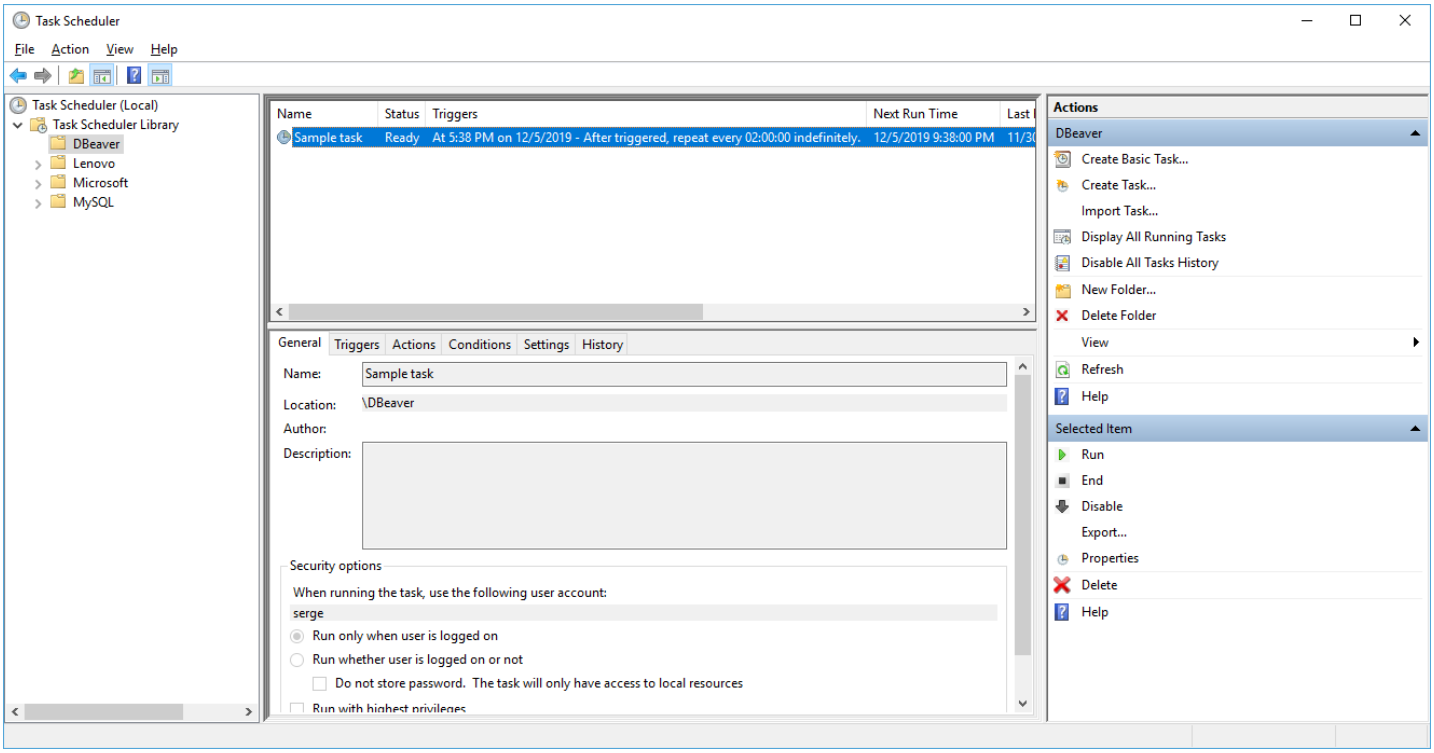
Click 'Yes' to proceed. The reason for that prompt is that macOS considers the `cron` settings (crontabs) to be system settings, and DBeaver will not be able to change them without your permission.

After that, you will see the confirmation message. Just like in Windows, you can change the scheduler settings at any moment by choosing the 'Edit scheduled task' command from the context menu, or cancel the schedule by clicking on 'Remove schedule'.

See schedule details

Windows

You can see and change the scheduled task details in the Windows Task Scheduler. Click on the Open scheduler settings command in the task view context menu:



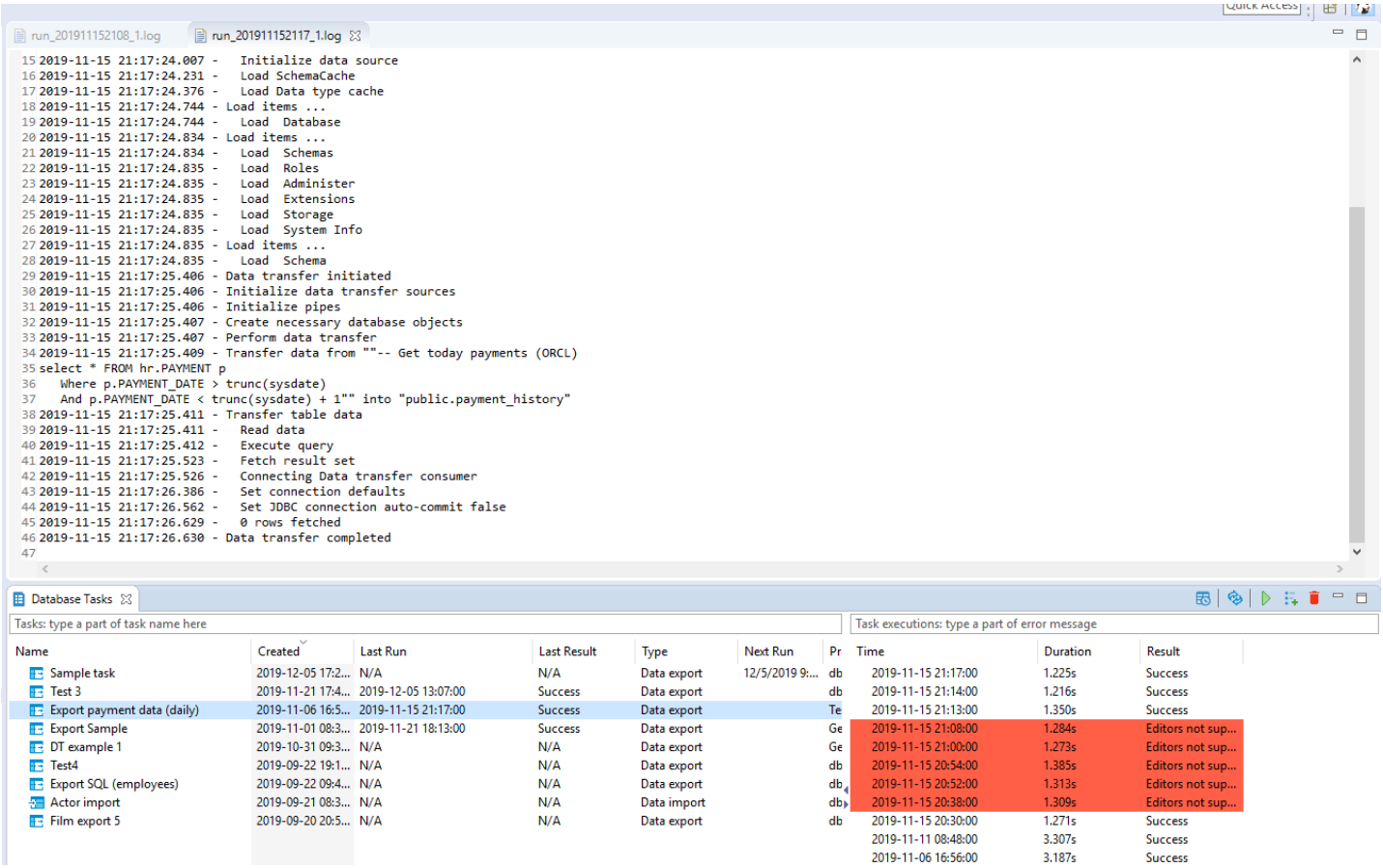
DBeaver stores all tasks in a folder called `DBeaver`.

macOS or GNU/Linux

You can take a look at the crontab DBeaver uses to schedule tasks in `cron` by clicking the 'Open scheduler settings' command in the task view context menu. You can also do it in the terminal by using the command `crontab -l`. Although you can also edit the crontab by using `crontab -e`, we strongly do not recommend it.

Monitoring for task execution (any OS)

You can look through the task execution logs on the right side of the tasks view. By double-clicking on a task run item, you can see the full log with all details, errors, and warnings:



DBeaver keeps the task run logs in the workspace directory, subfolder .metadata/task-stats.

Running tasks from the command line (any OS)

The task scheduler uses the DBeaver command line interface to perform task executions. Command-line parameter `-runTask TASK_ID` launches saved task executions (immediately). TASK_ID has the form `@projectName:taskName`. You can omit the project name part if you have only one project in your workspace. In Windows, you can use the `dbeaver-cli` executable to run tasks. Please note that if you use `dbeaver` executable (for any reason), you will need to add the command line parameter `-nosplash` to avoid a splash screen appearance.

Troubleshooting

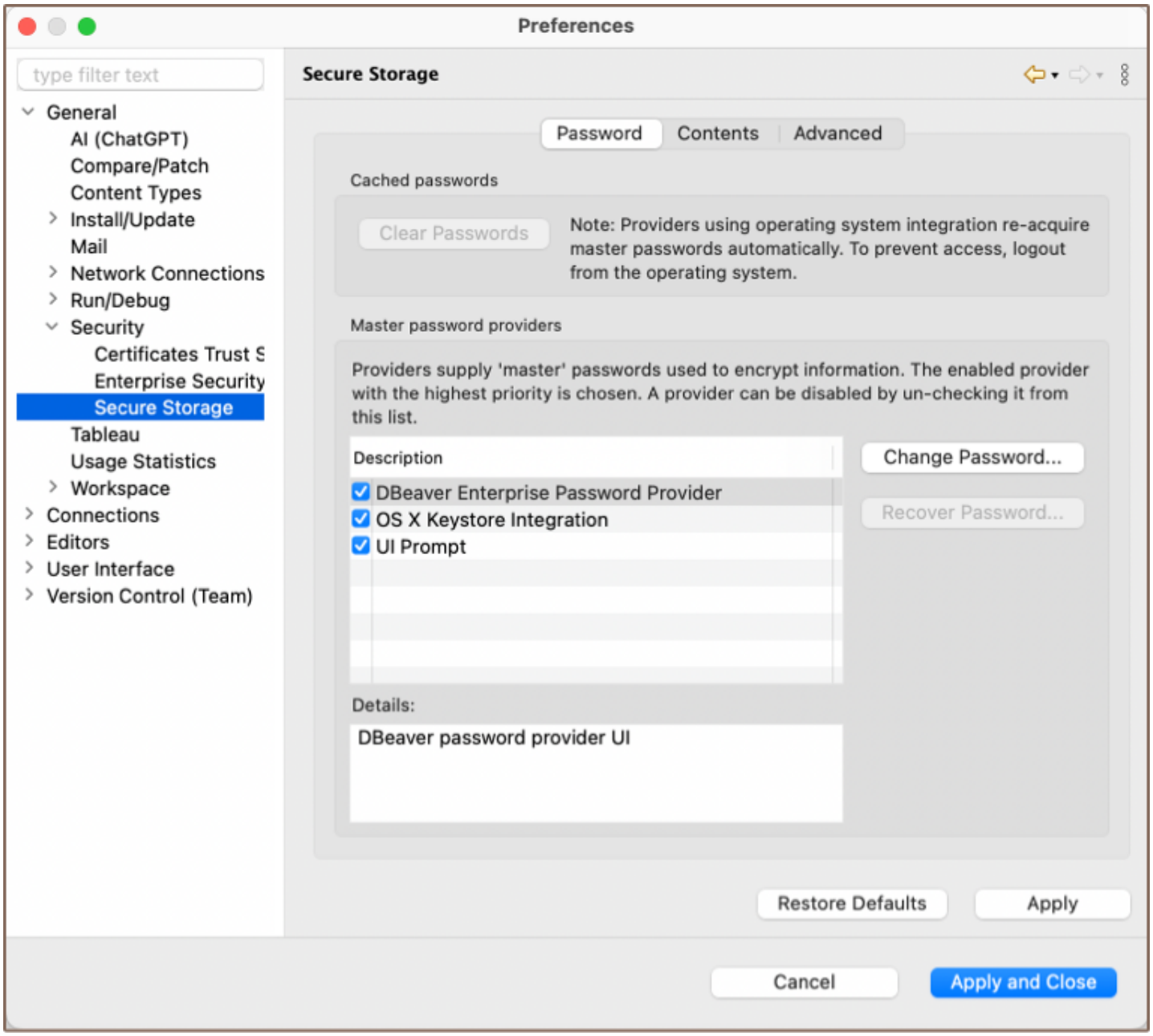
User credentials + enterprise security

When you

- Enable enterprise security for database credentials
- Use Mail server authentication to send data transfer results over email

Then these settings won't be accessible in scheduled tasks by default. But you can use default OS-specific encryption for that. Master password won't be used, so it is a bit less secure configuration (credentials are still encrypted, but they may be stolen if hacker will gain access to your computer).

To enable this you need to:



- Go to Preferences->General->Security->Secure Storage
- Disable "DBeaver Enterprise Password provider" and "UI prompt"
- Update credentials you want to use for scheduled task:

- For Mail profile you can open mail settings and click Ok
- For database credentials got to **Window** -> **Preferences** -> **General** -> **Security** -> **Enterprise security** and toggle "Use secure password storage" on and off to trigger stored credentials re-encryption

Windows scheduler overview

There are two implementations of the Windows scheduler present:

1. CLI-based (**Legacy**): uses `schtasks.exe` to communicate with the scheduler; sensitive to locale-dependent data, such as Unicode names and date-time format.
2. COM-based (**New**): uses COM API to communicate with the scheduler; more flexible and provides more features than the CLI version.

COM-based implementation is used by default, starting from the 21.1 version of DBeaver EE.

Windows Task Scheduler: COM exception

Non-legacy scheduler only

If you encounter an error in Windows which contains the following text:

`com.sun.jna.platform.win32.COM.COMException`,

do the following:

1. Open the file `dbeaver.ini` in the directory with your DBeaver installation
2. Place the line `-Ddbeaver.scheduler.windows.legacy=true` below the `-vmargs` line.

Windows Task Scheduler: incorrect date format

Legacy scheduler only

If you encounter an error in Windows which looks like this:

`ERROR: Invalid Start Date (Date should be in %some_format% format).`,

do the following:

1. Open the file `dbeaver.ini` in the directory with your DBeaver installation
2. Place the line `-Ddbeaver.scheduler.windows.dateFormat=%some_format%` (where %some_format% is a format from the error message) below the `-vmargs` line.

This flag is available starting from the 7.3.4 EA version of DBeaverEnterprise and might be removed in the future.

macOS 10.15+: Unable to read or write to crontab

When scheduling tasks on macOS 10.15 or newer versions, the OS will prompt you to elevate DBeaver's permissions to administer your computer. If you do not grant these permissions, DBeaver will fail to schedule your tasks with an error `Unable to read or write to crontab`. To bypass this, simply restart DBeaver and try to schedule the task again. The operating system will prompt you to elevate the permissions again. If macOS never prompted to do that in the first place, you could grant `Full disk access` permissions in the macOS settings. Here is how to do that:

1. Open `System Preferences`.
2. Click on `Security & Privacy`.
3. Choose the `Privacy` tab.
4. Choose the `Full Disk Access` folder.
5. Unlock the preferences lock to the bottom if it is locked.
6. Click the + button.
7. Select DBeaverEE in the file picker that opens.
8. Click `Open`.
9. Close the lock.

Tasks from password-protected projects cannot be run

You need to pass a password for one or more projects via the command-line interface.

To do so, you need to set the `dbeaver.project.password` parameter in the external configuration file like so:

```
# You can specify a single password for all projects:
dbeaver.project.password=p4$$w0rd

# Otherwise, you can specify a list of passwords for given projects:
dbeaver.project.password=@General:p4$$w0rd,@Other:12345
```

The syntax for a single entry is `@ <name of the project> : <password of the project>`; others are separated by the `,` symbol.

Please note that `@` and `:` symbols are mandatory.

Composite tasks

Note: This feature is available in Lite, Enterprise, Ultimate and Team editions only.

As the name suggests, the *composite task* is a type of task that consists of other tasks. Just like the other type of tasks, the composite tasks can be scheduled via Task Scheduler. Let's take a look at what they can offer.

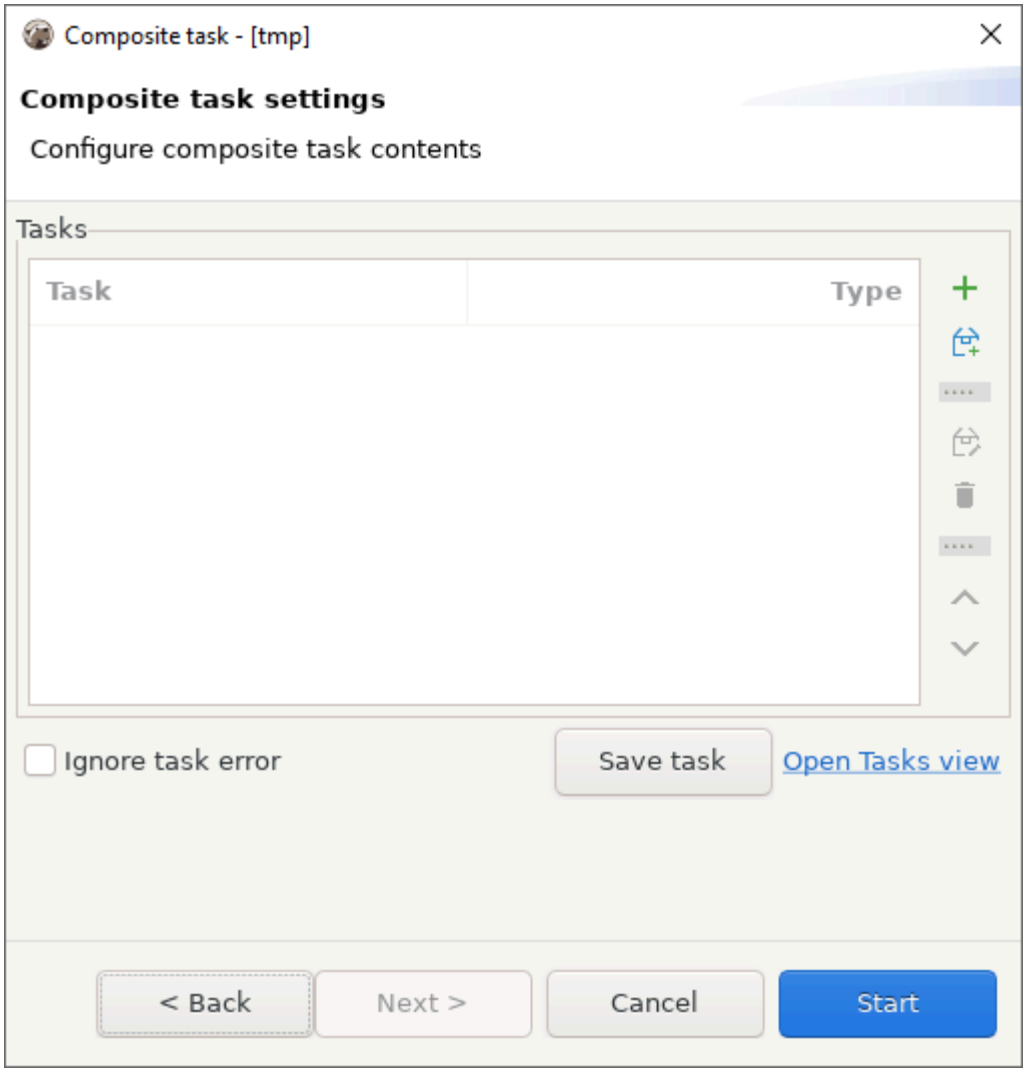
Creating a *composite task*

The first thing we need to open is the *Create a task* dialog. You can do it in multiple ways:

- From the context menu in the database navigator -> Tools -> Create new task... -> Composite task
- By clicking Database -> Tasks -> Create new task... -> Composite task
- From the context menu in the Database Tasks view.

Choose *Composite task*, enter the task name, description (optional), and hit *Next*.

You will be presented with the following dialog:



Setting up a *composite task*

When creating a composite task, you need to specify which tasks the composite task consists of.

This can be done:

1. By adding an existing task. To do that, click the button with the plus sign
2. By creating a new task and adding it simultaneously. To do that, click the button below the aforementioned button with the plus sign
3. By drag-and-dropping a task from the _Database tasks panel.

As a side note, you can add a composite task to your new composite task.

You can edit tasks in the same dialog, delete a task from a composite task, and change the execution order.

There is also a very important checkbox, *Ignore task error*. The tasks from the *composite task* are executed in the order they appear in the settings dialog. Executing a task from a *composite task* might produce an error that will block the next tasks from proceeding. The *Ignore task error* checkbox can be used to bypass this behavior.

Sending results by e-mail

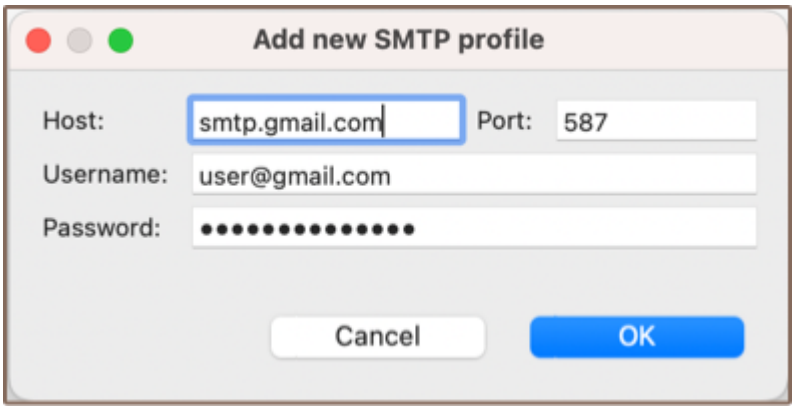
Note: The following feature is only available in Enterprise and Ultimate editions.

DBeaver offers a way to send data exported via Data Transfer by email.

SMTP profile configuration

First, you'll need to add an SMTP profile to send the email. Go to `Window ⇒ Preferences ⇒ General ⇒ Ma` and create a new profile.

Parameters `Host` and `Port` may depend on the mail service you use. Use the latter if the service offers SSL and TLS ports. Gmail, for example, uses host `smtp.gmail.com` and port `587`. An example of a configured profile:

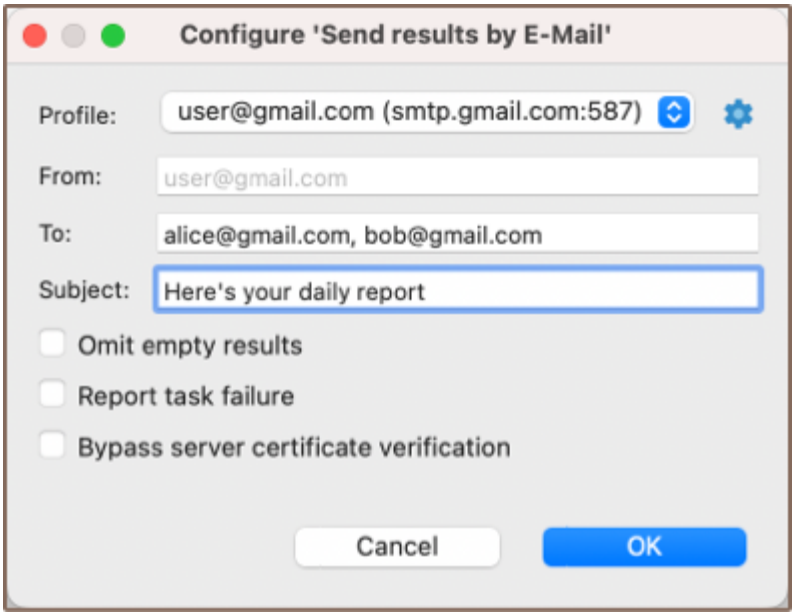


Then you can use the `Test connection` button to verify that the host and credentials are valid.

Please take a look at the troubleshooting section for more information on resolving common authorization problems.

Setting up data transfer

When at least one profile is present, you can set up email sending. Create a regular export task, go to the `Output` page, and make sure the `Send results by email` option is enabled. By pressing the `Configure` label near it, you can specify several recipients and the subject for your mail:



You also have control over the specifics of what you receive, as follows:

- **Omit Empty Results**: Enable this option to prevent emails from being sent when no results are displayed.
- **Report Task Failure**: Enable this option to receive an email alert whenever a task fails.
- **Bypass Server Certificate Verification**: This option can be used when working with self-signed certificates and you wish to bypass the standard certificate validation process.

That's it. After completing the task, the specified recipients will receive an email containing the exported file in a specified format during the data transfer.

Authorization troubleshooting

You may face various problems while setting up a new SMTP profile.

Several common errors when using Gmail and solutions for them are described below:

- `535-5.7.8 Username and Password not accepted`. Check that the username and password are correct. If you are confident that you have entered valid credentials, then try creating an app password.
  Read more at https://support.google.com/accounts/answer/185833
- `534-5.7.9 Application-specific password required`. You have two-factor authorization enabled. You will need to generate a special password for DBeaver to use.
  Read more at https://support.google.com/accounts/answer/185833

There were also several cases when the antivirus would block DBeaver from sending an email.

Uploading result to external storage

Using Data Transfer, you can export files to external storage, such as Google Drive.

Exporting files to Google Drive

1. Download and install Google Drive for desktop here
2. Configure Drive for streaming mode according to this article

After that, in the data transfer wizard, change the output directory to the Drive directory.

Exporting automatically

It can be achieved by using Task Scheduler.

Note: This feature is available in Enterprise, Ultimate and Team editions only.

Cloud Explorer

Note: This feature is available in Ultimate and Team editions, as well as in CloudBeaver Enterprise.

Overview

Cloud Explorer offers deep integration with popular cloud service providers. As of the latest version, it supports Amazon Web Services (AWS), Google Cloud Platform (GCP), and Microsoft Azure.

With **Cloud Explorer**, you can set up your cloud access once and then easily browse, connect, and manage all your cloud databases. It saves you from manually configuring each database connection, as it reads all database endpoint information directly from the cloud provider.

Authentication is centralized. You use your cloud account to access your cloud databases. Before you start using **Cloud Explorer**, you need to set up your cloud provider access. This setup includes access credentials, availability zones for database search, and other cloud-specific settings.

Cloud configuration

To begin the configuration process, click the **Cloud Explorer** button  located in the Toolbar. From there, you can start setting up your cloud provider access according to your needs. This involves configuring your access credentials, specifying the availability zones for database search, and adjusting other cloud-specific settings. The setup process varies depending on the cloud service provider you're using.

If this is your first time setting up, the **Create Cloud Configuration** dialog will open, guiding you through the setup process. Once you've set up your first cloud configuration, clicking the **Cloud Explorer** button will open a dialog displaying your cloud databases.

If you need to set up a new cloud configuration, you can do so by clicking the **New** button  in the **Cloud Explorer** dialog. You can start setting up your new cloud provider access according to your needs.

Cloud configuration is different for each cloud service provider.

Amazon Cloud Services

Name	Description	Additional article
Default Credentials	Allows AWS to determine credentials using the standard credential providers.	AWS Default credentials
AWS Profile	Allows you to choose which credentials profile you want to use.	AWS Profile
Access /Secret Key	Allows you to choose which credentials profile you want to use. If you have an AWS Secret set up, enter it in the Connection settings.	AWS Access /Secret Key
Single Sign-On (SSO)	Allows access to AWS resources without explicitly specifying user credentials in DBeaver connections configuration.	AWS SSO guide

About permissions, read AWS managed policies article.

Google Cloud Platform

Name	Description	Additional article
Default Credentials	Allows GCP to determine credentials using the standard credential providers.	GCP Default credentials
Web browser or **SSO**	Allows you to authorize your connection through your Google account without needing a service or user-based key file.	GCP SSO
Access key file	Allows you to provide the path to your service or user credentials files.	GCP Access key

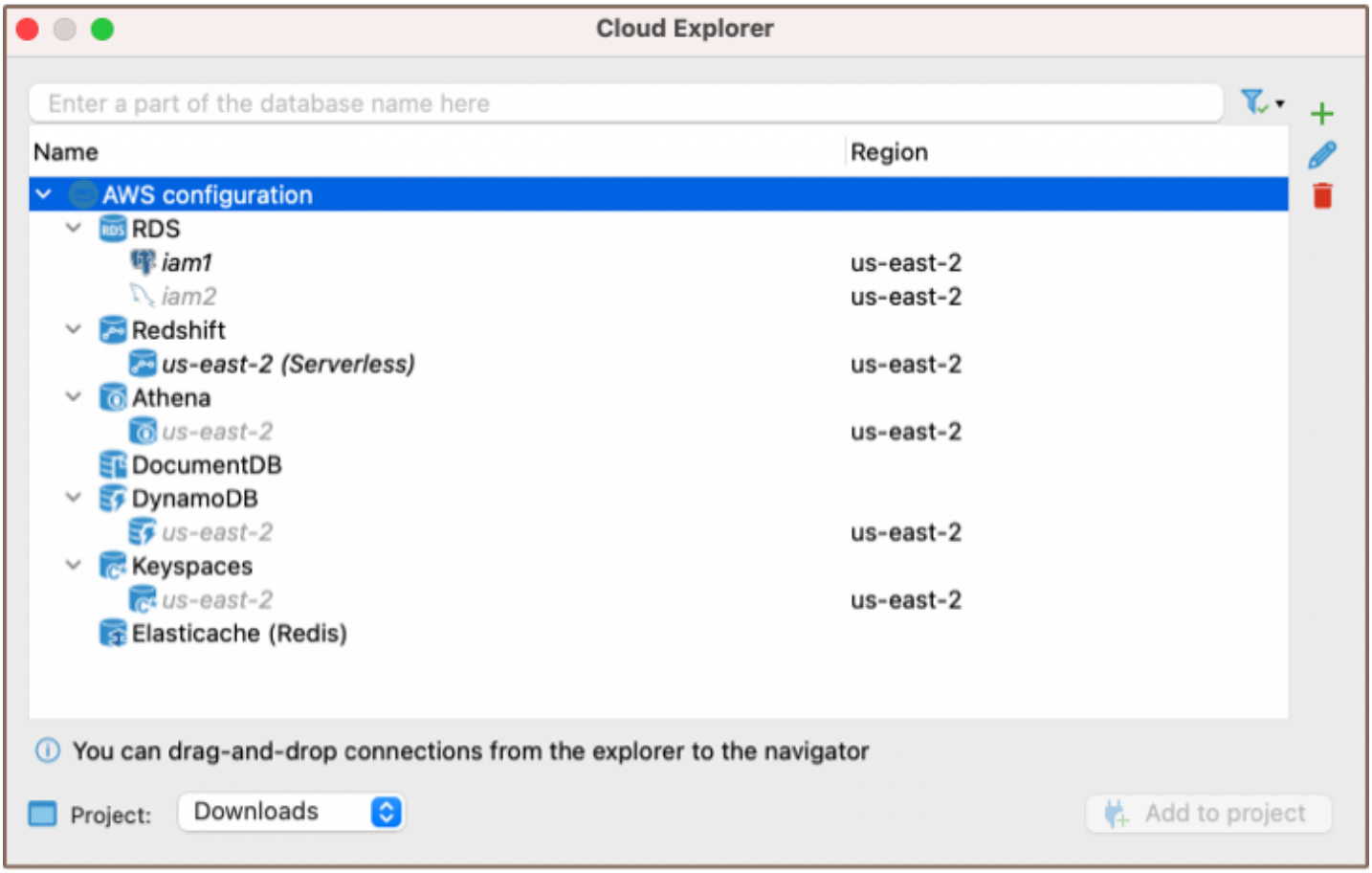
Microsoft Azure

Name	Description	Additional article
Client ID	Allows you to provide the ID of the application (client) used for authentication and database access.	Azure Cloud Explorer

Explorer

Once you've configured your cloud provider access, open the **Cloud Explorer** dialog and add database connections. In the top drop-down of the explorer dialog, you can select the active cloud configuration or click **Edit** button  to change the cloud configuration.

You can see cloud databases in the center of the dialog in a hierarchical view. All databases are grouped by database/service type. When you expand one of the top elements, DBeaver will search cloud databases in configured availability zones/regions.



If you have a large number of databases in your cloud, you can search or filter them using filter text above the cloud navigator.

You can drag-and-drop cloud databases directly to database navigator view or projects view. You can also add any number of databases to your project directly from the **Cloud Explorer**. To do this, click on the database you want to add. Once selected, click on the **Add to Project** button 

located in the bottom right corner of the **Cloud Explorer** dialog, or right-click on the database and select **Add to Project** from the context menu. This will add the selected database to your project, making it easier to manage and navigate your databases within specific projects.

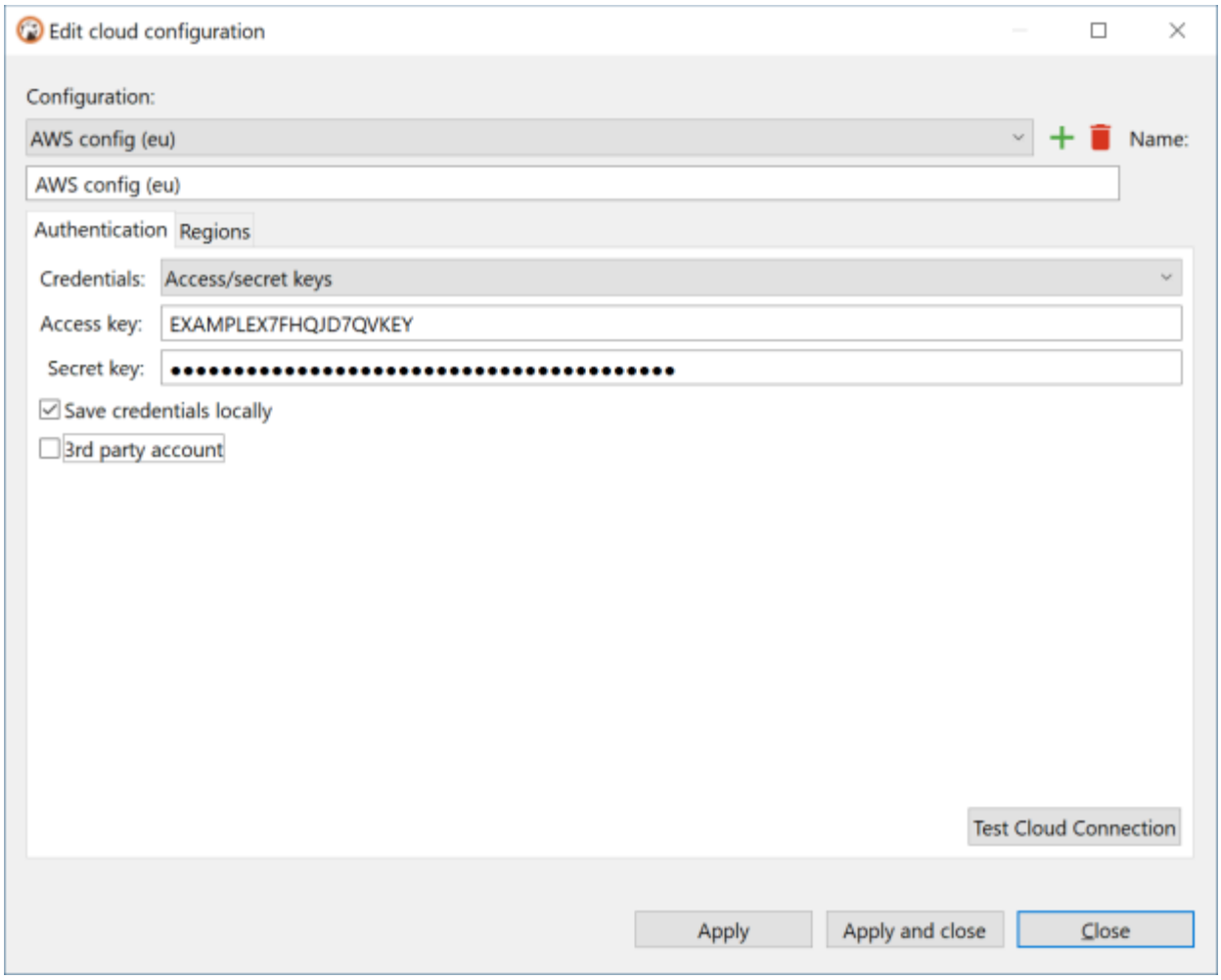
Database cloud information

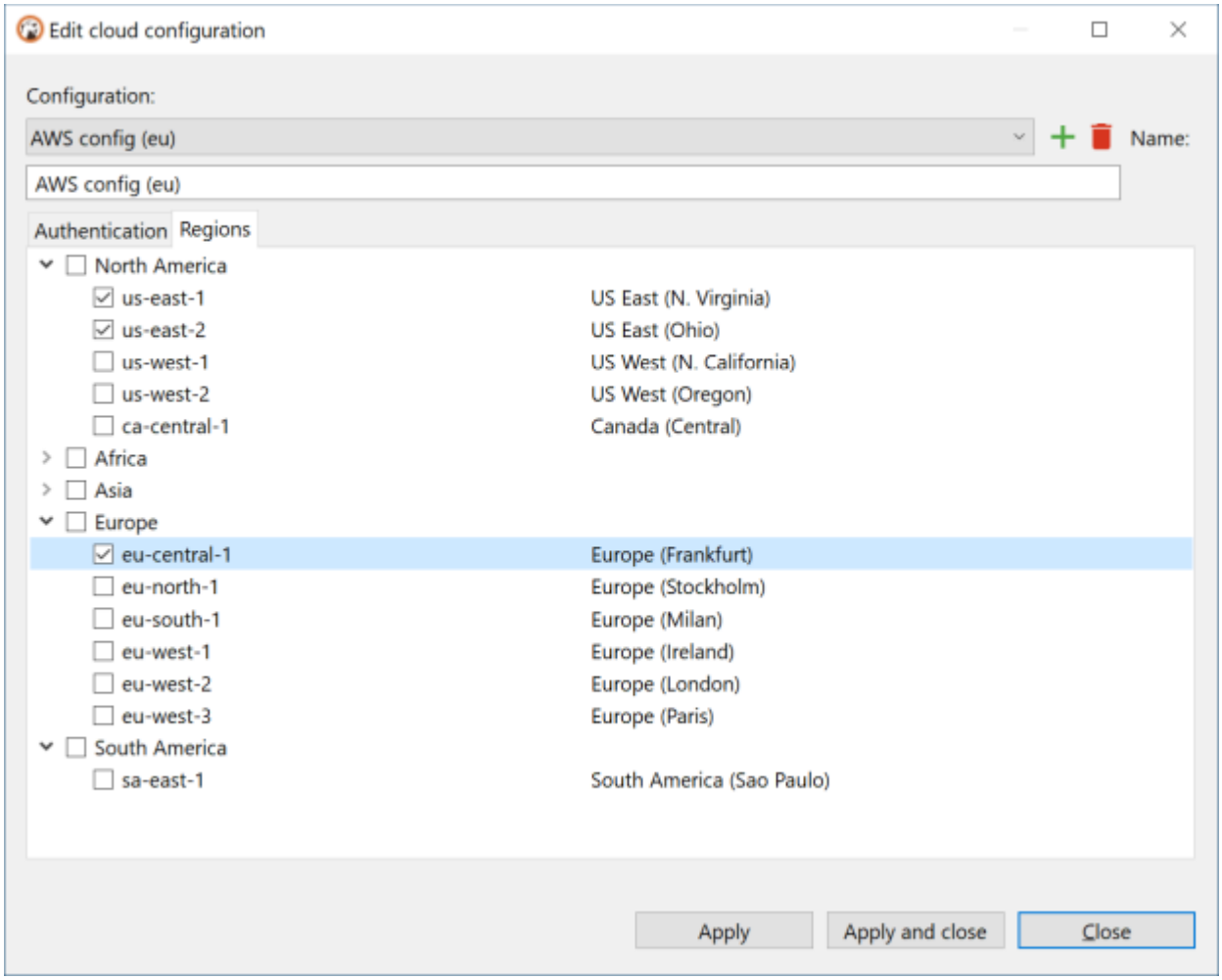
You can access your cloud database configuration directly from the **Database Navigator**. To do this, right-click on the database you are interested in and select **Edit Connection**. This action opens a special tab in the connection settings dialog. The information displayed in this tab is specific to the cloud and database type of the selected database.

AWS

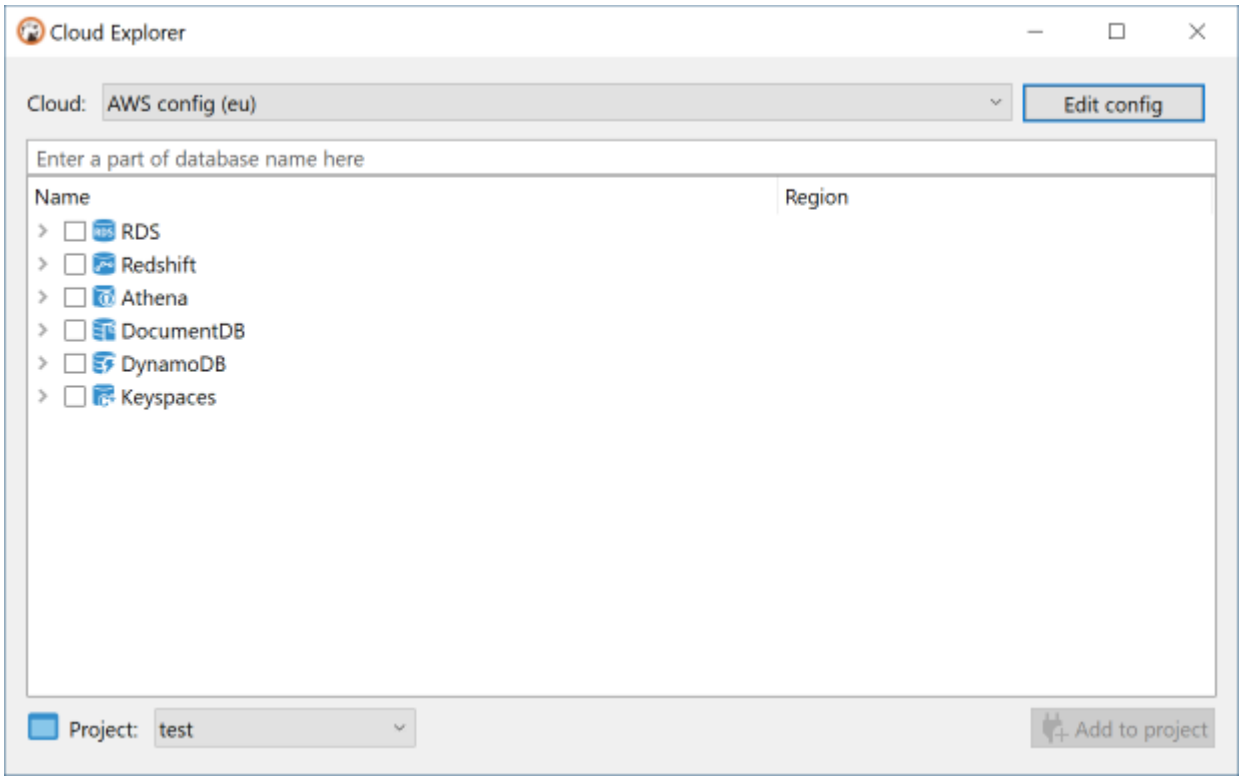
AWS (Amazon Web Services) Cloud Explorer

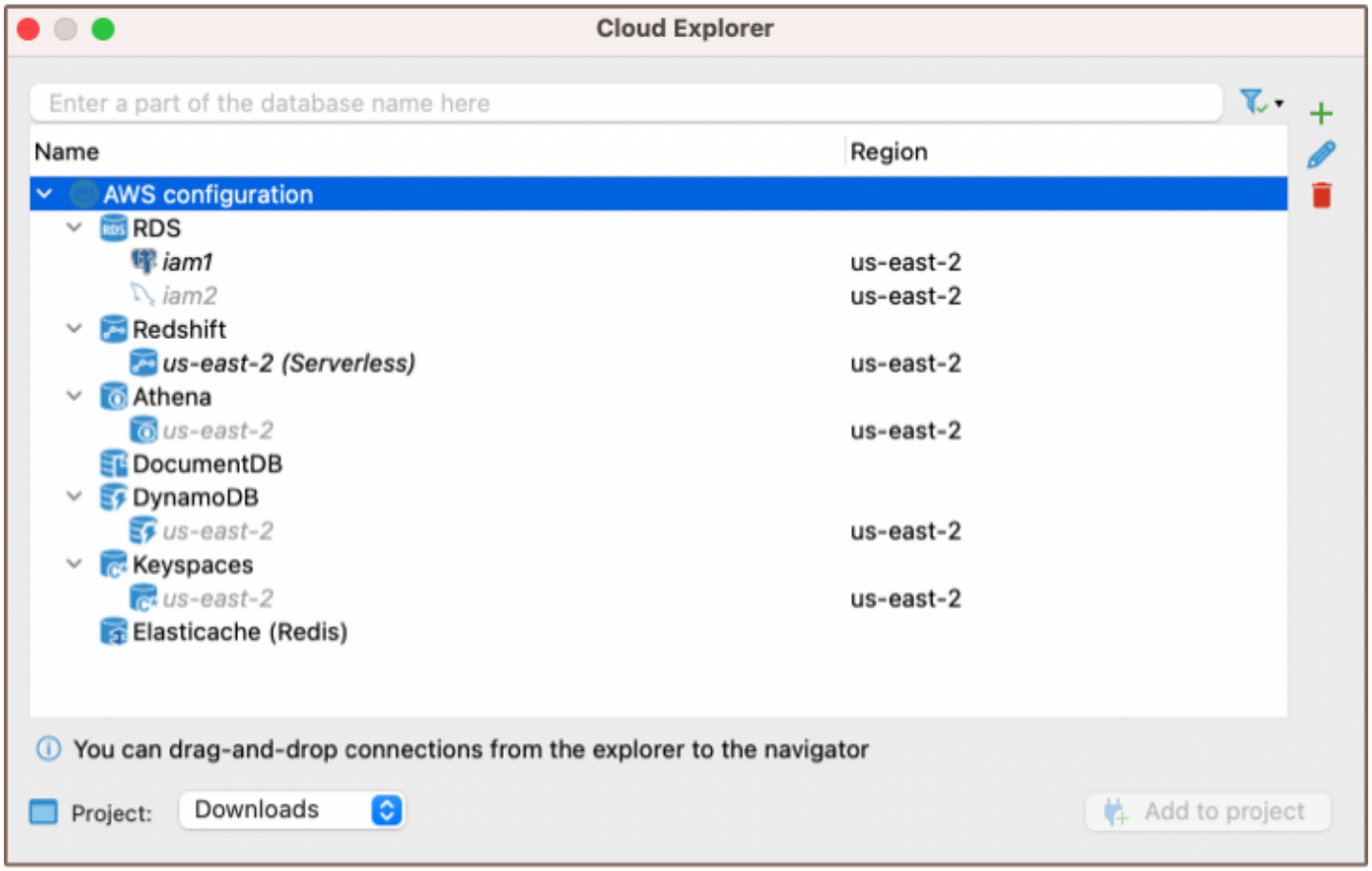
Cloud configuration


Edit cloud configuration
Configuration:
AWS config (eu)
Name:
AWS config (eu)
Authentication
Regions
Credentials:
Access/secret keys
Access key:
EXAMPLEX7FHQJD7QVKEY
Secret key:
Save credentials locally
3rd party account
Test Cloud Connection
Apply
Apply and close
Close


Edit cloud configuration
Configuration:
AWS config (eu)
Name:
AWS config (eu)
Authentication
Regions
North America
us-east-1
US East (N. Virginia)
us-east-2
US East (Ohio)
us-west-1
US West (N. California)
us-west-2
US West (Oregon)
ca-central-1
Canada (Central)
Africa
Asia
Europe
eu-central-1
Europe (Frankfurt)
eu-north-1
Europe (Stockholm)
eu-south-1
Europe (Milan)
eu-west-1
Europe (Ireland)
eu-west-2
Europe (London)
eu-west-3
Europe (Paris)
South America
sa-east-1
South America (Sao Paulo)
Apply
Apply and close
Close

Supported cloud databases





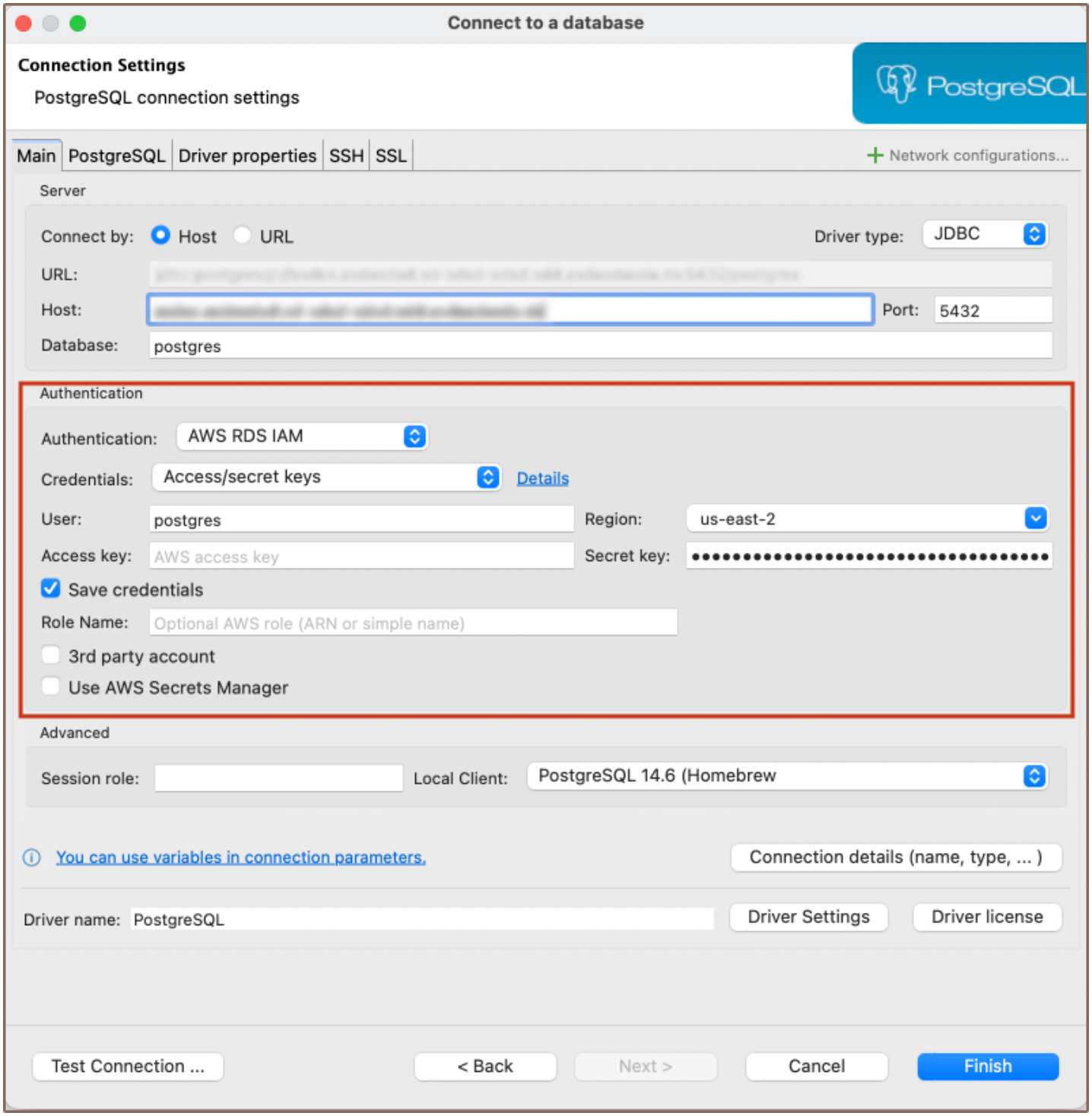
AWS Credentials

Note: This feature is available in Lite, Enterprise, Ultimate and Team editions only.

Overview

DBeaver is integrated with AWS RDS IAM authentication, providing you with the ability to authenticate in AWS to access your cloud databases. There are numerous ways to authorize and authenticate users in DBeaver AWS IAM, and DBeaver supports all the basic ones.

The instructions provided here are intended for the client machine where DBeaver is installed. It is assumed that the necessary configuration on the server side, including the setup of AWS RDS IAM, has already been completed.



Settings

To use IAM authentication in DBeaver, you need to select **AWS RDS IAM** as your authentication method in the connection settings.

Setting	Description
Credentials	Choose between Access/secret keys, AWS profile, or Default credentials for the type of IAM credentials configuration.

User	Input your username.
Region	Optional field to specify the AWS region. If not specified, the global region will be used.
Access key	AWS Access key.
Secret key	AWS Secret key.
Role name	Input the name of the role you want to use.
Save credentials locally	Check this box if you want to save your password locally.
3rd party account	Check this box if you want to access using a 3rd party role-based account.
Use AWS Secrets Manager	Check this box if you want to connect to AWS Secrets Manager to access the database credentials. When selected, a field **Secret Name** will appear where you can input the name of your **AWS Secret**.

Credentials

Choose the type of credentials by selecting the appropriate option from the **Credentials** dropdown menu.

Default credentials

When you use Default Credentials, AWS will then try to determine credentials by using the standard credential providers chain:

1. Java system properties
2. Environment variables
3. Web identity token from AWS STS
4. The shared credentials and config files
5. Amazon ECS container credentials

6. Amazon EC2 instance profile credentials
7. Amazon SSO credentials

Using default credentials is the easiest way to integrate with various Single Sign-On (SSO) and web identity providers, as these providers typically supply credentials through configuration files.

For a more detailed explanation, please refer to the AWS credentials documentation.

To use Default credentials, simply enter your username in the **User** field and select your AWS region.

Access keys

Using the IAM user access key and secret key is the simplest way to authenticate. You just need to input these two keys. You have the option to save them locally for convenience, or, for better security, you can choose to enter them each time you connect to a database.

As with the Default configuration, you need to enter your username and select the AWS region. If you've checked the **Save credentials locally** box, you'll need to fill in the **Access key** and **Secret key** fields. If you haven't checked this box, you'll be prompted to fill in these fields each time you connect to the database.

For more detailed instructions on managing access keys for IAM users, you can refer to the official AWS guide.

AWS Profiles

Just like with default credentials, you also have the option to select a specific credentials profile.

To do this, first select the profile you've already configured. If you need information on how to configure a profile, you can find it below. After selecting a profile, fill in the **User** field and select your AWS region, just like you would with the default credentials.

For more detailed instructions, you can refer to the official AWS guide on credentials config files.

SSO

If you've set up an SSO portal on your AWS account, you can use it for web-based SSO authorization. This SSO support can be activated for both Default and Profile-based AWS authorization types. To use this feature, you need to enable the **Enable SSO** option.

AWS Secrets Manager

If you've set up an **AWS Secret**, you can use it to access your database. This method can be used for both RDS databases and Redshift. You can find instructions on how to create an **AWS Secret** in official guide. Remember, you'll need to fill in the **Password** field.

To use this feature, check the **Use AWS Secrets Manager** box and then fill in the **Secret Name** field.

Note: Make sure that the secret in the same region as the database.

AWS SSO

Note: This feature is available in Lite, Enterprise, Ultimate and Team editions only.

AWS Single Sign-On is a cloud-based single sign-on (SSO) service that makes it easy to centrally manage SSO access to AWS resources.
You do not need to specify any user credentials explicitly in DBeaver connections configuration. All authorization is performed in a web browser in a 3rd party SSO provider, e.g. Google workspace, Microsoft AD portal, Facebook, etc.

AWS CLI

You need to install AWS CLI (Command Line Interface) utilities to enable SSO authorization.

AWS CLI installation

AWS CLI version 2.2 is recommended.

AWS SSO configuration

If you are in a corporate environment where all AWS configurations are provided by system administrators then you do not need to configure SSO parameters. Otherwise, you need to open the command shell (`win+R`), enter `aws configure sso`, press enter, and provide the required parameters. Read configuration instructions for the details.

Restart DBeaver after the AWS CLI SSO configuration will be finished.

Connection configuration

In the DBeaver database connection dialog you need to:

- Set Authentication to `AWS IAM`.

- Set Credentials to `AWS Profile`.
- Choose the profile which was configured with AWS SSO (see the previous chapter).
- Click on the `Enable SSO` check.

Now you can connect. DBeaver will open a web browser with SSO authorization.

AWS Permissions

AWS managed policies you need to assign to user for make CloudExplorer works.

Service	Read Policy	Write Policy
Login	-	
Cloud Explorer	-	
S3	AmazonS3ReadOnlyAccess	AmazonS3FullAccess
Redshift	AmazonRedshiftReadOnlyAccess	AmazonRedshiftFullAccess
RDS	AmazonRDSReadOnlyAccess	AmazonRDSFullAccess
DynamoDB	AmazonDynamoDBReadOnlyAccess	AmazonDynamoDBFullAccess
DocumentDB	AmazonDocDBReadOnlyAccess	AmazonDocDBFullAccess
Keyspaces	AmazonKeyspacesReadOnlyAccess	AmazonKeyspacesFullAccess
Athena	"lakeformation:Describe*", *"lakeformation:Get*"*, "lakeformation:List*", *"sns:ListTopics"*, *"sns:GetTopicAttributes"*, *"s3:Describe*"*, "s3:Get*", *"s3:List*"*, "s3:Put*", *"glue:Get*"*, "glue:List*", *"glue:BatchGet*"*, "athena:List*", *"athena:Get*"*, "athena:Batch*",	AmazonAthenaFullAccess

	"athena:StartQueryExecution", "athena:StopQueryExecution"	
Redis	AmazonElastiCacheReadOnlyAccess	AmazonElastiCacheFullAccess
SSO		

GCP Credentials

DBeaver is integrated with Google Cloud IAM authentication.

Thus it provides the possibility to authenticate in GCP to access your cloud databases.

Default credentials

When you use Default Credentials, Google Cloud will then try to determine credentials by using the standard credential providers chain:

1. Environment variables (GOOGLE_APPLICATION_CREDENTIALS)
2. Identity token from GCP CLI
3. The shared user or service credentials and config files (usually application_default_credentials.json in AppData)
4. Google Compute Engine

Using default credentials is essentially the simplest way to integrate with various SSO providers and web identity providers as they usually provide credentials through config files.

Please read the GCP authentication documentation for a detailed explanation.

Access key file

You can provide the path to your service credentials or user credentials file in the "Configuration" field.

You can read more about User and Service authentication here

Web browser or Single Sign On

Google Cloud Shell will be used for authethication. If your GCP account has a configured SSO portal, you can use a web-based SSO authorization.

GCP SSO

Note: This feature is available in Lite, Enterprise, Ultimate and Team editions only.

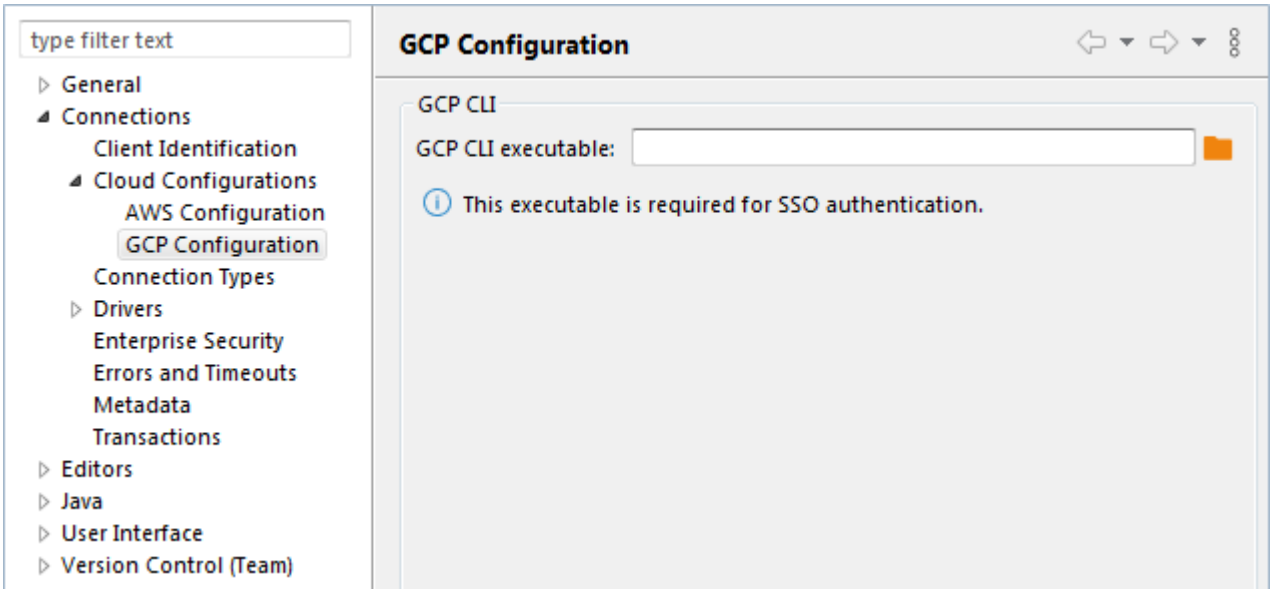
GCP web browser-based authentication allows you not to use a service or user-based key file, or other default authorization methods. You can provide access to your connection by authorization through the Google account.

GCP CLI

To enable SSO authorization, you need to install GCP CLI (Command Line Interface) utilities.

GCP CLI installation

If DBeaver didn't find your executable CLI file (error message: "Authentication error: Error running GCP CLI. Is it installed on the local machine?)", you can add the path for your executable file manually in the Preferences -> Connections -> Cloud Configurations -> GCP Configuration



GCP web browser-based authentication configuration

First `gcloud auth print-access-token` request to your CLI will be used. If token will be not empty, then this token will be used for the authentication. If token will be empty, then command `gcloud auth login` will be executed, which will open the web-browser and offer to choose your Google account.

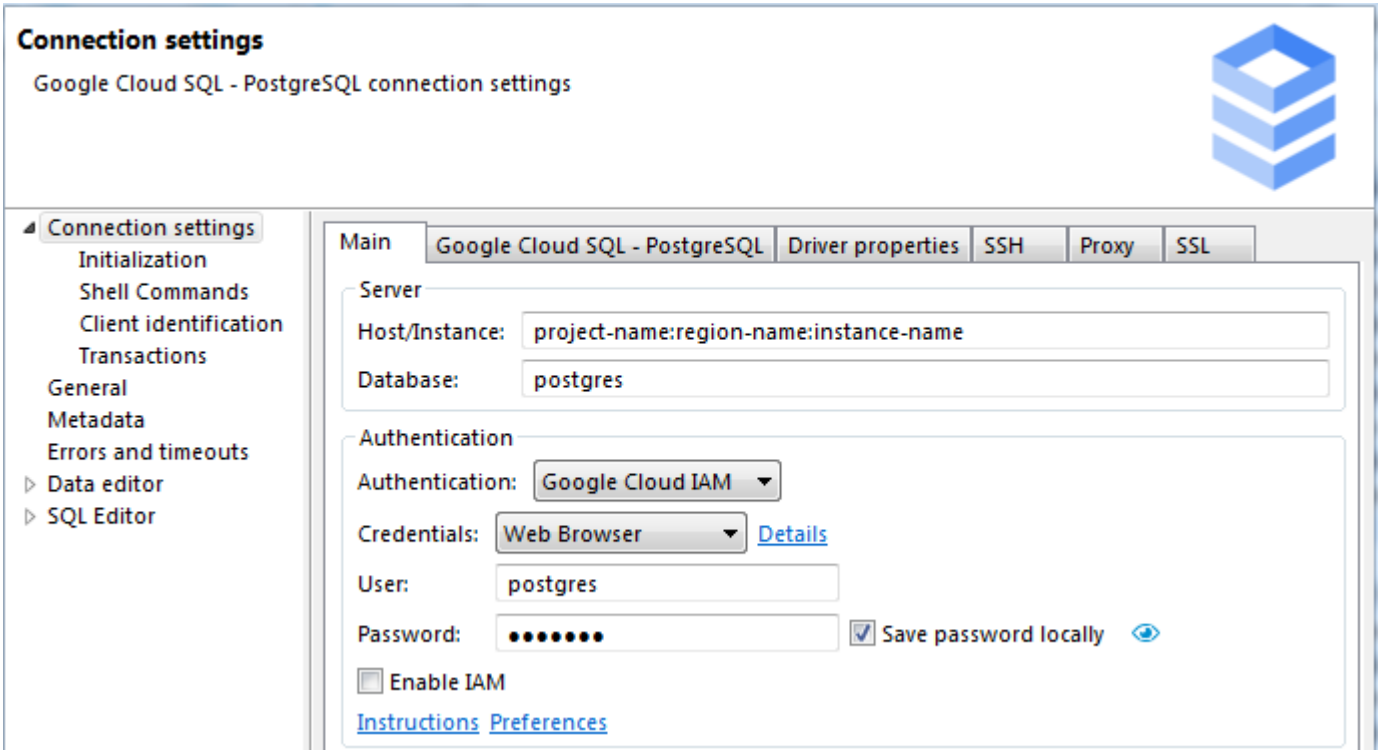
Connection configuration

In the DBeaver database connection dialog, you need to:

- Set Authentication to `Google Cloud IAM`.
- Set Credentials to `Web Browser`.
- Add user and password info if they needed

Now you can connect. DBeaver will open a web browser with SSO authorization. Choose your account with Goggle projects on the web page.

As example:



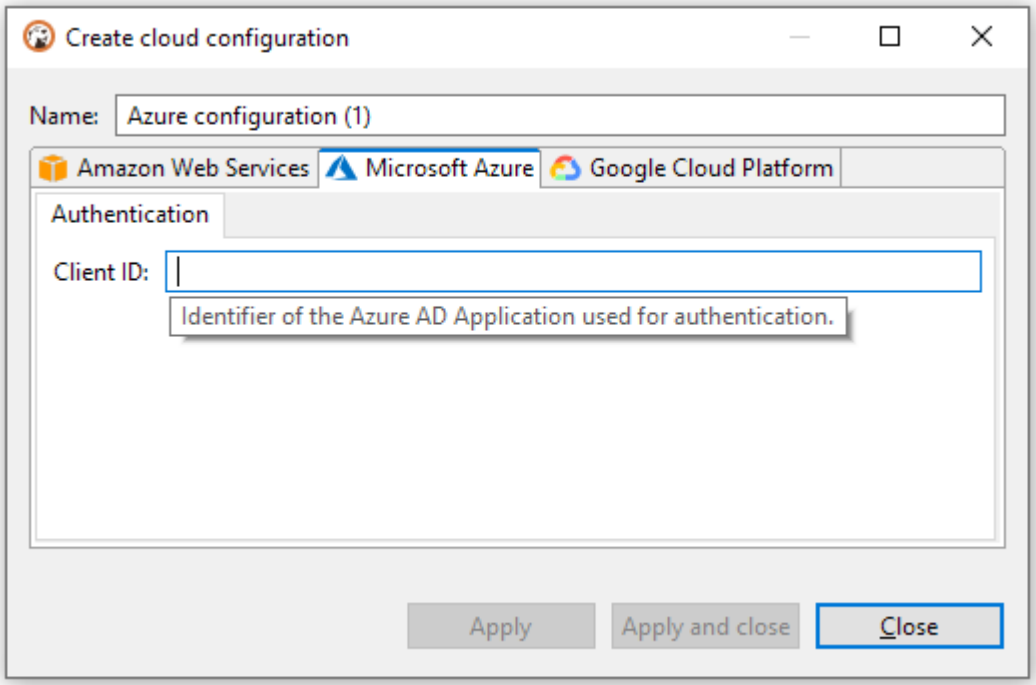
Azure

Azure Cloud Explorer

Note: This functionality is available in Ultimate and Team editions only.

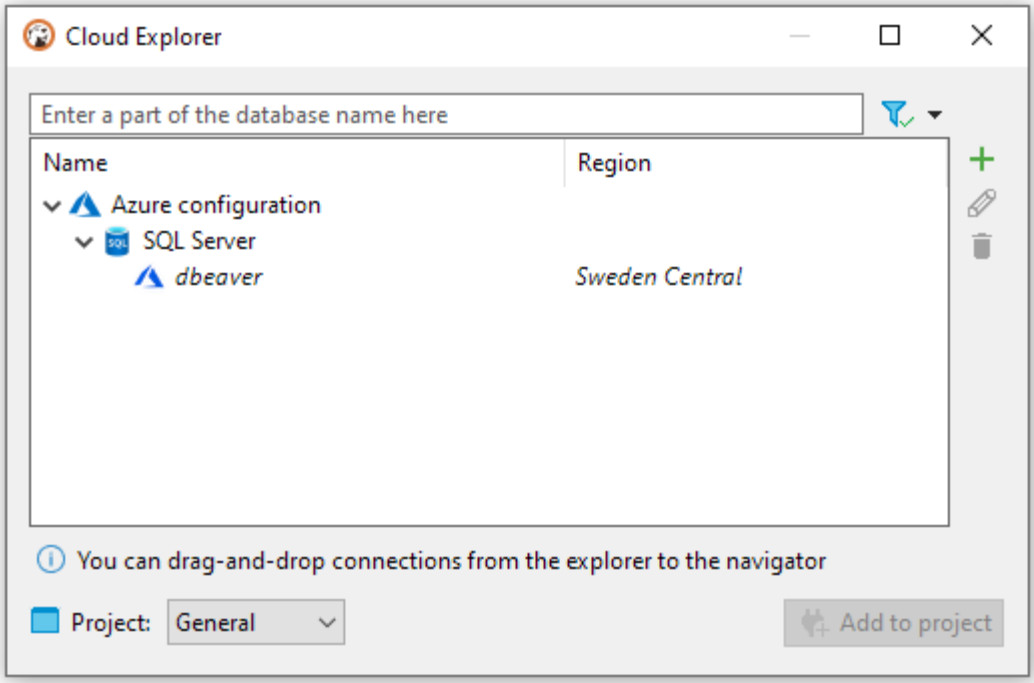
Cloud configuration

You have to provide just the ID of the application (client) that is used for authentication and database access:



Supported services

As of DBeaver 22.3, the only supported service is Azure SQL.


Cloud Explorer
Enter a part of the database name here
Name
Region
Azure configuration
SQL Server
dbeaver
Sweden Central
You can drag-and-drop connections from the explorer to the navigator
Project:
General
Add to project

Enterprise Edition

DBeaver Enterprise Edition (EE) is a commercial version of DBeaver CE. The EE license is perpetual and comes with one or two years of premium support.

DBeaver Enterprise is a comprehensive database management tool with advanced features for database management and administration, SQL editing and execution, and development tools. The EE version includes all features of the CE version plus:

- All popular JDBC drivers are included in the EE distribution so you will not need to download/configure them separately.
- Support of Databases:

Cloud Databases	NoSQL Databases	Embedded
Apache Cassandra	Google BigTable	WMI (Windows Management Instrumentation)
MongoDB	AWS Keyspaces	
InfluxDB	AWS DocumentDB	
Couchbase	AWS Timestream	
CouchDB	AWS DynamoDB	
Yugabyte CQL	AWS Neptune	
	Redis	

- Additional EE plugins:
  - Office formats support (XLS) for data import
  - Visual Query Builder
  - Schema/table compare, diff DDL generation
  - Data compare

- Analytical charts rendering
- Persistent query database. Search in query history
- Eclipse Marketplace (provides easy installation of any additional Eclipse plugins)
- Mock data generators
- Version control support
- Automatic proxy configuration (PAC)
- ERD edit mode
- Retaining query history (after restart)
- SQL execution plan diagram
- Tasks (Composite Tasks, Email results, Task scheduler)
- Kerberos UI Authentication
- Spelling
- Command Line Interface
- A possibility for us to support and develop the Community version, add new features faster, provide better support, and much more.

Even if you do not need enterprise features you can purchase a license as a form of donation. Thank you!

Apache Hive/Spark/Impala

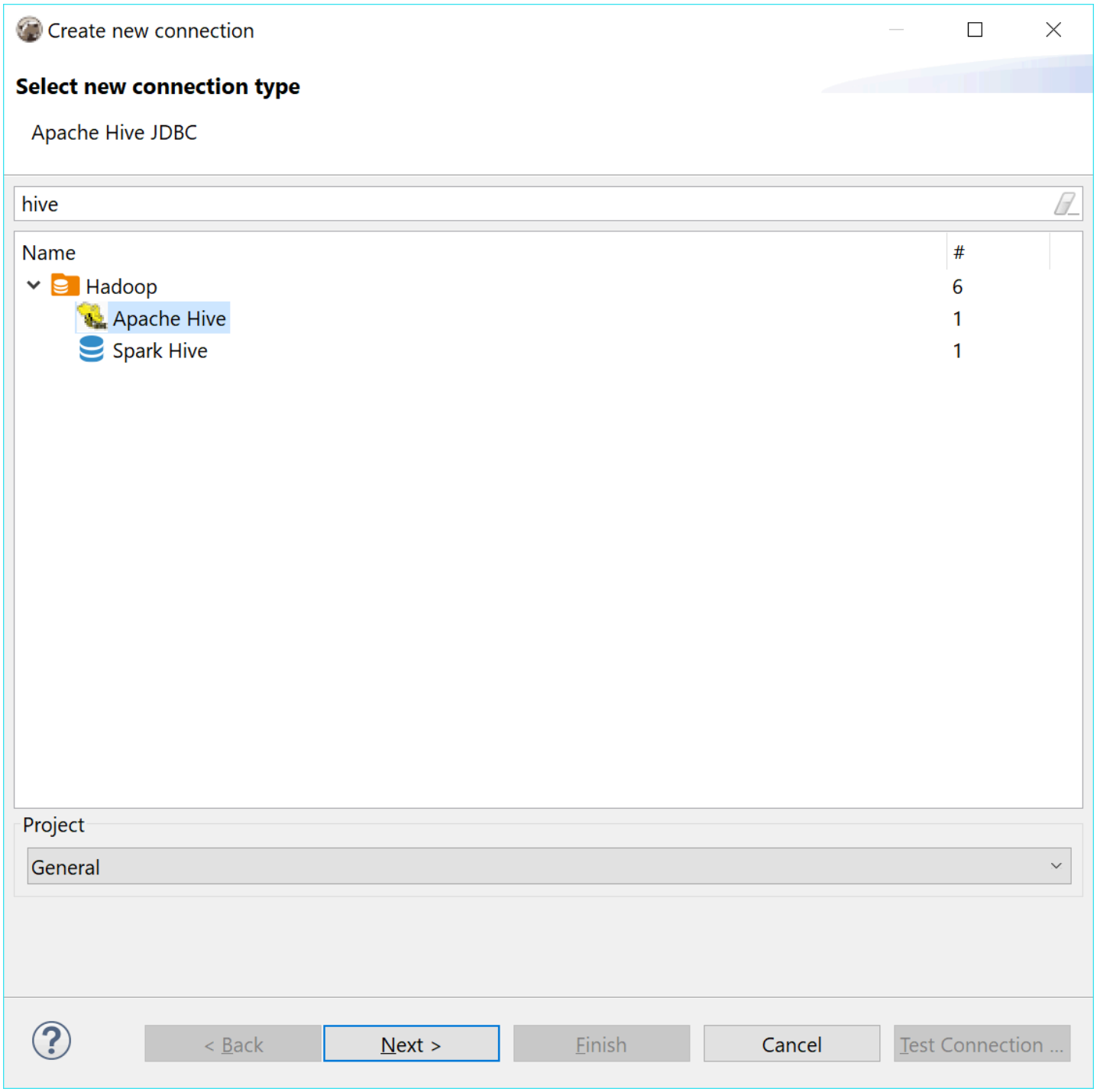
Apache Hive

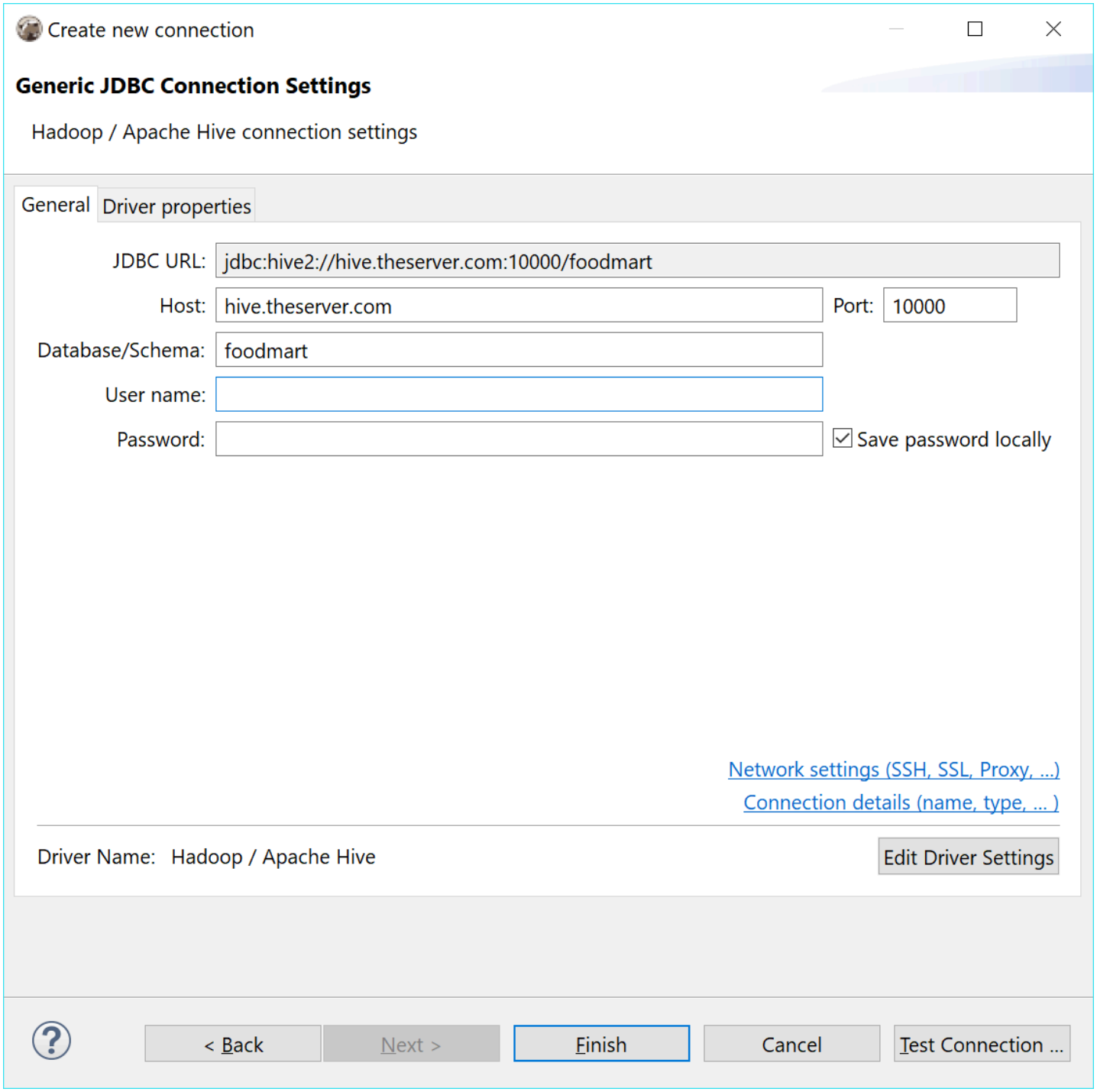
Hive is a Hadoop-based storage system. Hive uses a special SQL dialect (HiveQL) to operate with data and metadata. Generally, it is quite similar to SQL.

There are multiple implementations of storage systems which utilize Hive on the server-side - including Apache Spark, Impala, etc. Most of them support the standard Hive JDBC driver which is used in DBeaver to communicate with the server.

DBeaver uses a so-called Hive JDBC Uber Jar driver (https://github.com/timveil/hive-jdbc-uber-jar) which includes all necessary dependencies. You do not need to download anything - DBeaver will download everything automatically (if you have internet access).

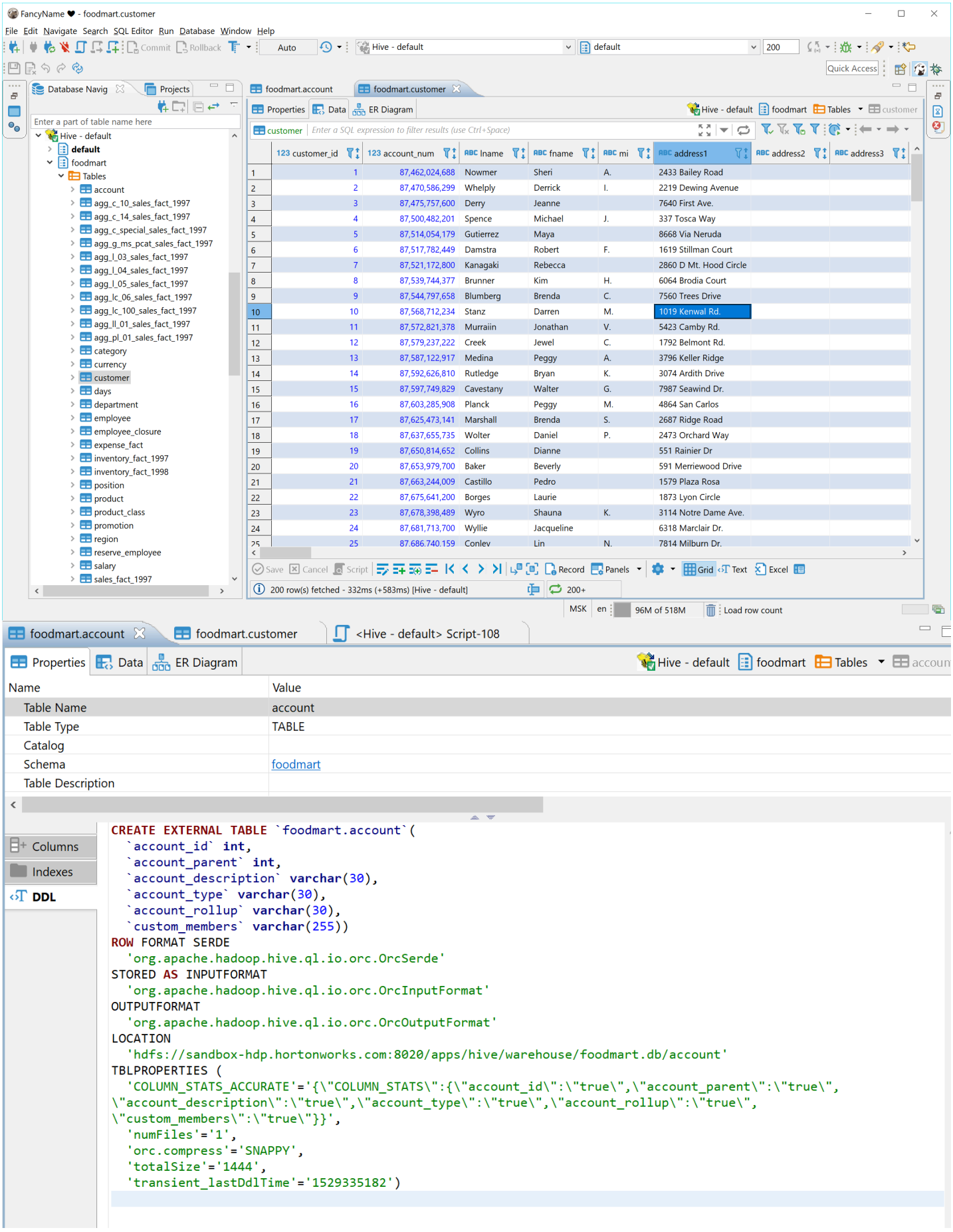
Connection setup


Create new connection
Select new connection type
Apache Hive JDBC
hive
Name
#
Hadoop
6
Apache Hive
1
Spark Hive
1
Project
General
< Back
Next >
Finish
Cancel
Test Connection ...





Schema/data browser



Limitations

Hive does not support referential integrity so you will not see primary keys or foreign keys. ER diagrams also do no make much sense.

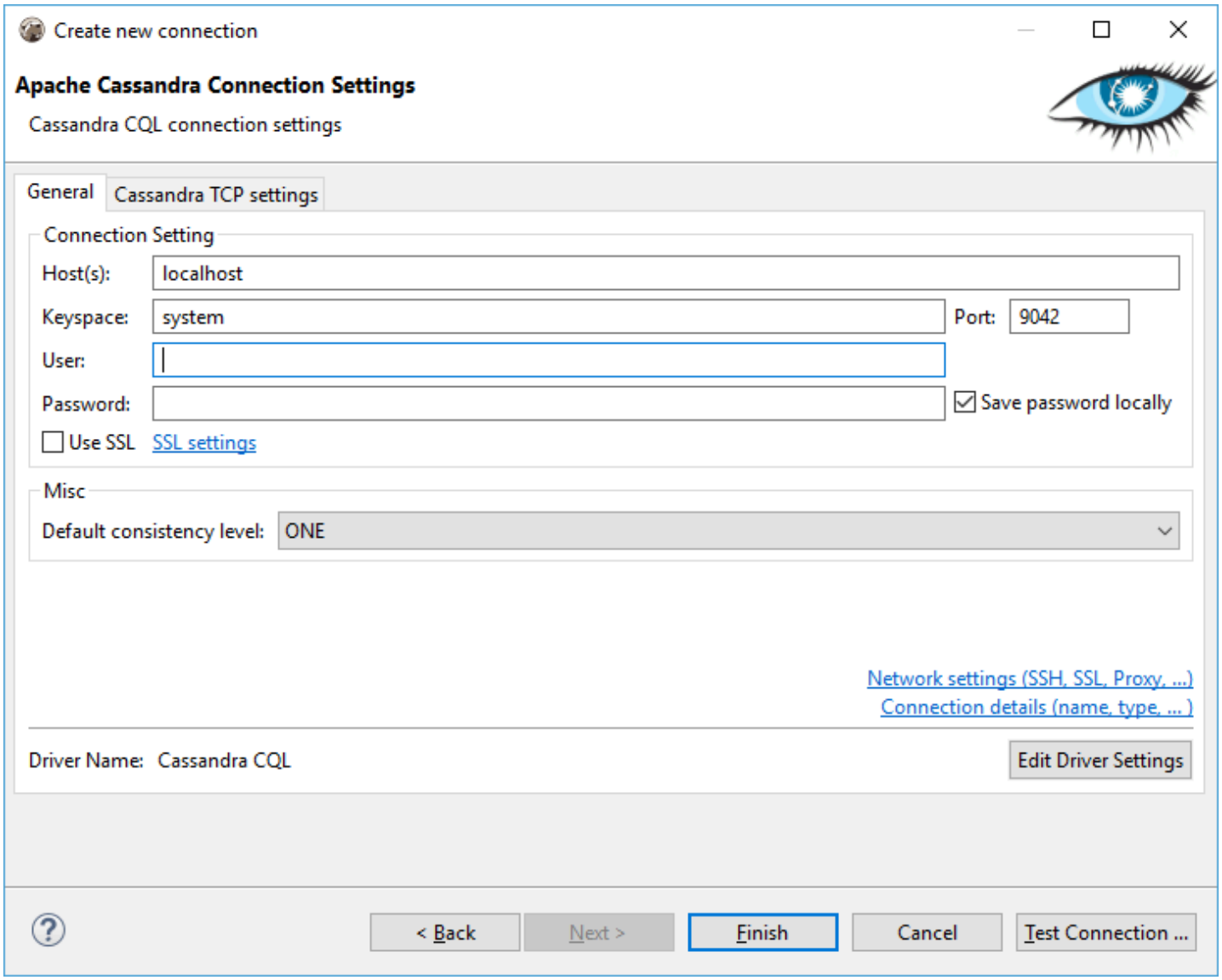
Cassandra

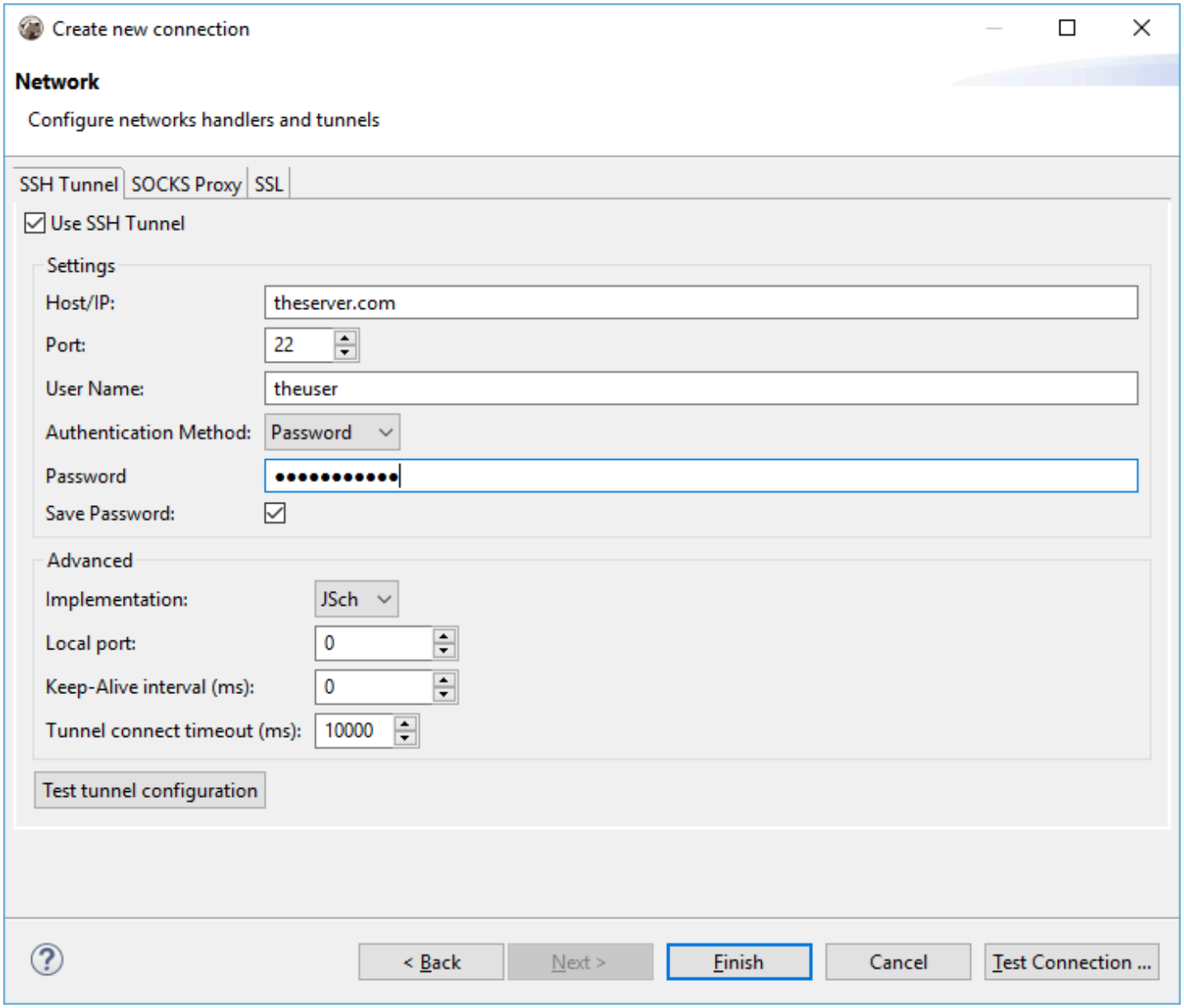
Overview

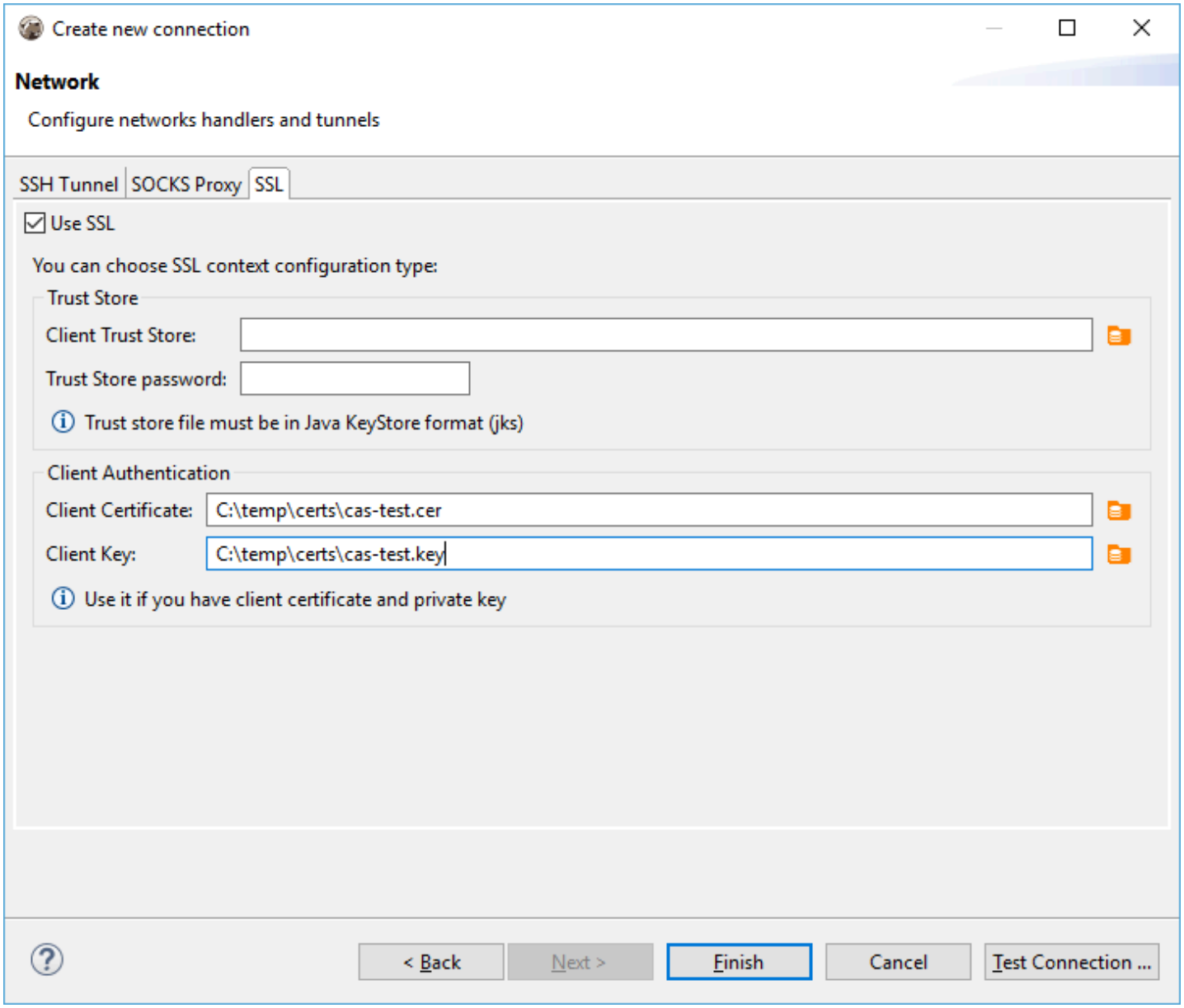
DBeaver EE supports Cassandra schema browser, data viewer and CQL queries execution. It also supports various administrative tools.

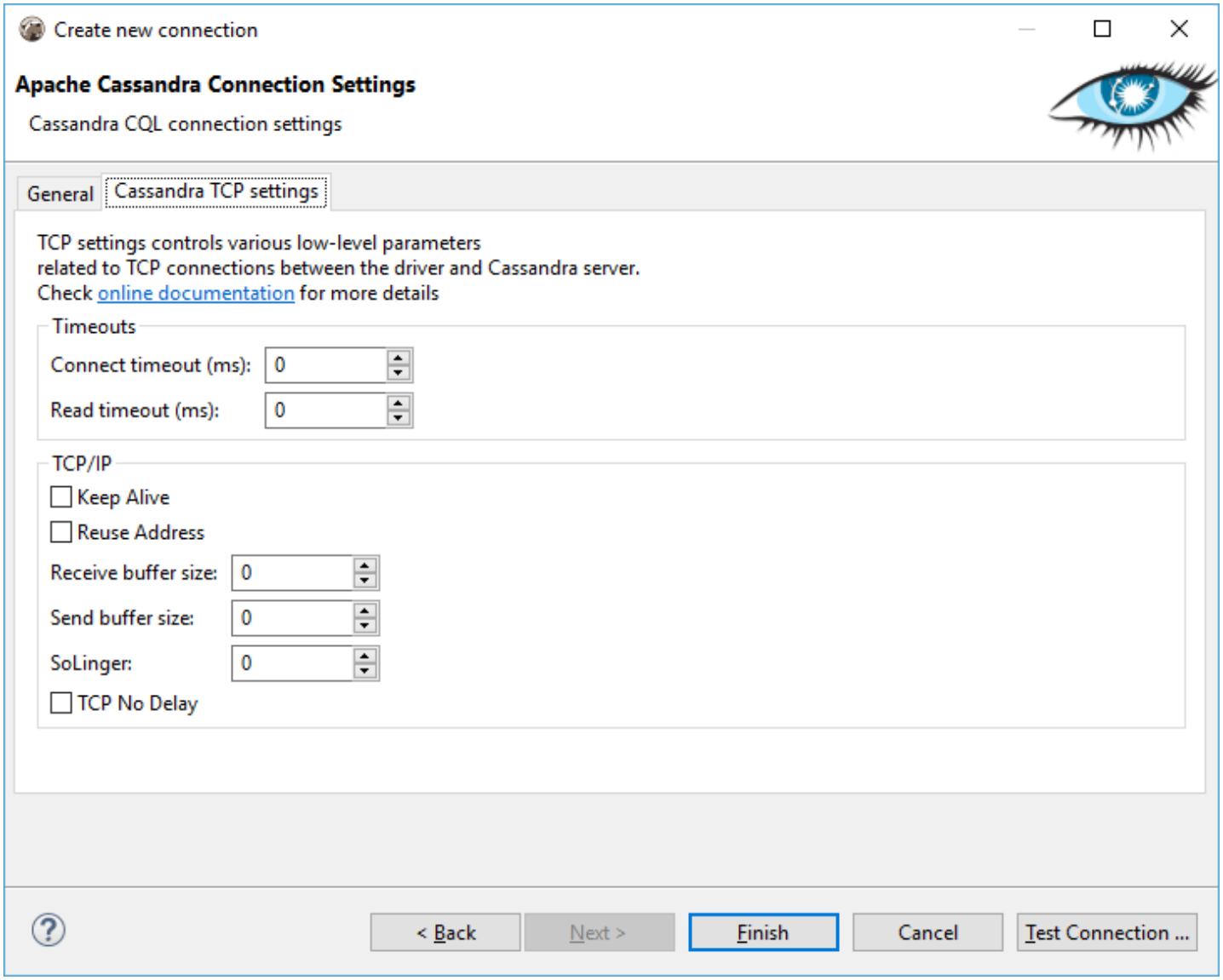
Connecting to Cassandra cluster

You can connect directly to a server or use SSH tunneling or SOCKS proxy. DBeaver uses the DataStax Java driver to operate with a server. It supports Cassandra servers 2.x, 3.x or higher.


Create new connection
Apache Cassandra Connection Settings
Cassandra CQL connection settings
General
Cassandra TCP settings
Connection Setting
Host(s):
localhost
Keyspace:
system
Port:
9042
User:
Password:
Save password locally
Use SSL
SSL settings
Misc
Default consistency level:
ONE
Network settings (SSH, SSL, Proxy, ...)
Connection details (name, type, ...)
Driver Name: Cassandra CQL
Edit Driver Settings
< Back
Next >
Finish
Cancel
Test Connection ...


Create new connection
Network
Configure networks handlers and tunnels
SSH Tunnel
SOCKS Proxy
SSL
Use SSH Tunnel
Settings
Host/IP:
theserver.com
Port:
22
User Name:
theuser
Authentication Method:
Password
Password
Save Password:
Advanced
Implementation:
JSch
Local port:
0
Keep-Alive interval (ms):
0
Tunnel connect timeout (ms):
10000
Test tunnel configuration
< Back
Next >
Finish
Cancel
Test Connection ...

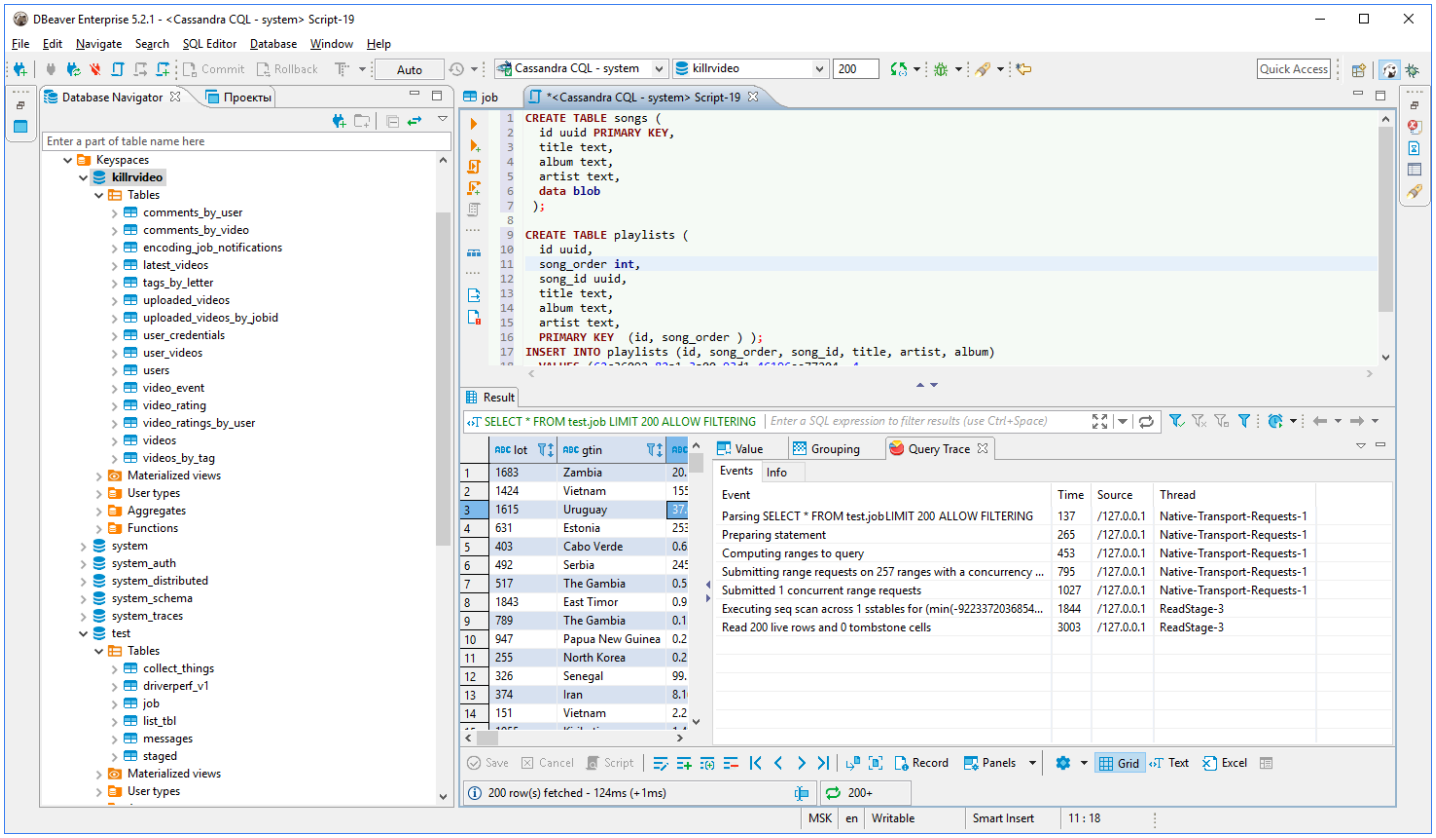

Create new connection
Network
Configure networks handlers and tunnels
SSH Tunnel
SOCKS Proxy
SSL
Use SSL
You can choose SSL context configuration type:
Trust Store
Client Trust Store:
Trust Store password:
Trust store file must be in Java KeyStore format (jks)
Client Authentication
Client Certificate:
C:\temp\certs\cas-test.cer
Client Key:
C:\temp\certs\cas-test.key
Use it if you have client certificate and private key
< Back
Next >
Finish
Cancel
Test Connection ...

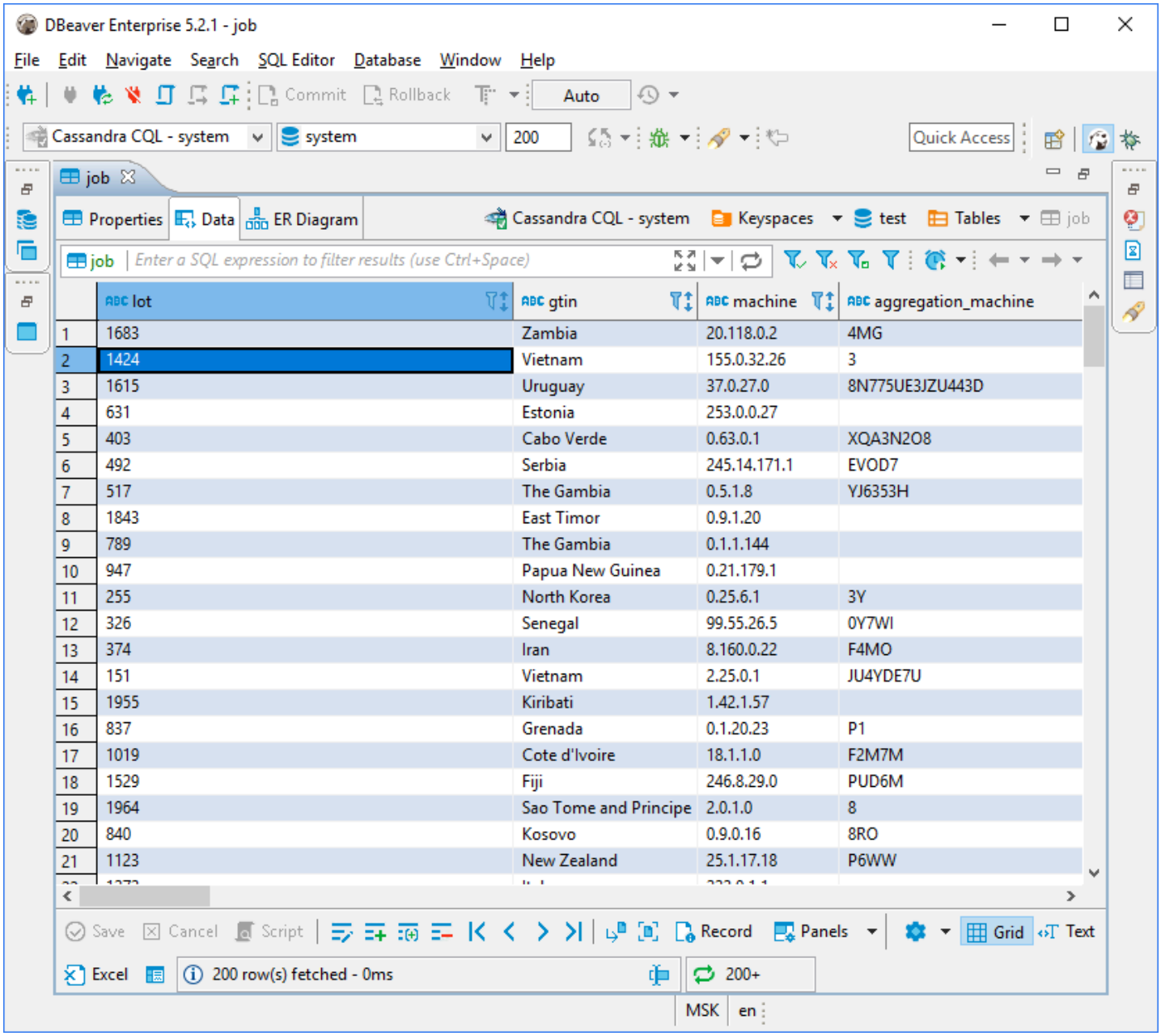


Browsing Cassandra tables

You can browse, view, edit and filter Cassandra tables the same way as with regular (relational) tables. However, being a distributed key-value database, Cassandra does not support any kind of referential integrity. There are no foreign keys, references, etc.

You should note that Cassandra has a very advanced (comparing to relational databases) data type system. Each column may be a collection, map, or set of values (with very big number of values). In some cases this makes browsing data in the "Grid" mode inconvenient.


DBeaver Enterprise 5.2.1 - <Cassandra CQL - system> Script-19
File Edit Navigate Search SQL Editor Database Window Help
Database Navigator
Enter a part of table name here
Keyspaces
killrvideo
Tables
comments_by_user
comments_by_video
encoding_job_notifications
latest_videos
tags_by_letter
uploaded_videos
uploaded_videos_by_jobid
user_credentials
user_videos
users
video_event
video_rating
video_ratings_by_user
videos
videos_by_tag
Materialized views
User types
Aggregates
Functions
system
system_auth
system_distributed
system_schema
system_traces
test
collect_things
driverperf_v1
job
list_tbl
messages
staged
CREATE TABLE songs (
id uuid PRIMARY KEY,
title text,
album text,
artist text,
data blob
);
CREATE TABLE playlists (
id uuid,
song_order int,
song_id uuid,
title text,
album text,
artist text,
PRIMARY KEY (id, song_order));
INSERT INTO playlists (id, song_order, song_id, title, artist, album)
Result
SELECT * FROM test.job LIMIT 200 ALLOW FILTERING
Query Trace
Events
Info
Parsing SELECT * FROM test.job LIMIT 200 ALLOW FILTERING
Preparing statement
Computing ranges to query
Submitting range requests on 257 ranges with a concurrency ...
Submitted 1 concurrent range requests
Executing seq scan across 1 sstables for (min(-9223372036854...
Read 200 live rows and 0 tombstone cells
200 row(s) fetched - 124ms (+1ms)

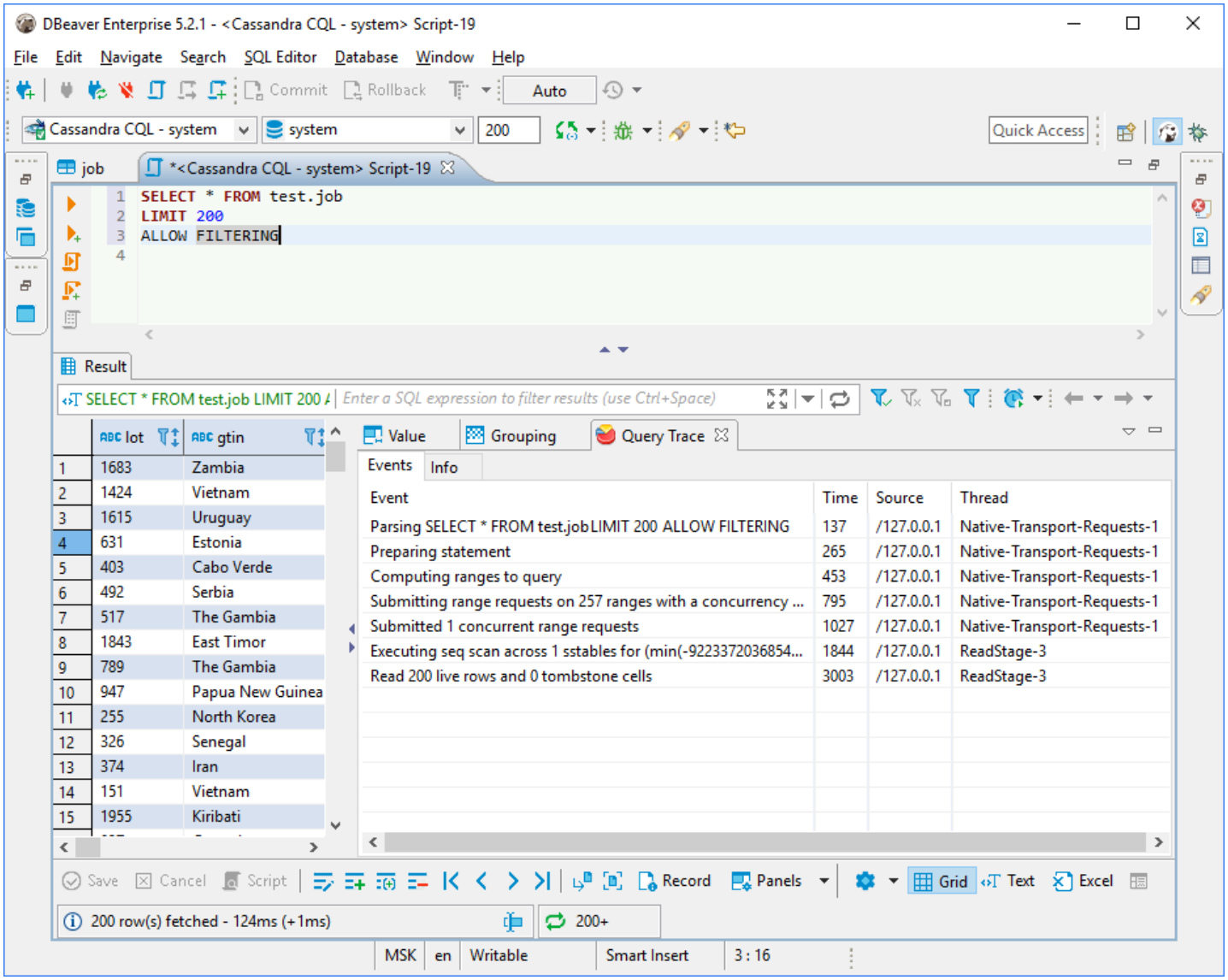


Executing CQL

CQL Cassandra Query Language is a very simple SQL language dialect.

It supports simple SELECT queries, DDL statements (like CREATE TABLE) and some other query types.

You can use the standard DBeaver SQL editor to execute CQL queries. DBeaver supports Cassandra query execution, results scrolling, data export/import, mock data generation and other features. Data viewer (of individual tables or custom CQL query results) query tracing is supported.



ERD

Physical ERD (Entity Relation Diagram) does not make much sense for Cassandra as there are no foreign keys.

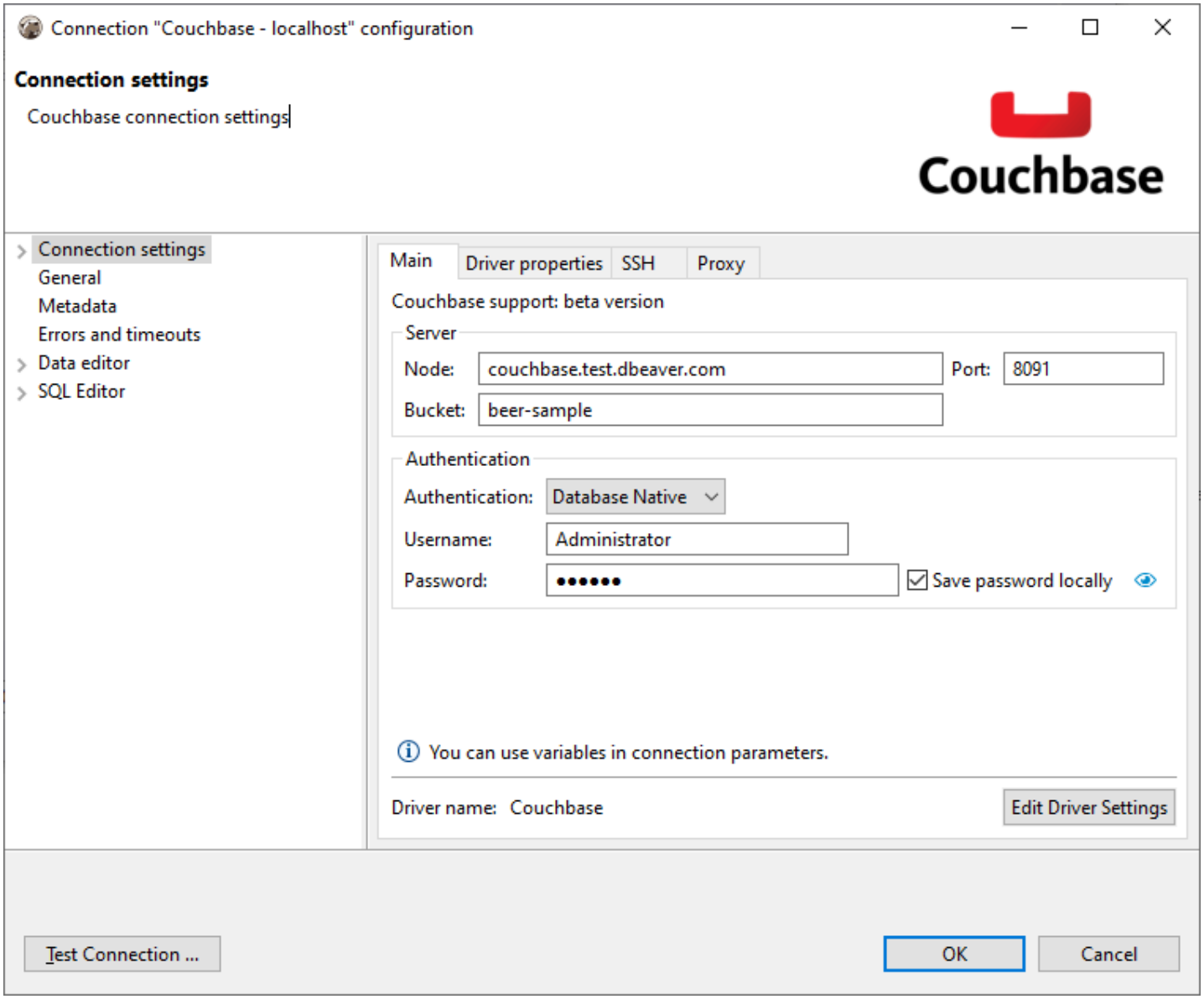
However, you can make you own custom ERD and connect an actual Cassandra table with each other using logical associations.

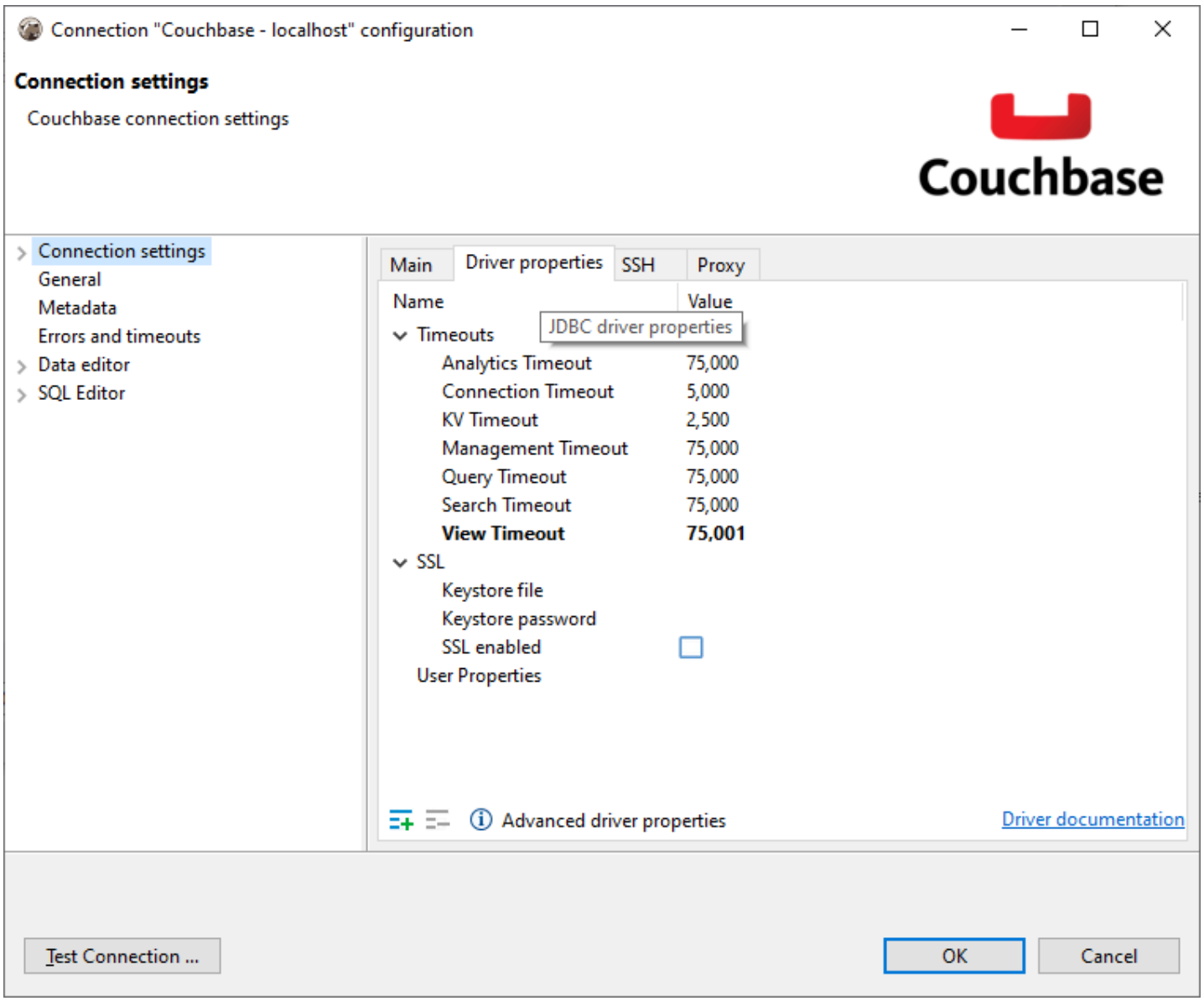
Couchbase

Connecting to Couchbase server

Note: This feature is available in Enterprise, Ultimate and Team editions only.

Couchbase client uses multiple ports to connect to a cluster (8091-8096, 9140, etc). Some of these ports are dynamic (i.e. depend on server settings) and cannot be overwritten. It makes SSH tunnelling impossible. Thus, if you work with a remote Couchbase deployed behind a firewall, you will need to setup a VPN connection or SOCKS proxy.

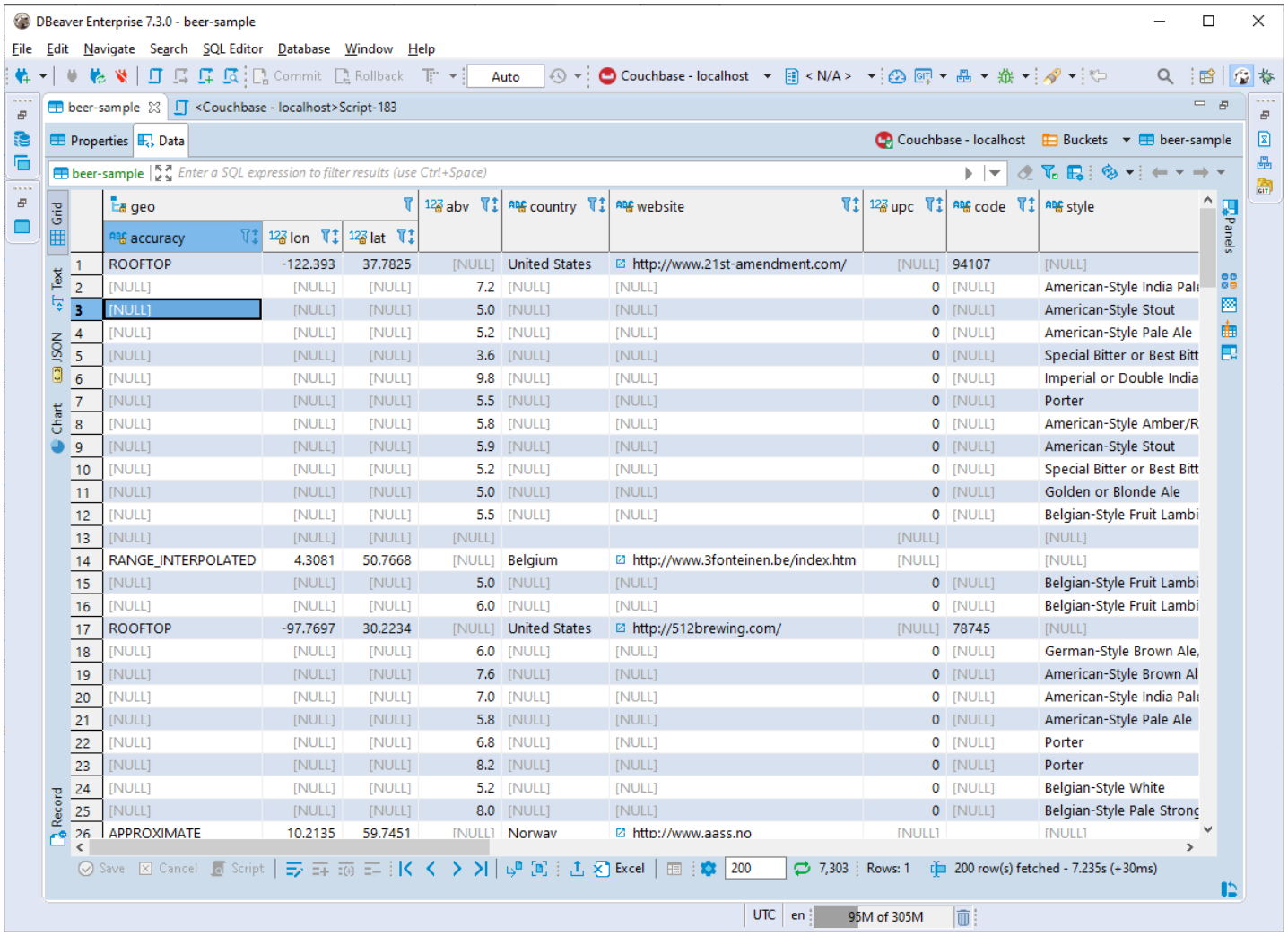

Connection "Couchbase - localhost" configuration
Connection settings
Couchbase connection settings
Couchbase
Connection settings
General
Metadata
Errors and timeouts
Data editor
SQL Editor
Main
Driver properties
SSH
Proxy
Couchbase support: beta version
Server
Node:
couchbase.test.dbeaver.com
Port:
8091
Bucket:
beer-sample
Authentication
Authentication:
Database Native
Username:
Administrator
Password:
Save password locally
You can use variables in connection parameters.
Driver name: Couchbase
Edit Driver Settings
Test Connection ...
OK
Cancel



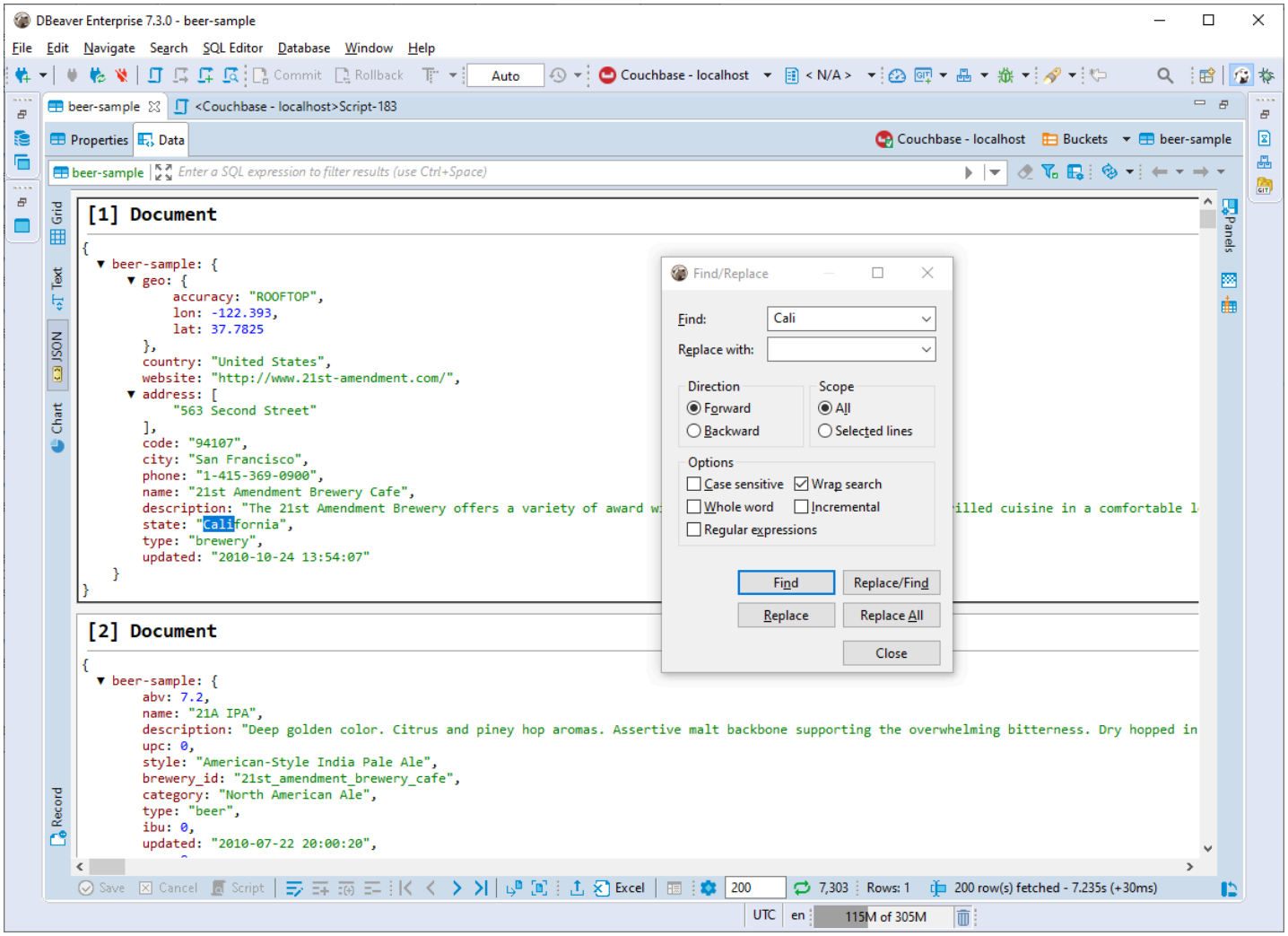
Viewing and editing Couchbase tables

Couchbase is a document-oriented database. It means that all documents may have different structures.

You can view/edit buckets content, such as standard relational tables (grid/plain text presentations) or JSON documents.



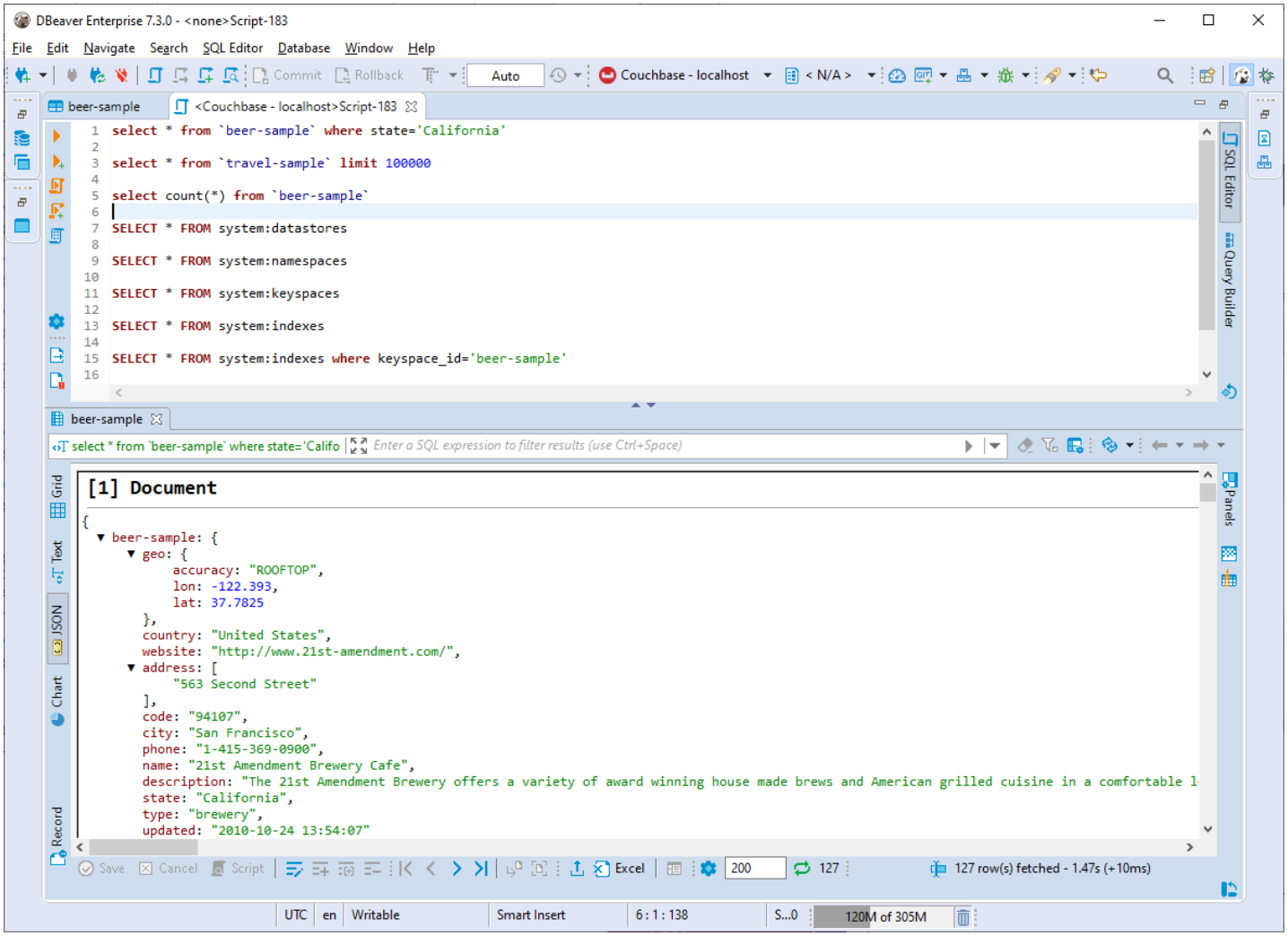
	geo			abv	country	website	upc	code	style
	accuracy	lon	lat						
1	ROOFTOP	-122.393	37.7825	[NULL]	United States	http://www.21st-amendment.com/	[NULL]	94107	[NULL]
2	[NULL]	[NULL]	[NULL]	7.2	[NULL]	[NULL]	0	[NULL]	American-Style India Pale
3	[NULL]	[NULL]	[NULL]	5.0	[NULL]	[NULL]	0	[NULL]	American-Style Stout
4	[NULL]	[NULL]	[NULL]	5.2	[NULL]	[NULL]	0	[NULL]	American-Style Pale Ale
5	[NULL]	[NULL]	[NULL]	3.6	[NULL]	[NULL]	0	[NULL]	Special Bitter or Best Bitt
6	[NULL]	[NULL]	[NULL]	9.8	[NULL]	[NULL]	0	[NULL]	Imperial or Double India
7	[NULL]	[NULL]	[NULL]	5.5	[NULL]	[NULL]	0	[NULL]	Porter
8	[NULL]	[NULL]	[NULL]	5.8	[NULL]	[NULL]	0	[NULL]	American-Style Amber/R
9	[NULL]	[NULL]	[NULL]	5.9	[NULL]	[NULL]	0	[NULL]	American-Style Stout
10	[NULL]	[NULL]	[NULL]	5.2	[NULL]	[NULL]	0	[NULL]	Special Bitter or Best Bitt
11	[NULL]	[NULL]	[NULL]	5.0	[NULL]	[NULL]	0	[NULL]	Golden or Blonde Ale
12	[NULL]	[NULL]	[NULL]	5.5	[NULL]	[NULL]	0	[NULL]	Belgian-Style Fruit Lambi
13	[NULL]	[NULL]	[NULL]	[NULL]			[NULL]		[NULL]
14	RANGE_INTERPOLATED	4.3081	50.7668	[NULL]	Belgium	http://www.3fonteinen.be/index.htm	[NULL]		[NULL]
15	[NULL]	[NULL]	[NULL]	5.0	[NULL]	[NULL]	0	[NULL]	Belgian-Style Fruit Lambi
16	[NULL]	[NULL]	[NULL]	6.0	[NULL]	[NULL]	0	[NULL]	Belgian-Style Fruit Lambi
17	ROOFTOP	-97.7697	30.2234	[NULL]	United States	http://512brewing.com/	[NULL]	78745	[NULL]
18	[NULL]	[NULL]	[NULL]	6.0	[NULL]	[NULL]	0	[NULL]	German-Style Brown Ale,
19	[NULL]	[NULL]	[NULL]	7.6	[NULL]	[NULL]	0	[NULL]	American-Style Brown Al
20	[NULL]	[NULL]	[NULL]	7.0	[NULL]	[NULL]	0	[NULL]	American-Style India Pale
21	[NULL]	[NULL]	[NULL]	5.8	[NULL]	[NULL]	0	[NULL]	American-Style Pale Ale
22	[NULL]	[NULL]	[NULL]	6.8	[NULL]	[NULL]	0	[NULL]	Porter
23	[NULL]	[NULL]	[NULL]	8.2	[NULL]	[NULL]	0	[NULL]	Porter
24	[NULL]	[NULL]	[NULL]	5.2	[NULL]	[NULL]	0	[NULL]	Belgian-Style White
25	[NULL]	[NULL]	[NULL]	8.0	[NULL]	[NULL]	0	[NULL]	Belgian-Style Pale Strong
26	APPROXIMATE	10.2135	59.7451	[NULL]	Norway	http://www.aass.no	[NULL]		[NULL]



Executing Couchbase queries

Couchbase uses N1QL language for queries. It is very similar to the standard SQL language.

```
SELECT country FROM `travel-sample` WHERE name = "Excel Airways";
```


DBeaver Enterprise 7.3.0 - <none>Script-183
File Edit Navigate Search SQL Editor Database Window Help
Commit Rollback Auto Couchbase - localhost < N/A >
beer-sample
<Couchbase - localhost>Script-183
1 select * from `beer-sample` where state='California'
3 select * from `travel-sample` limit 100000
5 select count(*) from `beer-sample`
7 SELECT * FROM system:datastores
9 SELECT * FROM system:namespaces
11 SELECT * FROM system:keyspaces
13 SELECT * FROM system:indexes
15 SELECT * FROM system:indexes where keyspace_id='beer-sample'
SQL Editor
Query Builder
beer-sample
select * from `beer-sample` where state='Califo
Enter a SQL expression to filter results (use Ctrl+Space)
Grid
Text
JSON
Chart
Record
Panels
[1] Document
{
beer-sample: {
geo: {
accuracy: "ROOFTOP",
lon: -122.393,
lat: 37.7825
},
country: "United States",
website: "http://www.21st-amendment.com/",
address: [
"563 Second Street"
],
code: "94107",
city: "San Francisco",
phone: "1-415-369-0900",
name: "21st Amendment Brewery Cafe",
description: "The 21st Amendment Brewery offers a variety of award winning house made brews and American grilled cuisine in a comfortable l
state: "California",
type: "brewery",
updated: "2010-10-24 13:54:07"
Save Cancel Script Excel 200 127
127 row(s) fetched - 1.47s (+10ms)
UTC en Writable Smart Insert 6 : 1 : 138 S...0 120M of 305M

InfluxDB

Overview

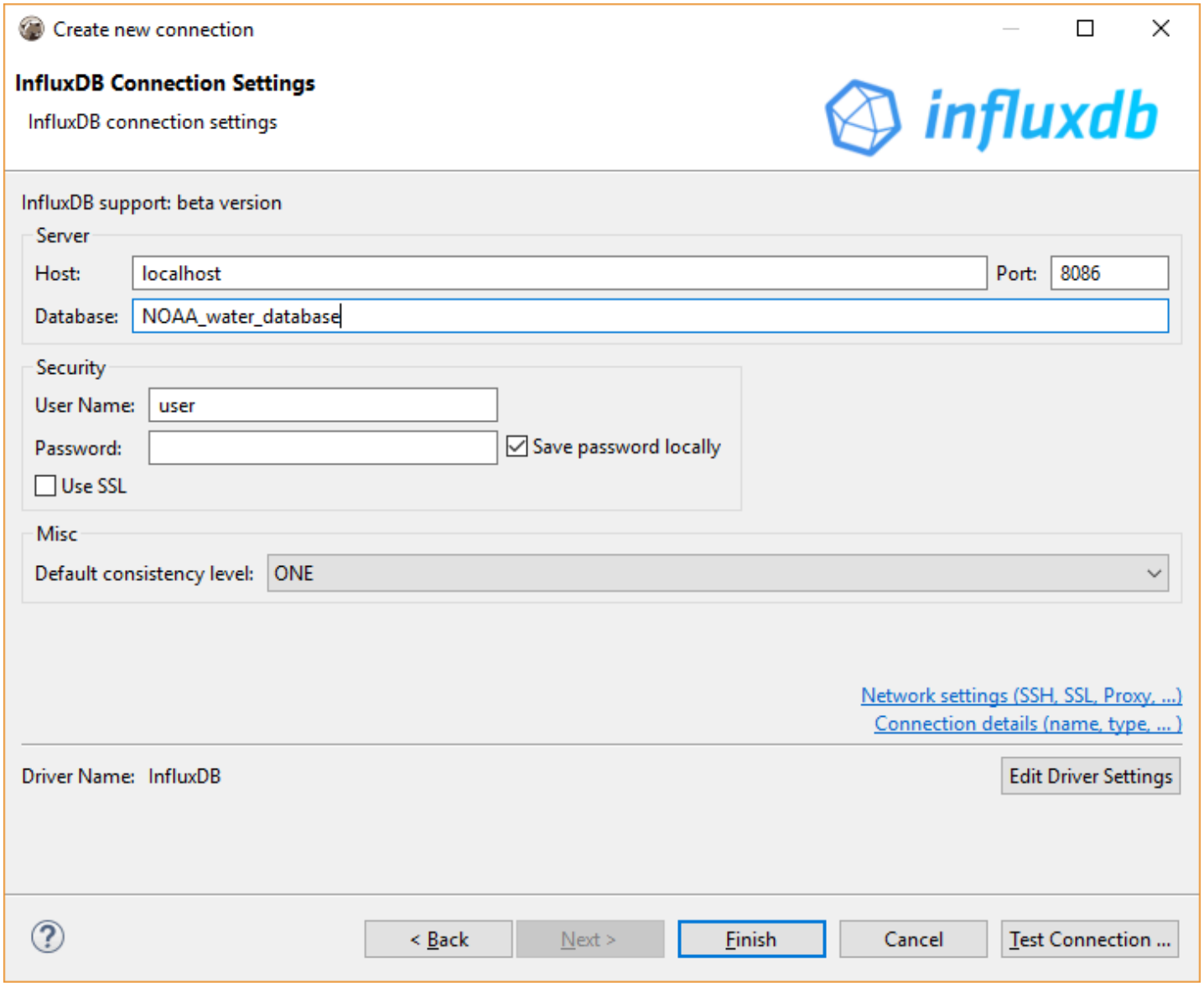
DBeaver EE supports InfluxDB schema browser, data viewer and InfluxQL queries execution.

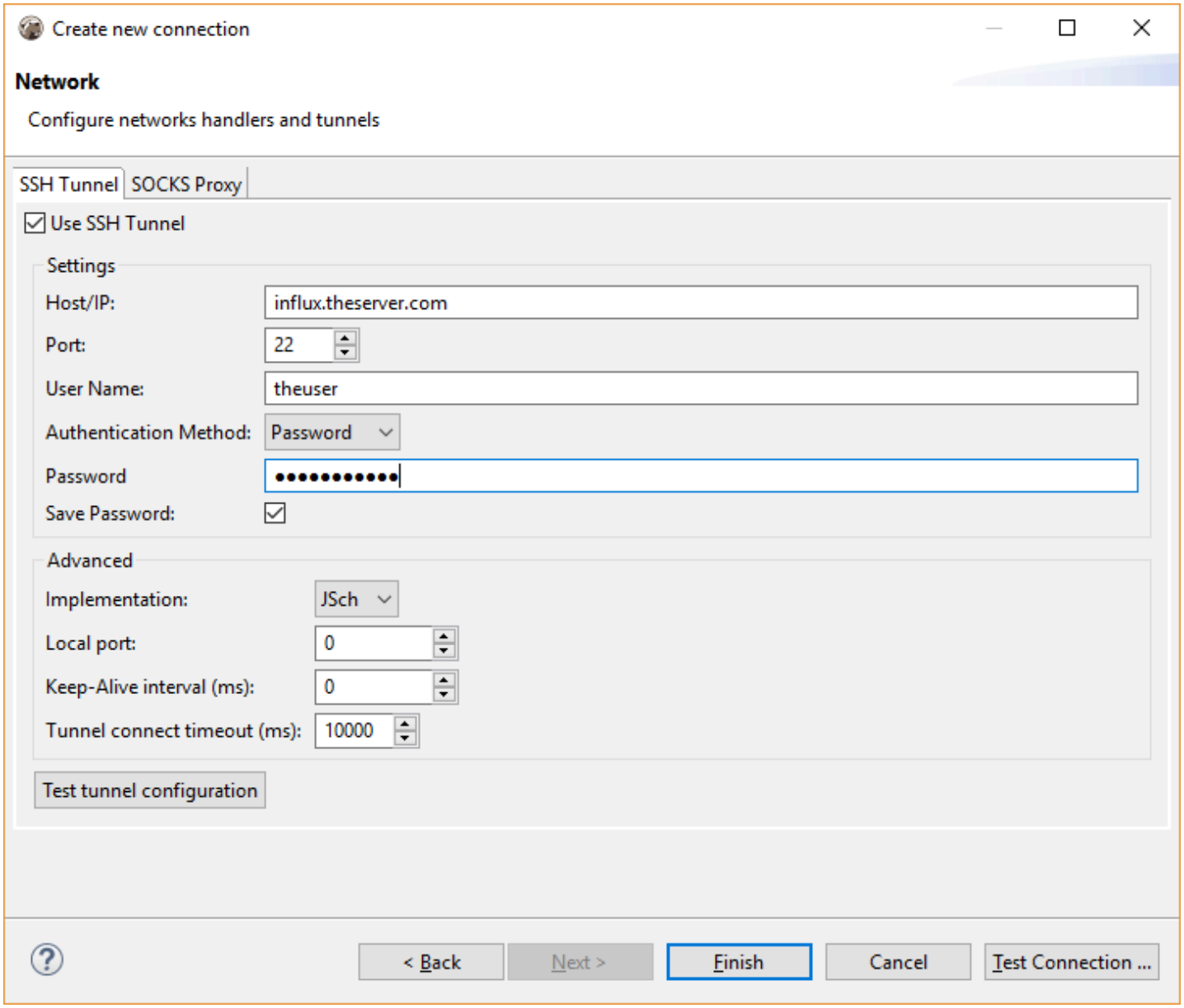
DBeaver uses InfluxDB Java driver 2.12 to operate with the server over HTTP/HTTPS (standard InfluxDB protocol).

It supports InfluxDB servers of any version (in the moment of writing).

Connecting to Influx Server

You can connect directly to a server or use SSH tunneling or SOCKS proxy.





Browsing InfluxDB schema

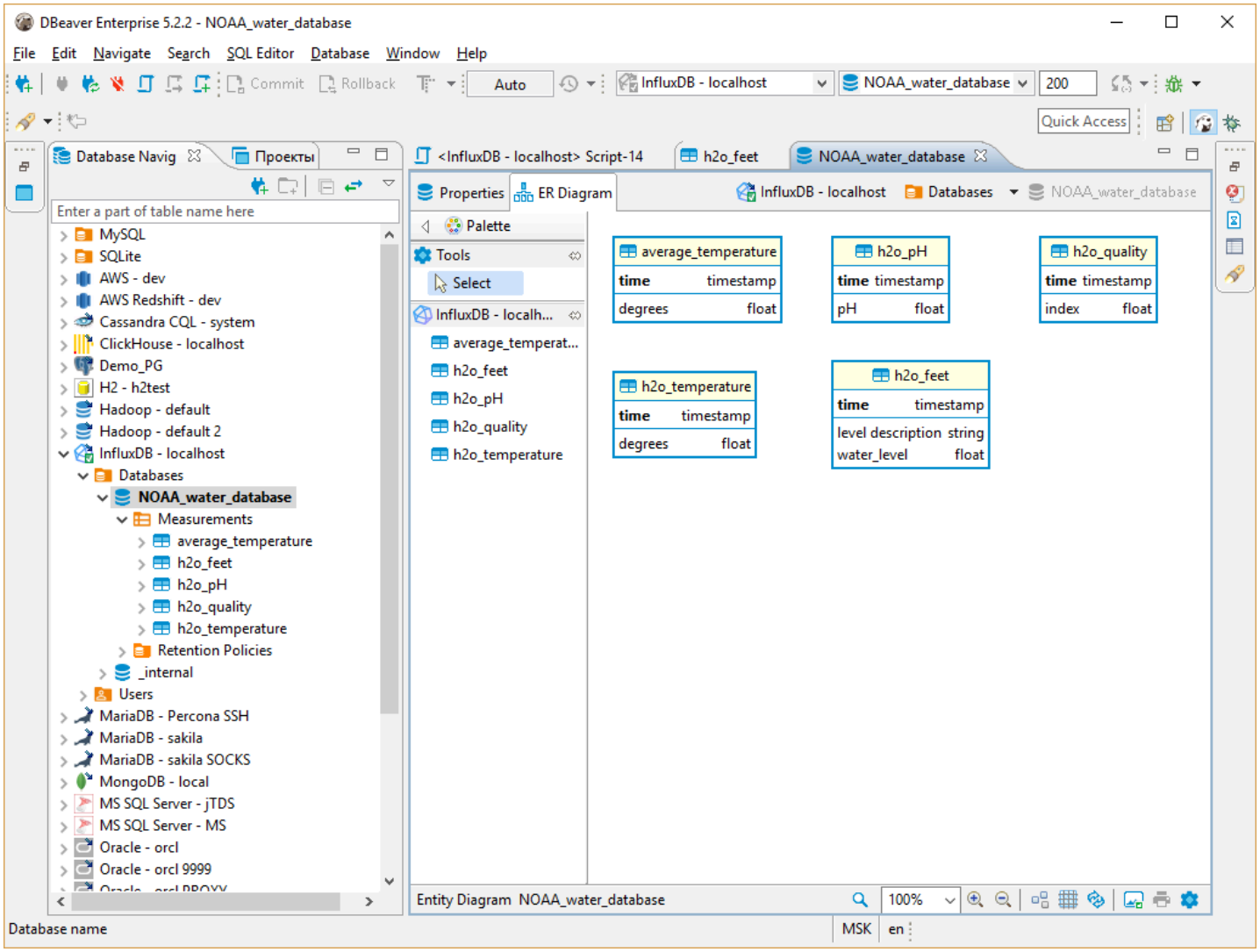
InfluxDB is TimeSeries database, it does not support tables, foreign keys and other relational entities.

DBeaver does not support data insert/update in InfluxDB. Database is basically a in read-only state for DBeaver.

You can browse schema and view/analyse data.

While data itself is loaded by various sensors/data collectors in real time.

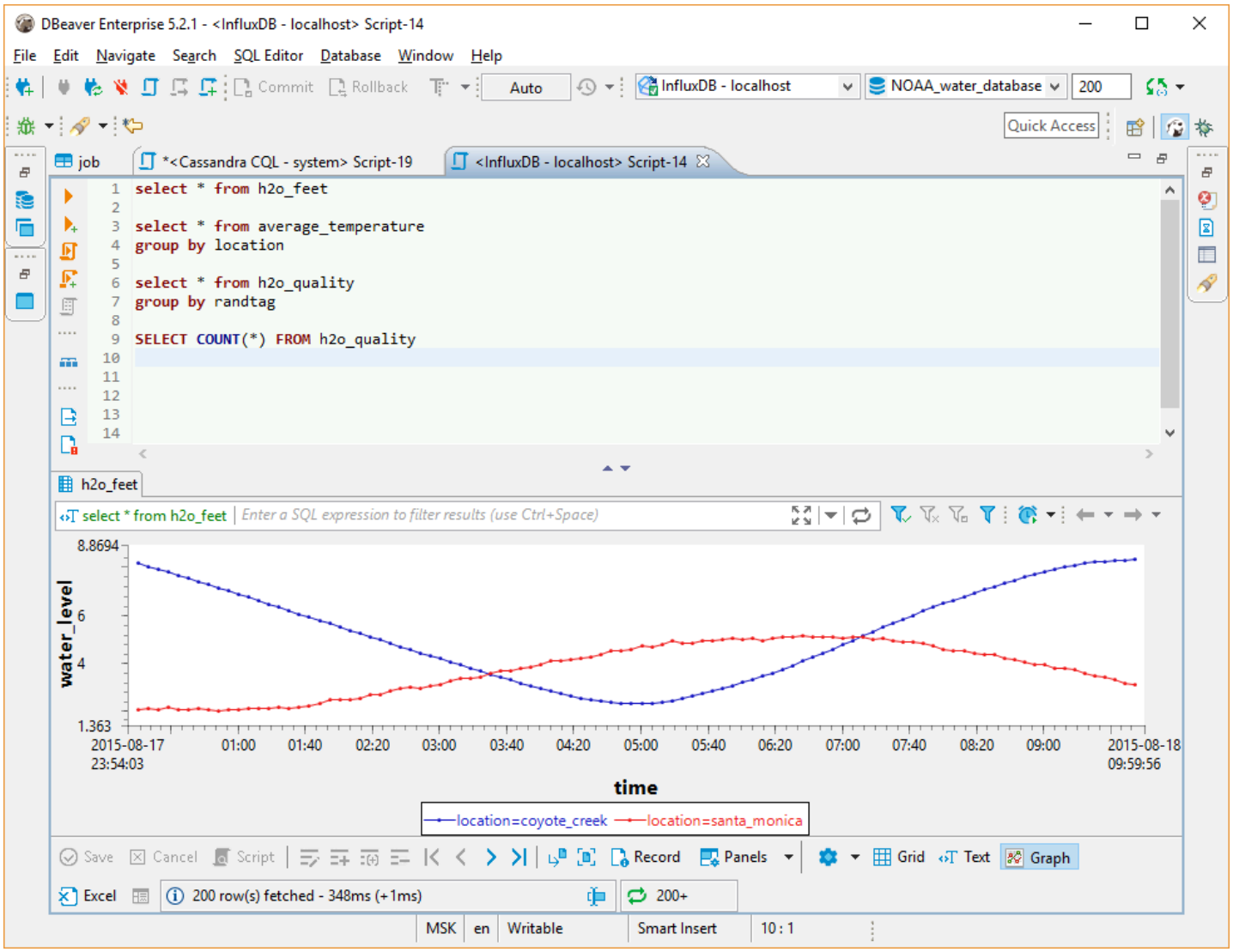
Instead of tables InfluxDB has measurements. Instead of columns it has fields and tags.

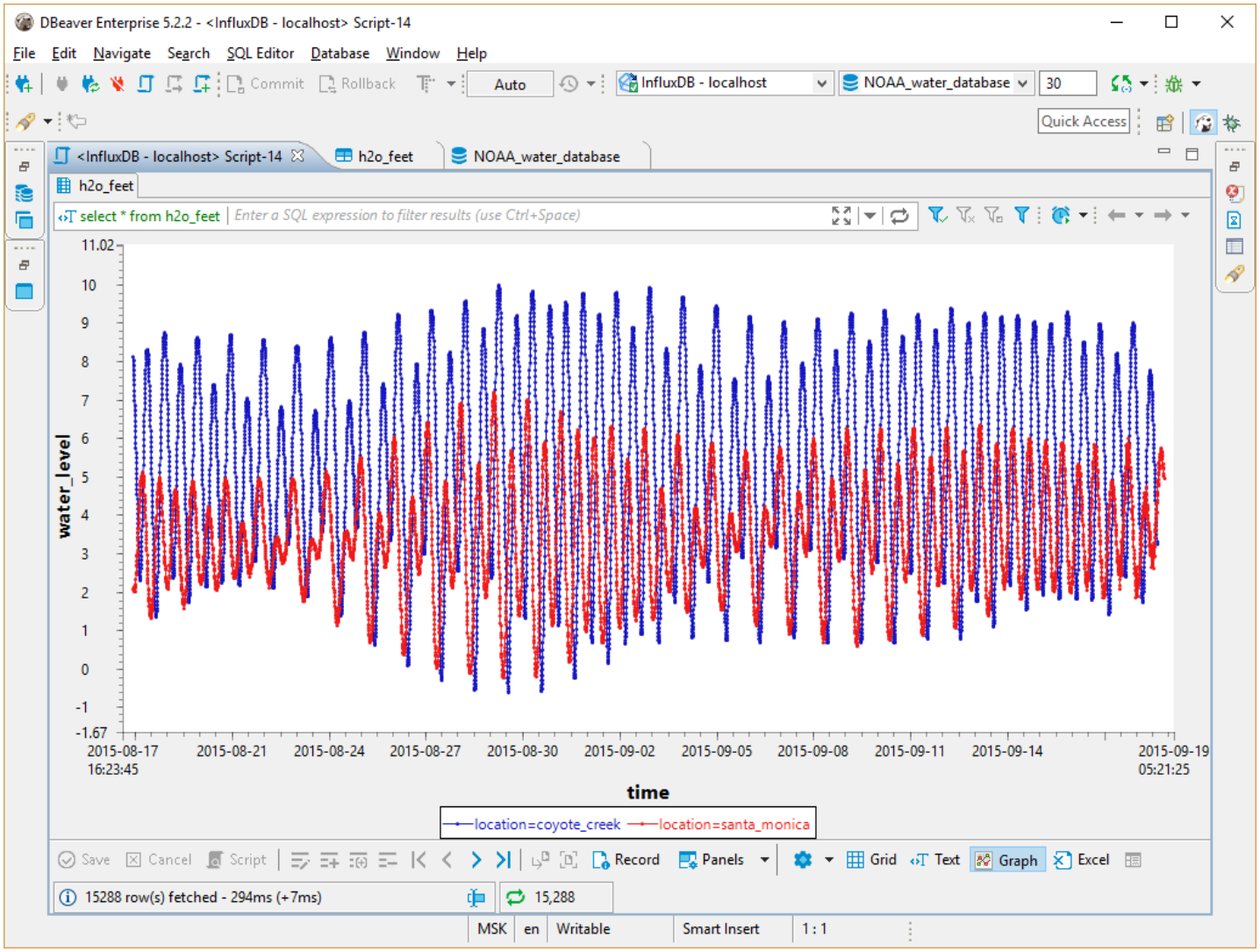


Executing InfluxQL (versions < 2)

InfluxQL is a query language similar to SQL.

DBeaver fully supports all InfluxQL statements. Query results are presented as grid or as graphs:





Executing Flux (versions > 2)

Flux is a query language that is used in the new versions of Influx. It is not similar to InfluxQL and doesn't use SQL syntax.

Query example

```
from(bucket: "example-bucket")
    |> range(start: -1d)
    |> filter(fn: (r) => r._measurement == "example-measurement")
    |> mean()
    |> yield(name: "_results")
```

MongoDB

Overview

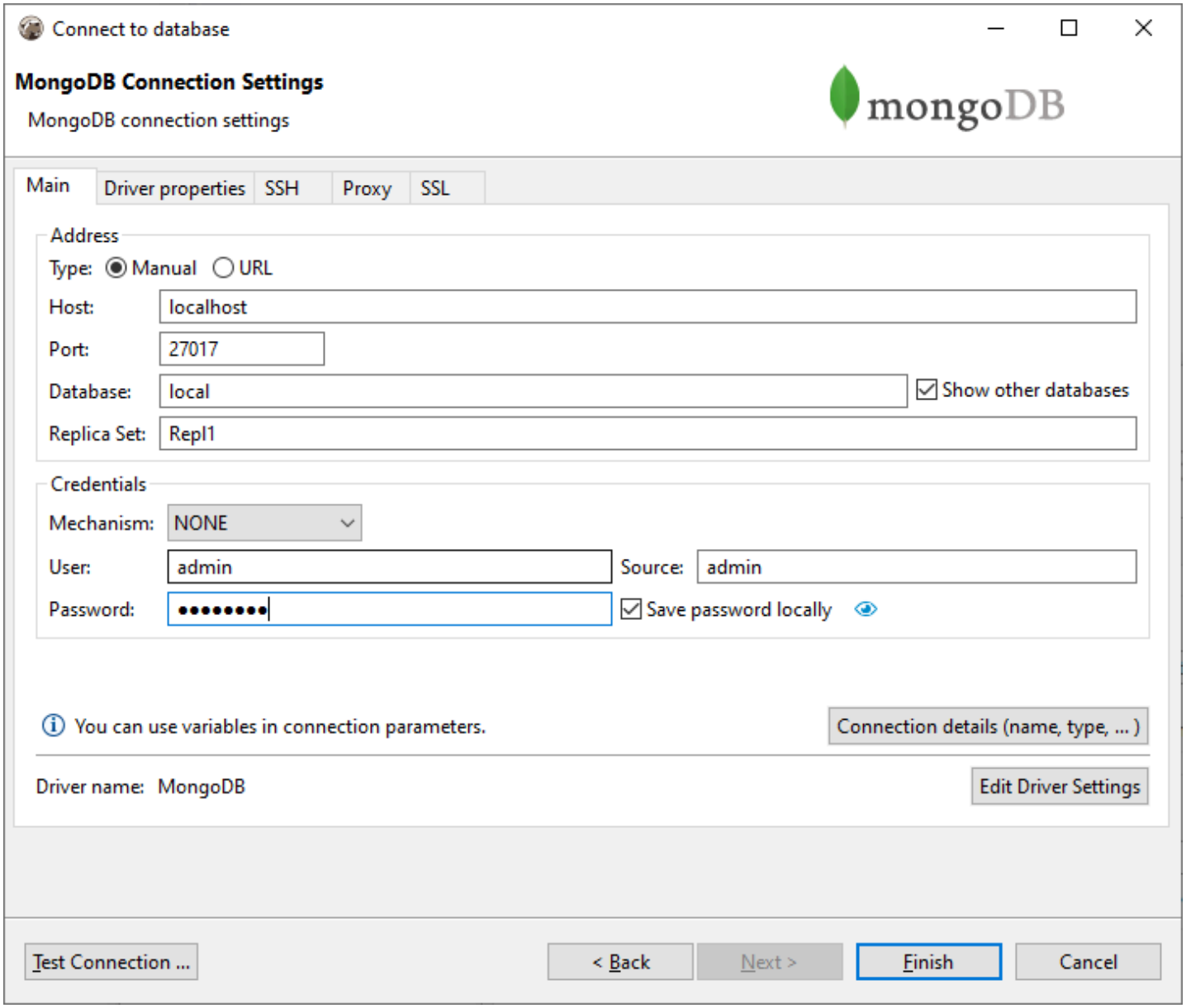
DBeaver EE supports MongoDB schema browser, data viewer, SQL and JavaScript queries execution. It also supports various administrative tools (like server sessions manager).

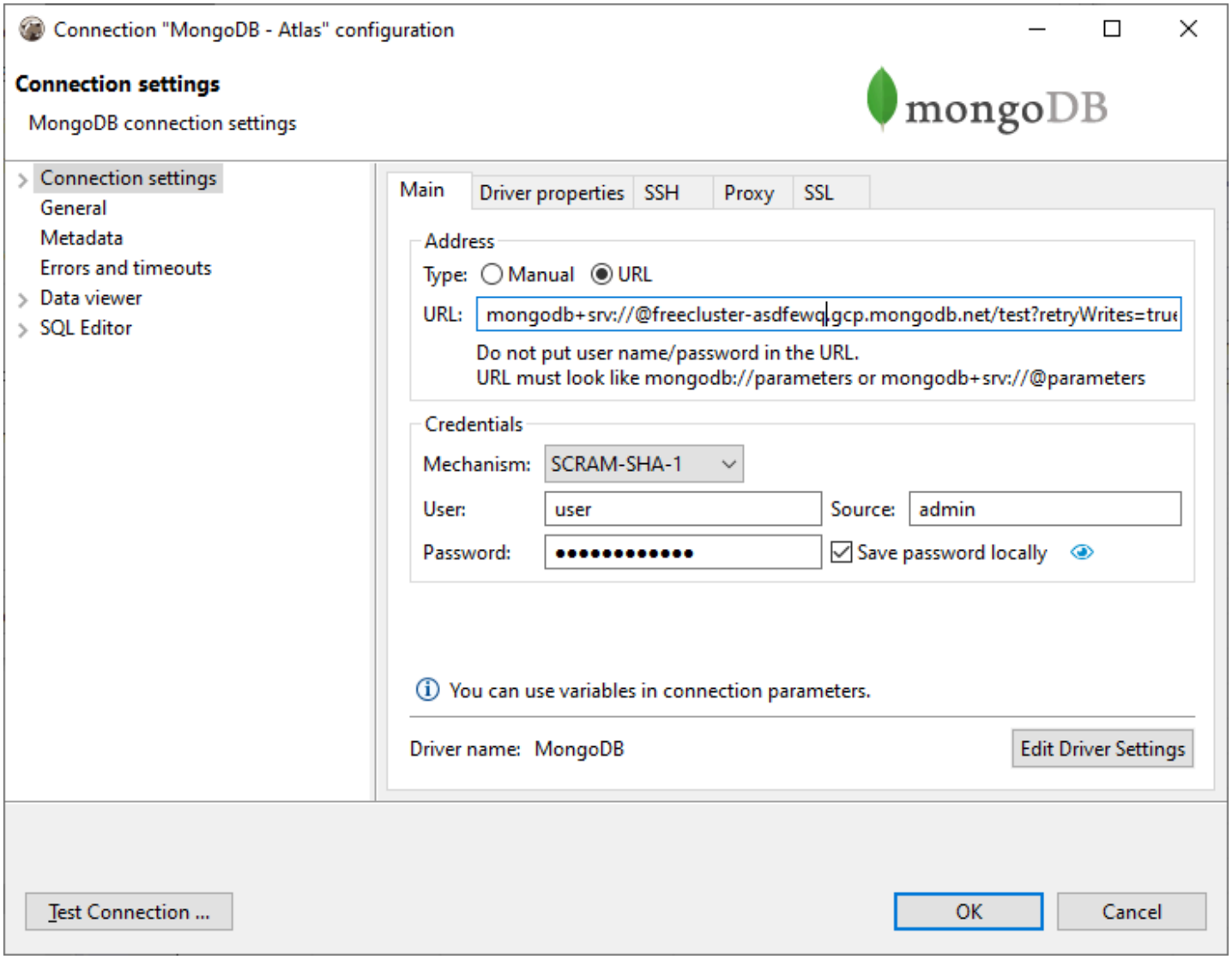
DBeaver uses MongoDB Java driver 3.8.0 to operate with a server. It supports MongoDB servers from 2.x to 4.x.

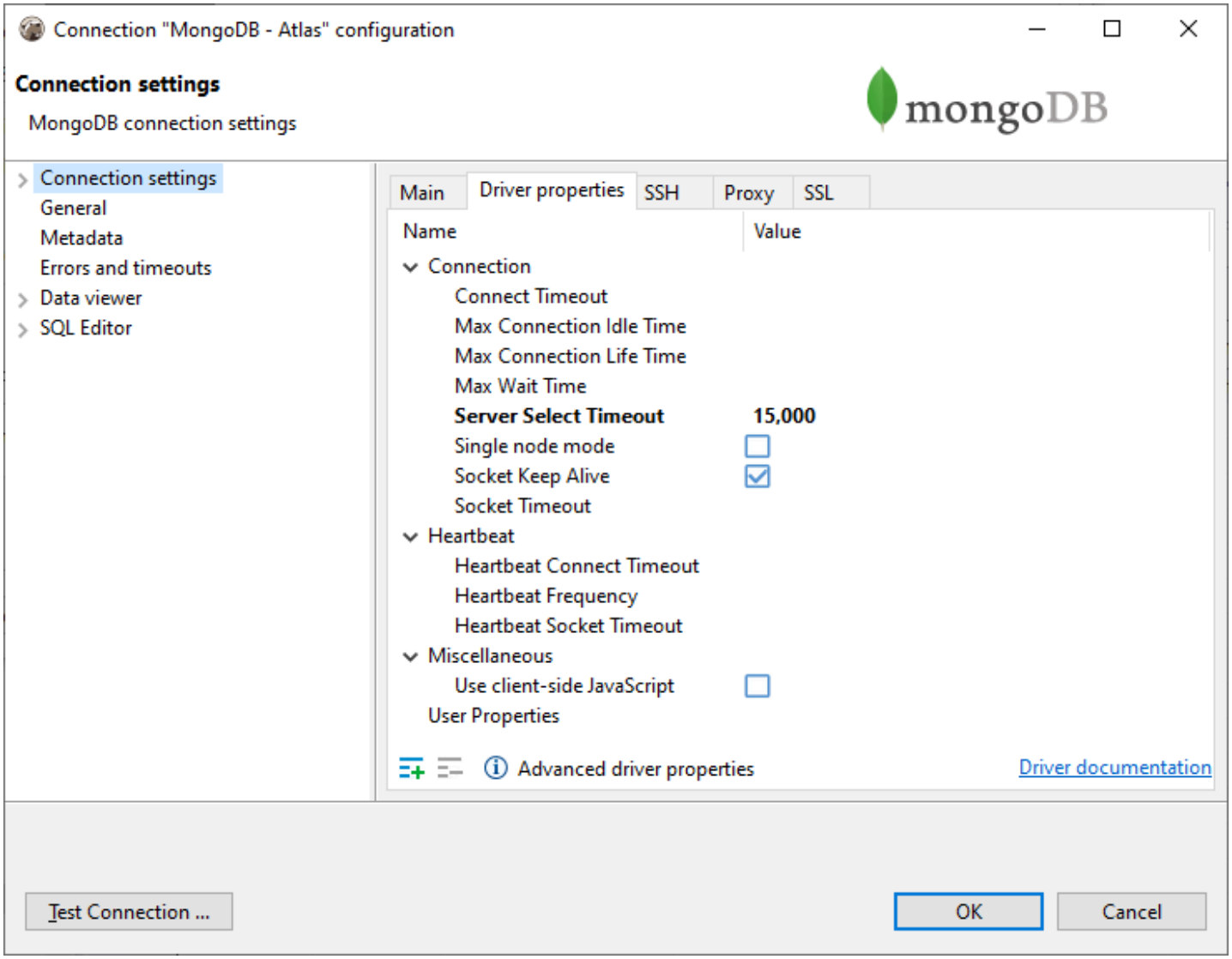
Connecting to MongoDB Server

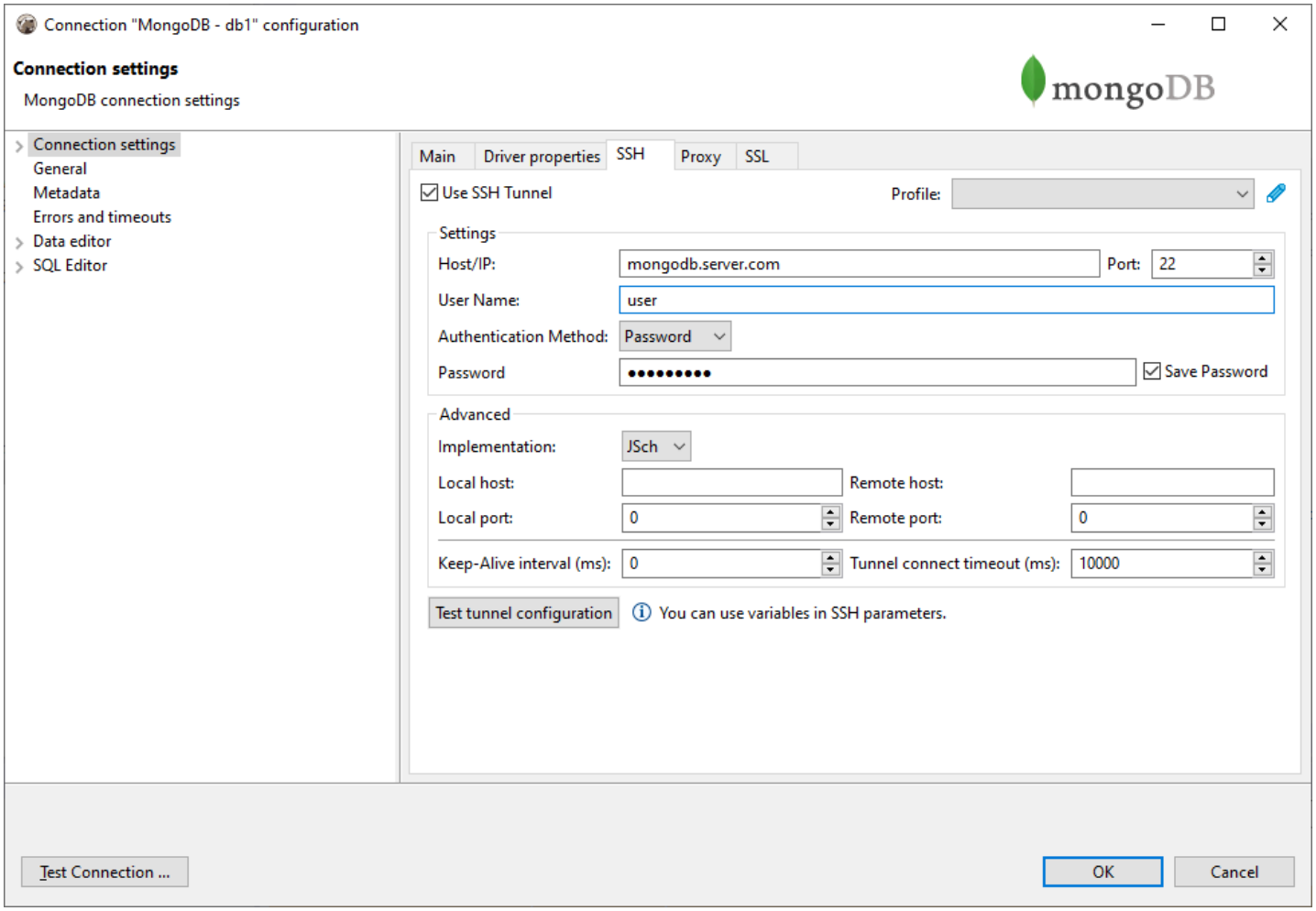
You can connect directly to a server or use SSH tunneling or SOCKS proxy.

You can specify server address as a host/port/database configuration or you can enter the target database URL with all necessary parameters:


Connect to database
MongoDB Connection Settings
MongoDB connection settings
mongoDB
Main
Driver properties
SSH
Proxy
SSL
Address
Type: Manual URL
Host: localhost
Port: 27017
Database: local
Show other databases
Replica Set: Repl1
Credentials
Mechanism: NONE
User: admin
Source: admin
Password: ••••••••
Save password locally
You can use variables in connection parameters.
Connection details (name, type, ...)
Driver name: MongoDB
Edit Driver Settings
Test Connection ...
< Back
Next >
Finish
Cancel


Connection "MongoDB - Atlas" configuration
Connection settings
MongoDB connection settings
mongoDB
Connection settings
General
Metadata
Errors and timeouts
Data viewer
SQL Editor
Main
Driver properties
SSH
Proxy
SSL
Address
Type: Manual URL
URL: mongodb+srv://@freecluster-asdfewq.gcp.mongodb.net/test?retryWrites=true
Do not put user name/password in the URL.
URL must look like mongodb://parameters or mongodb+srv://@parameters
Credentials
Mechanism: SCRAM-SHA-1
User: user
Source: admin
Password: ••••••••••••
Save password locally
You can use variables in connection parameters.
Driver name: MongoDB
Edit Driver Settings
Test Connection ...
OK
Cancel


Connection "MongoDB - Atlas" configuration
Connection settings
MongoDB connection settings
mongoDB
Connection settings
General
Metadata
Errors and timeouts
Data viewer
SQL Editor
Main
Driver properties
SSH
Proxy
SSL
Name
Value
Connection
Connect Timeout
Max Connection Idle Time
Max Connection Life Time
Max Wait Time
Server Select Timeout
15,000
Single node mode
Socket Keep Alive
Socket Timeout
Heartbeat
Heartbeat Connect Timeout
Heartbeat Frequency
Heartbeat Socket Timeout
Miscellaneous
Use client-side JavaScript
User Properties
Advanced driver properties
Driver documentation
Test Connection ...
OK
Cancel

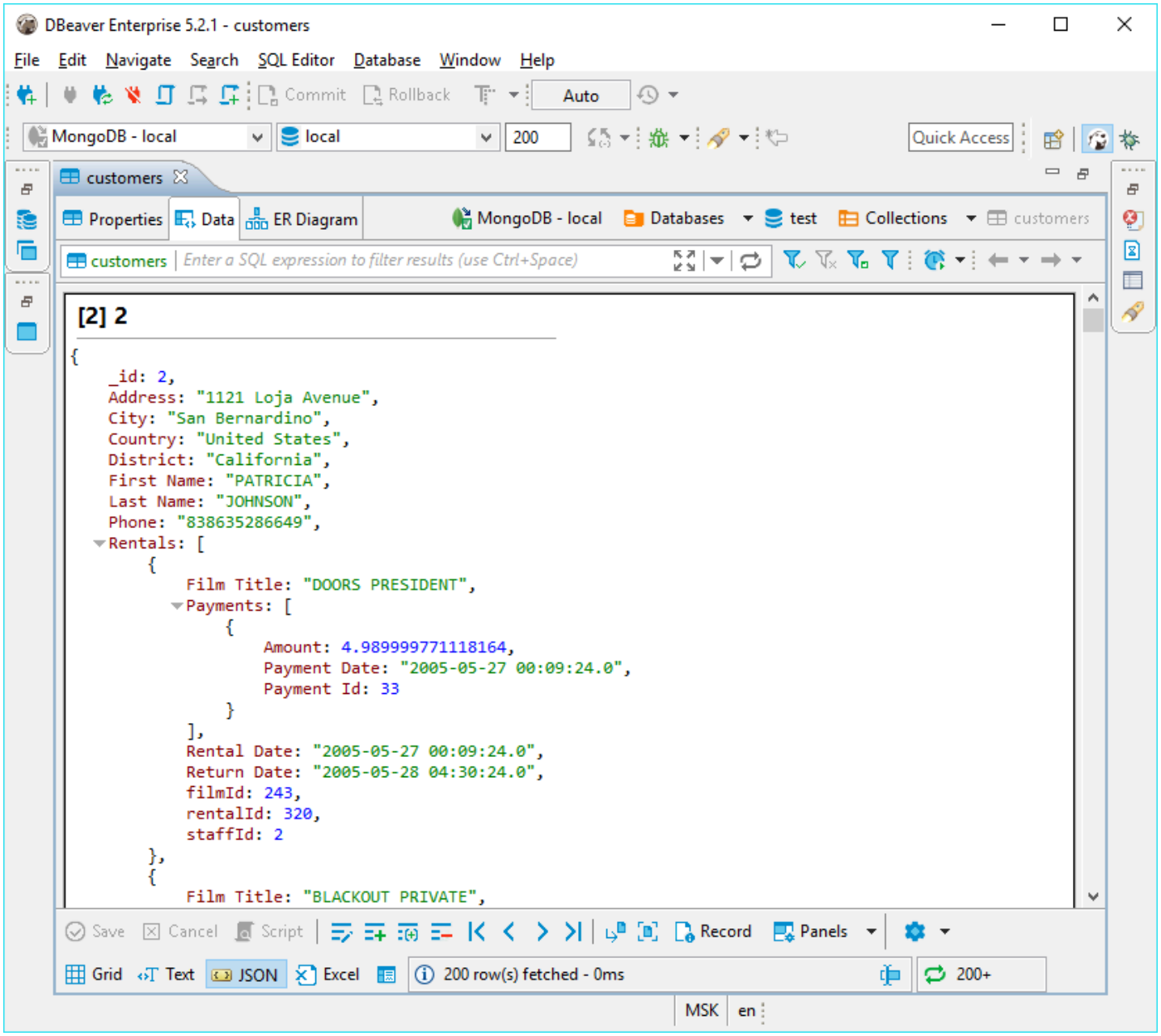


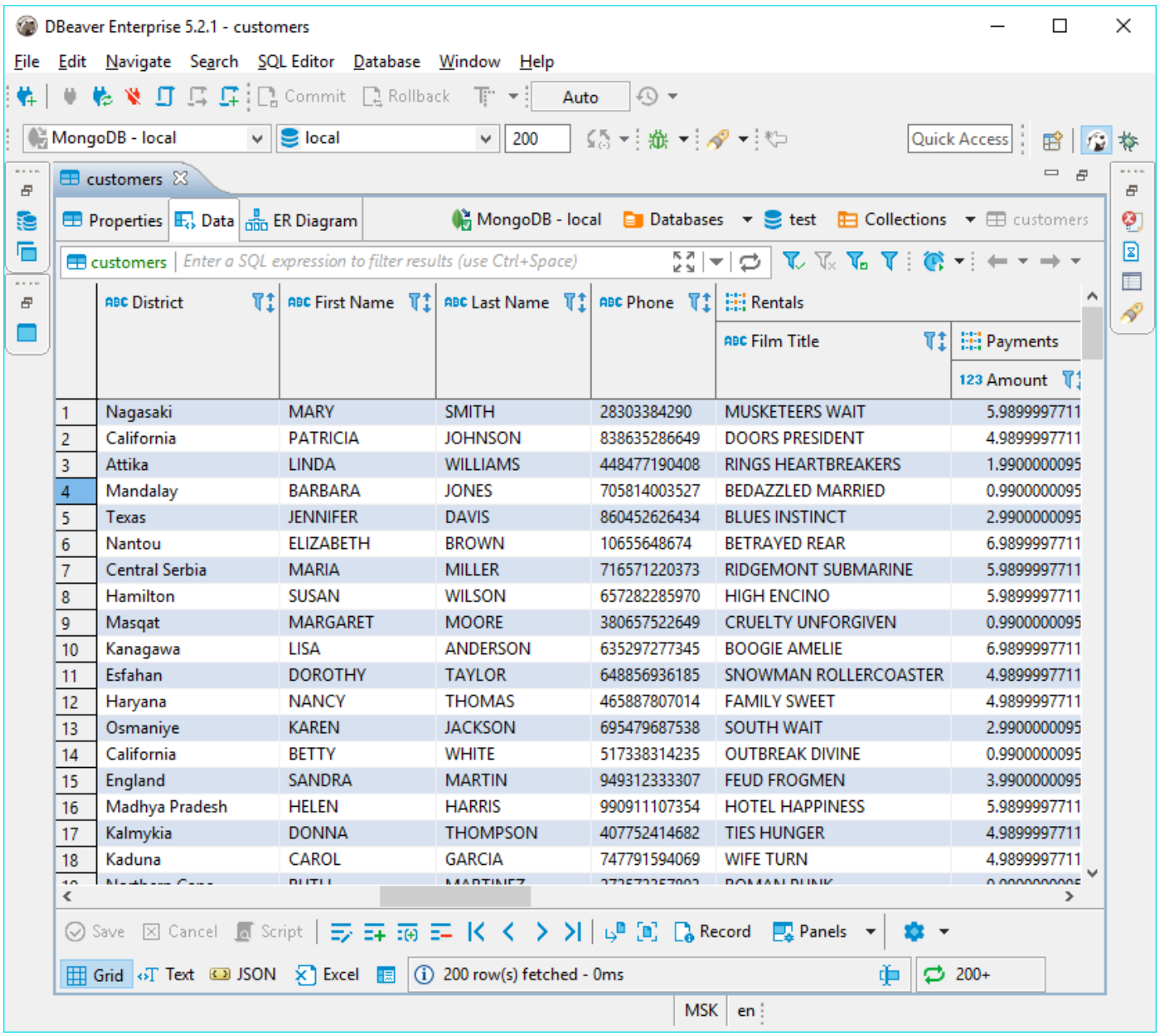
Browsing Mongo collections

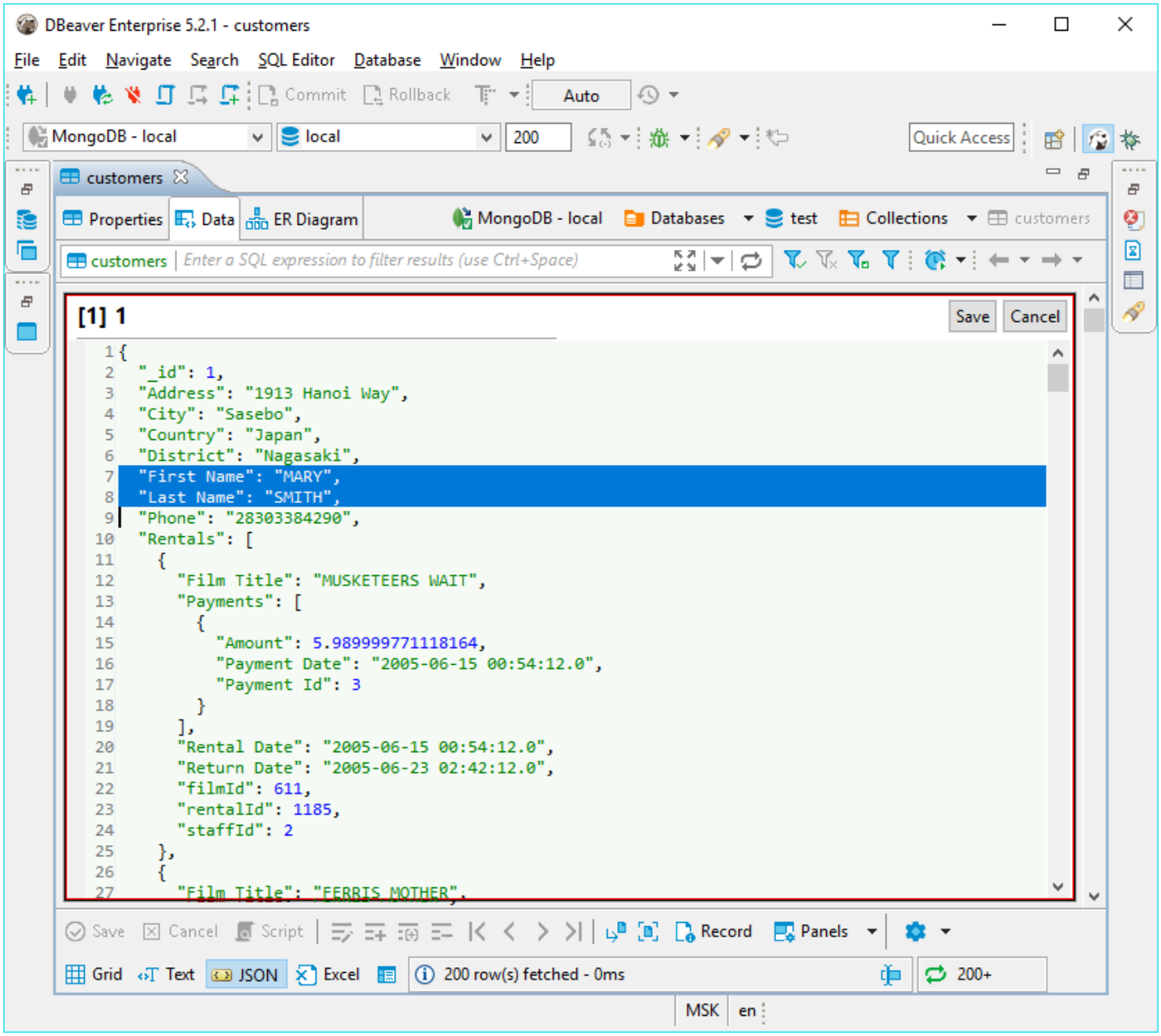
You can view/edit MongoDB collections content as standard relational tables (grid/plain text presentations) or as JSON documents.

The presentation can be switched in the Results Viewer toolbar.

In a grid, DBeaver will try to unify all documents in some particular collection (as they have the same structure/the same set of properties).


DBeaver Enterprise 5.2.1 - customers
File Edit Navigate Search SQL Editor Database Window Help
Commit Rollback Auto
MongoDB - local local 200 Quick Access
customers
Properties Data ER Diagram MongoDB - local Databases test Collections customers
customers Enter a SQL expression to filter results (use Ctrl+Space)
[2] 2
{
_id: 2,
Address: "1121 Loja Avenue",
City: "San Bernardino",
Country: "United States",
District: "California",
First Name: "PATRICIA",
Last Name: "JOHNSON",
Phone: "838635286649",
Rentals: [
{
Film Title: "DOORS PRESIDENT",
Payments: [
{
Amount: 4.989999771118164,
Payment Date: "2005-05-27 00:09:24.0",
Payment Id: 33
}
],
Rental Date: "2005-05-27 00:09:24.0",
Return Date: "2005-05-28 04:30:24.0",
filmId: 243,
rentalId: 320,
staffId: 2
},
{
Film Title: "BLACKOUT PRIVATE",
Save Cancel Script Record Panels
Grid Text JSON Excel 200 row(s) fetched - 0ms 200+
MSK en


DBeaver Enterprise 5.2.1 - customers
File Edit Navigate Search SQL Editor Database Window Help
Commit Rollback Auto
MongoDB - local local 200 Quick Access
customers
Properties Data ER Diagram MongoDB - local Databases test Collections customers
customers Enter a SQL expression to filter results (use Ctrl+Space)
District First Name Last Name Phone Rentals
Film Title Payments
Amount
1 Nagasaki MARY SMITH 28303384290 MUSKETEERS WAIT 5.9899997711
2 California PATRICIA JOHNSON 838635286649 DOORS PRESIDENT 4.9899997711
3 Attika LINDA WILLIAMS 448477190408 RINGS HEARTBREAKERS 1.9900000095
4 Mandalay BARBARA JONES 705814003527 BEDAZZLED MARRIED 0.9900000095
5 Texas JENNIFER DAVIS 860452626434 BLUES INSTINCT 2.9900000095
6 Nantou ELIZABETH BROWN 10655648674 BETRAYED REAR 6.9899997711
7 Central Serbia MARIA MILLER 716571220373 RIDGEMONT SUBMARINE 5.9899997711
8 Hamilton SUSAN WILSON 657282285970 HIGH ENCINO 5.9899997711
9 Masqat MARGARET MOORE 380657522649 CRUELTY UNFORGIVEN 0.9900000095
10 Kanagawa LISA ANDERSON 635297277345 BOOGIE AMELIE 6.9899997711
11 Esfahan DOROTHY TAYLOR 648856936185 SNOWMAN ROLLERCOASTER 4.9899997711
12 Haryana NANCY THOMAS 465887807014 FAMILY SWEET 4.9899997711
13 Osmaniye KAREN JACKSON 695479687538 SOUTH WAIT 2.9900000095
14 California BETTY WHITE 517338314235 OUTBREAK DIVINE 0.9900000095
15 England SANDRA MARTIN 949312333307 FEUD FROGMEN 3.9900000095
16 Madhya Pradesh HELEN HARRIS 990911107354 HOTEL HAPPINESS 5.9899997711
17 Kalmykia DONNA THOMPSON 407752414682 TIES HUNGER 4.9899997711
18 Kaduna CAROL GARCIA 747791594069 WIFE TURN 4.9899997711
Save Cancel Script Record Panels
Grid Text JSON Excel 200 row(s) fetched - 0ms 200+
MSK en



Executing JavaScript

JS statements can be executed in the SQL editor as usual. DBeaver supports all JS queries for MongoDB 2 and 3 as well as a subset of the `mongo` shell queries.

The following example creates a user in the current database.

```
db.createUser({
    user: 'testuser',
    pwd: 'test',
    roles: []
})
```

This example returns all documents in the collection 'test_col':

```
db.test_col.find().toArray()
```

Note: the script will be executed in the current database.

You can not set an explicit database name in your query.

The current database can be changed on the SQL Editor toolbar or on the Database Navigator.

Executing SQL

You can use standard SQL statements (`SELECT` , `INSERT` , `UPDATE` , `DELETE`) to manipulate Mongo data.

SELECT queries

SELECT queries support `WHERE` , `ORDER BY` , `GROUP BY` , `JOIN` and `HAVING` clauses.

MongoDB dialect doesn't support SQL sub-queries.

```
SELECT * FROM test_col
WHERE propName.subProp='value'

UPDATE FROM test_col
SET propsName.val1=123
WHERE propName.subProp='value'
```

Conditions

SELECT queries with `WHERE` support `AND` , `OR` , `<` , `<=` , `>` , `>=` , `=` and `!=` operators:

```
SELECT * FROM Employees
WHERE (Country = 'CA' OR Country = 'RU') AND Age > 20;
```

Please note that `AND` has higher precedence than `OR` and will evaluate first, so you need to surround it with parentheses.

Nested fields

Nested JSON fields can be divided by dot. If your field contains any special characters (e.g. spaces, dashes, etc.), you must enclose it with double quotes. For example:

```
SELECT title FROM movies WHERE info."imdb-details".rating > 6
```

Working with object IDs

When you need to find a document by ID, you must use the function `ObjectId`:

```
SELECT * FROM documents
WHERE _id = ObjectId('5f9c458018e3c69d0adc0fbd')
ORDER BY value DESC
```

Working with JOINs

Currently, SQL dialect for MongoDB supports `LEFT JOIN` and `INNER JOIN`:

```
SELECT
    ar.Name as Artist,
    al.Title as Album,
    SUM(tr.Milliseconds) as Duration
FROM Track tr
INNER JOIN Album al ON tr.AlbumId = al.AlbumId
INNER JOIN Artist ar ON al.ArtistId = ar.ArtistId
GROUP BY Artist, Album
ORDER BY Duration DESC
```

The only limitation is that you have to specify aliases for both source and target tables in a particular order:

```
SELECT *
FROM <source> <source-alias>
INNER JOIN <target> <target-alias> ON <source-alias>.column = <target-alias>.column
```

Note that executing the following script will not result in a merged document, but it will result in separate documents for `Track` and `Album`:

```
SELECT *
FROM Track tr
INNER JOIN Album al ON tr.AlbumId = al.AlbumId
```

Aggregate functions

In version 22.x only COUNT function is supported.

INSERT statement

You cannot use condition in INSERTS so just basic form is supported:

```
    INSERT INTO <collection-name> (field1, field2) VALUES (val1, val2);
```

UPDATE statement

You can use any expressions in WHERE clause but you cannot use sub-selects or joins.

```
    UPDATE <collection-name> SET field2=val3 WHERE field1=val1;
```

DELETE statement

You can use any expressions in WHERE clause, but you cannot use sub-selects or joins.

```
    DELETE FROM <collection-name> WHERE field1=val1;
```

CREATE TABLE statement

You cannot specify column list in CREATE TABLE. Only collection name can be specified.

```
    CREATE TABLE  <collection-name>;
```

DROP TABLE statement

```
    DROP TABLE  <collection-name>;
```

Working with dates

If you need to operate with dates then you must specify them in an ISO format. It is possible in both the JavaScript and SQL dialect:

```
db.dates.insert([
    { value: new Date('2016-05-18T16:00:00Z') },
    { value: new Date('2017-05-18T16:00:00Z') },
    { value: new Date('2018-05-18T16:00:00Z') },
    { value: new Date('2019-05-18T16:00:00Z') },
    { value: new Date('2020-05-18T16:00:00Z') }
])
```

Querying data in JavaScript:

```
db.dates.find({
    value: { $gte: new Date('2018-05-18T16:00:00Z') }
}).toArray()
```

Querying data in the SQL dialect (ISO and UNIX timestamp, in milliseconds):

```
SELECT value FROM dates
WHERE value > ISODate('2018-05-18T16:00:00.000Z')
ORDER BY value DESC

SELECT value FROM dates
WHERE value > ISODate(1526659200000)
ORDER BY value DESC
```

MySQL

Overview

This guide provides instructions on how to set up and use MySQL with DBeaver.

Before you start, you must create a connection in DBeaver and select MySQL. If you have not done this, please refer to our Database Connection article.

DBeaver interacts with the MySQL server using a specific driver. It supports all versions of MySQL, but the correct driver must be selected: use MySQL (old) for versions 4.x and earlier, MySQL 5 for 5.x versions, and MySQL for version 8 and later. DBeaver also supports MySQL extensions such as NDB Cluster, OceanBase, and TiDB. For Google Cloud SQL, there is a dedicated Google Cloud SQL (MySQL) driver that supports MySQL version 8 and later. You must select the appropriate driver in the **Connect to a database** window for these extensions.

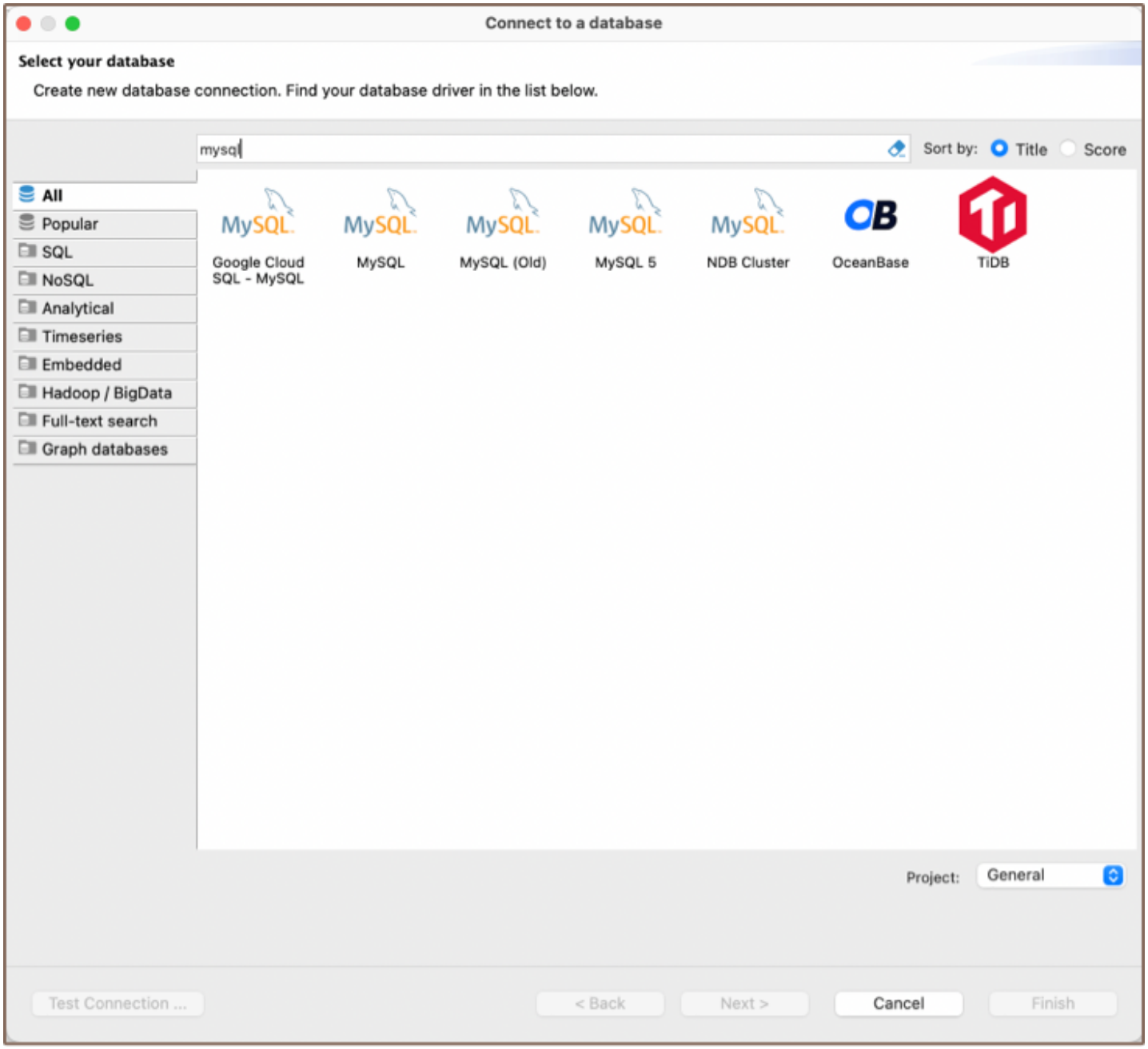


Table of contents:

- Setting Up
  - MySQL's connection settings
    - Connection details
  - MySQL driver properties
  - ODBC and JDBC Driver Configuration
  - Secure Connection Configurations

- Powering MySQL with DBeaver.
  - MySQL database objects
  - MySQL Features in DBeaver

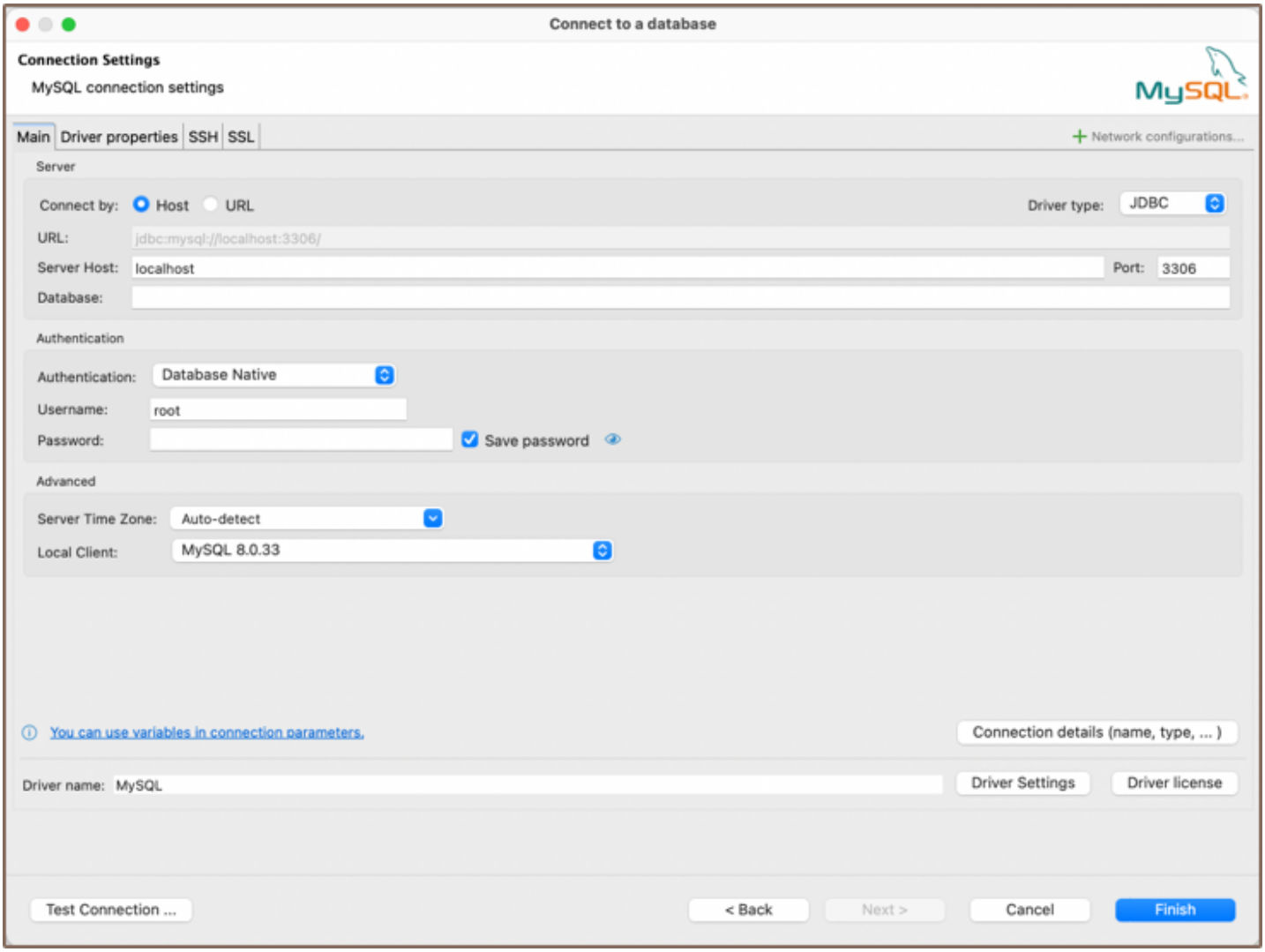
Setting Up

This section provides an overview of DBeaver's settings for establishing a direct connection and the configuration of secure connections using SSH, proxies, SSL, and the setup of ODBC/JDBC drivers for MySQL.

MySQL connection settings

In this subsection, we will outline the settings for establishing a direct connection to a MySQL database using DBeaver. Correctly configuring your connection ensures seamless interaction between DBeaver and your MySQL database.

The page of the connection settings requires you to fill in specific fields to establish the initial connection.



Field	Description
Connect by (Host/URL)	Choose whether you want to connect using a host or a URL.
Driver Type	Select the appropriate driver for your MySQL database.
URL	If you are connecting via URL, enter the URL of your MySQL database here. This field is disabled if you're connecting via the host.
Host	If you are connecting via host, enter the host address of your MySQL database here.
Database	Enter the name of the MySQL database you want to connect to.
Port	Enter the port number for your MySQL database. The default MySQL port is `3306`.
	Choose the type of authentication you want to use for the connection. For detailed guides on authentication types, please refer to the following articles: - Native Database Authentication

Authentication	- AWS RDS IAM Authentication ★ - Azure AD Authentication for MySQL ★ - DBeaver Profile Authentication ★ - MySQL Kerberos Authentication ★ - Two-factor Authentication ★ You can also read about security in DBeaver PRO.
Local Client	Specify the local client, see our article about Local client.
Server Time Zone	This setting is responsible for determining the time zone used by the MySQL server.
Connection Details	Provide additional connection details if necessary.
Driver Name	This field will be auto-filled based on your selected driver type.
Driver Settings	If there are any specific driver settings, configure them here.

Connection details

The **Connection Details** section in DBeaver allows you to customize your experience while working with MySQL database. This includes options for adjusting the **Navigator View**, setting up **Security measures**, applying **Filters**, configuring **Connection Initialization** settings, and setting up **Shell Commands**. Each of these settings can significantly impact your database operations and workflow. For detailed guides on these settings, please refer to the following articles:

- Connection Details Configuration
- Database Navigator
- Security Settings Guide
- Filters Settings Guide
- Connection Initialization Settings Guide
- Shell Commands Guide

MySQL driver properties

The settings for MySQL **Driver properties** enable you to adjust the performance of the MySQL JDBC and ODBC driver. These adjustments can influence the efficiency, compatibility, and features of your MySQL database.

For a complete walkthrough on setting up MySQL JDBC and ODBC drivers properties, you can refer to the official MySQL JDBC documentation and MySQL ODBC documentation. These guides detail each driver's properties and how they can be used to optimize MySQL database connections.

You can customize the MySQL driver in DBeaver via the **Edit Driver** page, accessible by clicking on the **Driver Settings** button on the first page of the driver settings. This page offers a range of settings that can influence your MySQL database connections. For a comprehensive guide on these settings, please refer to our Database drivers article.

ODBC and JDBC Driver Configuration

DBeaver provides extensive capabilities for managing MySQL database connections via ODBC/JDBC drivers. This functionality enables you to connect to your MySQL database using native ODBC drivers, offering an alternative when specific DBeaver drivers are not available.

This process is similar across most databases that support ODBC/JDBC drivers. For a comprehensive, step-by-step guide on how to install the driver manager, set up drivers, configure data sources, and establish connections in DBeaver, you can refer to our ODBC Driver Configuration article.

Secure Connection Configurations

DBeaver supports secure connections to your MySQL database. Guidance on configuring such connections, specifically **SSH**, **Proxy**, and **SSL** connections, can be found in various referenced articles. For a comprehensive understanding, please refer to these articles:

- SSH Configuration.
- Proxy Configuration.
- SSL Configuration.

Powering MySQL with DBeaver

DBeaver provides a host of features designed for MySQL databases. This includes the ability to view and manage databases, along with numerous unique capabilities aimed at optimizing database operations.

MySQL database objects

DBeaver lets you view and manipulate a wide range of MySQL database objects. DBeaver has extensive support for various MySQL metadata types, allowing you to interact with a wide variety of database objects, such as:

- Databases
  - Tables
    - Columns
    - Constraints
    - Indexes
    - Foreign Keys
    - References
    - Triggers
  - Views
  - Indexes
  - Procedures
  - Triggers
  - Events
- Users
- Administer

- Session Manager

MySQL Features in DBeaver

DBeaver is not limited to typical SQL tasks. It also includes numerous unique features specifically for MySQL.

Beyond regular SQL operations, DBeaver provides a range of MySQL-specific capabilities, such as:

Category	Feature
Data Types	MySQL Enumerations and Sets
Security	MySQL User Management
Storage Engines	InnoDB, MyISAM, MRG_MyISAM, Archive, Blackhole, CSV, Memory, Performance_schema
Database Management	MySQL Events
	MySQL Stored Procedures

Additional features compatible with MySQL, but not exclusive to it:

Category	Feature
Data Transfer	Data Import
	Data Export
Session Management	Session Manager
Backup and Restore	How to Backup/Restore data
Schema Management	Schema Compare
Data Visualization	GIS Guide
	ERD Guide

Oracle

Overview

This guide provides instructions for setting up and using Oracle with DBeaver.

Before you start, you must create a connection in DBeaver and select Oracle. If you haven't done this, please refer to our Database Connection article.

DBeaver interacts with the Oracle server using a specific driver. It also supports Oracle cloud database extensions such as Oracle Cloud JSON and Oracle NetSuite, both of which can be connected through Cloud Explorer.

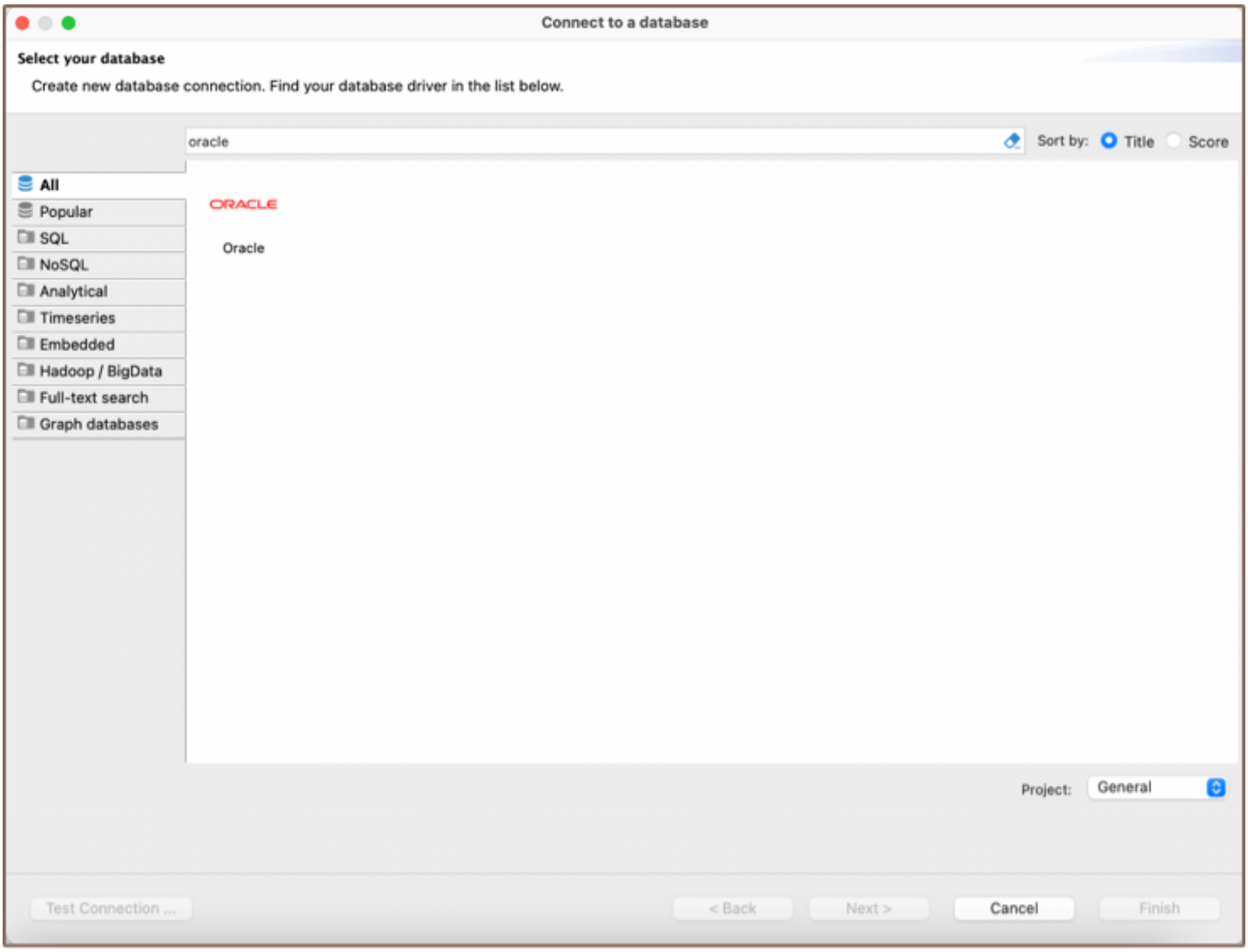


Table of contents:

- Setting Up
  - Oracle connection settings
    - Connection details
  - Oracle driver properties
  - Secure Connection Configurations
- Powering Oracle with DBeaver.
  - Oracle database objects
  - Oracle Features in DBeaver

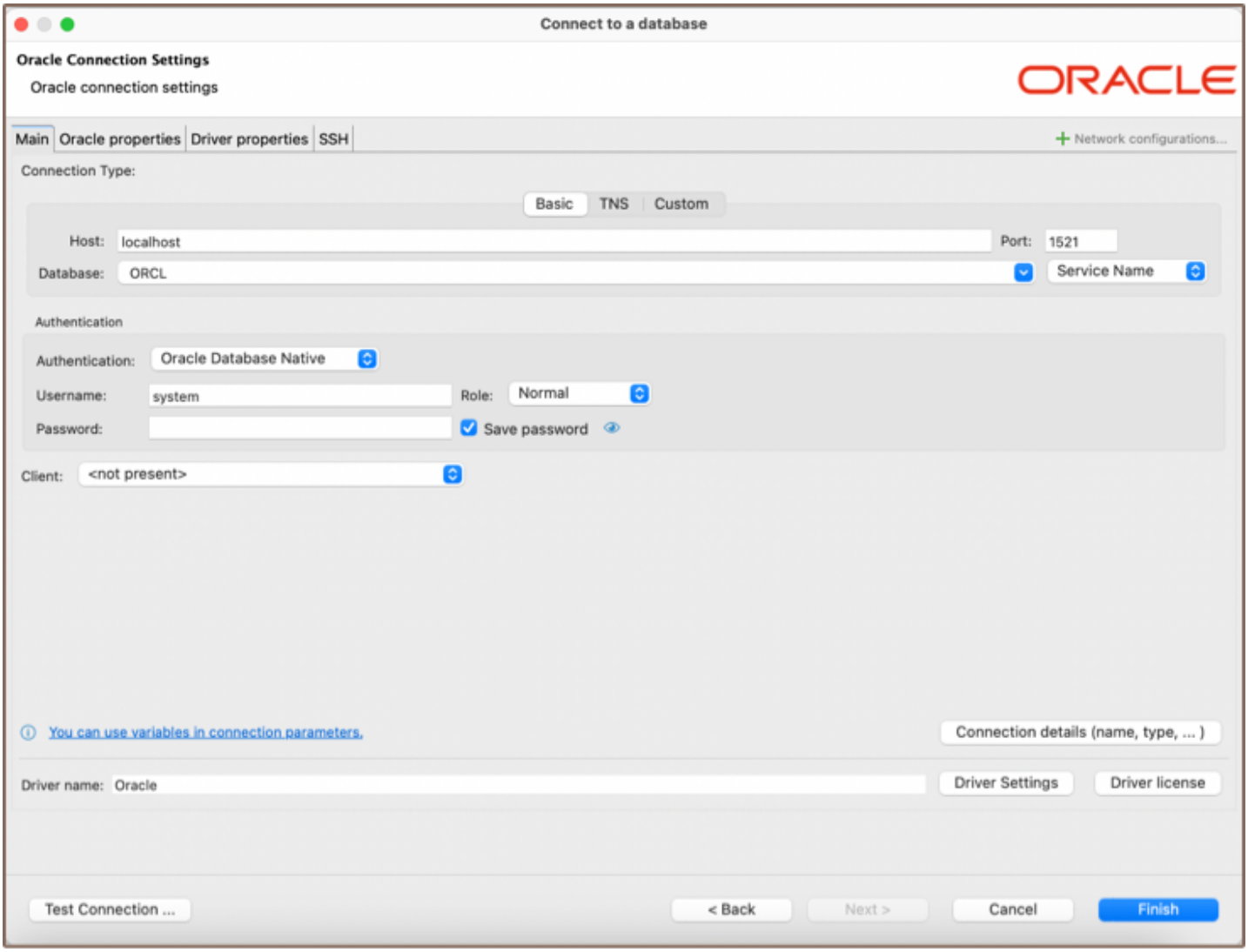
Setting Up

This section provides an overview of DBeaver's settings for establishing a direct connection and the configuration of secure connections using SSH, proxies, and the driver setup for Oracle.

Oracle connection settings

In this subsection, we'll outline the settings for establishing a direct connection to a Oracle database using DBeaver. Correctly configuring your connection ensures seamless interaction between DBeaver and your Oracle database.

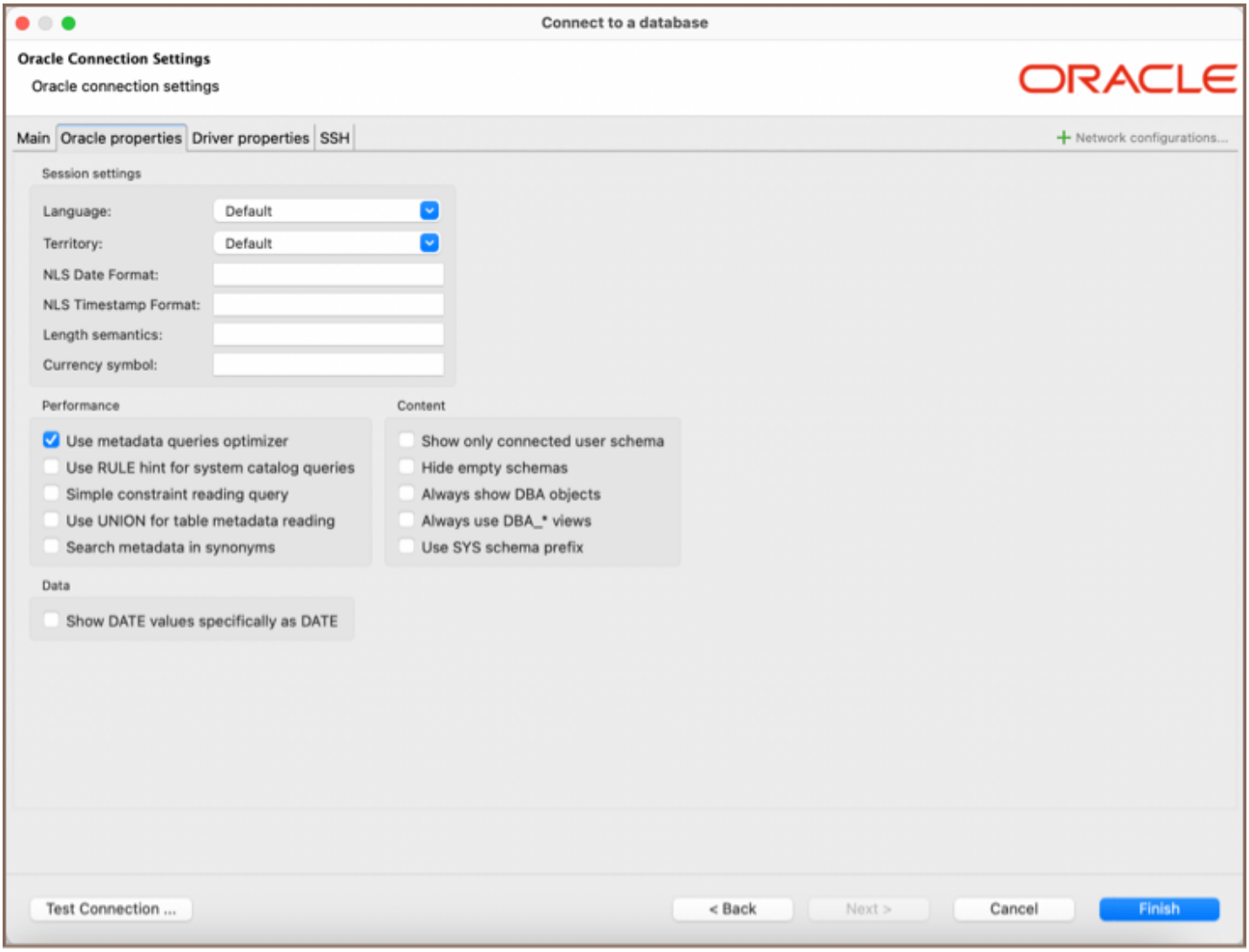
1) The first page of the connection settings requires you to fill in specific fields to establish the initial connection.



Field	Description
Host	If you're connecting via host, enter the host address of your Oracle database here.
Database	Enter the name of the Oracle database you want to connect to.
Port	Enter the port number for your Oracle database. The default Oracle port is `1521`.
Connection identifiers	Choose whether you want to connect using a `Service name` or a `SID` identifier.
Authentication	Choose the type of authentication you want to use for the connection. For detailed guides on authentication types, please refer to the following articles: - Oracle Database Native - OS Authentication ★ - DBeaver Profile Authentication ★ - Oracle Kerberos Authentication ★ - Oracle Wallet ★

	You can also read about security in DBeaver PRO.
Role	Choose whether you want to connect using a `Normal`, `SYSDABA` or a `SYSOPER` role. For more details, you can refer to the official Administering User Accounts and Security article.
Connection Details	Provide additional connection details if necessary.
Driver Name	This field will be auto-filled based on your selected driver type.
Driver Settings	If there are any specific driver settings, configure them here.

2) The second page of the connection settings offers additional options that allow you to customize your further connection to the Oracle database.



Field	Description
Language	Specify the Session Language.

Territory	Specify the Session Territory.
NLS Date Format	Specify NLS (National Language Support) Date Format. The default value of the `NLS_DATE_FORMAT` parameter is determined by the `TERRITORY` parameter.
NLS Timestamp Format	Specify NLS Timestamp format. The default value of the `NLS_TIMESTAMP_FORMAT` parameter is determined by the `TERRITORY` parameter.
Length semantics	specify the length semantics for `VARCHAR2` and `CHAR` table columns, user-defined object attributes, and PL/SQL variables in database objects created in the session. The length semantics can be specified as either `BYTE` or `CHAR`.
Currency symbol	Specify the Currency symbol.You can find out which currency symbol your current session uses by querying the `V$NLS_PARAMETERS` view.
Show only connected user schema	Show only a scheme of a connected user in the Database Navigator.
Hide empty schemas	Check existence of objects within schema and do not show empty schemas in tree. Enabled by default but it may cause performance problems on databases with very big number of objects.
Always show `DBA` objects	Always shows DBA-related metadata objects in tree even if user do not has `DBA` role.
**Always use `DBA_ views\*\*` | Use DBA * `views instead of` ALL*` views wherever it is possible	
Use SYS schema prefix	Use SYS schema prefix in all metadata queries. Otherwise use view names without explicit schema.
Simple constraint reading query	Use simple metadata queries. May work slower but it is more stable for all Oracle versions.
Use UNION for table metadata reading	Use legacy table metadata query. With `UNION` instead `JOIN`. It helps in some cases speed up reading of table data.
Search metadata in synonyms	Search for metadata in synonyms among other places. May significantly slow down metadata search as well as autocompletion.
Use RULE hint for system catalog queries	Adds `RULE` hint for some system catalog queries (like columns and constraints reading).It significantly increases performance on some Oracle databases (and decreases on others).

Show DATE values specifically as DATE	Show `DATE` data type values specifically as `DATE`, not as a `TIMESTAMP`.This setting will not work with disabled date/time formatting.
Use metadata queries optimizer	Use metadata queries optimizer. May significantly improve metadata reading performance on some systems.

Connection details

The **Connection Details** section in DBeaver allows to customize your experience while working with Oracle database. This includes options for adjusting the **Navigator View**, setting up **Security** measures, applying **Filters**, configuring **Connection Initialization** settings, and setting up **Shell Commands**. Each of these settings can significantly impact your database operations and workflow. For detailed guides on these settings, please refer to the following articles:

- Connection Details Configuration
- Database Navigator
- Security Settings Guide
- Filters Settings Guide
- Connection Initialization Settings Guide
- Shell Commands Guide

Oracle driver properties

The settings for Oracle **Driver properties** enable you to adjust the performance of the Oracle driver. These adjustments can influence the efficiency, compatibility, and features of your Oracle database.

For a complete walkthrough on setting up Oracle JDBC driver properties, you can refer to the official Oracle JDBC documentation. These guide detail each driver's properties and how they can be used to optimize Oracle database connections.

You can customize the Oracle driver in DBeaver via the **Edit Driver** page, accessible by clicking on the **Driver Settings** button on the first page of the driver settings. This page offers a range of settings that can influence your

Oracle database connections. For a comprehensive guide on these settings, please refer to our Database drivers article.

Secure Connection Configurations

DBeaver supports secure connections to your Oracle database. Guidance on configuring such connections, specifically **SSH** and **Proxy** connections, can be found in various referenced articles. For a comprehensive understanding, please refer to these articles:

- SSH Configuration.
- Proxy Configuration.

Powering Oracle with DBeaver

DBeaver provides a host of features designed for Oracle databases. This includes the ability to view and manage databases, along with numerous unique capabilities aimed at optimizing database operations.

Oracle database objects

DBeaver lets you view and manipulate a wide range of Oracle database objects. DBeaver has extensive support for various Oracle metadata types, allowing you to interact with a wide variety of database objects, such as:

- Schemas
  - Tables
    - Columns
    - Constraints
    - Foreign Keys
    - References
    - Triggers

        - Indexes
        - Dependencies
    - Views
    - Materialized Views
    - Indexes
    - Sequences
    - Queues
    - Types
    - Packages
    - Procedures
    - Functions
    - Synonyms
    - Schema Triggers
    - Table Triggers
    - Database Links
    - Java
    - Jobs
    - Scheduler
        - Jobs
        - Programs
    - Recycle bin
- Global metadata

- Types
- Public Synonyms
- Public Database Links
- User Recycle bin

- Storage
  - Tablespaces
    - Files
    - Objects

- Security
  - Users
  - Roles
  - Profiles

- Administer
  - Session Manager
  - Locks Manager

Oracle Features in DBeaver

DBeaver isn't limited to typical SQL tasks. It also includes numerous unique features specifically for Oracle. Beyond regular SQL operations, DBeaver provides a range of Oracle-specific capabilities, such as:

Category	Feature
Data Types	Oracle Nested Tables
PL/SQL Support	PL/SQL Procedures
	PL/SQL Functions
Security	Oracle Permissions

	Oracle Profiles
	Oracle Roles
Data Organization	Oracle Partitions
Database Management	Oracle Dependencies
Performance Tuning	Oracle Performance Reports
	Oracle Execution Plans

Additional features compatible with Oracle, but not exclusive to it:

Category	Feature
Data Transfer	Data Import
	Data Export
Session Management	Session Manager
Backup and Restore	How to Backup/Restore data
Schema Management	Schema Compare
Data Visualization	GIS Guide
	ERD Guide

PostgreSQL

Overview

This guide provides instructions on how to set up and use PostgreSQL with DBeaver.

DBeaver offers extensive support for PostgreSQL, but it might not cover all possible functionalities. If DBeaver is missing a feature you need, please submit a request through our GitHub page. If you are a developer, you have the option to enhance DBeaver's capabilities by contributing your own code. For guidelines on how to do this, check out our Contribution Guide.

Before you start, you must create a connection in DBeaver and select PostgreSQL. If you have not done this, please refer to our Database Connection article.

DBeaver interacts with the PostgreSQL server using a specific driver, it supports all versions of PostgreSQL, but the correct driver must be selected: use **PostgreSQL (old)** for versions below 8.x and **PostgreSQL** for newer versions. DBeaver also supports PostgreSQL extensions such as **EDB**, **Google Cloud SQL for PostgreSQL** and **Fujitsu Enterprise Postgres**. You must select the appropriate driver in the **Connect to a database** window for these extensions.

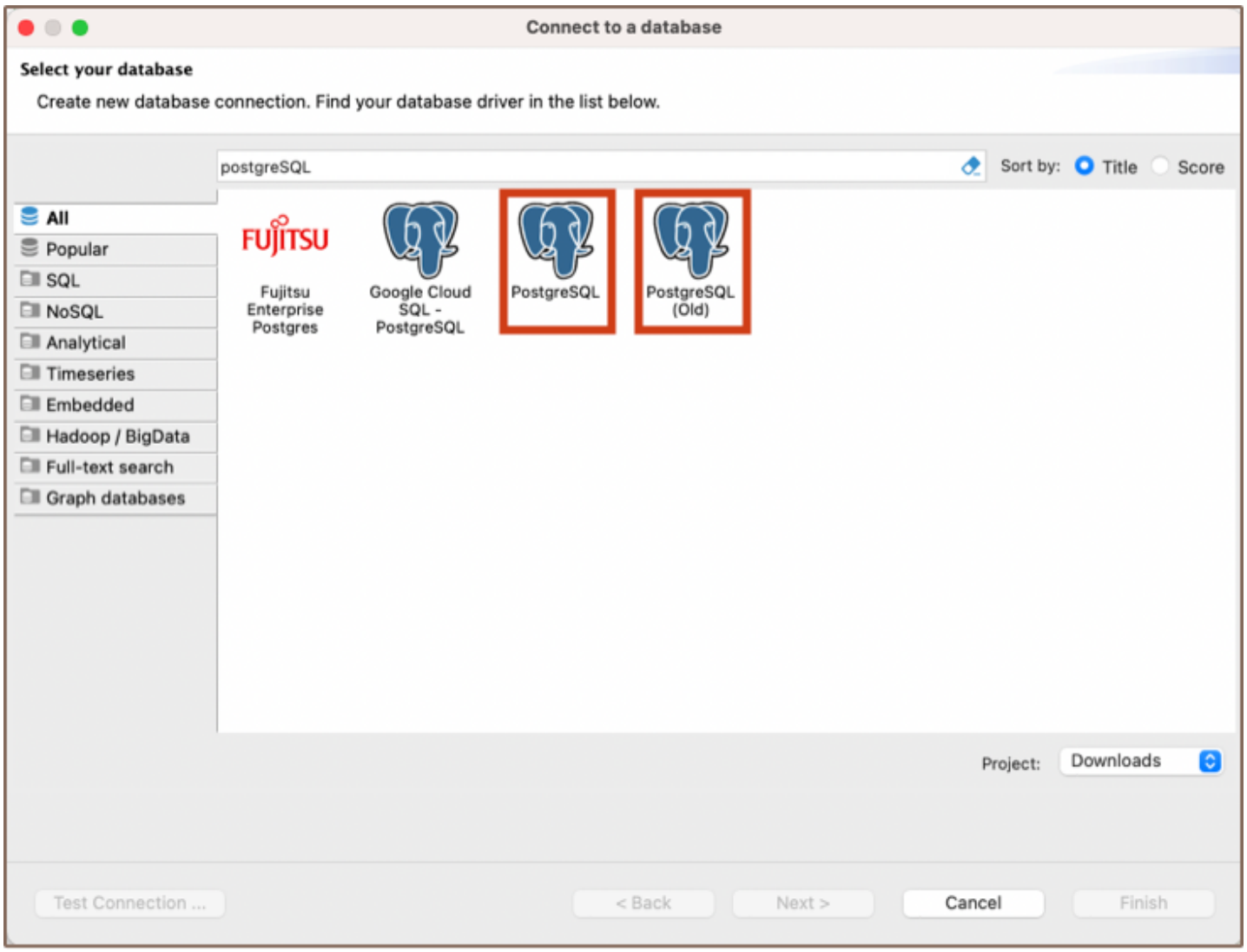


Table of contents:

- Setting Up
  - PostgreSQL's connection settings
    - Connection details
  - PostgreSQL driver properties
  - ODBC and JDBC Driver Configuration
  - Secure Connection Configurations
- Powering PostgreSQL with DBeaver.
  - PostgreSQL database objects

- PostgreSQL Features in DBeaver

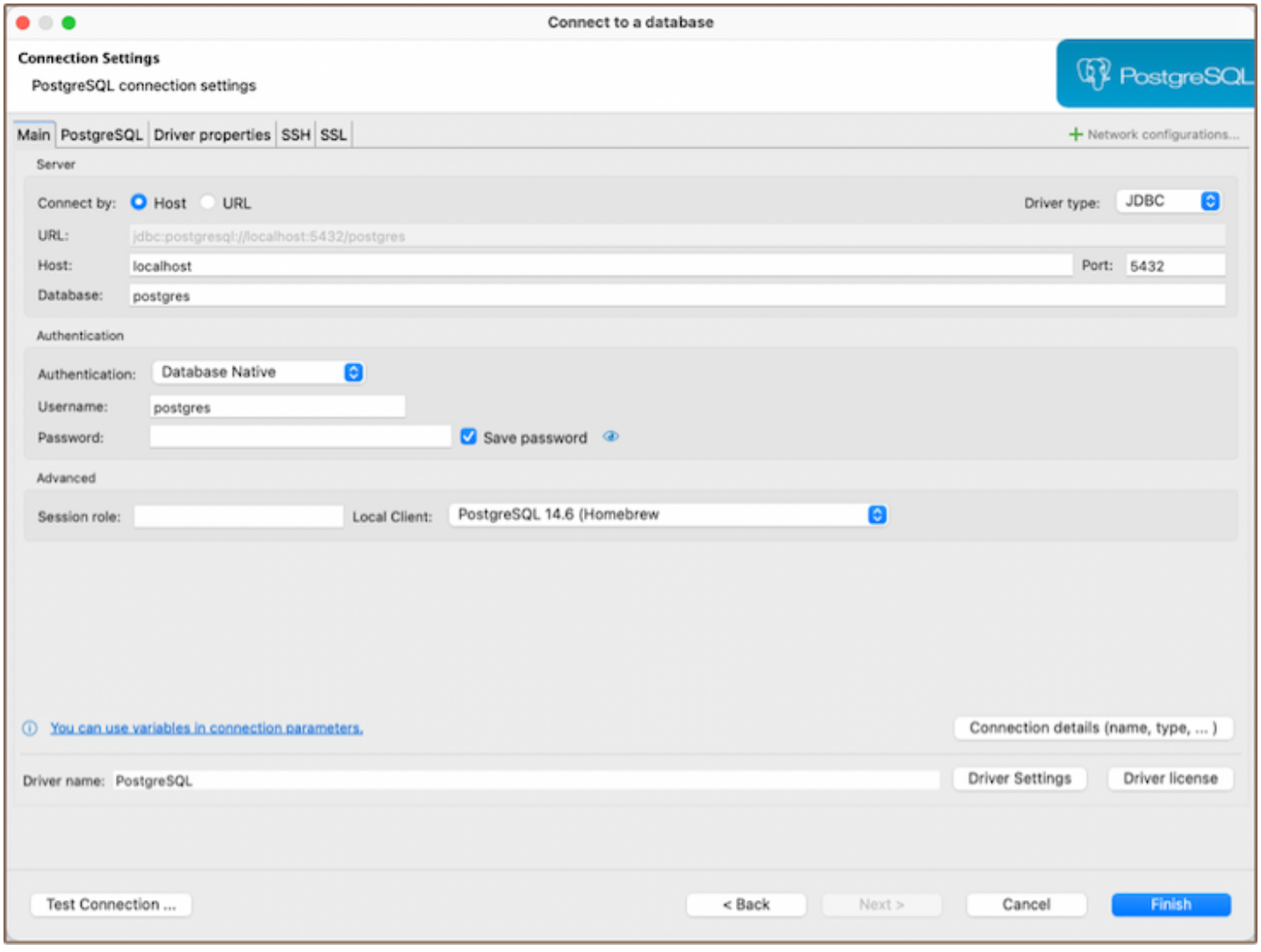
Setting Up

This section provides an overview of DBeaver's settings for establishing a direct connection and the configuration of secure connections using SSH, proxies, SSL, and the setup of ODBC/JDBC drivers for PostgreSQL.

PostgreSQL connection settings

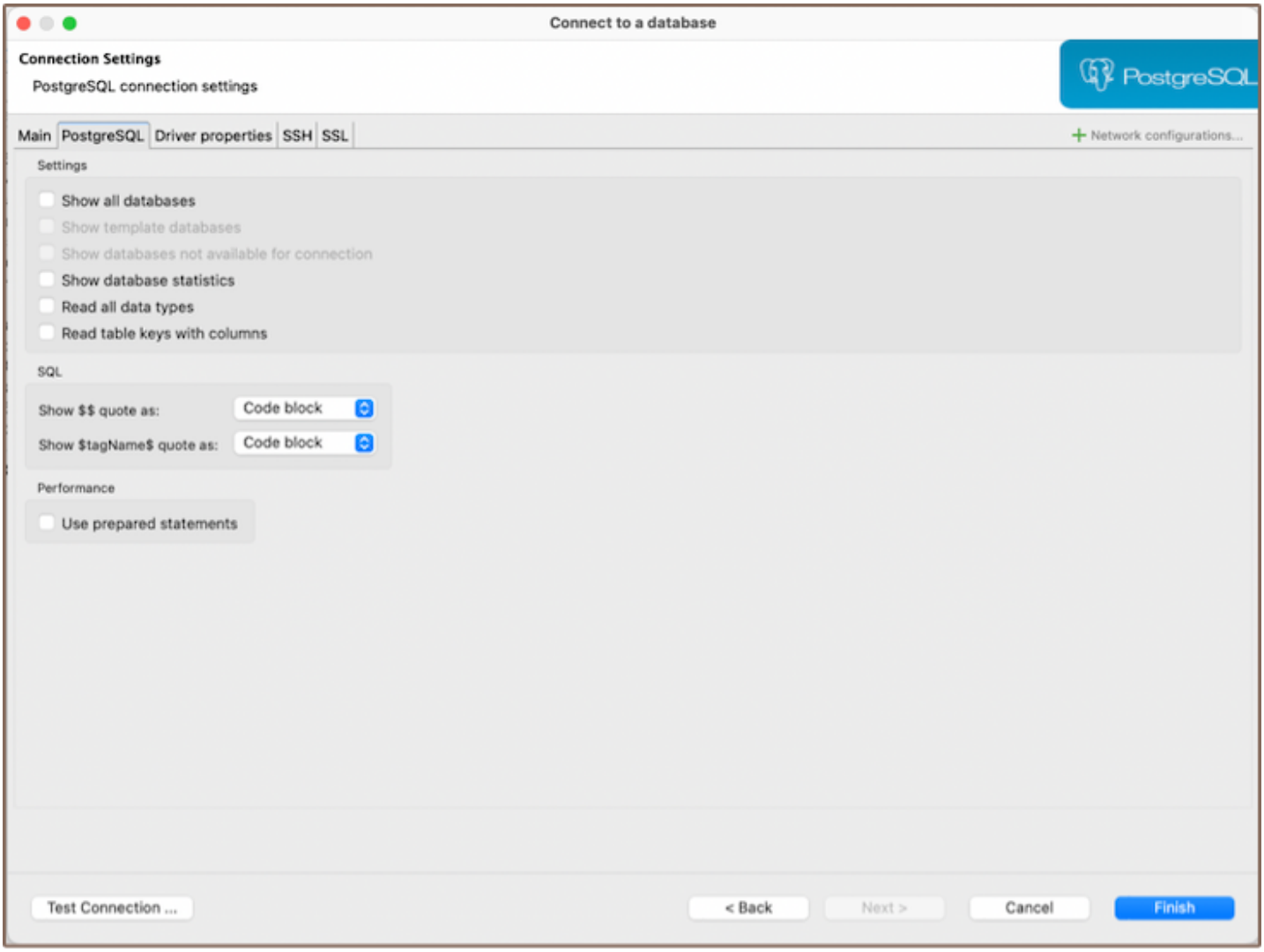
In this subsection, we will outline the settings for establishing a direct connection to a PostgreSQL database using DBeaver. Correctly configuring your connection ensures seamless interaction between DBeaver and your PostgreSQL database.

1) The first page of the connection settings requires you to fill in specific fields to establish the initial connection.



Field	Description
Connect by (Host/URL)	Choose whether you want to connect using a host or a URL.
Driver Type	Select the appropriate driver for your PostgreSQL database.
URL	If you are connecting via URL, enter the URL of your PostgreSQL database here. This field is hidden if you are connecting via the host.
Host	If you are connecting via host, enter the host address of your PostgreSQL database here.
Database	Enter the name of the PostgreSQL database you want to connect to.
Port	Enter the port number for your PostgreSQL database. The default PostgreSQL port is `5432`.
Authentication	Choose the type of authentication you want to use for the connection. For detailed guides on authentication types, please refer to the following articles: - - AWS RDS IAM Authentication ★ - Azure AD Authentication for PostgreSQL ★ - DBeaver Profile Authentication ★ - PostgreSQL Kerberos Authentication ★ - PostgreSQL PgPass Authentication - PostgreSQL SSPI Authentication ★ You can also read about security in DBeaver PRO.
Local Client	Specify the local client, see our article about Local client.
Connection Details	Provide additional connection details if necessary.
Driver Name	This field will be auto-filled based on your selected driver type.
Driver Settings	If there are any specific driver settings, configure them here.

2) The second page of the connection settings offers additional options that allow you to customize your further connection to the PostgreSQL database.



Field	Description
Show all databases	Displays all databases in the database navigator.
Show template databases	Displays template databases. This option is available when **Show all databases** is checked.
Show databases unavailable for connection	Displays databases unavailable. This option is available when **Show all databases** is checked.
Show databases statistics	Displays statistics for databases.
Read all data types	Reads all data types from the server.
Read table keys with columns	Reads table keys with columns.
Show `$$` quote as	Displays `$$` quotes as code blocks or strings.
Show `$tagName$` quote as	Displays `$tagName$` quotes as code blocks or strings.
Use prepared statements	Uses prepared statements for SQL execution.

Connection details

The **Connection Details** section in DBeaver allows for further customization of your PostgreSQL connection. This includes options for adjusting the **Navigator View**, setting up **Security measures**, applying **Filters**, configuring **Connection Initialization** settings, and setting up **Shell Commands**. Each of these settings can significantly impact your database operations and workflow. For detailed guides on these settings, please refer to the following articles:

- Connection Details Configuration
- Database Navigator
- Security Settings Guide
- Filters Settings Guide
- Connection Initialization Settings Guide
- Shell Commands Guide

PostgreSQL driver properties

The settings for PostgreSQL **Driver properties** enable you to adjust the performance of the PostgreSQL JDBC and ODBC driver. These adjustments can influence the efficiency, compatibility, and features of your PostgreSQL database.

For a complete walkthrough on setting up PostgreSQL JDBC and ODBC drivers properties, you can refer to the official PostgreSQL JDBC documentation and PostgreSQL ODBC documentation. These guides detail each driver's properties and how they can be used to optimize PostgreSQL database connections.

You can customize the PostgreSQL driver in DBeaver via the **Edit Driver** page, accessible by clicking on the **Driver Settings** button on the first page of the driver settings. This page offers a range of settings that can influence your PostgreSQL database connections. For a comprehensive guide on these settings, please refer to our Database drivers article.

ODBC and JDBC Driver Configuration

DBeaver provides extensive capabilities for managing PostgreSQL database connections via ODBC/JDBC drivers. This functionality enables you to connect to your PostgreSQL database using native ODBC drivers, offering an alternative when specific DBeaver drivers are not available.

You can find a comprehensive, step-by-step guide on how to install the driver manager, set up drivers, configure data sources, and establish connections in DBeaver in our ODBC Driver Configuration article.

Secure Connection Configurations

DBeaver supports secure connections to your PostgreSQL database. Guidance on configuring such connections, specifically **SSH**, **Proxy**, and **SSL** connections, can be found in various referenced articles. For a comprehensive understanding, please refer to these articles:

- SSH Configuration.
- Proxy Configuration.
- SSL Configuration.

Powering PostgreSQL with DBeaver

DBeaver provides a host of features designed for PostgreSQL databases. This includes the ability to view schemas, along with numerous unique capabilities aimed at optimizing database operations.

PostgreSQL database objects

DBeaver lets you view and manipulate a wide range of PostgreSQL database objects. DBeaver has extensive support for various PostgreSQL metadata types, allowing you to interact with a wide variety of database objects, such as:

- Databases
    - Schemas
        - Data types

- Tables
  - Columns
  - Constraints
  - Indexes
  - Foreign Keys
  - Dependencies
  - References
  - Partitions
  - Triggers
  - Rules
  - Policies
- Procedures and functions
- Views
- Materialized Views
- Sequences
- Aggregate functions
- Data types
- Roles
- Storage
  - Tablespaces
- Event Triggers
- Extensions

- Administer
  - Jobs
  - Session Manager
  - Lock Manager

For a practical guide on creating new tables and working with various database objects in PostgreSQL, check out our tutorial.

PostgreSQL Features in DBeaver

DBeaver is not limited to typical SQL tasks. It also includes numerous unique features specifically for PostgreSQL. Beyond regular SQL operations, DBeaver provides a range of PostgreSQL-specific capabilities, such as:

Category	Feature
Data Types	PostgreSQL Arrays
	PostgreSQL Structures
Extensions	PostgreSQL Extensions
Security	PostgreSQL Permissions
	PostgreSQL Policies
	PostgreSQL Roles
Data Organization	PostgreSQL Partitions
Database Management	PostgreSQL Dependencies
	PostgreSQL Tools

Additional features compatible with PostgreSQL, but not exclusive to it:

Category	Feature
Data Transfer	Data Import
	Data Export

Session Management	Session Manager
Lock Management	Lock Manager
Backup and Restore	How to Backup/Restore data
Schema Management	Schema Compare
Data Visualization	GIS Guide
	ERD Guide

Redis

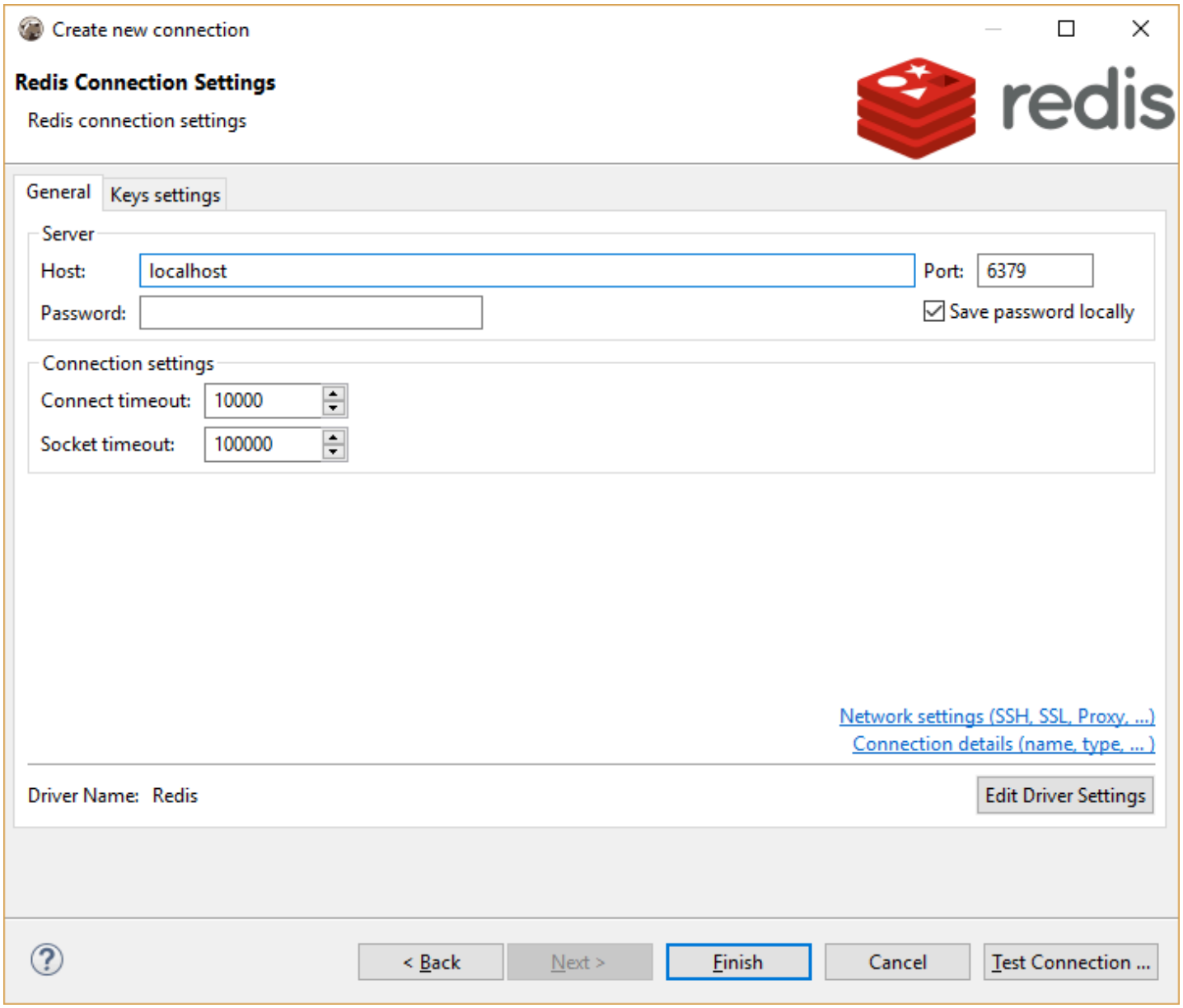
Overview

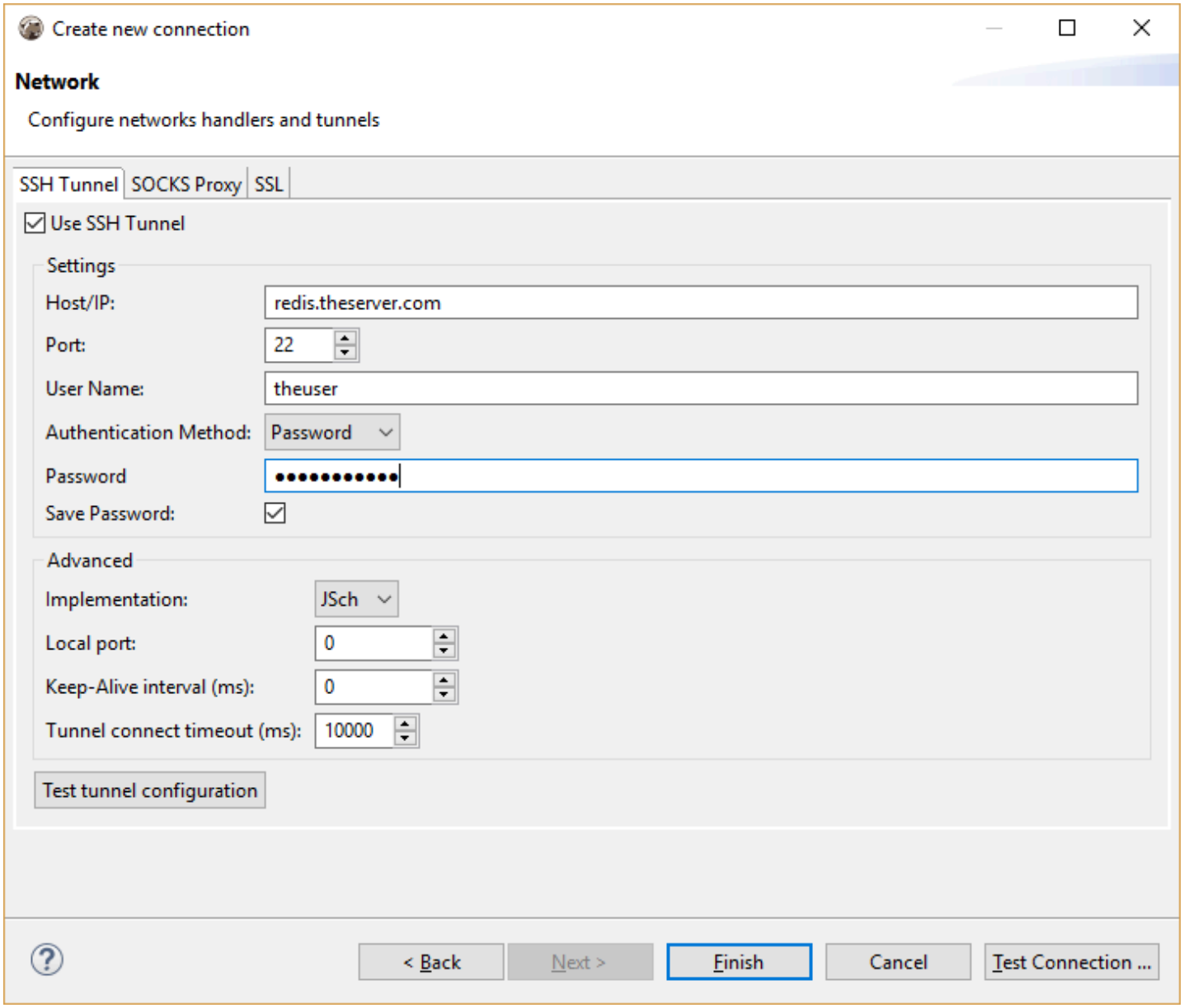
DBeaver EE supports Redis key browser, key value viewer and Redis commands shell.

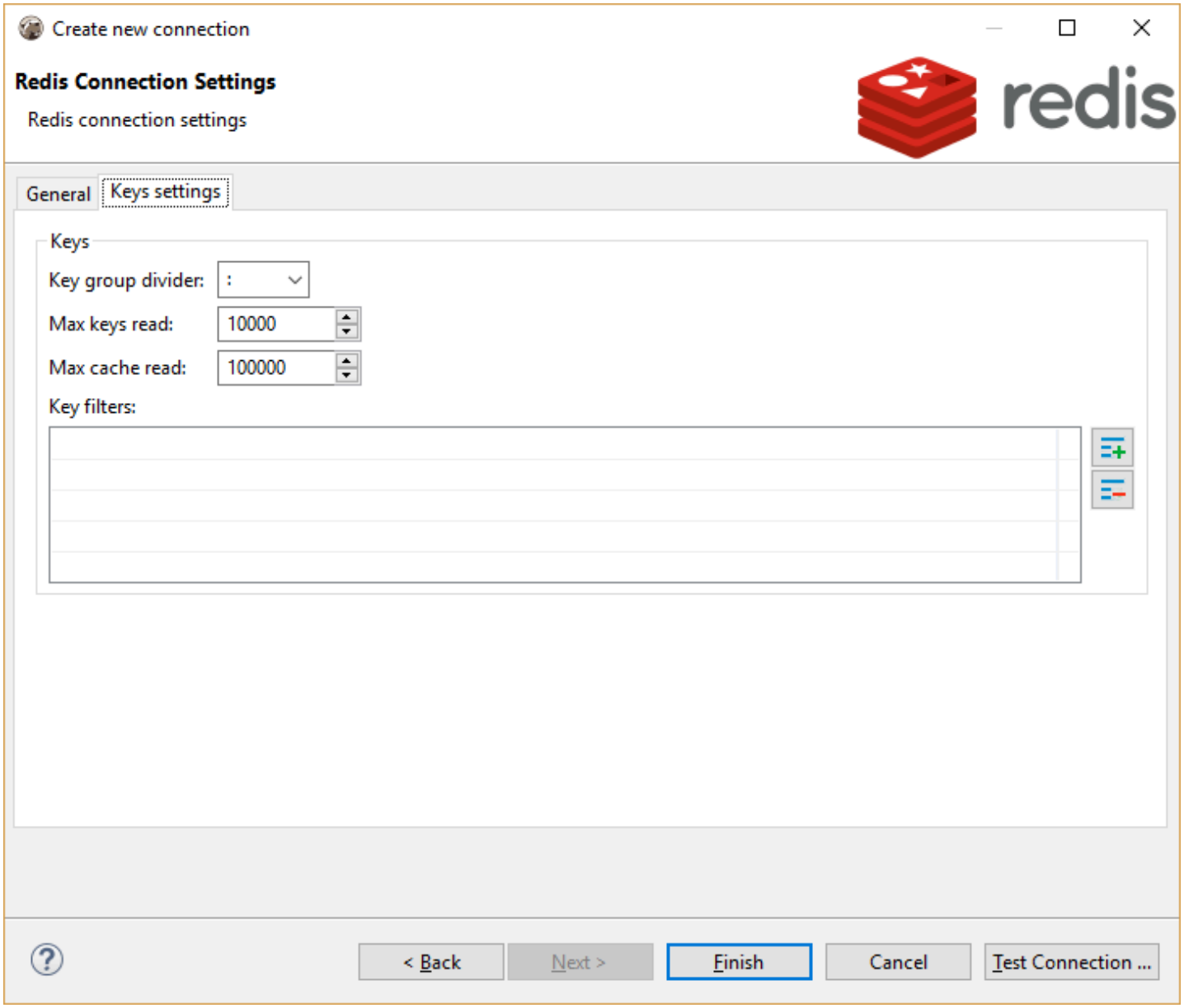
DBeaver uses Jedis driver 2.9.0 to operate with Redis server. It supports Redis servers of any version.

Connecting to Redis Server

You can connect directly to a server or use SSH tunneling or SOCKS proxy.


Create new connection
Redis Connection Settings
Redis connection settings
redis
General
Keys settings
Server
Host:
localhost
Port:
6379
Password:
Save password locally
Connection settings
Connect timeout:
10000
Socket timeout:
100000
Network settings (SSH, SSL, Proxy, ...)
Connection details (name, type, ...)
Driver Name: Redis
Edit Driver Settings
< Back
Next >
Finish
Cancel
Test Connection ...

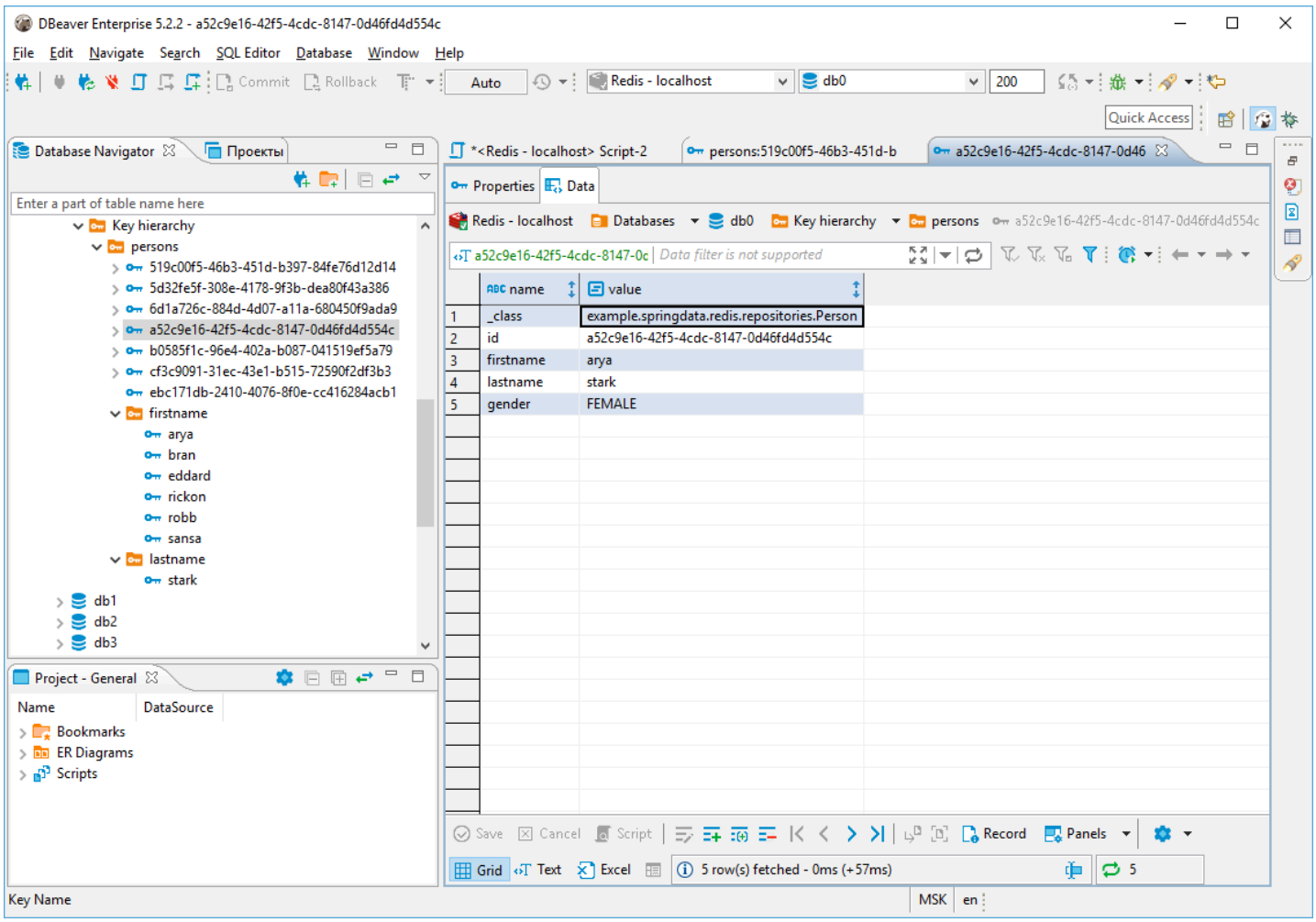

Create new connection
Network
Configure networks handlers and tunnels
SSH Tunnel
SOCKS Proxy
SSL
Use SSH Tunnel
Settings
Host/IP:
redis.theserver.com
Port:
22
User Name:
theuser
Authentication Method:
Password
Password
Save Password:
Advanced
Implementation:
JSch
Local port:
0
Keep-Alive interval (ms):
0
Tunnel connect timeout (ms):
10000
Test tunnel configuration
< Back
Next >
Finish
Cancel
Test Connection ...

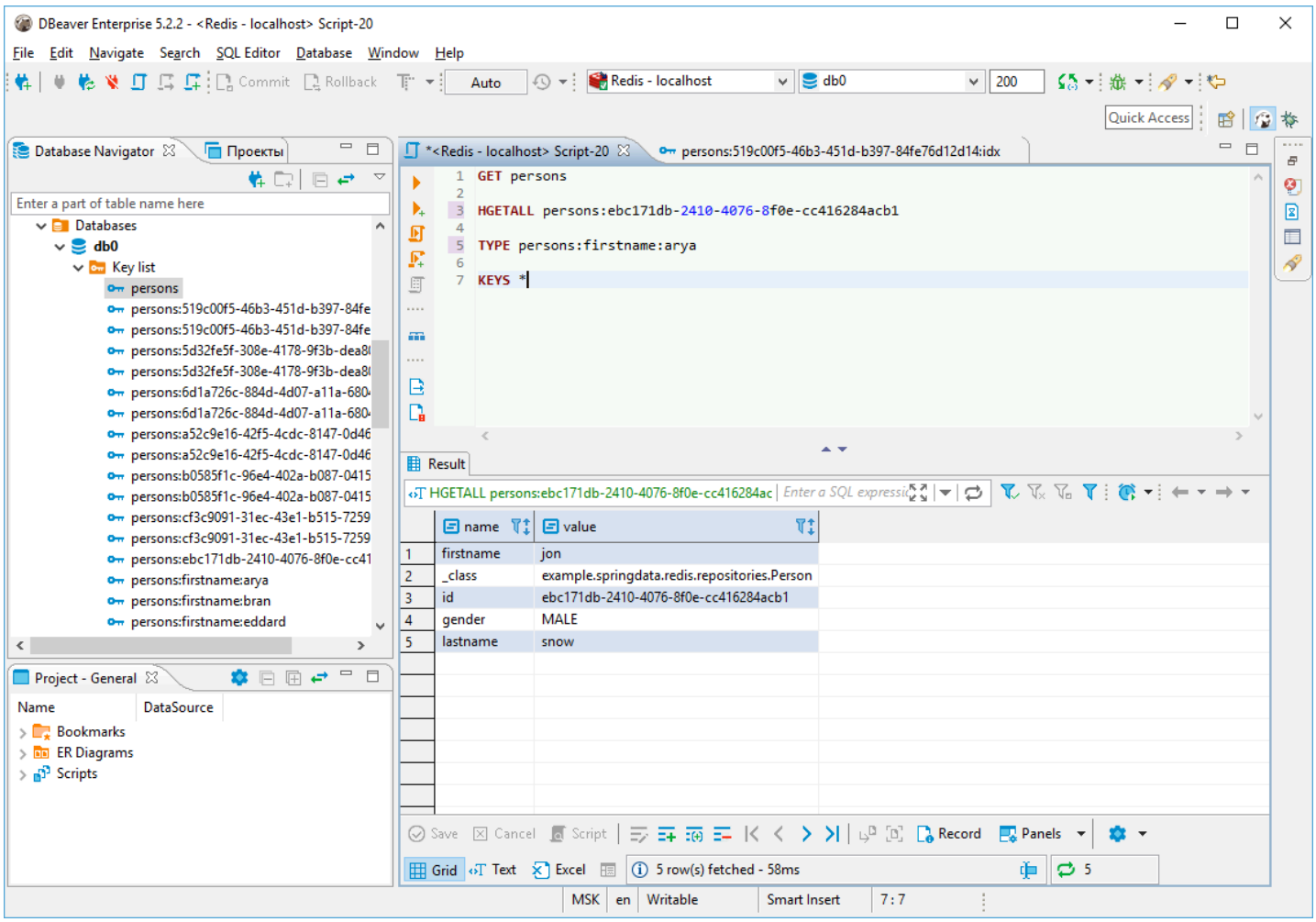


Browsing Redis keys

You can view/edit Redis keys as a plain list. However the Redis database usually contains a lot of keys (millions or even billions) and using list presentation is not convenient (or it is not possible).
DBeaver supports a hierarchy presentation of keys. Internally Redis does not support hierarchies but application level key names may be divided into groups using a character (e.g. coma, dash or colon). DBeaver uses this pattern to show hierarchy. Group separator can be configured in connection properties.

Key browser may be convenient in some cases but in the case of big databases it is very difficult to find your key in the navigator, so the SQL editor should be used instead. Redis commands is the most flexible way to operate with keys.


DBeaver Enterprise 5.2.2 - a52c9e16-42f5-4cdc-8147-0d46fd4d554c
File Edit Navigate Search SQL Editor Database Window Help
Commit Rollback Auto Redis - localhost db0 200
Quick Access
Database Navigator Проекты
Enter a part of table name here
Key hierarchy
persons
519c00f5-46b3-451d-b397-84fe76d12d14
5d32fe5f-308e-4178-9f3b-dea80f43a386
6d1a726c-884d-4d07-a11a-680450f9ada9
a52c9e16-42f5-4cdc-8147-0d46fd4d554c
b0585f1c-96e4-402a-b087-041519ef5a79
cf3c9091-31ec-43e1-b515-72590f2df3b3
ebc171db-2410-4076-8f0e-cc416284acb1
firstname
arya
bran
eddard
rickon
robb
sansa
lastname
stark
db1
db2
db3
Project - General
Name DataSource
Bookmarks
ER Diagrams
Scripts
*<Redis - localhost> Script-2
persons:519c00f5-46b3-451d-b
a52c9e16-42f5-4cdc-8147-0d46
Properties Data
Redis - localhost Databases db0 Key hierarchy persons a52c9e16-42f5-4cdc-8147-0d46fd4d554c
a52c9e16-42f5-4cdc-8147-0c Data filter is not supported
name value
1 _class example.springdata.redis.repositories.Person
2 id a52c9e16-42f5-4cdc-8147-0d46fd4d554c
3 firstname arya
4 lastname stark
5 gender FEMALE
Save Cancel Script Record Panels
Grid Text Excel 5 row(s) fetched - 0ms (+57ms) 5
Key Name
MSK en


DBeaver Enterprise 5.2.2 - <Redis - localhost> Script-20
File Edit Navigate Search SQL Editor Database Window Help
Commit Rollback Auto Redis - localhost db0 200
Quick Access
Database Navigator Проекты
Enter a part of table name here
Databases
db0
Key list
persons
persons:519c00f5-46b3-451d-b397-84fe
persons:519c00f5-46b3-451d-b397-84fe
persons:5d32fe5f-308e-4178-9f3b-dea8
persons:5d32fe5f-308e-4178-9f3b-dea8
persons:6d1a726c-884d-4d07-a11a-680
persons:6d1a726c-884d-4d07-a11a-680
persons:a52c9e16-42f5-4cdc-8147-0d46
persons:a52c9e16-42f5-4cdc-8147-0d46
persons:b0585f1c-96e4-402a-b087-0415
persons:b0585f1c-96e4-402a-b087-0415
persons:cf3c9091-31ec-43e1-b515-7259
persons:cf3c9091-31ec-43e1-b515-7259
persons:ebc171db-2410-4076-8f0e-cc41
persons:firstname:arya
persons:firstname:bran
persons:firstname:eddard
Project - General
Name DataSource
Bookmarks
ER Diagrams
Scripts
*<Redis - localhost> Script-20
persons:519c00f5-46b3-451d-b397-84fe76d12d14:idx
1 GET persons
2
3 HGETALL persons:ebc171db-2410-4076-8f0e-cc416284acb1
4
5 TYPE persons:firstname:arya
6
7 KEYS *
Result
HGETALL persons:ebc171db-2410-4076-8f0e-cc416284ac Enter a SQL expressi
name value
1 firstname jon
2 _class example.springdata.redis.repositories.Person
3 id ebc171db-2410-4076-8f0e-cc416284acb1
4 gender MALE
5 lastname snow
Save Cancel Script Record Panels
Grid Text Excel 5 row(s) fetched - 58ms 5
MSK en Writable Smart Insert 7 : 7

Executing Redis commands

Redis does not support SQL or any other query language. Instead, it supports build-in commands and LUA scripts.

Redis commands can be executed in the same way as in a Redis command line shell:

```
In order to execute a command, run it using CTRL+Enter or ALT+X. All standard DBeaver SQL editor shortcuts

In order to execute a LUA script, surround it with {} brackets and run it as a single statement. If the sc
```lua
{
    return {1,2,{3,'Hello World!'}}
}
```
```

Salesforce

Note: This driver is available in Lite, Enterprise, Ultimate and Team editions only.

Overview

This guide provides instructions on how to set up and use Salesforce with DBeaver.

Before you start, you must create a connection in DBeaver and select Salesforce. If you have not done this, please refer to our Database Connection article.

DBeaver interacts with the Salesforce server using a specific driver. To use it, choose the right Salesforce driver in the Connect to a database window.

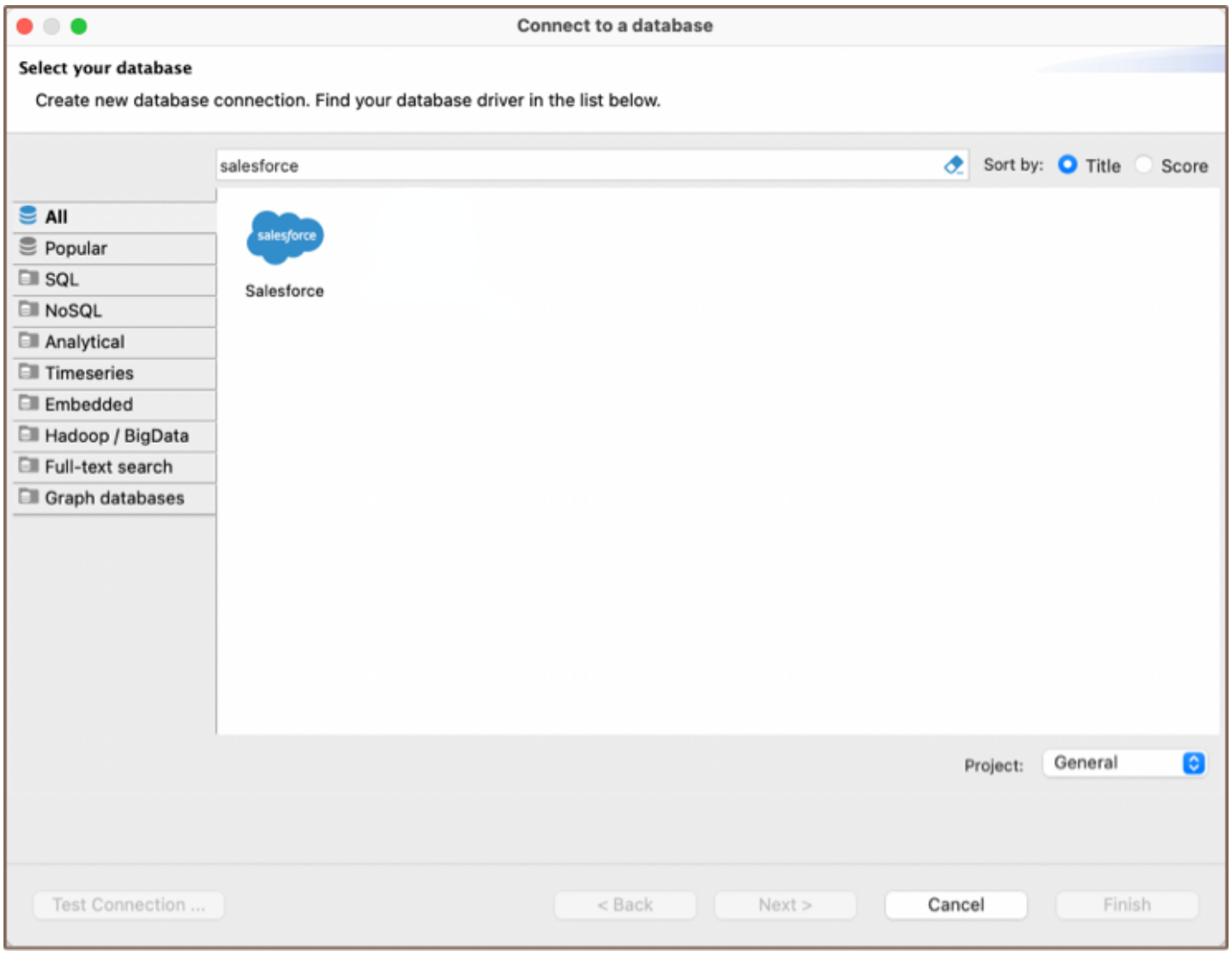


Table of contents:

- Setting Up
  - Salesforce's connection settings
    - Connection details
  - Salesforce driver properties
  - Secure Connection Configurations
- Powering Salesforce with DBeaver.
  - Salesforce database objects
- Salesforce Object Query Language
  - SOQL vs SQL: Key Differences
  - Examples
    - Simple Select
    - Select with Filtering
    - Select with Grouping

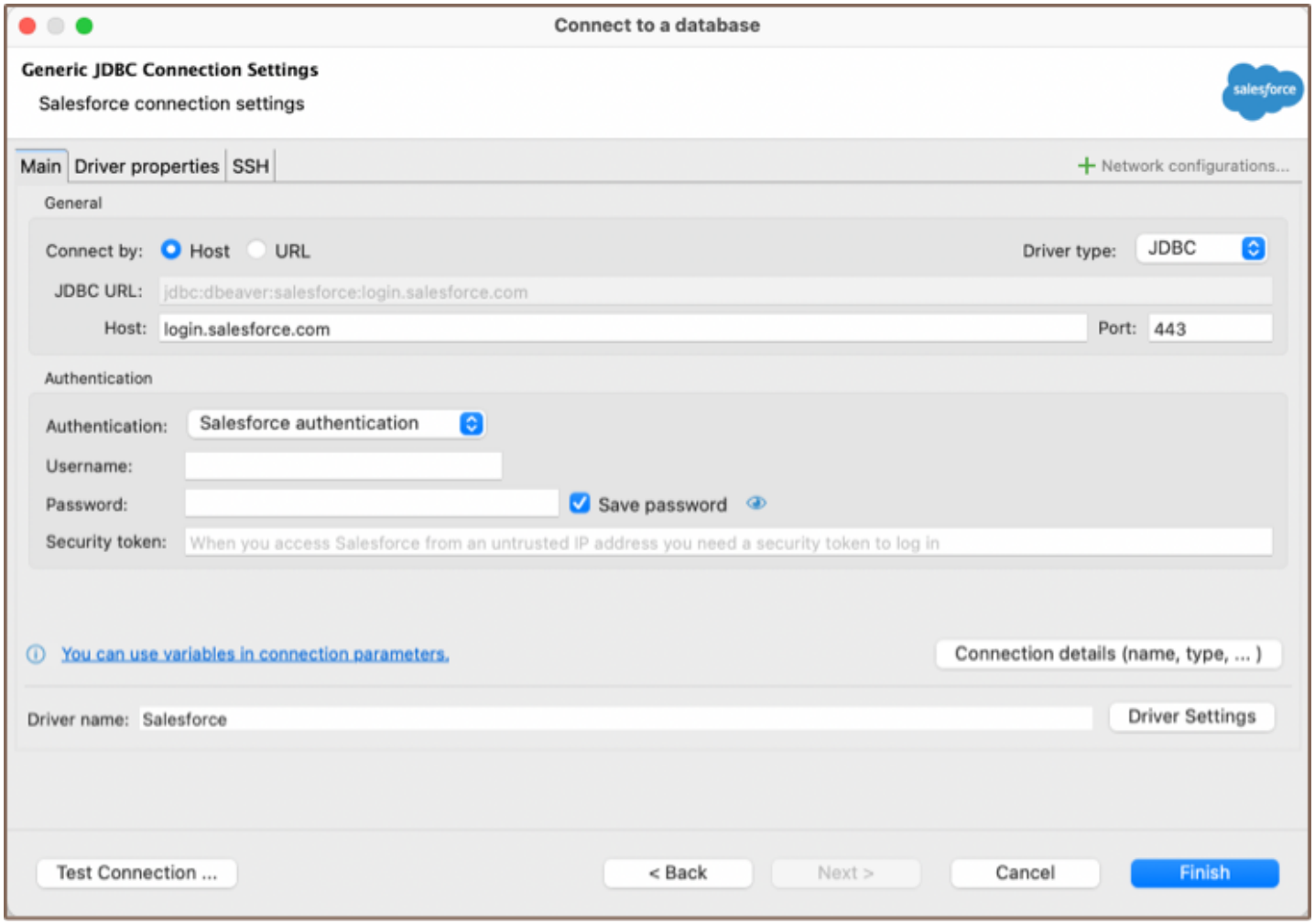
Setting Up

This section provides an overview of DBeaver's settings for establishing a direct connection and the configuration of secure connections using SSH and proxies.

Salesforce connection settings

In this subsection, we will outline the settings for establishing a direct connection to a Salesforce database using DBeaver.

The page of the connection settings requires you to fill in specific fields to establish the initial connection.



Field	Description
Connect by (Host/URL)	Choose whether you want to connect using a host or a URL.
Driver Type	Select the appropriate driver for your Salesforce database.
URL	If you are connecting via URL, enter the URL of your Salesforce database here. This field is disabled if you're connecting via the host.
Host	If you are connecting via host, enter the host address of your Salesforce database here.
Port	Enter the port number for your Salesforce database. The default Salesforce port is `443`.
Authentication	Choose the type of authentication you want to use for the connection. For detailed guides on authentication types, please refer to the following articles: - DBeaver Profile Authentication - Salesforce authentification (article coming soon)

	You can also read about security in DBeaver PRO.
Connection Details	Provide additional connection details if necessary.
Driver Name	This field will be auto-filled based on your selected driver type.
Driver Settings	If there are any specific driver settings, configure them here.

Connection details

The **Connection Details** section in DBeaver allows you to customize your experience while working with Salesforce database. This includes options for adjusting the **Navigator View**, setting up **Security measures**, applying **Filters**, configuring **Connection Initialization** settings, and setting up **Shell Commands**. Each of these settings can significantly impact your database operations and workflow. For detailed guides on these settings, please refer to the following articles:

- Connection Details Configuration
- Database Navigator
- Security Settings Guide
- Filters Settings Guide
- Connection Initialization Settings Guide
- Shell Commands Guide

Salesforce driver properties

Salesforce is not a traditional Relational Database Management System (RDBMS), and as such, it does not come with a standard JDBC driver. DBeaver uses REST API for connecting to the Salesforce server.

There are no particular properties for configuration at this time. The driver is designed with the current functionality of the Salesforce platform in mind, focusing solely on providing the ability to read data.

Secure Connection Configurations

DBeaver supports secure connections to your Salesforce database. Guidance on configuring such connections, specifically **SSH** and **Proxy** connections, can be found in various referenced articles. For a comprehensive understanding, please refer to these articles:

- SSH Configuration.
- Proxy Configuration.

Powering Salesforce with DBeaver

At the current version of DBeaver, the Salesforce database driver allows read-only access to its data, enabling you to execute `SELECT` queries to view data. This includes the ability to use filters, groupings, data export, ER diagrams and other functionalities of DBeaver for viewing and analyzing the data.

Salesforce database objects



DBeaver lets you view Salesforce database objects and has support for a limited number of Salesforce metadata types, allowing you to interact with a select variety of database objects, such as:

- Tables
  - Columns
  - Keys

    - Foreign Keys
    - References
- Data Types

Salesforce Object Query Language

SOQL is a query language used to search Salesforce data. Within DBeaver, you can utilize SOQL to execute read-only `SELECT` queries on Salesforce data.

SOQL vs SQL

SOQL is specifically tailored for querying Salesforce data and has some distinct differences from traditional SQL:

- Read-Only Queries: SOQL is used primarily for querying data, not for data manipulation. In Salesforce, data insertion and manipulation are typically handled through the platform's user interface, APIs, or specialized data loading tools, rather than through SOQL.
- Limited `JOIN` Operations: Unlike SQL, SOQL does not support all types of `JOIN` operations.
- No Arbitrary Expressions: SOQL does not allow arbitrary expressions or calculations in the `SELECT` clause, unlike SQL.

Examples

Here are some brief examples, you can find more details in the official SOQL tutorial:

Simple Select

Retrieve the `Name` and `Id` of all accounts:

```
SELECT Id, Name
FROM Account
```

Select with Filtering

Retrieve accounts with a specific status:

```
SELECT Id, Name
FROM Account
WHERE Status = 'Active'
```

Select with Grouping

Group accounts by `status` and count them:

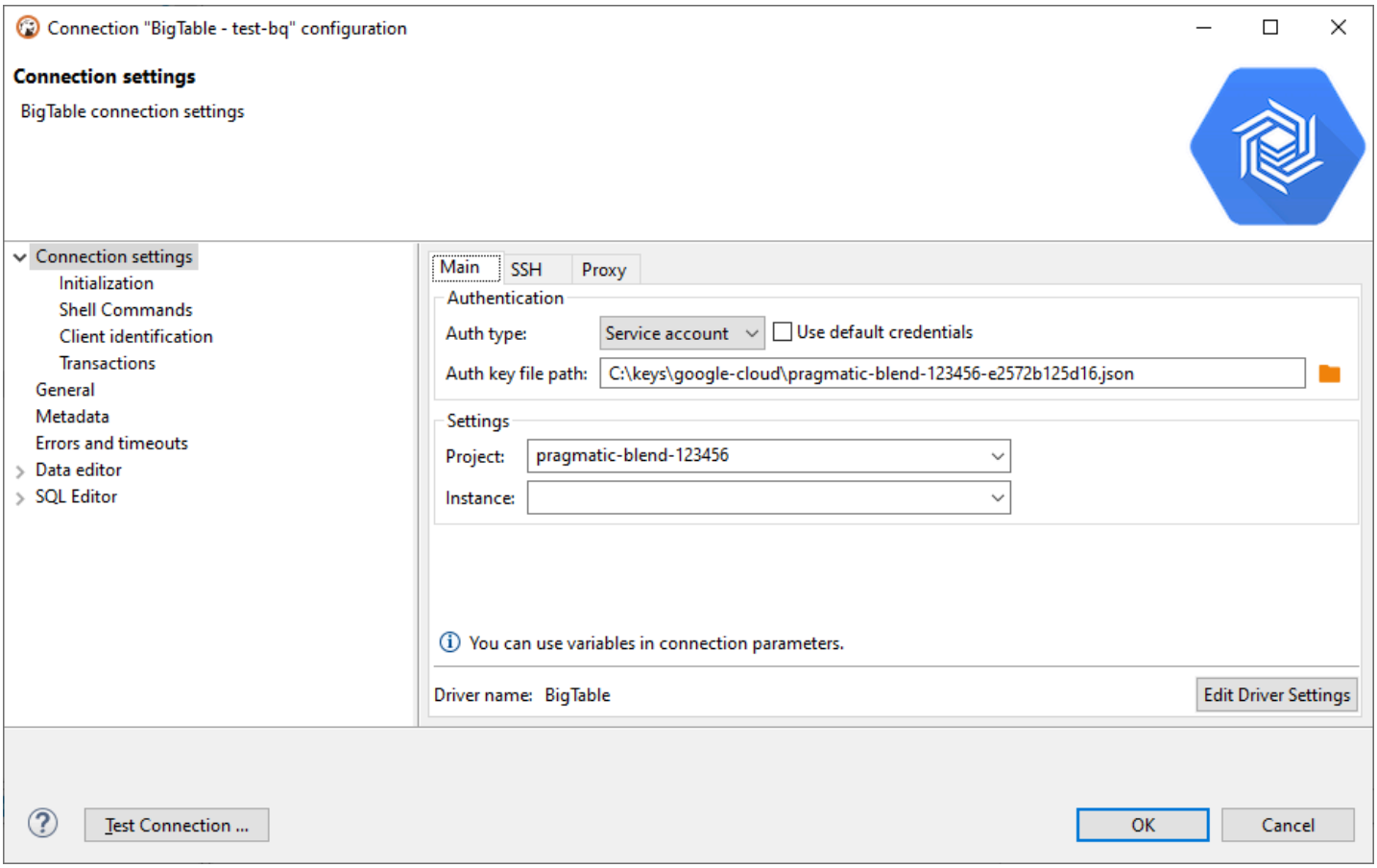
```
SELECT Status, COUNT(Id)
FROM Account
GROUP BY Status
```

Bigtable

Supported features:

- Table data view
- Table data edit in document (json) mode
- Data filters
- SQL queries execution
- Data export and import

Bigtable connection



DocumentDB

AWS DocumentDB is based on the MongoDB engine.

It has several minor differences in the query processing and network configuration.

However, most features which work for MongoDB will work for DocumentDB as well. Please refer to the MongoDB article.

Connections

AWS restricts direct access to DocumentDB clusters from outside of the cloud (region). So you can connect to it directly (using a cluster host name) only when DBeaver is deployed on the EC2 instance.

In other cases you will need to use the SSH tunnel through a proxy machine to access DocumentDB instance.

Please read the AWS Documentation about proxy configurations: https://docs.aws.amazon.com/documentdb/latest/developerguide/connect-from-outside-a-vpc.html

In DBeaver you can use the SSH tab on the connection settings page. Just enter proxy host, user name and specify a private key file (it is provided by AWS as a keypair).

Queries

DBeaver processes DocDB SQL queries exactly like in MongoDB. It supports SELECT, UPDATE, INSERT and DELETE queries.

SELECT queries support WHERE, ORDER BY, GROUP BY and HAVING clauses.

DocumentDB restricts the `eval` function so all JavaScript queries will be parsed on the client's side and then evaluated at a DocDB cluster one by one. Most JS functions work exactly like in Mongo Shell.

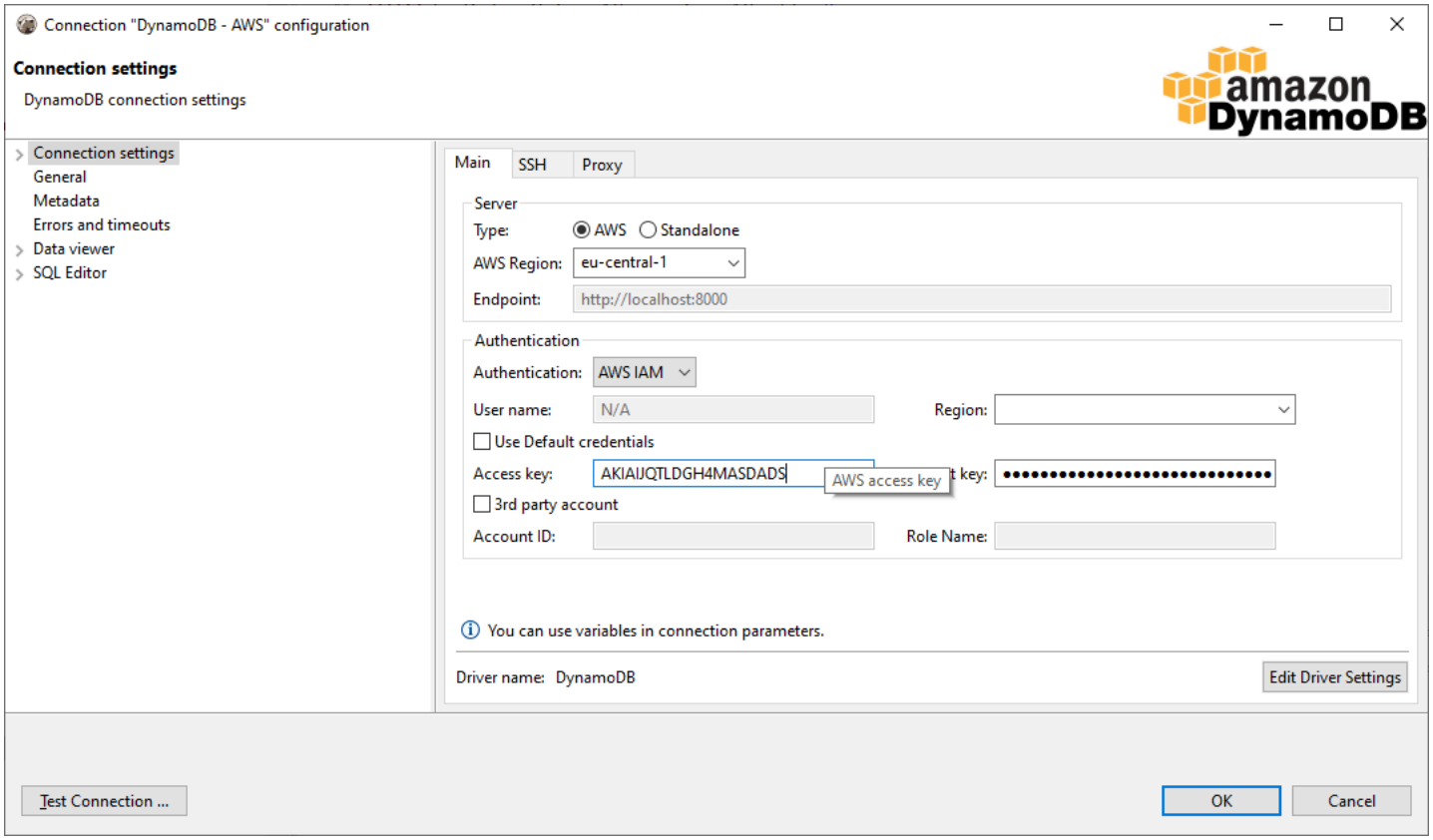
DynamoDB

Note: This feature is available in Enterprise, Ultimate and Team editions only.

Supported features:

- Table data view
- Table data edit in document (json) mode
- Data filters
- SQL queries execution
- JSON queries execution
- Data export and import

DynamoDB connection



DBeaver supports AWS Cloud and Standalone versions of DynamoDB.

For standalone server you need to enter endpoint (http or https URL).

For cloud server you must enter the AWS region. DynamoDB exists in all available regions in your AWS account but the tables are different.

AWS Access Key and Secret Key are used for authentication.

For 3rd-party account access you must specify the 3rd party account ID (12-digits number) and the 3rd party role name. This role will be used for permission management. You account must be added to the whitelist in the 3rd party account.

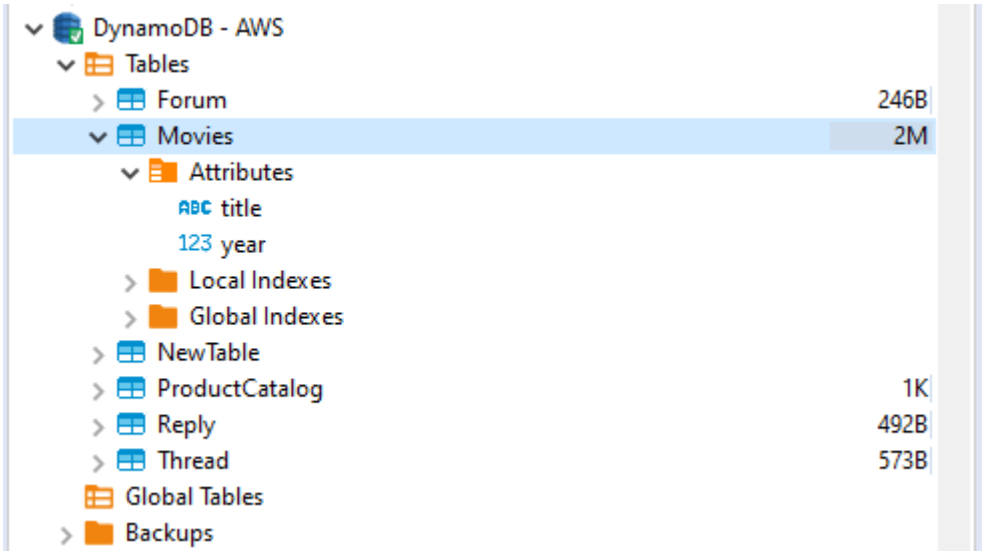
Press "Test Connection" to validate your connection settings.

Database navigation

DynamoDB has a simple metadata structure. Basically, you can only access Table and Global tables.

Table has primary attributes (a kind of primary key) and indexes.

DynamoDB is a document-oriented database. Each table may have its own set of attributes and sub-attributes.

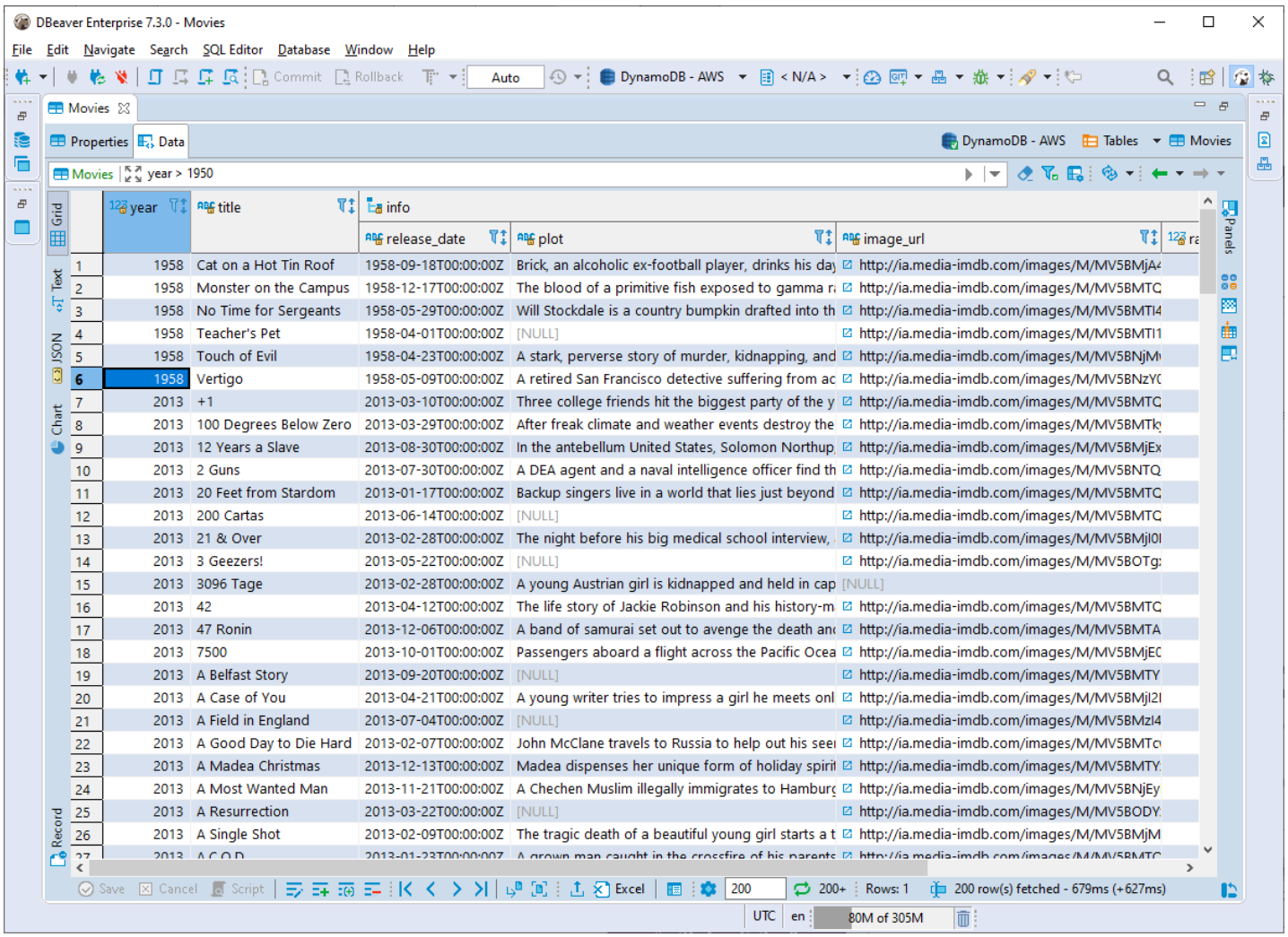


Viewing table data

You can open table editor and see the table data.

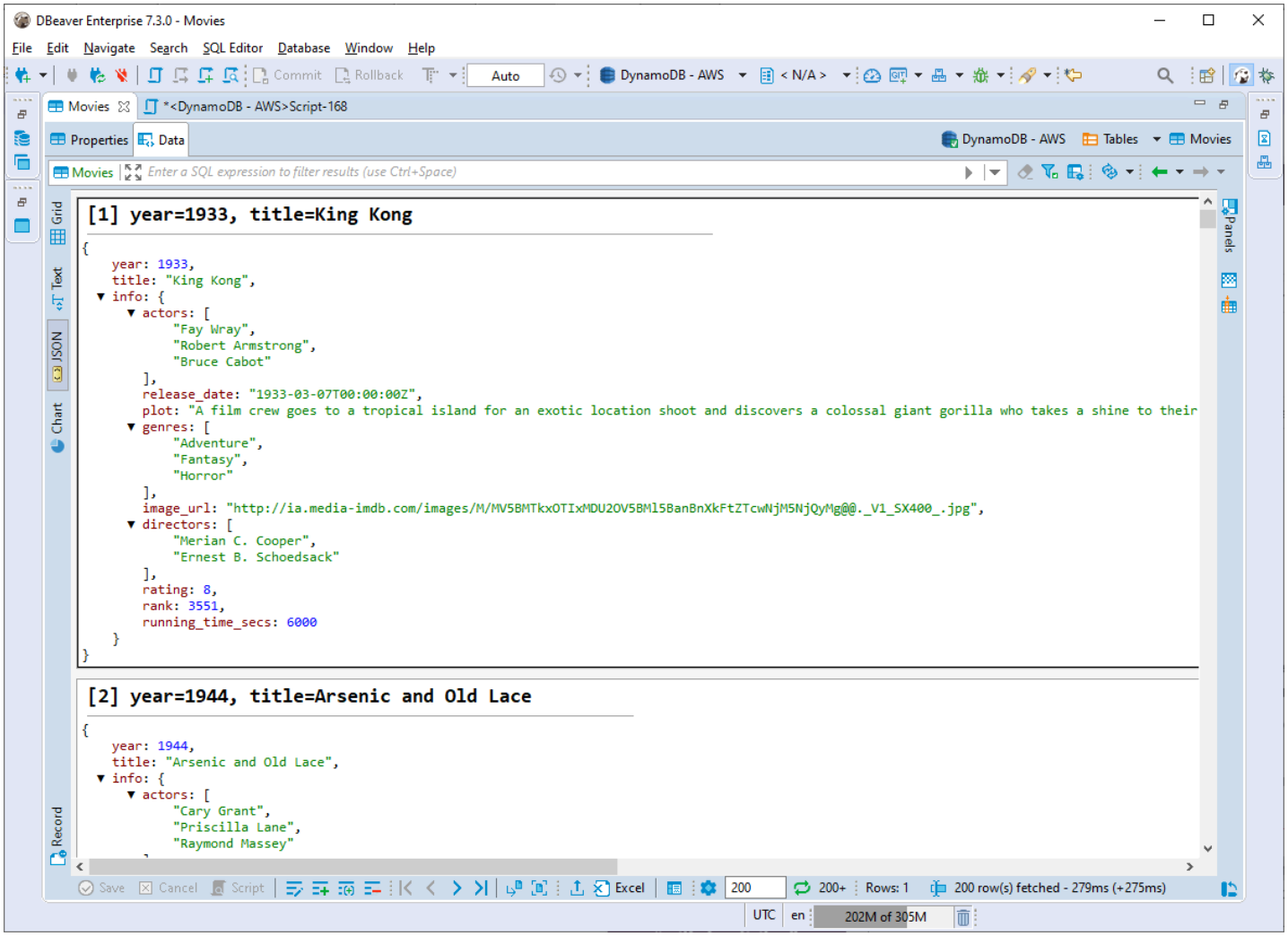
You may need to switch to the "Data" tab. DBeaver converts DynamoDB documents into a table format by default, but you can switch to another data representation.

You can use data filters in order to find documents.



Viewing data in JSON document format

You view, search and edit JSON documents. Double-click on a document to activate the editor.

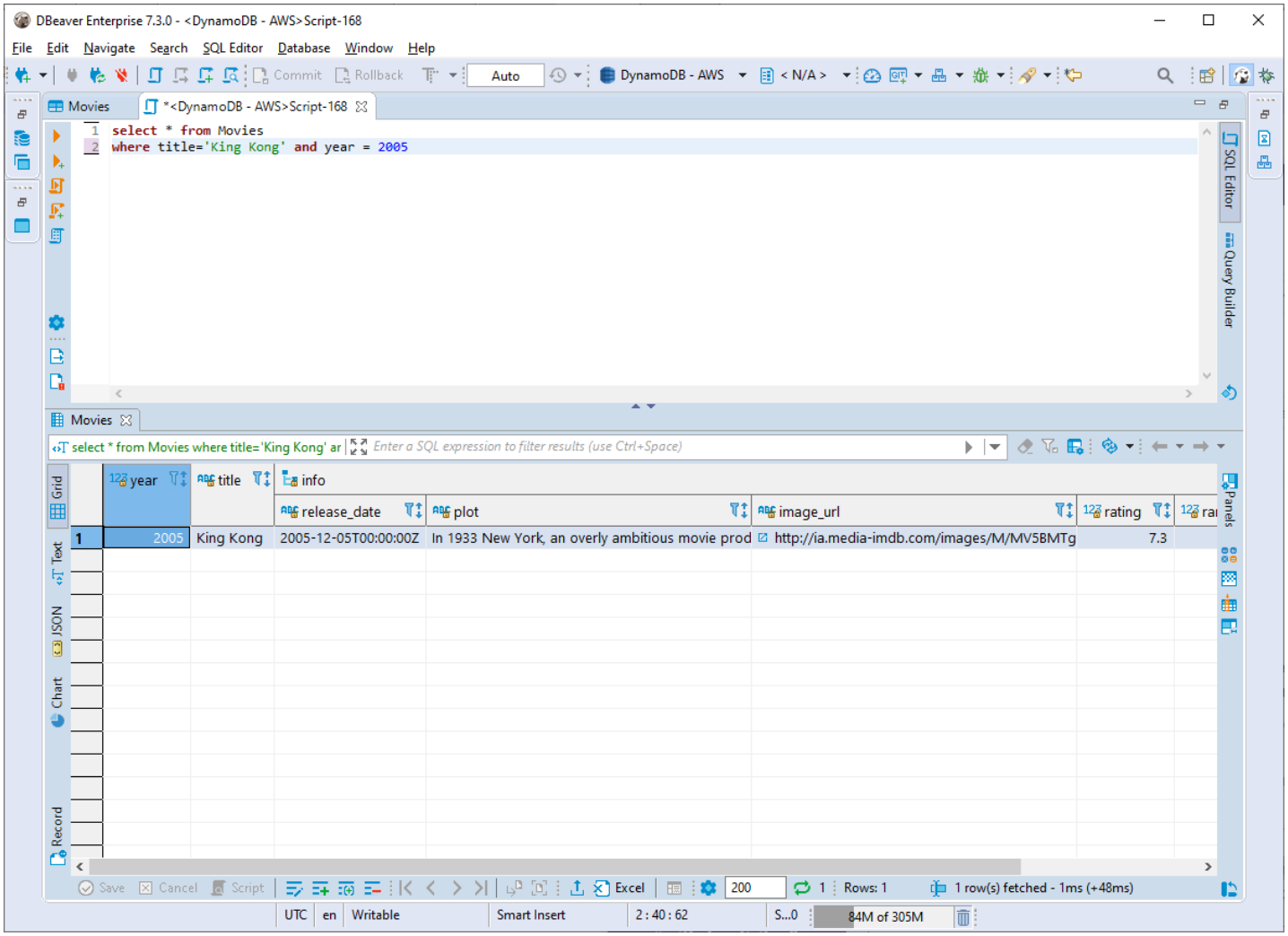


Executing queries

DBeaver supports simple SQL dialect for DynamoDB.

You can use the WHERE clause in the same fashion as in regular SQL in order to find or filter documents.

You can also use JSON requests syntax to query documents. See Amazon DynamoDB query reference.



Exporting and importing data

You can export data from a DynamoDB table in different file formats (CSV, XLSX, XML, JSON, etc,) or export data directly to another table.

Keyspaces

Overview

AWS Keyspaces is a key-value database based on Apache Cassandra.

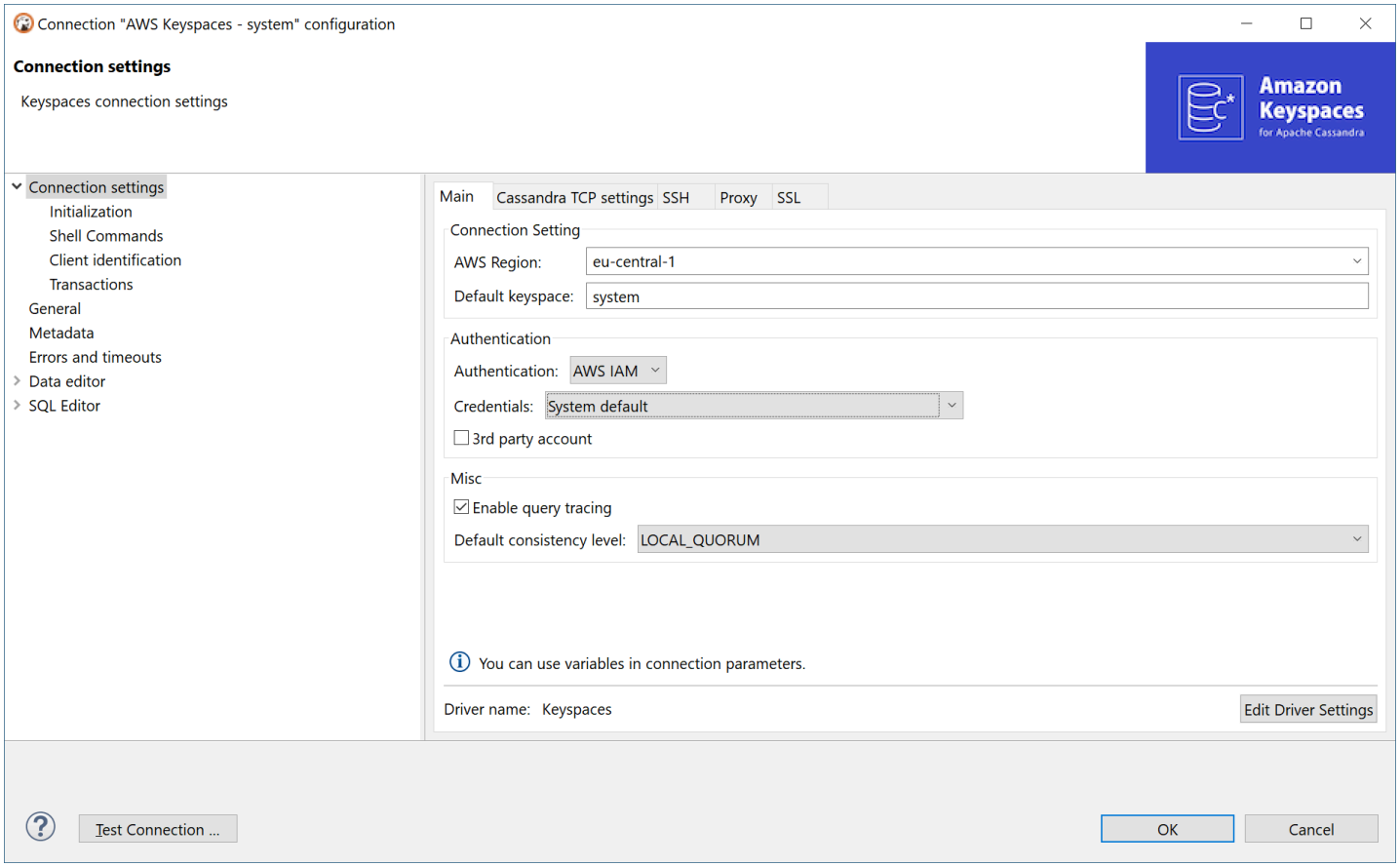
DBeaver EE supports the Keyspaces schema browser, data viewer and CQL queries execution. It also supports various administrative tools.

Connecting to Keyspaces

AWS Keyspaces uses AWS IAM authentication.

You need to specify your IAM credentials and AWS region. There is no specific endpoint - there is only one cluster per account per AWS region.

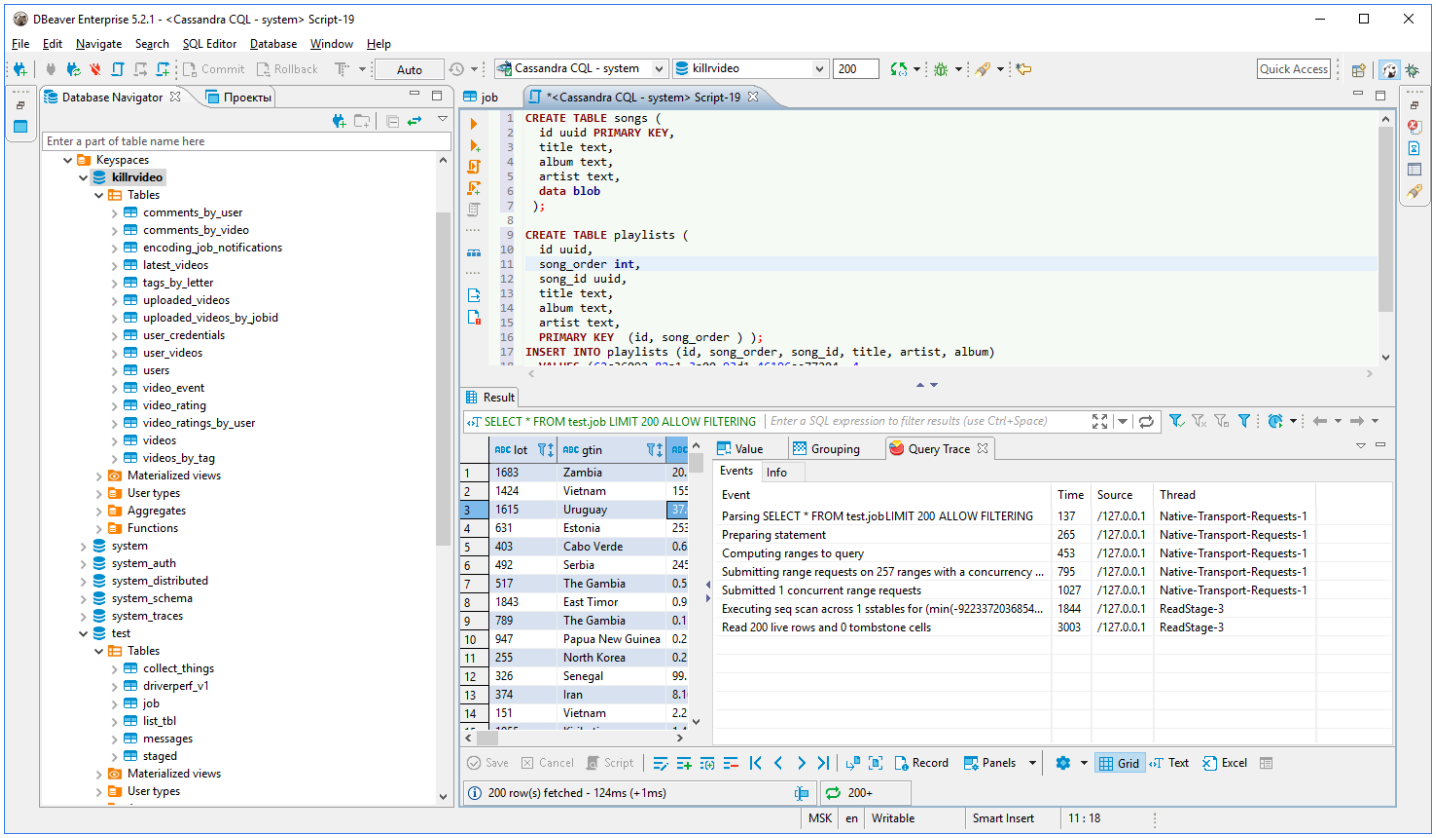
There is also no need to configure SSH or SSL - DBeaver uses default AWS settings to access the Keyspace cluster.

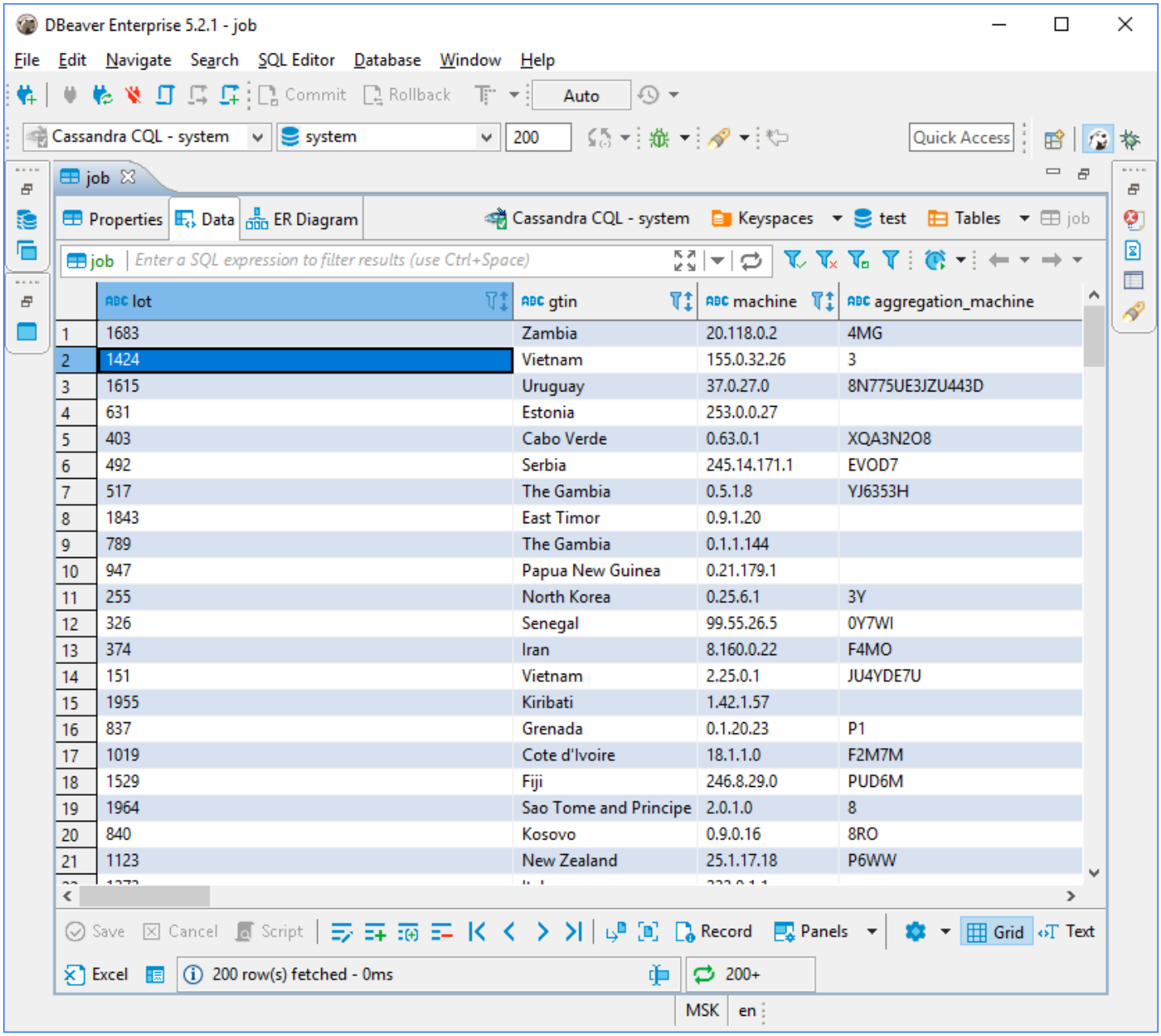


Browsing Keyspaces tables

You can browse, view, edit and filter Cassandra tables the same way as with regular (relational) tables. However, being a distributed key-value database, Keyspaces does not support any kind of referential integrity. There are no foreign keys, references, etc.

You should note that Cassandra has a very advanced (comparing to relational databases) data type system. Each column may be a collection, map, or set of values (with a very big number of values). In some cases this makes browsing data in the "Grid" mode inconvenient.


DBeaver Enterprise 5.2.1 - <Cassandra CQL - system> Script-19
File Edit Navigate Search SQL Editor Database Window Help
Database Navigator
Enter a part of table name here
Keyspaces
killrvideo
Tables
comments_by_user
comments_by_video
encoding_job_notifications
latest_videos
tags_by_letter
uploaded_videos
uploaded_videos_by_jobid
user_credentials
user_videos
users
video_event
video_rating
video_ratings_by_user
videos
videos_by_tag
Materialized views
User types
Aggregates
Functions
system
system_auth
system_distributed
system_schema
system_traces
test
collect_things
driverperf_v1
job
list_tbl
messages
staged
CREATE TABLE songs (
id uuid PRIMARY KEY,
title text,
album text,
artist text,
data blob
);
CREATE TABLE playlists (
id uuid,
song_order int,
song_id uuid,
title text,
album text,
artist text,
PRIMARY KEY (id, song_order));
INSERT INTO playlists (id, song_order, song_id, title, artist, album)
Result
SELECT * FROM test.job LIMIT 200 ALLOW FILTERING
Enter a SQL expression to filter results (use Ctrl+Space)
lot
gtin
1 1683 Zambia
2 1424 Vietnam
3 1615 Uruguay
4 631 Estonia
5 403 Cabo Verde
6 492 Serbia
7 517 The Gambia
8 1843 East Timor
9 789 The Gambia
10 947 Papua New Guinea
11 255 North Korea
12 326 Senegal
13 374 Iran
14 151 Vietnam
Value
Grouping
Query Trace
Events
Info
Event Time Source Thread
Parsing SELECT * FROM test.job LIMIT 200 ALLOW FILTERING 137 /127.0.0.1 Native-Transport-Requests-1
Preparing statement 265 /127.0.0.1 Native-Transport-Requests-1
Computing ranges to query 453 /127.0.0.1 Native-Transport-Requests-1
Submitting range requests on 257 ranges with a concurrency ... 795 /127.0.0.1 Native-Transport-Requests-1
Submitted 1 concurrent range requests 1027 /127.0.0.1 Native-Transport-Requests-1
Executing seq scan across 1 sstables for (min(-9223372036854... 1844 /127.0.0.1 ReadStage-3
Read 200 live rows and 0 tombstone cells 3003 /127.0.0.1 ReadStage-3
Save Cancel Script Record Panels Grid Text Excel
200 row(s) fetched - 124ms (+1ms)
MSK en Writable Smart Insert 11 : 18



Executing CQL

CQL Cassandra Query Language is a very simple kind of SQL language dialect.

It supports simple SELECT queries, DDL statements (like CREATE TABLE) and some other query types.

You can use the standard DBeaver SQL editor to execute CQL queries. DBeaver supports Cassandra query execution, results scrolling, data export/import, mock data generation and other features.

ERD

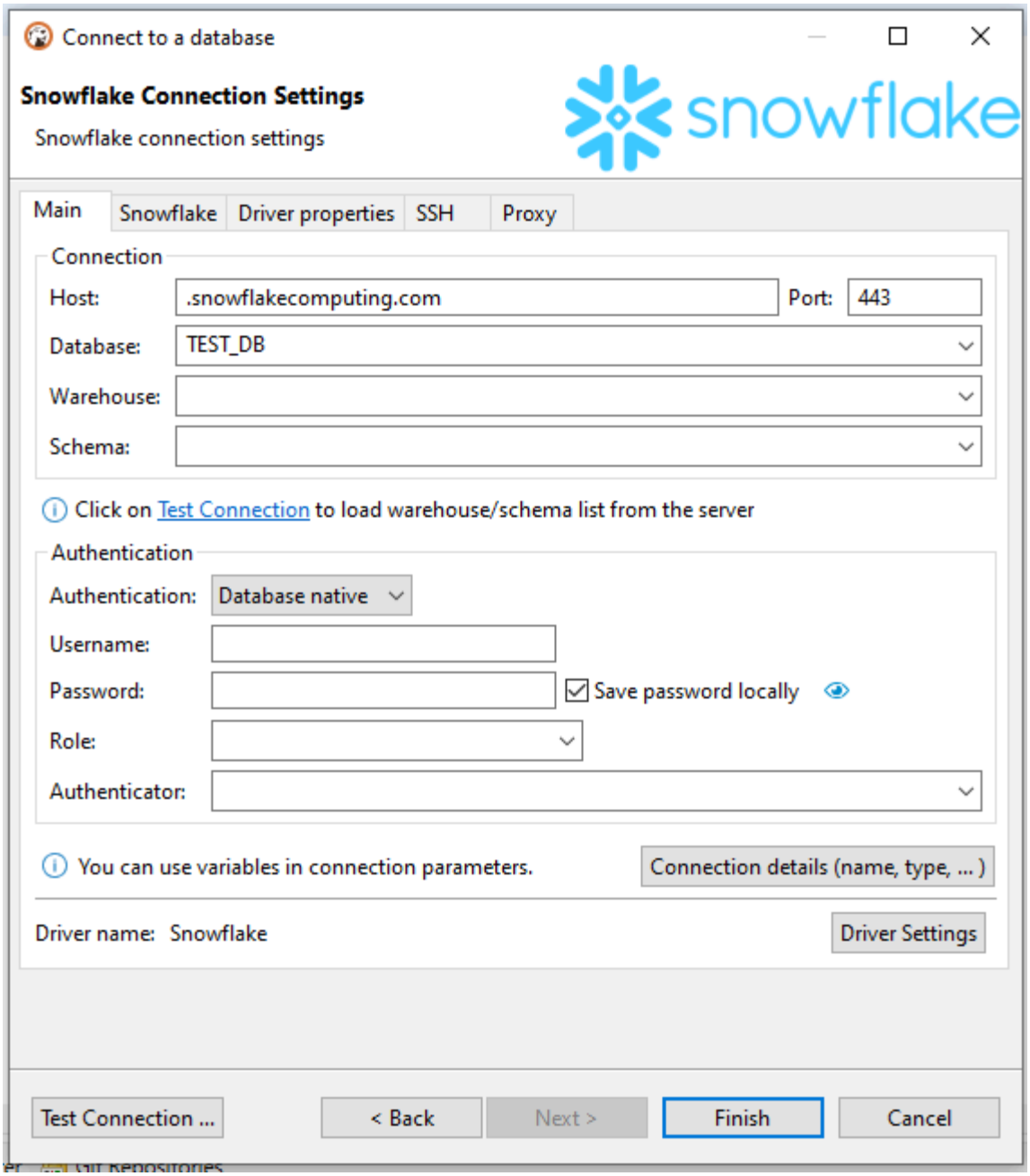
Physical ERD (Entity Relation Diagram) does not make much sense for Keyspaces as there are no foreign keys. However, you can make you own custom ERD and connect Keyspaces tables with each other using logical associations.

Snowflake

You can read more about the Snowflake database in the Snowflake documentation.

Connecting to the Snowflake database.

There are several ways to configure a database connection.



Connection parameters

Parameter	Description	Example
Host	Server host name	https://account.region.snowflakecomputing.com
Post number	Server listener port	443 (default)
Database*	Database name	SNOWFLAKE_SAMPLE_DATA
Warehouse*	Cluster of compute resources in Snowflake. Warehouses are required for queries, as well as all DML operations, including loading data into tables	SNOWFLAKE
Schema*	Schema name	SAMPLE_DATA

*Click on the "Test Connection" button to load the warehouse/database/schema list from the server.

Authentication

Database

Parameter	Description	Example
Username	Database user name	ADMIN
Password	Database user password	*****
Save password	Saves the user/password information in the local DBeaver configuration	
Role	Role for connection	SYS
Authenticator	Authenticator parameter/option	https://.okta.com

Private key ★

Note: This feature is available in Lite, Enterprise, Ultimate and Team editions only.

Parameter	Description	Example
Username	Database user name	ADMIN

Private key	Path to encrypted or unencrypted private key	C:/Users/admin/Downloads/rsa_key.p8
Key Password	Private key password	*****
Save password	Saves the user/password information in the local DBeaver configuration	
Role	Role for connection	SYS

SSO (Browser) ★

Note: This feature is available in Lite, Enterprise, Ultimate and Team editions only.

Connection with the help of an external browser will be used.

Parameter	Description	Example
Username	Database user name	ADMIN
Role	Role for connection	SYS

DBeaver Profile ★

You can use preconfigured connection profile here.

Metadata

DBeaver provides an opportunity to view and manipulate such types of metadata:

- Databases
- Schemas
- Data types
- Tables

- Views
- Procedures and functions
- Sequences ★
- Stages ★
- Pipes ★
- Streams ★
- Tasks ★

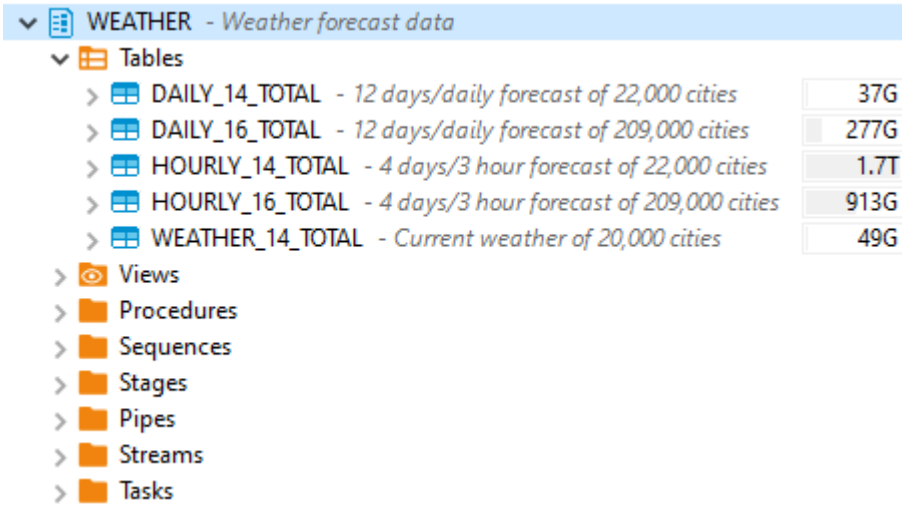
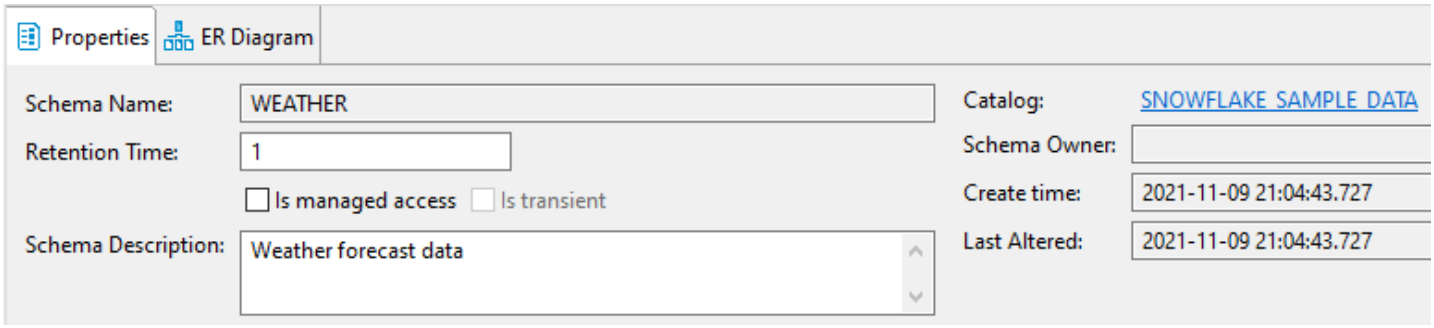


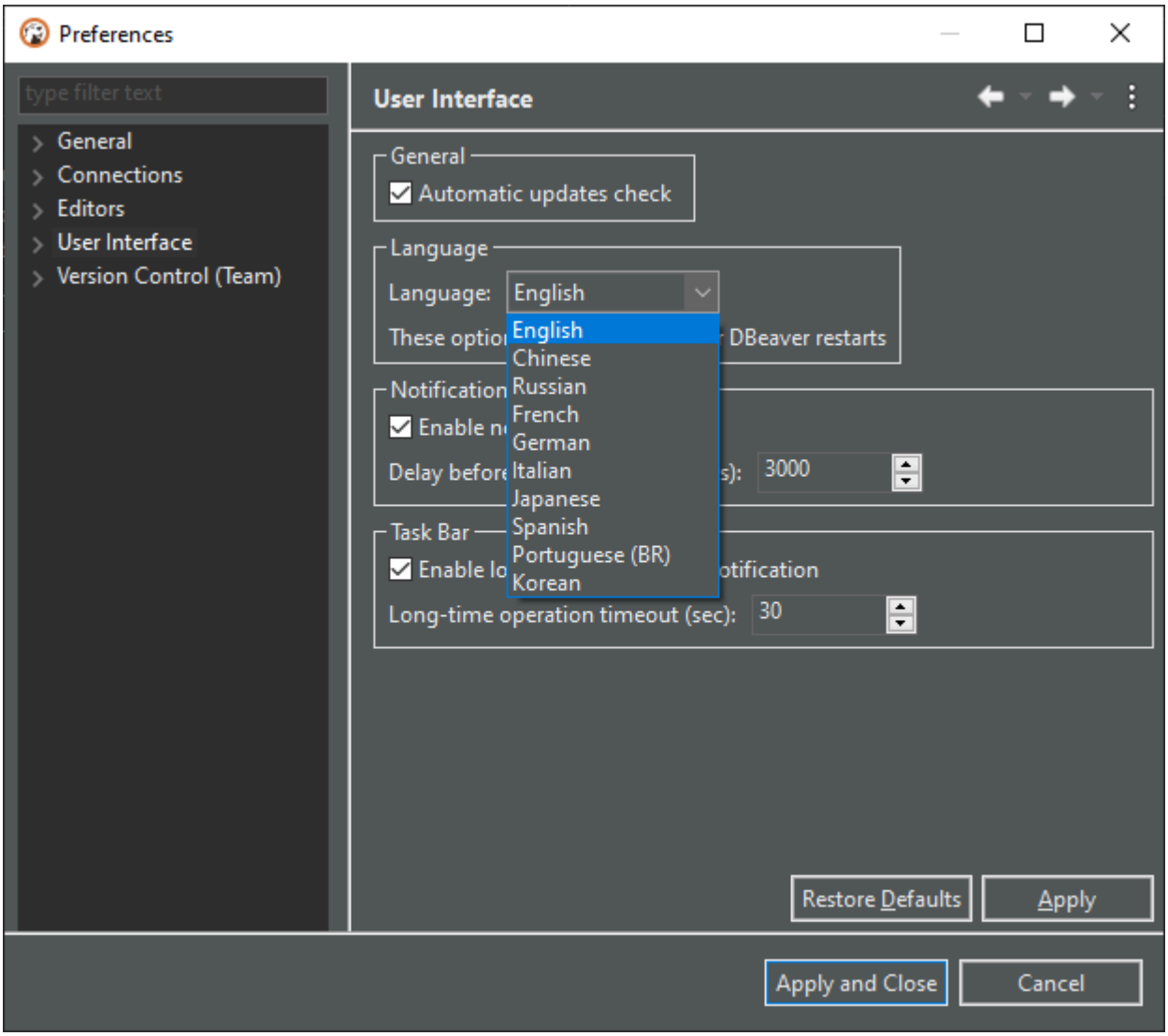
Table statistics are also available only in PRO DBeaver versions. As well as expanded metadata about tables and schemes.



Changing interface language

Changing interface language in Preferences

Go to Preferences->User Interface:



Select your language in the drop-down list and click the "Apply and Close" button.

If DBeaver is installed in a read-only directory, the automatic language change is not possible. In this case, try to edit the configuration file (see below).

Changing interface language in configuration file

Locate the `dbeaver.ini` file. It is in the same directory where DBeaver is installed.

Open `dbeaver.ini` in a text editor and add the following lines before the line `-vmargs`

```
-nl
XX
```

where XX is two-letter language code:

Language	Code
English	en
Chinese	zh
French	fr
Italian	it
Japanese	jp
German	de
Korean	ko
Portuguese (BR)	pt_BR
Russian	ru
Spanish	es

Installing extensions - Themes, version control, etc

You can install a lot of optional extensions (plugins) in DBeaver. Most of the extensions can be found on the Eclipse Marketplace website.

DBeaver-specific extensions

- Office formats support (XLSX)
- Vector graphics support (SVG)
- SSHJ and advanced cryptography (since version 21 it is included in the base distribution)
- Git support - Git version control integration
- SQL debugger

Popular 3rd party extensions for Eclipse and DBeaver

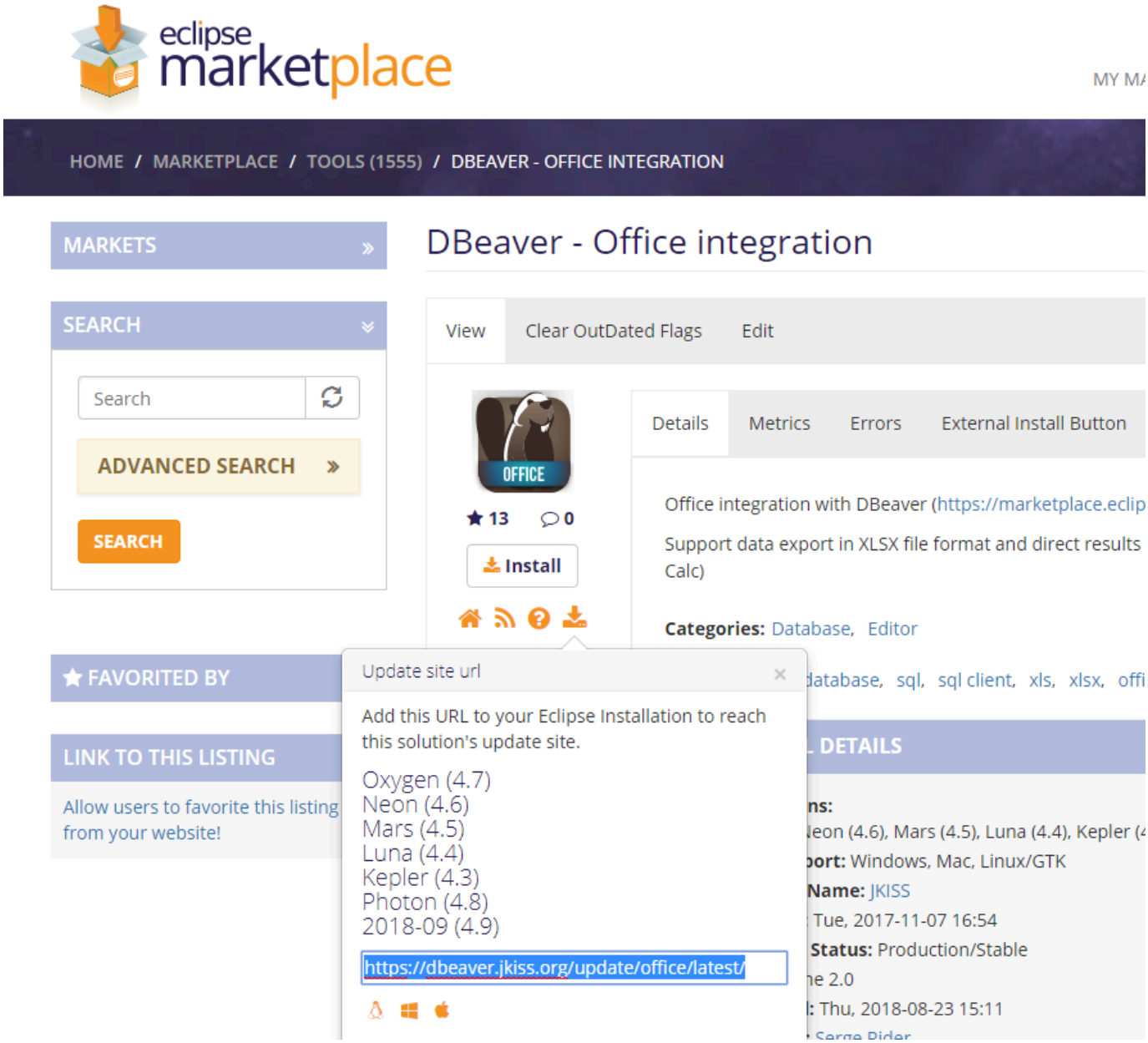
- Darkest Dark theme - the best Dark theme for DBeaver
- Eclipse Color Theme - if for some reason you do not like the Darkest Dark theme, you can use this one
- Subversion support - Subversion integration
- Embedded Shell - Allows you to run shell commands directly from DBeaver
- Editor vertical indents - Adds vertical indents to all text editors
- CodeTogether - Allows you to share the IDE and collaborate

Install Process

In DBeaver EE you can use drag-n-drop from the Marketplace web site (see button `Install`) in the DBeaver main window. This will launch the Marketplace installation wizard automatically. In the DBeaver Community or other DBeaver-based products which do not include marketplace clients, you can use the following instructions:

Extension installation in CE version:

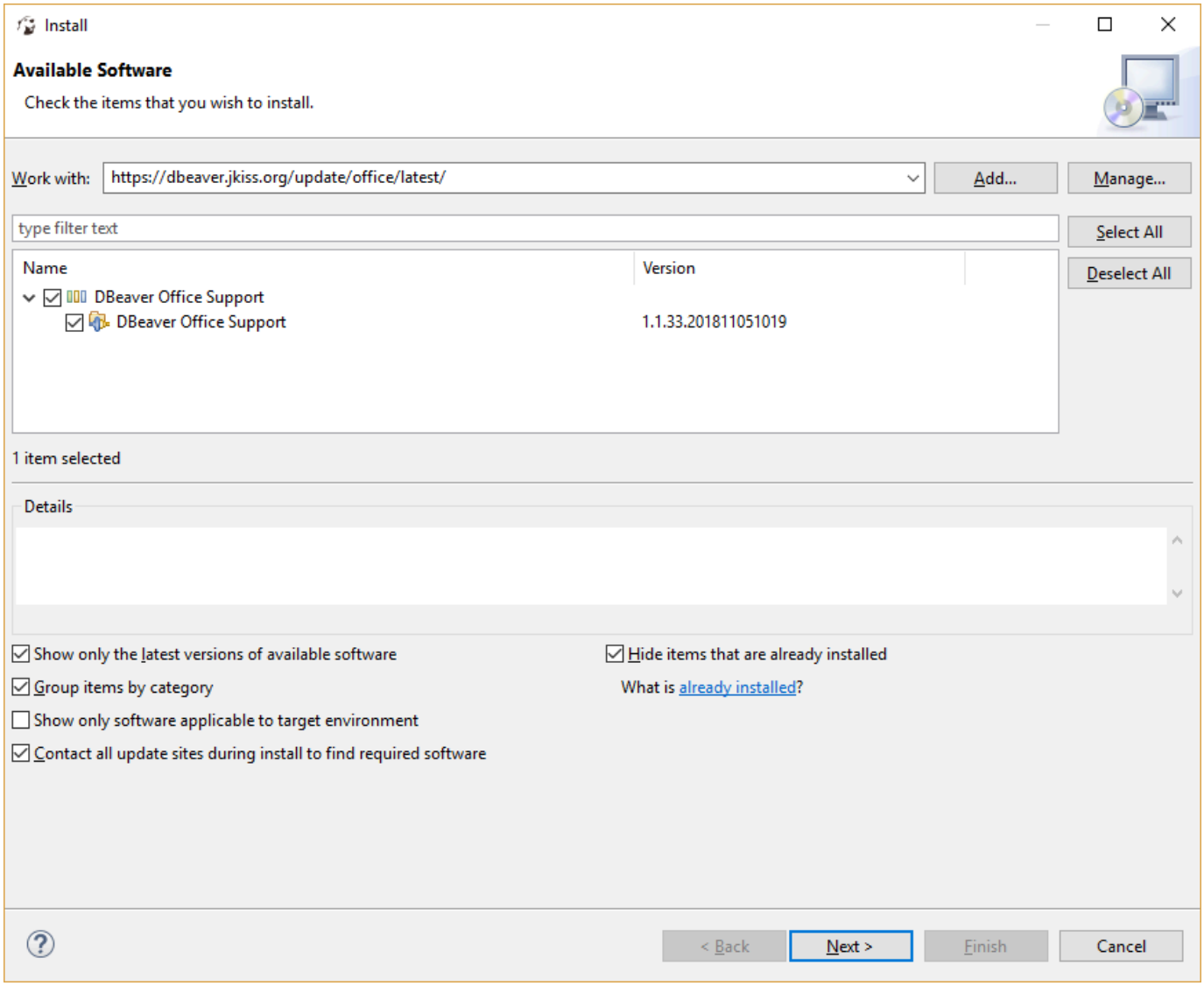
1. Copy URL of extension update site:



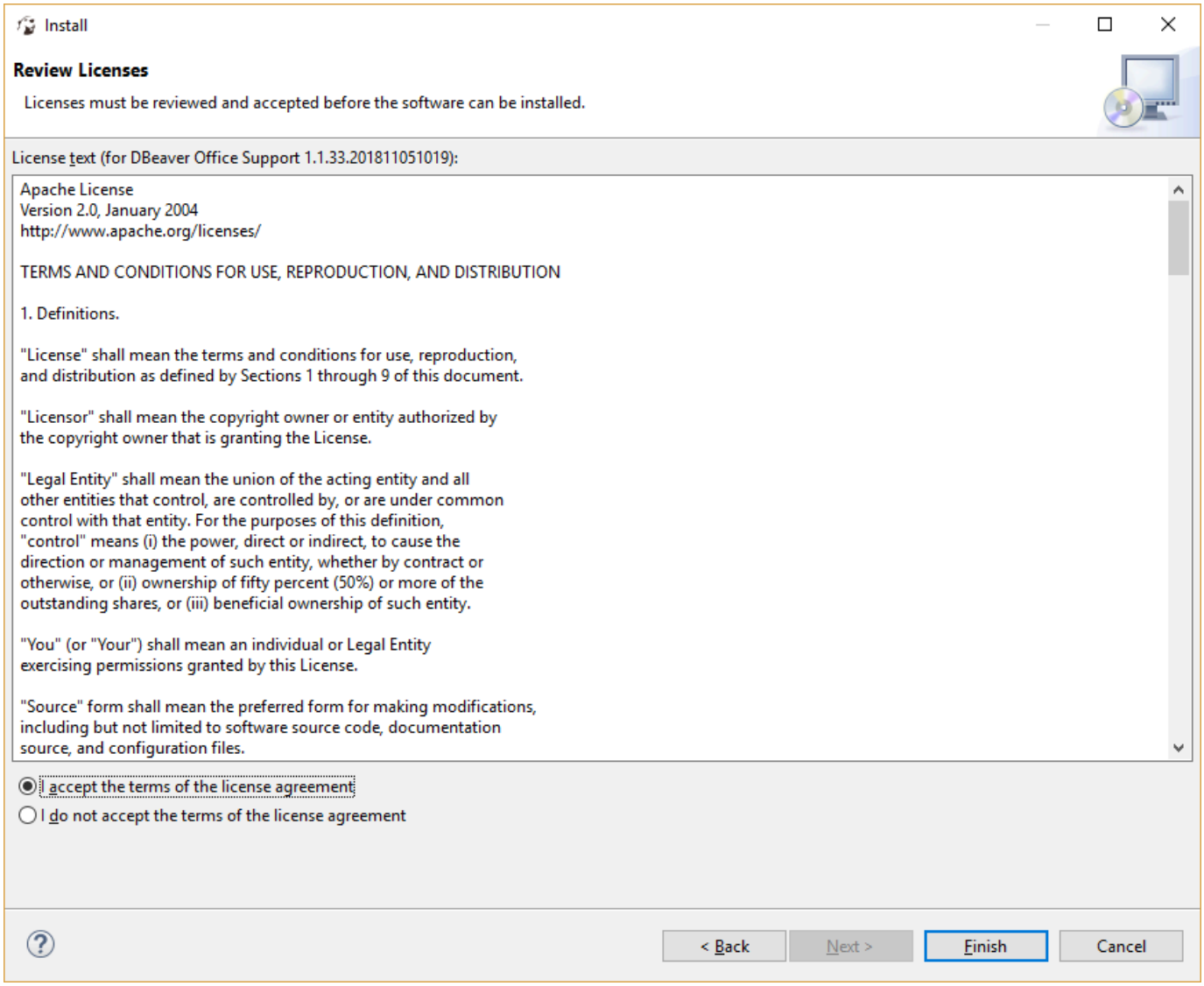
2. In the DBeaver main menu open `Help -> Install New Software`

3. Paste update site URL into `Work with` field and press `Enter`

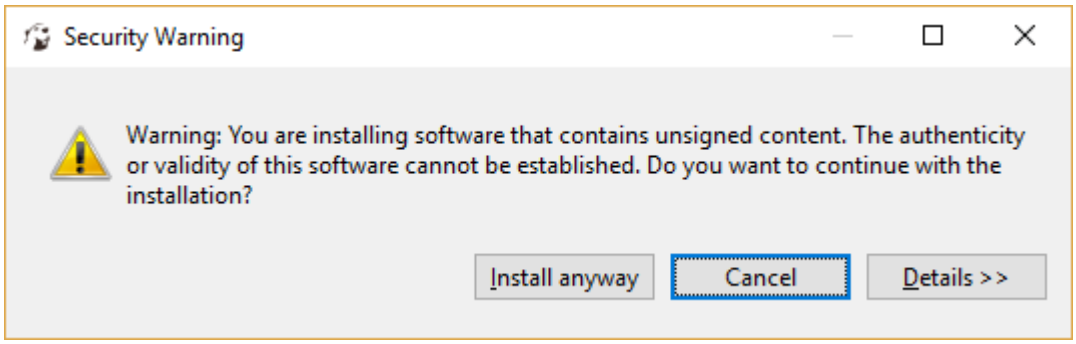
4. Check items you wish to install (in most cases just all items)



5. Click Next. You may need to accept the extension license before installing



6. Some extensions may contain unsigned bundles. Only install such extensions if you really trust the author.



7. Click Next->Finish. The installation will take some time. Restart DBeaver.

User Interface Themes

The interface themes governs the look and feel of the user interface's windows, dialogs, buttons, and all other visual components. By default, DBeaver is configured to use the **Light** theme.

Table of contents

- Change the DBeaver theme
- Managing the High Contrast Theme in Windows
  - Activation of High Contrast theme
  - Deactivation of High Contrast theme
  - High Contrast theme switching limitations

Change the DBeaver theme

1) Go to **Window** -> **Preferences** -> **User Interface** -> **Appearance**.

2) Select from the following available themes:

- **Light**
- **Dark**
- **Classic**

Note: To fully apply the selected theme, you must restart DBeaver.

Managing the High Contrast theme in Windows

Windows users have the additional option of using **High Contrast** themes. This section provides details on how DBeaver interacts with these themes.

Activation of High Contrast theme

When the **High Contrast** theme is activated in Windows, DBeaver automatically switches its theme to **High Contrast**. If DBeaver is already running when the system theme changes, a notification will appear suggesting a restart for the theme to apply correctly.

Deactivation of High Contrast theme

If you deactivate the **High Contrast** theme in Windows, it will automatically be removed from DBeaver's list of available themes after the restart.

Note: After disabling the **High Contrast** theme in Windows, DBeaver will revert to the theme that was previously selected.

High Contrast theme switching limitations

The theme switcher in DBeaver is disabled when the **High Contrast** theme is active in Windows. You cannot switch to a **non-High Contrast** theme in DBeaver unless you disable the **High Contrast** theme in Windows.

Command Line

Command line parameters can be passed directly to the DBeaver executable. The way to do this depends on your operating system:

Windows:

You can use the `dbeaver-cli.exe [parameters]` executable. This executable does not spawn a new window, so you can see the output messages.

Mac:

Parameters can be passed to DBeaver in one of two ways:

- Use the open command followed by the `-a` flag and the name of the application, along with any necessary arguments. Depending on the version of DBeaver, you might need to specify `DBeaver.app`, `DBeaverLite.app`, `DBeaverEE.app`, `DBeaverUltimate.app`, or `DBeaverTeam.app`. For example:
  `open -a "DBeaver.app" --args [parameters]`.
  Note: This method does not redirect logging messages, stdout, and stderr to the terminal.
- Pass parameters directly to the DBeaver executable in the terminal using the path where the `.app` file is located. For example: `/Application/DBeaver.app/Content/MacOS/dbeaver [parameters]`. This method redirects logging messages, stdout, and stderr to the terminal.

For both Windows and Mac, parameters can also be added in the `dbeaver.ini` configuration file. These should be written at the beginning of the file, with each parameter on its own line.

Command line parameters

DBeaver control

Name	Value	Example
-help	Prints help message	
-stop	Quits DBeaver	
-dump	Prints DBeaver thread dump	
-f	Opens the file in DBeaver UI, if the command has -con argument, connects it to datasource	`-f c:\some-path\some-file.sql`
-con	Opens database connection in DBeaver UI	See connection parameters table
-closeTabs	Closes all open editor tabs	
-disconnectAll	Closes all open connections	
-reuseWorkspace	Forces reuse of single workspace by multiple DBeaver instances	
-newInstance	Forces new DBeaver instance creation (do not try to reuse already running one)	
-bringToFront	Brings the DBeaver window on top of other applications	
-var ★	Customs variables for runTask. You can change existing variables in the task. You cannot add new task variables with this parameter. You can add several parameters at once to the command line, each starting with "-var". Used right before -runTask. Template: `-var variableName=variableValue`	`-var film=sakila.film` `-var actor=sakila.actor` `-runTask "exportFromSakila"` PRO versions only.
-vars	Path to a property file with variables	`-vars c:\path\to\file.properties` For more information see the main article
-runTask ★	Executes specified task	`-runTask "@projectName:taskName"` . PRO versions only. See task scheduler.
-license ★	Path to the license file	`-license "/etc/licenses/dbeaver.txt"` . PRO versions only.

System parameters

Name	Value	Example
-nl	Locale	en_US
-data	Workspace path	c:\ProgramData\MyWorkspace
-nosplash	Omits splash screen	true
-clean	Clears all Eclipse caches. Use it if DBeaver fails to start after it upgrades.	
-vmargs	VM parameters	See VM arguments table

VM arguments

You can pass any advanced Java parameters supported by your local JVM. Parameters supported by HotSpot JVM (17): `https://docs.oracle.com/en/java/javase/17/docs/specs/man/java.html`

Parameters supported by all JVMs:

Name	Value	Example
-Xms	Sets initial memory available for DBeaver	`-Xmx1000m`
-Xmx	Sets maximum memory available for DBeaver	`-Xmx4000m`

Connection parameters

All connection parameters must be supplied as a single command line argument. The parameters are divided by pipe (`|`). The parameter name and value is divided by `=` .

Example:

```
-con "driver=sqlite|database=C:\db\SQLite\Chinook.db|name=SQLiteChin|openConsole=true|folder=SQLite"
```

Name	Description	Example
name	Connection name	`Test connection`

driver	Driver name or ID	`driver=sqlite` , `driver=mysql` , etc
url	Connection URL. Optional (JDBC URL may be constructed by a driver from other parameters)	`url=jdbc:sqlite:C:\db\SQLite\Chinook.db`
host	Database host name (optional)	`host=localhost`
port	Database port number (optional)	`port=1534`
server	Database server name (optional)	`server=myserver`
database	Database name or path (optional)	`database=db-name`
user	User name (optional)	`user=root`
password	User password (optional)	`password=mysecret`
auth	Authentication model ID. See Auth models	`auth=postgres_pgpass`
authProp. propName	Custom authentication parameters (depends on the driver and auth model)	`authProp.oracle.net.wallet_location=C:/temp/ora-wallet`
savePassword	Does not ask user for a password on connection	`savePassword=true`
showSystemObjects	Shows/Hides system schemas, tables ,etc	`showSystemObjects=true`
showUtilityObjects	Shows/Hides utility schemas, tables ,etc	`showUtilityObjects=true`
folder	Puts a new connection in a folder	`folder=FolderName`
autoCommit	Sets connection auto commit flag (default value depends on driver)	`autoCommit=true`
	Advanced connection	

prop.propName	parameters (depend on driver)	`prop.connectTimeout=30`
id	Connection id	`oracle_thin-16a88e815bd-70598e648cedd28c` (useful in conjunction with `create=false`)
connect	Connects to this database	`connect=false`
openConsole	Opens the SQL console for this database (sets `connect` to true)	`openConsole=true`
create	Creates new connection	`create=false` (true by default). If it is set as false, then an existing connection configuration will be used. The name or id parameter must be specified.
save	Saves new connection	When `create=true`, then `save=false` (default) makes new connection temporary, `save=true` means that new connection will be saved and accessible between DBeaver launches.

Declare external variables in a file

See the main article

Reset UI settings

Reset settings

You might want to reset settings in the following cases:

- Shortcuts suddenly stop working
- Theme colors are messed up
- Broken or invalid localization
- Other UI elements aren't shown anymore

This can happen due to the following reasons:

- After multiple version upgrades
- After switching between newer and older versions
- After an incorrect shutdown

You can perform a reset using the `Help ⇒ Reset Settings...` action.

> **Be careful**: this will reset **all** UI settings and other user preferences, including:
>
> - resource cache, imported projects
> - layout of menus, toolbars, windows, editors
> - theme, colors, and fonts
> - other settings from installed third-party plugins

After choosing the appropriate options, DBeaver will restart and greet you with a fresh workspace.

Manual reset

If you cannot launch DBeaver, you can reset the workspace manually.

> **Be careful**: only delete any other files if you know what you're doing, and follow the steps below.

To do so, do the following:

1. - On **Windows**, open Explorer and navigate to `%APPDATA%\DBeaverData\workspace6\.metadata\.plugins`
   - On **macOS**, open Finder and navigate to `~/Library/DBeaverData/workspace6/.metadata/.plugins`
   - On **Linux**, open file explorer and navigate to `$XDG_DATA_HOME/DBeaverData/workspace6/.metadata/.plugins`
2. First, you can try deleting the file `workbench.xmi` located in the `org.eclipse.e4.workbench` directory
3. If it didn't help, try deleting all files in the `.plugins` directory or that directory itself

Clear History

You can clean up the workspace by deleting redundant files that accumulate as you use DBeaver.

You do so, use the `Help ⇒ Clear History...` action.

You will need to choose one or more options provided below:

Name	Description
Task run history	Contains information about previously run tasks along with logs collected during their execution
Query log history ★	Contains information about previously run (user and meta) SQL queries

When you're happy with the options, click the `Apply and Restart` button to continue.

Reset workspace

The article has been moved to Reset UI settings.

Linux

GTK

DBeaver is an Eclipse RCP application. Therefore, it may have issues with GTK. Here are known workarounds regarding various issues which may arise.

Fixing screen flickering.

- Add `export GTK_IM_MODULE=ibus` to `~/.profile`.

Parts of DBeaver are white/dark on dark/white theme

To fix the issue, we recommended having a graphical theme which is similar to system one in terms of the color palette.

No more handles because there is no underlying browser available

You need to install `libwebkit2gtk` library to your system, this will solve the issue and allow you to use DBeaver browser functionality

`GTK-WARNING xxx:Theme parsing error`

This issue requires you to change the GTK-program-style `system settings-appearance-program style` (the setting location may vary for different systems).

`gtk_box_gadget_distribute: assertion size 'size >=0 0' failed in GtkScrollbar`

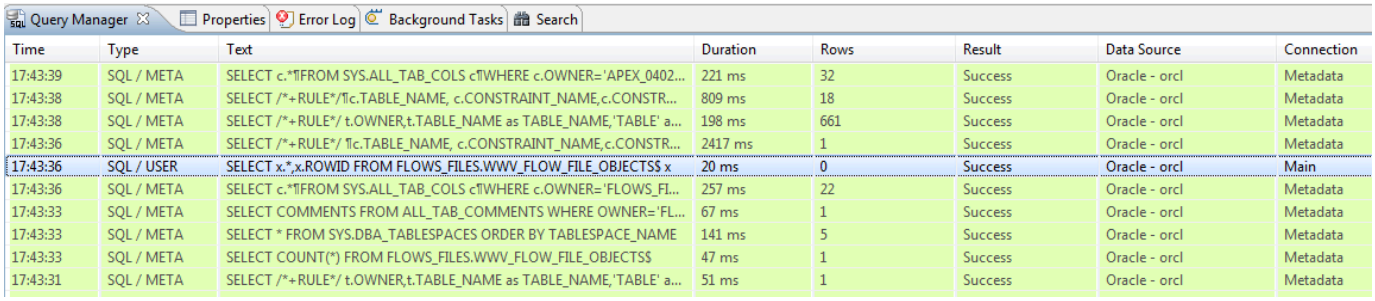
To fix this issue, you need to remove the overlay scrollbars in `~/.config/gtk-3.0/settings.ini`.

```
[Settings]
gtk-overlay-scrolling = false
```

Posting issues

A few tips.

- Check existing issues for your issue (including closed ones). Duplicating an issue is slower for both parties so search through the open and closed issues to see if what you are running into has already been addressed.
- Be clear about what your problem is: what was the expected outcome, what happened instead? Detail how someone else can recreate the problem.
- If you posting a bug report check "Error Log" view. If there are any errors related to your bug then post a complete stacktrace. Sometimes there are no errors in Error Log - if so, try to find them in log files.
- If your issue is related to database data or metadata management - check the Query Manager view. It contains information about all queries DBeaver executes (explicitly or implicitly). To see more detailed information you can configure Query Manager in Preferences.



- Depending on the nature of your bug report provide information about:
  - Operating system
  - Window manager (for Linux)
  - Database (name and version)
  - Database driver (name and version)
- Do not write issue type in the issue title (like Feature Request:, Bug: etc). We will review your issue and assign a corresponding label.

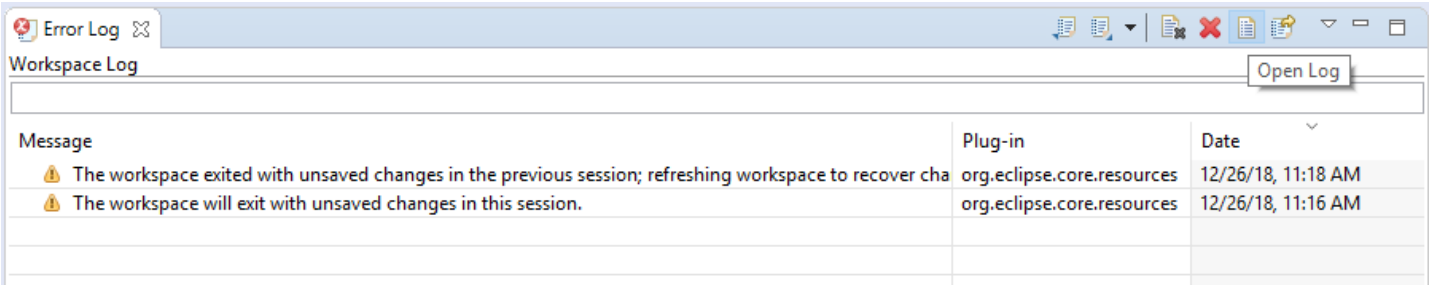
Log files

Error Log view

There is an Error Log view (main menu Window->Show View->Error Log) which contains all errors which occur during the DBeaver runtime.

You can double click on the warning/error in the log viewer and see the error stacktrace. Please attach it to the bug report.

Also, you can open the full log (all error messages) if you need:



Log files

DBeaver writes different log files. Most of them are Eclipse logs.

Log files usually reside in the workspace/workspace6/.metadata .

- In Windows open Explorer and paste path `%APPDATA%\DBeaverData\workspace6\.metadata`.
- In Linux just type `cd $XDG_DATA_HOME/DBeaverData/workspace6/.metadata`
- In MacOS open path `~/Library/DBeaverData/workspace6/.metadata` in Finder.
    - To view hidden folders press `Cmd+Shift+.` in the folder view.

Two standard log files:

- workspace/workspace6/.metadata/.log - all warnings and errors which happen during normal work
- workspace/workspace6/.metadata/dbeaver-debug.log - the same as `.log` plus debug information

In special cases log files can be written in other directories. A special case is an emergency situation when DBeaver cannot start and there is no workspace. Two typical places to find emergency logs:

- `<install-path>/configuration`
- `${HOME}/.eclipse/org.jkiss.dbeaver.product_<dbeaver-version>`

If you are reporting an error, please attach the applicable part of the log - not the complete file.

Logs are very useful. Many errors cannot be reproduced and fixed without a full error stacktrace (all the details).

Java fatal logs

On the rare occasion that the DBeaver process dies, it does not leave any valuable logs. This is caused by a Java VM crash.

JVM creates a fatal log file for each crash (log gile `hs_err_PID.log`). This log usually resides in the same directory where the DBeaver launcher is (e.g. dbeaver.exe).

But in some cases it is a write-protected directory and the log file will be created in other folder.

Instructions on how to find the Java fatal log file: https://docs.oracle.com/javase/9/troubleshoot/fatal-error-log.htm

Log files for Team Edition

Log files usually reside in the workspace/team-workspace/.metadata .

- In Windows open Explorer and paste path `%APPDATA%\DBeaverData\team-workspace\.metadata`.
- In Linux just type `cd $XDG_DATA_HOME/DBeaverData/team-workspace/.metadata`
- In MacOS open path `~/Library/DBeaverData/team-workspace/.metadata` in Finder.
    - To view hidden folders press `Cmd+Shift+.` in the folder view.

JDBC trace

In some cases, custom JDBC drivers work incorrectly in DBeaver - they show the wrong metadata like table columns, constraints or foreign keys.
It usually happens because the driver is not compliant with the JDBC API specification and DBeaver cannot correctly interpret the metadata provided by the driver.

To understand what is going on inside the driver, you can enable JDBC tracing:

1. Find `dbeaver.ini` file (it is located in the same folder where DBeaver is installed)
2. Add line `-Ddbeaver.jdbc.trace=true` in the end of `dbeaver.ini`
3. Restart DBeaver
4. Connect to your database and browse the metadata in the database navigator/object editors.
5. In DBeaver Workspace go to `.metadata` folder
6. File `jdbc-api-trace.log` contains all JDBC API invocations and all queries with results.

Analyzing contents of `jdbc-api-trace.log` you can understand what is wrong with the metadata. Attach the piece of the trace file in the GitHub ticket if you think that something is wrong on DBeaver's side.

WARNING: disable JDBC tracing in your regular work. Enable it only for debugging. The trace generation decreases application performance and may produce huge log files.

Thread dump

Sometimes (due to some bug) DBeaver UI hangs, freezes, or works incorrectly. It is usually impossible to find the reason for such an issue without a thread dump. A thread dump is information about the internal execution state of the Java program. To get thread dump:

Mac and Linux

Run the following on your terminal:

```
jstack $(pgrep dbeaver) > ~/dbeaver-thread-dump.txt
```

Windows

Open the task manager (`CTRL+SHIFT+ESC`), find DBeaver in the process list, and copy its process ID value. In Windows 8+ you need to switch to the "Details" tab.

Open the command prompt (`Win+R` and type `cmd`, then press `ENTER`), run the following command:

```
jstack PID > thread-dump.txt
```

The produced file will be located in the user's home folder, e.g. `C:\Users\Username`.

Now you can attach thread-dump.txt to the GitHub issue.

Managing connections

This guide describes how to manage/secure the DBeaver database connections. It is designed for System administrators. Regular users should check this guide.

Provide predefined connections

DBeaver keeps connections information in the project folder. By default, all projects reside in the workspace.

The default project folder is workspace\workspace6\General.

DBeaver 6.1.3+

DBeaver keeps information about project connections in the `.dbeaver/data-sources.json` . file.

All secured information (user name, password, secret keys, etc) is stored in the encrypted file, `.dbeaver/credentials-config.json` .

DBeaver can load multiple connection files. Any files in the project folder matching the `.dbeaver/data-sources*.json` pattern will be loaded on the startup. So you can create a file, for example, `.dbeaver/data-sources-2.json` in the project folder and DBeaver will see it.

DBeaver < 6.1.3 (Legacy)

DBeaver keeps information about project connections in the `.dbeaver-data-sources.xml` . file.

DBeaver can load multiple connection files. Any files in the project folder matching the `.dbeaver-data-sources*.xml` pattern will be loaded on the startup. So you can create a file, for example, `.dbeaver-data-sources-2.xml` in the project folder and DBeaver will see it.

Importing connections from CSV/XML

You can import a connection from CSV or XML files.

The CSV file must have a header row (first line of file) with column names (see list of supported columns below).

The XML file should contain a top-level element and a set of nested elements. The connections config must be specified in the attributes of the nested elements. Attribute names are the same as the CSV column names.

Supported names:

Name	Meaning
name	Connection name
url	JDBC URL
host	Database server host name
port	Database server port
database	Database/schema name
user	User name
password	User password

You can only specify the URL or the host/port/etc setting.

User name/password are optional.

Sample CSV

```
name,host,port,server,database,url,user,password,type
Postgre Import XML 1,localhost,5432,,postgres,jdbc:postgresql://localhost:5432/postgres,postgres,postgres,
Postgre Import XML 2,localhost,5432,,postgres2,jdbc:postgresql://localhost:5432/postgres2,postgres2,postgr
```

Sample XML

```
<connections>
        <connection name="Postgre Import XML 1" host="localhost" port="5432" server="" database="postgres"
        <connection name="Postgre Import XML 2" host="localhost" port="5432" server="" database="postgres"
</connections>
```

Secure connections from editing

It is possible to set the connection settings as read-only (protected by password)

- Generate MD5 hash of your password. You can do it from the command line using Linux utility md5sum (`md5sum <<<"your password"`) or you can do it online - just google "MD5 hash online".
- Add field `lockPassword` in the connection descriptor (in `.dbeaver/data-sources.json` in `connections` element. So it will look like this:

```
"postgres-jdbc-161537836e8-3e0957d039995715": {
   "provider": "postgresql",
   "driver": "postgres-jdbc",
   "name": "PostgreSQL - postgres",
   "save-password": true,
   "show-system-objects": true,
   "read-only": false,
   "folder": "PG",
   "lockPassword": "2ba81a47c5512d9e23c435c1f29373cb"
...
}
```

- If the user will try to change connection settings now, he/she will be asked for a password.

Using environment variables

You can use references on environment variables in most of connection configuration properties.

For example:

```
"postgres-jdbc-161537836e8-3e0957d039995715": {
   "provider": "postgresql",
   "driver": "postgres-jdbc",
   "name": "PostgreSQL - postgres",
   "user": "${dbeaver.default-user}",
...
}
```

Managing variables

Variables

Disclaimer: this article does NOT **cover variables used in SQL editor.**

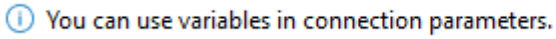
What are variables

A variable is a special template which is replaced with an associated value. Variables help keep your configuration clean and tidy by avoiding unnecessary repetitions: instead of manually replacing each occurrence of some value, you can replace the value of a variable just once and its occurrences will be kept intact.

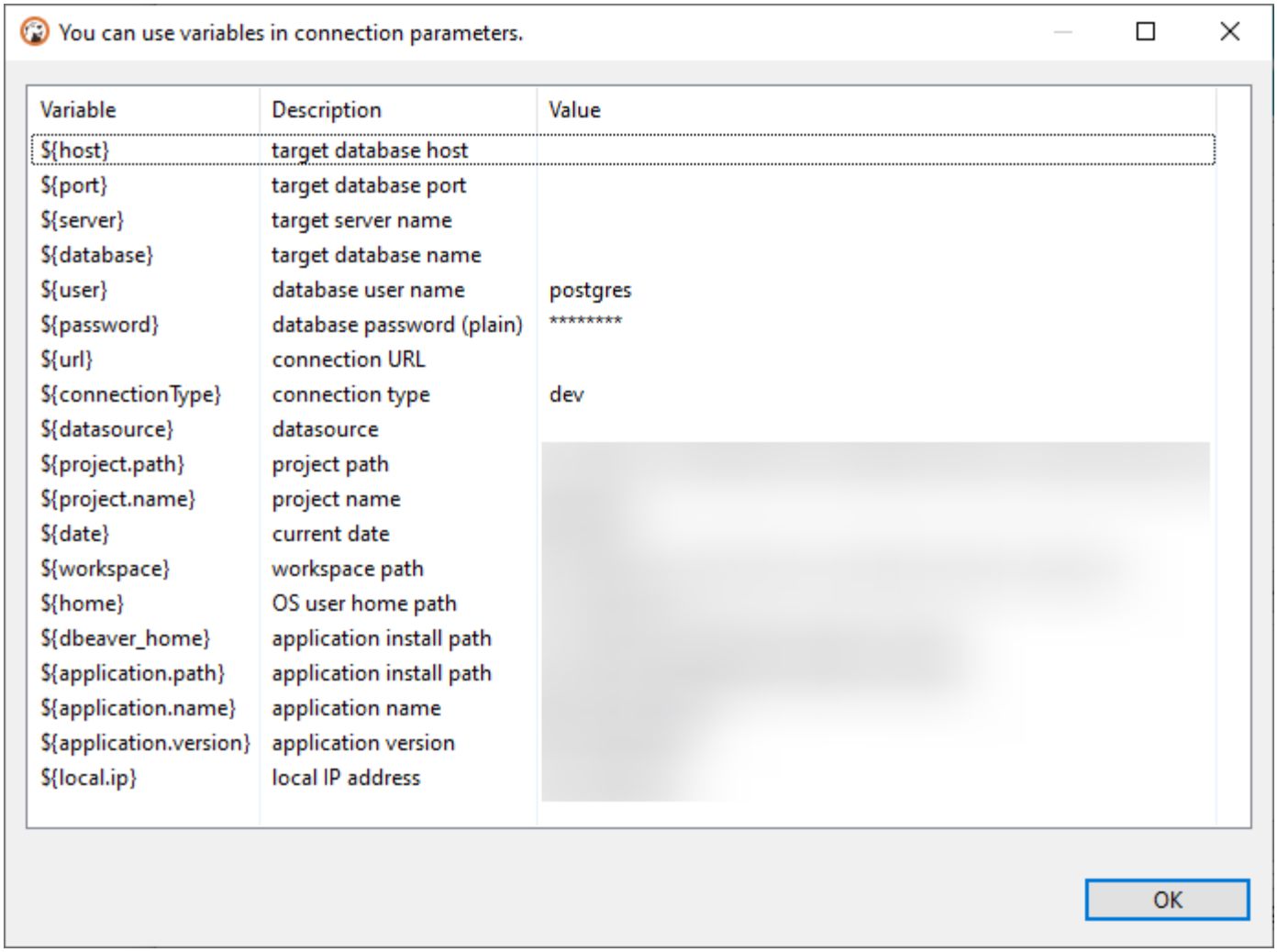
Variables are available in a lot of places. You might find them in:

1. The Connection settings
2. The SSH tunnel settings
3. Data Transfer and in other tasks
4. The Command Line Interface

Wherever you find variables available in the user interface, a hint will usually be present:



By clicking on it, a separate window will open showing all the available variables with their descriptions and values:



Using variables

To use the variables you need to utilize a special syntax:

```
${variable_name}
```

Where `variable_name` is a variable name which you want to use. The window shown above already contains `${` и `}` anchors so you do not need to specify them multiple times.

Variables may be a part of something bigger: e.g., a part of a path. For example, if some file lies under the user's root folder, you may want to use variable `home` : `${home}` : `${home}/path/to/file.txt` .

The Variable resolution

A value of a particular variable may be resolved from different places depending on its origin.

The resolution is performed in the following order (from the highest to the lowest priority):

1. Dialog, where a specific variable is used
2. Datasource-specific variables (`host` , `port` , `server` , `database` , etc.)
3. Application-specific variables (`application.name` , `application.version` , `home` , etc.)
4. Environment variables. For additional information see Wikipedia.
5. External configuration (see the below)

If you have an environment variable called `home` , it will be resolved in the application-specific variable because it has higher priority.

Declare external variables in a file

You can create a file and fill it with pairs of named values and pass it to DBeaver using the `-vars` command-line argument.

Variables from this file can be accessed by other command-line arguments, in the data transfer wizard, and in other places that support variable resolutions.

For example, you may want to put your credentials in that file to avoid showing them to everyone else:

```
# Lines that start with the `#` symbol are comments and therefore ignored.
sampleVar1=abc
someOtherVar=DBeaver is cool
password=P4$$w0r3
```

You can use them as such:

```
dbeaver.exe -vars C:\secrets.properties -con 'driver=<xxx>|url=<xxx>|password=${password}'
```

Here, the `-con` argument has the `${password}` variable that will be replaced with `P4$$w0r3` defined in the example file from above.

Managing drivers

Configure drivers with pre-installed jars

You can customize drivers configuration in the workspace/.metadata/.plugins/org.jkiss.dbeaver.core/drivers.xml file. If you have some pre-installed jar files you can reference them in drivers.xml. Example:

```
<library type="jar" path="absolute-jar-folder-path\driver-jar.jar" custom="true" />
```

Also in drivers.xml you can use the following variables to specify relative paths:

Variable	Meaning
drivers_home	Standard DBeaver drivers location - ($workspace/drivers by default)
dbeaver_home	DBeaver installation folder
home	User home folder
workspace	DBeaver workspace path

For instance:

```
<library type="jar" path="${workspace}\drivers\my-driver.jar" custom="true" />
```

Full drivers.xml example:

```
<?xml version="1.0" encoding="UTF-8"?>
<drivers>

        <provider id="postgresql">
                <driver id="postgres-jdbc" custom="false" embedded="false" name="PostgreSQL" class="org.po
                        <library type="jar" path="maven:/org.postgresql:postgresql:RELEASE" custom="false"
                                <file id="org.postgresql:postgresql" version="42.2.20" path="${drivers_hom
                        </library>
                        <library type="jar" path="maven:/net.postgis:postgis-jdbc:RELEASE" custom="false"
                                <file id="net.postgis:postgis-jdbc" version="2.5.0" path="${drivers_home}/
                        </library>
                        <library type="jar" path="maven:/net.postgis:postgis-jdbc-jtsparser:RELEASE" custo
                                <file id="net.postgis:postgis-jdbc-jtsparser" version="2.5.0" path="${driv
                        </library>
                </driver>
        </provider>

</drivers>
```

Provide predefined drivers configuration

In some cases you may need to provide a driver's configuration or driver `jar` files for a number of DBeaver installations automatically.

This can be done by adding a special parameter in the `dbeaver.ini` file:

```
-Ddbeaver.drivers.configuration-file=c:\some-path\dbeaver-drivers-config.xml
```

This file has the same structure as `drivers.xml` file (see above) and it will be loaded *before* `drivers.xml`.

You can specify partial driver configuration. For example if you need to configure only the jar path then it may look like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<drivers>
  <provider id="generic">
    <driver id="netezza">
      <library type="lib" path="X:\jdbc-drivers\netezza-jdbc.jar" />
    </driver>
  </provider>
</drivers>
```

Managing preferences

Overview

This guide describes how to set up an application preferences through configuration files without even booting it up.

Locating preference files

Preference files have the extension `.prefs` and can be found in the workspace directory under the following path: `.metadata/.plugins/org.eclipse.core.runtime/.settings`

You might not find a preference file you are interested in, if it has not been created yet. In this case, you will need to create one yourself. See the examples below to figure out which preference file you need.

Importing and Exporting workspace configuration

You can easily import and export your workspace configuration. Follow the steps below to perform these actions:

1. Go to **File** -> **Import** (for importing) or **Export** (for exporting).
2. In the opened window, select **DBeaver**, then choose **Preferences** from the nested menu.
3. In the new window:
    - For importing, navigate to the location where the configuration file is stored, select the file, and click **Finish**.
    - For exporting, choose the destination where you want to save the configuration file, and click **Finish**.
4. If you are importing, DBeaver will prompt you to restart the application to apply the changes. Confirm the restart to complete the import process.

Reading and modifying preference files

Preference files consist of key-value pairs glued together with `=` and separated with a line break:

```
# file org.jkiss.dbeaver.test.prefs
some.key=value1
some.other.key=value2
```

Table of contents:

- Default navigator view for new connections
- Miscellaneous navigator settings
- Default transaction settings
- Metadata settings
- Query manager settings
- Database Navigator settings
- Error Logs settings
- User Interface settings
- SQL Editor settings
- Data Editor settings
- Drivers settings

Preferences

You can place the following preferences in the `org.jkiss.dbeaver.core.prefs` file:

Default navigator view for new connections

Name	Type	Default Value	Allowed Values	Description
navigator.settings.default.preset	string	`simple`	`simple`, `advanced`	Sets the default view mode for a new connections. Do not specify this preference if you want to configure a custom preset.
navigator.settings.default.showSystemObjects	boolean	`false`	`true`, `false`	Controls whether system objects must be shown. Used if preset is not specified.
navigator.settings.default.showUtilityObjects	boolean	`false`	`true`, `false`	Controls whether utility objects must be shown. Used if preset is not specified.
navigator.settings.default.showOnlyEntities	boolean	`false`	`true`, `false`	Controls whether only schemas and tables must be shown. Used if preset is not specified.
navigator.settings.default.mergeEntities	boolean	`false`	`true`, `false`	Controls whether all tables must be shown in a single list (without any catalog or database). Used if preset is not specified.

For more information, please see Simple and Advanced View.

Miscellaneous navigator settings

Name	Type	Default Value	Allowed Values	Description
navigator.settings.default.connectionPattern	string	`host_or_database`	expression with variables	Pattern for new connections title.
navigator.show.folder.placeholders	boolean	`true`	`true`, `false`	Displays placeholders for special folders. Show special folders (e.g. Scripts) even if they were not created yet.
navigator.sort.case.insensitive	boolean	`true`	`true`, `false`	Sorts items in case-insensitive mode.
navigator.sort.forlers.first	boolean	`true`	`true`, `false`	Shows folders first.

Default transaction settings

Name	Type	Default Value	Allowed Values	Description
transaction.smart.commit	boolean	`false`	`true`, `false`	Enables smart commit mode.
transaction.smart.commit.recover	boolean	`true`	`true`, `false`	Returns to auto-commit mode after transaction end.
transaction.auto.close.enabled	boolean	`false`	`true`, `false`	Automatically ends (rollback) transaction after idle period.
transaction.auto.close.ttl	integer	`900`	integer value	Timeout before transaction close.
transaction.show.notifications	boolean	`true`	`true`, `false`	Shows transaction and notification.

Metadata settings

Name	Type	Default Value	Allowed Values	Description
database.meta.separate.connection	string	`DEFAULT`	`ALWAYS`, `NEVER`	Controls whether to open a separate connection for metadata read. Do not specify this preference if you want to use default settings.
database.meta.casesensitive	boolean	`false`	`true`, `false`	Specifies the usage of case-sensitive names in DDL statements.
database.props.expensive	boolean	`true`	`true`, `false`	Enables display of row count for tables.
database.meta.server.side.filters	boolean	`true`	`true`, `false`	Enables filtering on the server side.

Query manager settings

Name	Type	Default Value	Allowed Values	Description
qm.queryTypes	string	`USER`, `USER_FILTERED`, `USER_SCRIPT`	`USER`, `USER_FILTERED`, `USER_SCRIPT`, `UTIL`, `META`, `META_DDL`	View for different query types.
qm.objectTypes	string	`session`, `txn`, `query`	`session`, `txn`, `query`	Displays different object types. `txn` - Transactions.
qm.maxEntries	integer	`200`	integer value	Number of entries displayed per page.
qm.storeLogs	boolean	`false`	`true`, `false`	Enables the saving of logs.
qm.logDirectory	string	workspace directory	string value	Folder location for saving logs.
qm.historyDays	integer	`90`	integer value	Number of days for storing logs.

Database Navigator settings

Name	Type	Default Value	Allowed Values	Description
navigator.show.connection.host	boolean	`true`	`true`, `false`	Displays the conn
navigator.show.objects.description	boolean	`false`	`true`, `false`	Displays the objec
navigator.show.statistics.info	boolean	`true`	`true`, `false`	Displays the statis
navigator.show.node.actions	boolean	`true`	`true`, `false`	Displays the actio

navigator.show.folder.placeholders	boolean	`true`	`true` , `false`	Displays placehol
navigator.sort.forlers.first	boolean	`true`	`true` , `false`	Displays folders fi
navigator.sort.case.insensitive	boolean	`false`	`true` , `false`	Orders elements a
navigator.color.nodes.all	boolean	`false`	`true` , `false`	Sets connection c
navigator.show.objects.tips	boolean	`true`	`true` , `false`	Displays object br
navigator.show.tooltips	boolean	`true`	`true` , `false`	Displays informati
navigator.show.tooltips.file.contents	boolean	`false`	`true` , `false`	Allows to show a
navigator.object.doubleClick	string	`EDIT`	`EXPAND`	The behavior of d
navigator.connection.doubleClick	string	`EXPAND`	`EDIT` , `CONNECT` , `SQL_EDITOR` , `SQL_EDITOR_NEW`	The behavior of d
navigator.object.defaultEditorPage	string	`""`	`default.object.editor` , `org.jkiss.dbeaver.ui.editors.data.DatabaseDataEditor` , `org.jkiss.dbeaver.erd.ui.editor.ERDEditorEmbedded`	Defines which edi Properties Editor (Editor (`org.jkiss.dbeave`), or the ERD Edito `org.jkiss.dbeave`
navigator.expand.on.connect	boolean	`false`	`true` , `false`	Expands navigato
navigator.restore.filters	boolean	`false`	`true` , `false`	Determines wheth upon reopening.

navigator.long.list.fetch.size	integer	`5000`	integer values	Determines nodes
navigator.restore.state.depth	integer	`0`	integer values	Restores the navig depth.

Error Logs settings

Name	Type	Default Value	Allowed Values	Description
logs.debug.enabled	boolean	`true`	`true` , `false`	Enables debug logs.
logs.debug.location	string	`${workspace}/.metadata/dbeaver-debug.log`	string values	Location of the log file.
logs.files.output.maxSize	integer	`10485760`	integer values	Maximum log file size (KB).
logs.files.output.maxCount	integer	`3`	integer values	Maximum backup log files count.

User Interface settings

Name	Type	Default Value	Allowed Values	Description
ui.auto.update.check	boolean	`true`	`true` , `false`	Enables automatic updates check.
platform.language	string	`en`	`en` , `zh_CN` , `ru` , `fr` , `de` , `it` , `ja` , `es` , `pt` , `ko` , `zh_TW` ,	Platform language setting.
java.client.timezone	string	operating system time zone	time zones	Time zone setting.
notifications.				Determines whether notification popups

enabled	boolean	`true`	`true` , `false`	are allowed to appear.
notifications. closeDelay	integer	`3000`	integer values	Defines the timeout (in ms) before a notification will disappear.

SQL Editor settings

Name	Type	Default Value	Allowed Values	Description
database.editor. separate.connection	string	`DEFAULT`	`ALWAYS` , `NEVER`	Opening a separate connection for each editor.
database.editor.connect. on.activate	boolean	`true`	`true` , `false`	Connects on editors activation.
database.editor.connect. on.execute	boolean	`false`	`true` , `false`	Connects on query execution.
SQLEditor. autoSaveOnChange	boolean	`false`	`true` , `false`	Auto-saves after any modifications.
SQLEditor. autoSaveOnClose	boolean	`false`	`true` , `false`	Auto-saves editor on close.
SQLEditor. autoSaveOnExecute	boolean	`false`	`true` , `false`	Saves editor on query execution.
SQLEditor. autoSaveActiveSchema	boolean	`true`	`true` , `false`	Saves/restores active schema.
SQLEditor.resultSet. closeOnError	boolean	`false`	`true` , `false`	Closes results tab on error.
SQLEditor.resultSet. replaceCurrentTab	boolean	`true`	`true` , `false`	Replaces active result tab on single query execution.
SQLEditor.resultSet. orientation	string	`HORIZONTAL`	`VERTICAL`	Result view orientation.
SQLEditor.outputPanel. autoShow	boolean	`true`	`true` , `false`	Opens output viewer on new messages.

SQLEditor.outputPanel.autoShow	integer	`20`	integer values	Sets maximum of result tabs for single query.
SQLEditor.ContentAssistant.auto.activation.enable	boolean	`true`	`true`, `false`	Enables SQL Assistant auto-activation.
SQLEditor.ContentAssistant.activate.hippie	boolean	`false`	`true`, `false`	Uses Hippie Engine for autocompletion.
SQLEditor.ContentAssistant.auto.activation.delay	integer	`0`	integer values	Auto activation delay (ms).
SQLEditor.ContentAssistant.auto.keystrokes.activation	boolean	`true`	`true`, `false`	Activates SQL Assistant on typing.
SQLEditor.ContentAssistant.insert.single.proposal	boolean	`true`	`true`, `false`	Auto inserts proposal in SQL assistant if there is only one option.
SQLEditor.ContentAssistant.autocompletion.tab	boolean	`true`	`true`, `false`	Use Tab for autocompletion in SQL Assistant.
SQLEditor.ContentAssistant.insert.case	integer	`0`	`1`, `2`	SQL Assistant insertion case (1- Upper case, 2 - Lower case).
SQLEditor.ContentAssistant.replace.word	boolean	`false`	`true`, `false`	Sets up whether to replace the current word on accepting autocompletion proposal.
SQLEditor.ContentAssistant.proposals.short.name	boolean	`false`	`true`, `false`	Uses short object names in SQL Assistant.
SQLEditor.ContentAssistant.show.helpTopics	boolean	`false`	`true`, `false`	Shows server help topics in SQL Assistant.

SQLEditor.ContentAssistant.show.values	boolean	`true`	`true`, `false`	Proposes values for the given attribute. Note that it can drastically degrade performance when working with large tables, as it selects distinct values.
sql.proposals.insert.table.alias	string	`PLAIN`	`NONE`, `EXTENDED`	Inserts table aliases (in FROM clauses).
SQLEditor.format.activeQuery	boolean	`true`	`true`, `false`	Enables formatting of the query under cursor only.
script.auto.folders	boolean	`false`	`true`, `false`	Creates script folder for each connection.

Data Editor settings

Name	Type	Default Value	Allowed Values	Description
resultset.maxrows	integer	`200`	integer values	Determines the number of rows fetched from the database in a single batch.
resultset.autofetch.next.segment	boolean	`true`	`true`, `false`	Automatically loads the next data segment when the current one is fully processed or viewed.
resultset.reread.on.scroll	boolean	`true`	`true`, `false`	Refreshes the data when moving to the next page.
resultset.maxrows.sql	boolean	`false`	`true`, `false`	Modifies the source SQL query to scroll through or limit results. This is typically achieved using the LIMIT/OFFSET SQL clause.
resultset.order.mode	string	`SMART`	`CLIENT_SIDE`, `SERVER_SIDE`, `SMART`	Specifies the method of sorting returned query results.
				Retrieves information

resultset.read.metadata	boolean	`true`	`true` , `false`	about the table structure, specifically the unique keys.
resultset.read. references	boolean	`true`	`true` , `false`	Retrieves information about the table structure, specifically the foreign keys.
resultset.cancel. timeout	integer	`5000`	integer values	Defines the duration (in milliseconds) the system waits before cancelling a non-responsive query.
sql.query.filter.force. subselect	boolean	`false`	`true` , `false`	Forces the usage of subqueries when applying filters to the data.
resultset.column.label. ignore	boolean	`false`	`true` , `false`	Prioritizes column names over column labels when referencing data.
resultset.edit.key. use_all_columns	boolean	`false`	`true` , `false`	Forms a virtual composite key using all table columns for data saving. If set to `true` , this will determine the action taken if a key element is missing. If set to `false` , no action will be taken when a key element is missing.
resultset.edit. disableIfKeyMissing	boolean	`false`	`true` , `false`	Prevents data modification when a primary key is absent. This setting defines the action taken when the assumed primary key is missing.
keep.statement.open	boolean	`false`	`true` , `false`	Closes the result set after a selection. To view a cursor column value, the option should be enabled.

resultset.datetime. editor	boolean	`false`	`true` , `false`	Enables the advanced datetime editor in the data editor.
resultset.edit.new.row. after	boolean	`true`	`true` , `false`	Places newly inserted rows immediately following the currently selected row.
resultset.edit. refreshAfterUpdate	boolean	`true`	`true` , `false`	Automatically refreshes the data after making changes.
resultset.filter.use. navigator	boolean	`true`	`true` , `false`	Employs filters in the navigator to refine the displayed data.
resultset.confirm. beforeSave	boolean	`false`	`true` , `false`	Displays a prompt asking for a confirmation before saving changes.
resultset.show. errorDialog	boolean	`false`	`true` , `false`	Presents error messages in a separate dialog box.
resultset.mark.cell.value. occurrences	boolean	`false`	`true` , `false`	Highlights instances of the selected cell's value within the data set.
resultset.image. browser	boolean	`true`	`true` , `false`	Displays images in the web browser.
resultset.behavior. autoSwitchMode	boolean	`false`	`true` , `false`	Allow to switch between **Record mode** and **Grid** **mode** when dealing with single or multiple rows. When set to `true` , the system will automatically switch modes based on the

				number of rows selected (**Record mode** for a single row, **Grid mode** for multiple rows).
resultset.calc. columnWidthByValues	boolean	`false`	`true` , `false`	Displays the description for each column in the header.
resultset.show. columnDescriptio	boolean	`false`	`true` , `false`	Adjusts the width of each column based on the content of its cells.
resultset.show. connectionName	boolean	`false`	`true` , `false`	Displays the name of the connection in the status section.
resultset.transform. complex.type	boolean	`true`	`true` , `false`	Enables the visualization of complex columns (such as objects, structures and arrays).
resultset.show. rightJustifyNumbers	boolean	`true`	`true` , `false`	Alows right-justify numbers.
resultset.show. rightJustifyDateTime	boolean	`true`	`true` , `false`	Aligns date and time values to the right side of their respective cells or columns.
resultset.behavior. autoCompleteProposial	boolean	`true`	`true` , `false`	Activates the auto-completion feature in the data filter text field.
resultset.show. oddRows	boolean	`true`	`true` , `false`	Marks odd and even rows in the grid.
resultset.show. colorizeDataTypes	boolean	`false`	`true` , `false`	Colorized data types in the grid.
resultset.show.attIcons	boolean	`true`	`true` , `false`	Displays icons representing the data types of

				each column in the grid.
resultset.show.attFilters	boolean	`true`	`true` , `false`	Enables the display of attribute filters in the grid.
resultset.show. attrOrder	boolean	`false`	`true` , `false`	Shows asc/desc order icon in column header.
resultset.show. collections.inline	boolean	`false`	`true` , `false`	Enables the inline display of collections within the grid.
resultset.show.boolean. checkbox	boolean	`true`	`true` , `false`	Represents boolean values in the grid as checkboxes.
resultset.show. whitespace.characters	boolean	`false`	`true` , `false`	Displays whitespace characters within the grid.
resultset.grid.value. format	string	`UI`	`EDIT` , `NATIVE` , `UI`	Determines the format in which values are displayed within the grid.
resultset.grid.value. format	integer	`30`	integer values (from `0` to `100`)	Determines the maximum percentage of width that a column can occupy in the grid.
resultset.highlight. selectedRows	boolean	`true`	`true` , `false`	Enables the highlighting of entire rows when one or more cells within those rows are selected.
resultset.hScroll. smooth	boolean	`true`	`true` , `false`	Enables smooth scrolling within the DBeaver.
resultset.show.boolean. toggleOnClick	boolean	`true`	`true` , `false`	Allows the boolean values to be toggled or switched between true and false with a single click.

resultset.behavior.inlineEnter	boolean	`false`	`true` , `false`	Selects the next cell after the inline edit completed (or finished).
resultset.behavior.doubleClick	string	`INLINE_EDITOR`	`NONE` , `EDITOR` , `INLINE_EDITOR` , `COPY_VALUE` , `COPY_PASTE_VALUE`	Defines the action taken when a cell in a grid is double-clicked (Options: EDITOR, INLINE_EDITOR, COPY_VALUE, COPY_PASTE_VALUE, NONE).
resultset.show.row.batch.size	integer	`1`	integer values	Number of consequent rows with the same background color. Default = 1 means odd/even rows coloring.
resultset.text.tab.size	integer	`4`	integer values	Determines tab size in plain text.
resultset.text.max.column.size	integer	`255`	integer values	Determines the maximum length of a column's content when viewing data in plain text.
resultset.text.value.format	string	`EDIT`	`EDIT` , `UI` , `NATIVE`	Determines how plain text values are displayed in DBeaver.
resultset.text.show.nulls	boolean	`false`	`true` , `false`	Shows `NULL` if this setting is enabled or nothing if disabled.
resultset.text.line.number	boolean	`false`	`true` , `false`	Enables the display of line numbers.
resultset.text.delimiter.leading	boolean	`false`	`true` , `false`	Enables trailing delimiter in plain text for Grid Representation.
				Enables trailing delimiter in

resultset.text.delimiter.trailing	boolean	`true`	`true` , `false`	plain text for Grid Representation.
resultset.text.delimiter.top	boolean	`false`	`true` , `false`	Enables top delimiter in plain text for Grid Representation.
resultset.text.delimiter.bottom	boolean	`false`	`true` , `false`	Enables bottom delimiter in plain text for Grid Representation.
resultset.text.extra.spaces	boolean	`false`	`true` , `false`	Enables extra spaces in plain text for Grid Representation.
resultset.string.use.content.editor	boolean	`false`	`true` , `false`	Opens separate string editor instead of dialog.
resultset.binary.representation	string	`string`	`hex` , `base64` , `string`	Determines the format in which binary data is displayed.
resultset.binary.editor	string	`EDITOR`	`EDITOR` , `PANEL`	Enables to change the binary data editor. Choose between `EDITOR` (dedicated binary data editor) or `PANEL` (dialog window for viewing/editing binary data).
resultset.binary.stringMaxLength	integer	`32`	integer values (max: `3200`)	Determines the maximum length of binary strings that can be displayed in the binary editors.
content.memory.maxsize	integer	`10000`	integer values (max: `1000000000`)	Determines the maximum length of LOB (Large Object) data that

				will be kept in memory when working with binary editors.
content.hex.encoding	string	`UTF-8`	All available encodings from the current JVM	Allows to specify the character encoding used when interpreting binary data as strings.
hex.font.name	string	`Courier New`	`Courier New`, `Andale Mono`, `Menio`, `Monaco`, `PT Mono`	Allows to specify the name of the font to be used in the data editor.
hex.font.style	integer	`0`	`0` - Regular, `1` - Bold, `2` - Italic, `3` - Bold Italic	Allows to choose the font style for the data editor.
hex.font.size	integer	`10`	`6` - `14`, `16`, `18`, `22`, `32`, `72`	Allows to choose the font style for the data editor.
default.hex.width	integer	`8`	`4`, `8`, `16`	Allows to choose the default width for the data editor.
resultset.json. maxObjectMembers	integer	`200` (max: `2000000000`)	integer values	Specifies the threshold for collapsing JSON objects in the data editor.
resultset.json. maxArrayElements	integer	`200` (max: `2000000000`)	integer values	Determines the threshold for collapsing arrays in the JSON data editor.
resultset.json. maxElementNesting	integer	`4` (max: `400000000`)	integer values	Determines which objects in JSON will be collapsed (hidden) if their nesting

				level exceeds a specified threshold.
dataformat.type.number.nativeSpecialValues	boolean	`false`	`true` , `false`	Instructs the data editor to utilize the native formatting for special values.
dataformat.type.number.useGrouping	boolean	`false`	`true` , `false`	Indicates whether grouping is used to display numbers.
dataformat.type.number.useTypeScale	boolean	`false`	`true` , `false`	Uses column/attribute data type scale as minimum fraction digits number.
dataformat.type.number.roundingMode	string	`UNNECESSARY`	`UP` , `DOWN` , `CEILING` , `FLOOR` , `HALF_UP` , `HALF_DOWN` , `HALF_EVEN` , `UNNECESSARY`	Determines the rounding mode used for numerical values.
dataformat.type.number.minIntegerDigits	integer	`1`	integer values	Determines the minimum number of digits to display for integer values.
dataformat.type.number.minFractionDigits	integer	`0`	integer values	Determines the minimum number of fractional digits displayed.
dataformat.type.number.maxIntegerDigits	integer	`2147483647`	integer values	Defines the maximum number of digits allowed for an integer value.
dataformat.type.number.maxFractionDigits	integer	`10`	integer values	Specifies the maximum number of fraction digits allowed.

dataformat.type. number.groupingSize	integer	`3`	integer values	Determines the grouping size for displaying numeric values.
dataformat.type. timestamp.pattern	string	`yyyy-MM-dd HH:mm:ss.SSSS`	timestamp format	Determines the format for displaying and editing timestamp data.
dataformat.type. timestamptz.pattern	string	`yyyy-MM-dd HH:mm:ss.SSS Z`	timestamp with timezone format	Determines the format for displaying timestamp values with timezone information.
dataformat.type.date. pattern	string	`yyyy-MM-dd`	date format	Determines the format used for displaying and editing dates.
dataformat.type.time. pattern	string	`HH:mm:ss`	time format	Determines the format used for displaying and editing time.
dataformat.type.timetz. pattern	string	`HH:mm:ss Z`	time with timezone format	Determines the format used for displaying and editing time with timezone.

The preferences of **GIS Viewer** from **Data Editor** you can find in the `org.jkiss.dbeaver.data.gis.view.prefs` file:

Name	Type	Default Value	Allowed Values	Description
gis.view. minZoomLevel	integer	`0`	integer values	Determines the maximum level of zoom.
gis.view. maxObjectsRender	integer	`10000`	integer values	Determines the maximum number of objects that can be displayed.
gis.view.defaultSRID	integer	`4326`	integer values	Determines the default Spatial Reference Identifier (SRID).

Drivers settings

Name	Type	Default Value	Allowed Values	Description
ui.drivers.home	string	`path/to/DBeaverData/drivers`	string values	Specifies the local folder for driver storage.
ui.proxy.host	string	``	string values	Identifies the host for the HTTP proxy when downloading drivers.
ui.proxy.port	integer	`1080`	integer values	Specifies the port number used by the HTTP proxy during driver downloads.
ui.proxy.user	string	``	string values	Determines the username for accessing proxy during driver downloads.
ui.proxy.password	string	``	string values	Sets the password required for the proxy during driver downloads.
ui.drivers.sources	string	`https://dbeaver.io/files/jdbc/`	string values	Provides the URL where driver files can be downloaded.
ui.drivers.global.libraries	string	``	string values	Defines the global classpath from which global libraries can be loaded.
ui.drivers.version.update	boolean	`false`	`true`, `false`	Determines whether the system checks for new driver versions.

Windows Silent Install

It is possible to install DBeaver in silent mode using the Windows Installer command line parameters.

This might be very useful for mass install automation (SSCM and other similar systems).

Installer was improved in DBeaver 5.3.3, special thanks to the https://github.com/Drizin/NsisMultiUser team.

Parameters

Command line parameters supported by DBeaver installer:

Parameter	Description
/S	silent mode, requires /allusers or /currentuser, case-sensitive
/D=path	(installer only) set install directory, must be last parameter, without quotes, case-sensitive
/allusers	(un)install for all users, case-insensitive
/currentuser	(un)install for current user only, case-insensitive
/uninstall	(installer only) run uninstaller, requires /allusers or /currentuser, case-insensitive

In order to install with the `/allusers` parameter the current user must have the administrator's permission.

Installer return codes (decimal):

Code	Meaning
0	normal execution (no error)
1	(un)installation aborted by user (Cancel button)
2	(un)installation aborted by script
666660	invalid command-line parameters
666661	elevation is not allowed by defines
666662	uninstaller detected there is no installed version

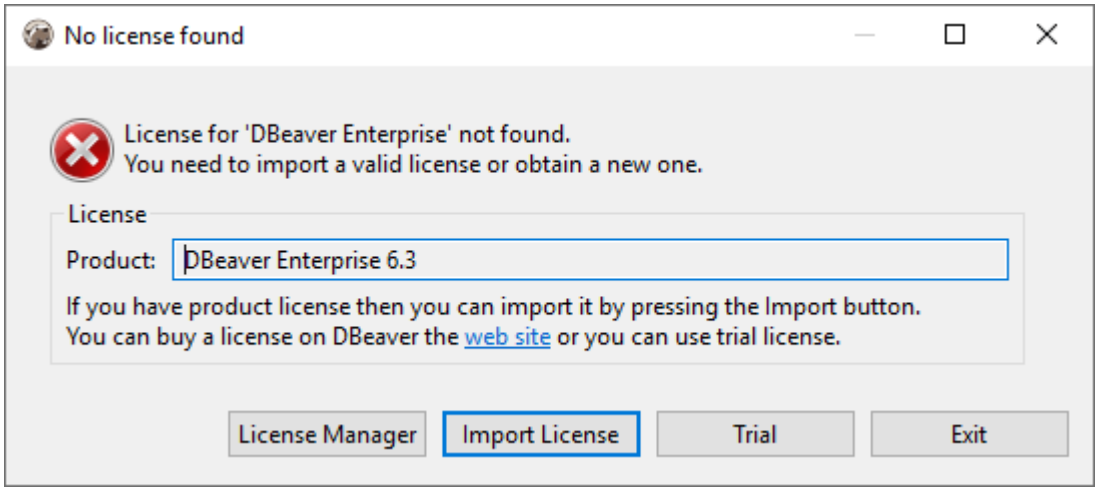
666663	executing uninstaller from the installer failed
666666	cannot start elevated instance
other	Windows error code when trying to start elevated instance"

License Administration

Note: This feature is available in Lite, Enterprise, Ultimate and Team editions only.

Manual license import

Commercial versions of DBeaver ask the user to import the license file if they cannot find it locally. It is the most simple and convenient way to import the product license for individual users.



License management automation

There are several ways to automate the license management process. It makes sense for a multi-user environment.

Put the license file to the predefined locations

1. Install DBeaver
2. Save license text to a file. It should be named as `.dbeaver-%PRODUCT_PREFIX%-license.dat` or `.dbeaver-%PRODUCT_PREFIX%-license.txt`.
3. Add the license file to one of the following locations:
   - Windows
     - `%HOMEPATH%\`
     - `%APPDATA%\DBeaverData\workspace6\.metadata\`

    - `%AppData%/.jkiss-lm`
- MacOS X
    - `~/` (user home)
    - `~/Library/DBeaverData/workspace6/.metadata/`
    - `~/Library/.jkiss-lm`
- Linux
    - `~/` (user home)
    - `$XDG_DATA_HOME/DBeaverData/workspace6/.metadata/`
    - `$XDG_DATA_HOME/.jkiss-lm`

4. Launch DBeaver from Start Menu

Product prefixes:

Product name	Prefix
DBeaver Enterprise	ee
DBeaver Lite	le
DBeaver Ultimate	ue

Passing license file through command line

You can add the command line parameter `license <license-path>` to the DBeaver EE shortcut. Also, you can add this parameter to the `dbeaver.ini` config file.

Command line reference.

How to Import License

Note: This feature is available in Lite, Enterprise, Ultimate and Team editions only.

- Import from email
- Import from the personal account
- Insert the License key into the License Manager
- Import of Subscription license
- Import of License extension
- License Manager

To start using commercial versions of DBeaver you can

- request a Trial license for 2 weeks;
- request an Academic license if you are a student or a teacher;
- buy a Subscription license, Standard DBeaver license or DBeaver license extension.

After purchasing the DBeaver license or getting the Trial/Academic license, you will receive a License text by email. It will also be available in your personal account on our site. This License text will contain your License ID e. g. DB-841MRZHY-ZH54, the start date and license owner's name and company name. It is very important to import your License correctly.

Import from email

You can just copy-paste the License Key to import the license into the License Manager. Please note that you need to copy-paste the full license text (not just the license ID). The license text starts with "–" and ends with "==" characters.

Hello,

Thank you for your interest to DBeaver.

We are glad to see you among DBeaver users. Please, find your license below:

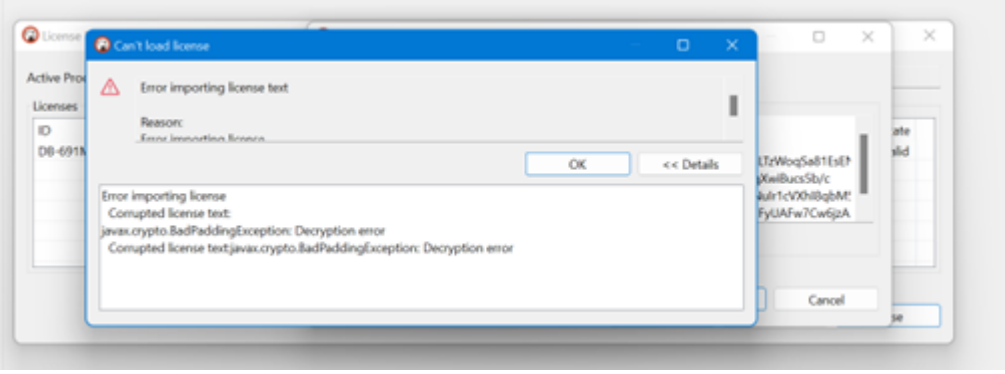
```
-- DBeaver EE LICENSE - DB-841MRZHJ-ZH53
-- Starts at Tue Oct 11 10:14:33 UTC 2022 to Louis Anderson //
PK2uj9eCXU4Zmgd1wvMufQyC2Jk+cSTtePOXewaUoS2/jYvuaX1opFJ0G/H81sLTzWoqSa81EsEN
BG3Z6TJ0VFPko5bvxkm2r0hRXjRusal8gM47snPK/NaN/7fg/CEzHhy3+NrstrtRqXwiBucs5b/c
lRgIZm3Z4B5qg8G1lV8K2gG99p8K0e690gTB6vWY1hECRfs3vdrgz4x5QHnQrNuIr1cVXhI8qbM5
l3L9/7e3QgBMQya+XF9yVeGiVRQGdQDP8GNwY1WMoHBzpGKuhmltwBj+jF7FyUAFw7Cw6jzAk0es
gb3Fg2REPkLwvIq2yxangUpnNGDUE1bH7VXoSw==
```

Enjoy your database research with DBeaver and good luck!

Regards,

DBeaver team

Sometimes an email client can corrupt the formatting of the License Key that can cause an error.

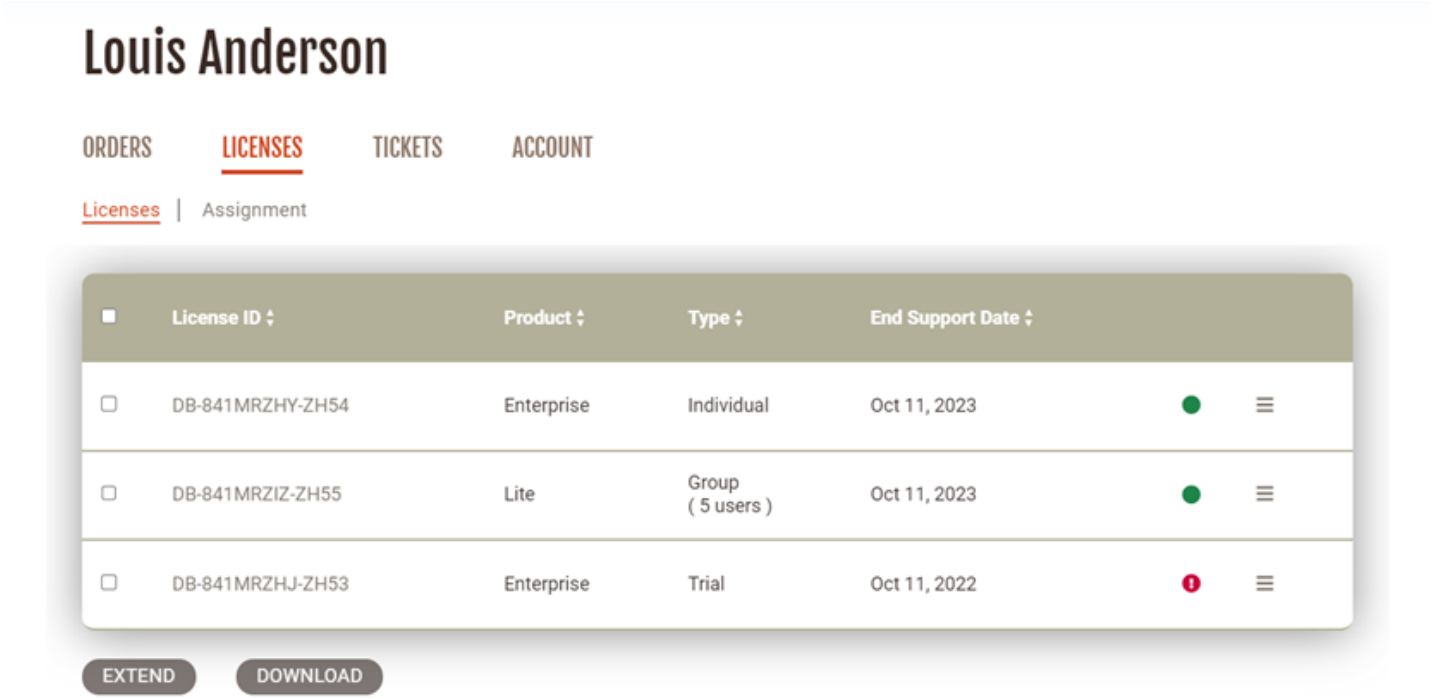


Therefore, you need to import your License Key from your personal account on our site https://dbeaver.com/.

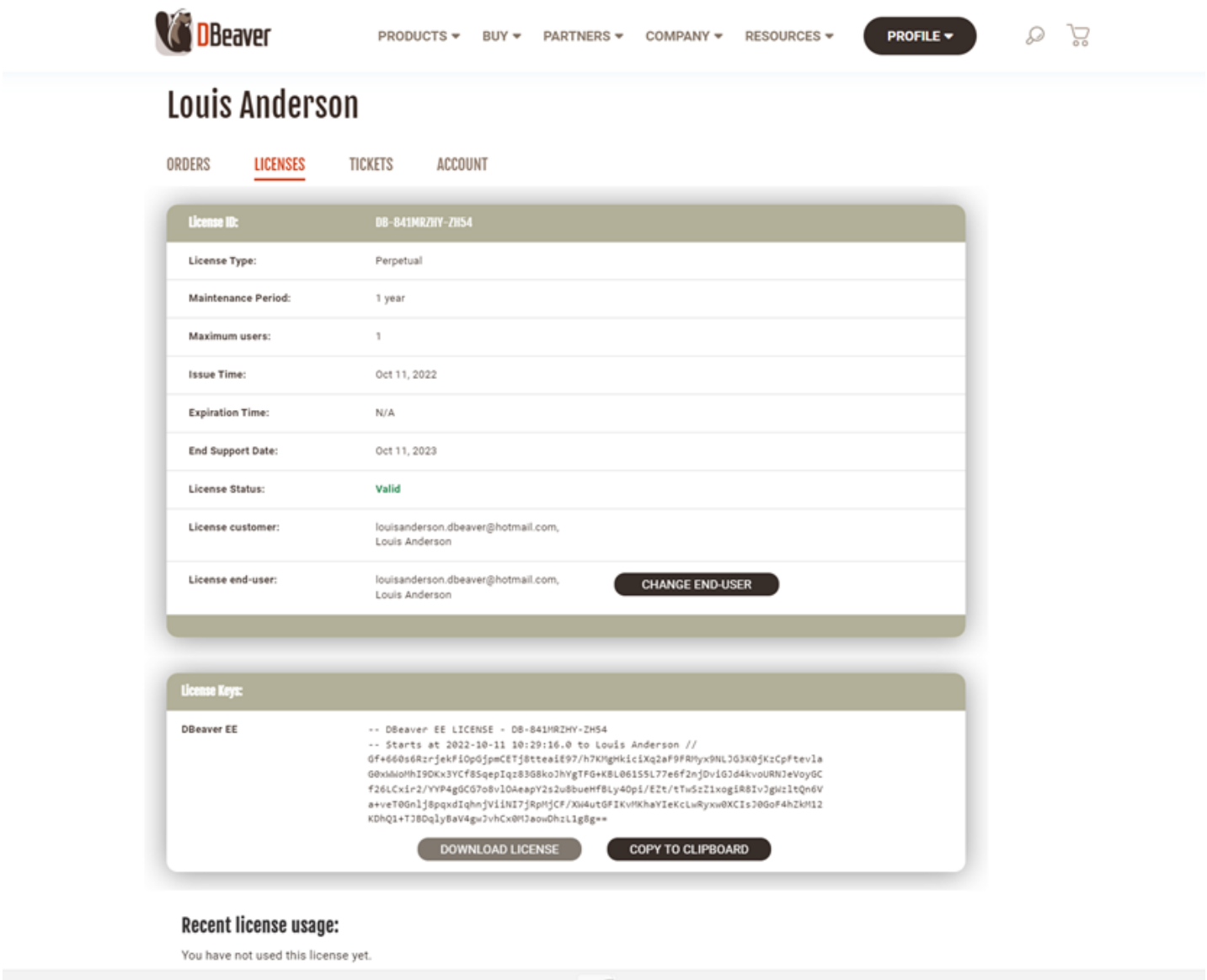
Import from the personal account

Firstly, you need to Sign in.

Secondly, you should open the **Licenses** tab, where you can find all your licenses.

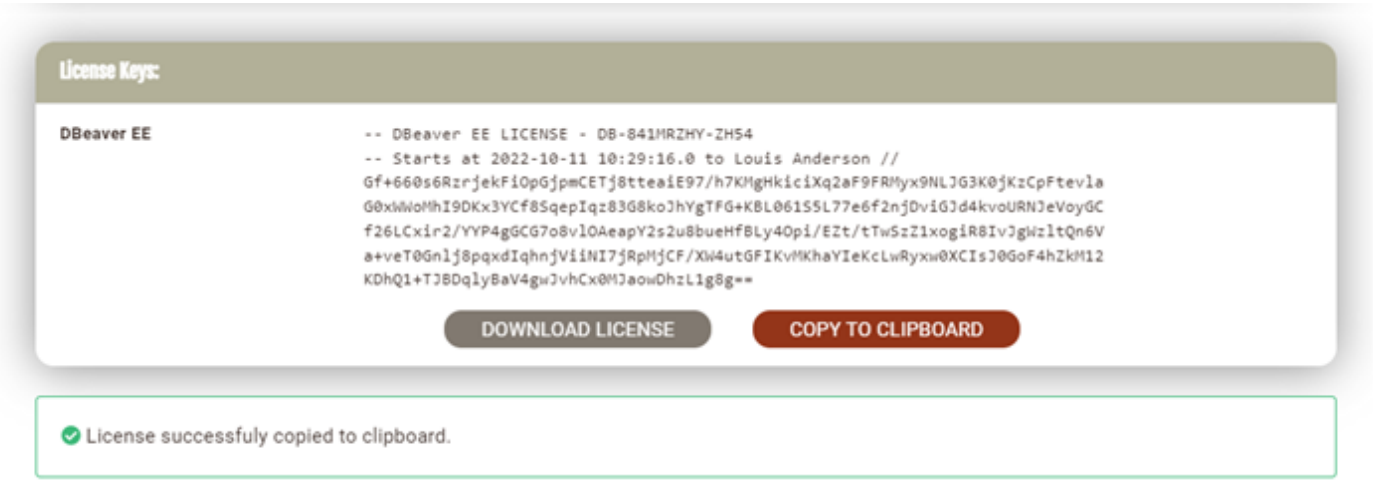


To open License details and copy the license key text click the license ID link. Here you can find your license status, type, maintenance period, and end support date.



At the bottom of the page you can find the License Key required to start using DBeaver. There are two options how to copy your License Key from the personal account:

1)Press the **COPY TO CLIPBOARD** button, then press OK. The license text will be copied to the clipboard.



2)Press the **DOWNLOAD LICENSE** button, then press OK.

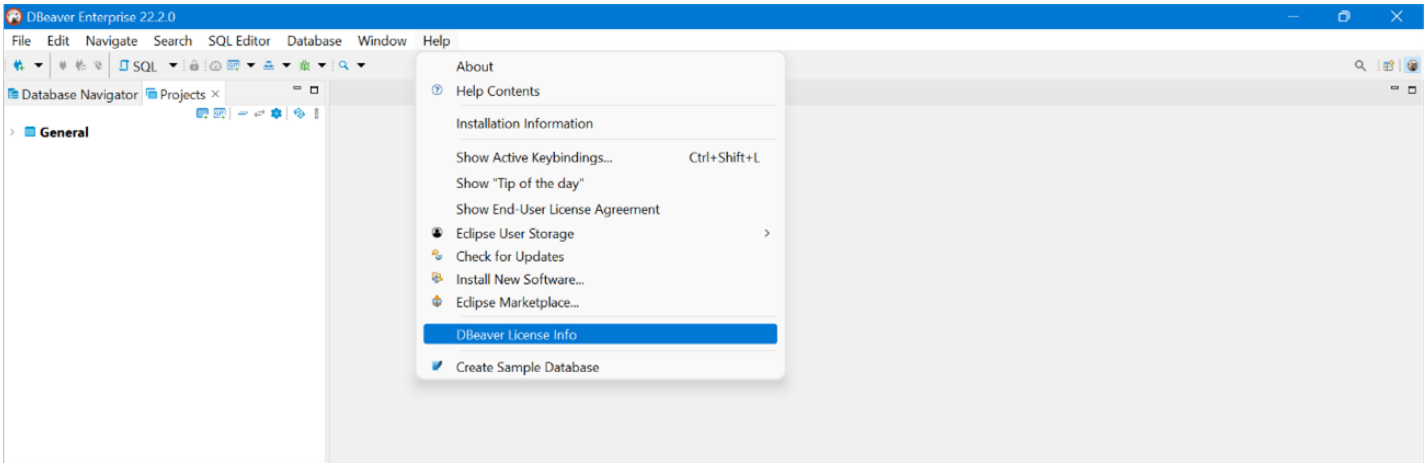
.txt file with your License Key will be downloaded to your download folder. The file name is License ID, e. g. DB-841MRZHY-ZH54.



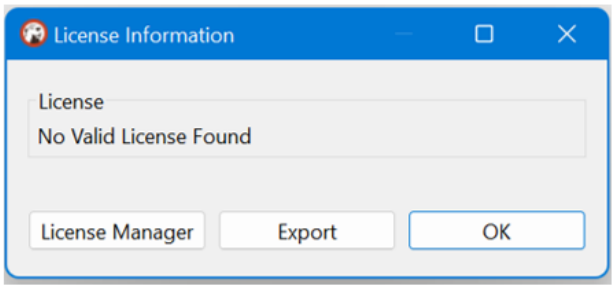
Then you need to insert the copied License Key to License Manager in DBeaver.

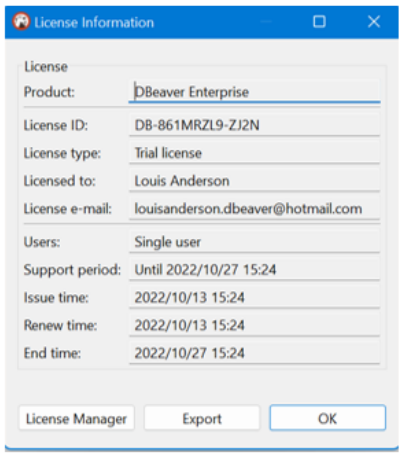
Insert the License key into the License Manager

To start using commercial versions of DBeaver with your License Key you need to open License Manager in DBeaver: Help -> DBeaver License Info

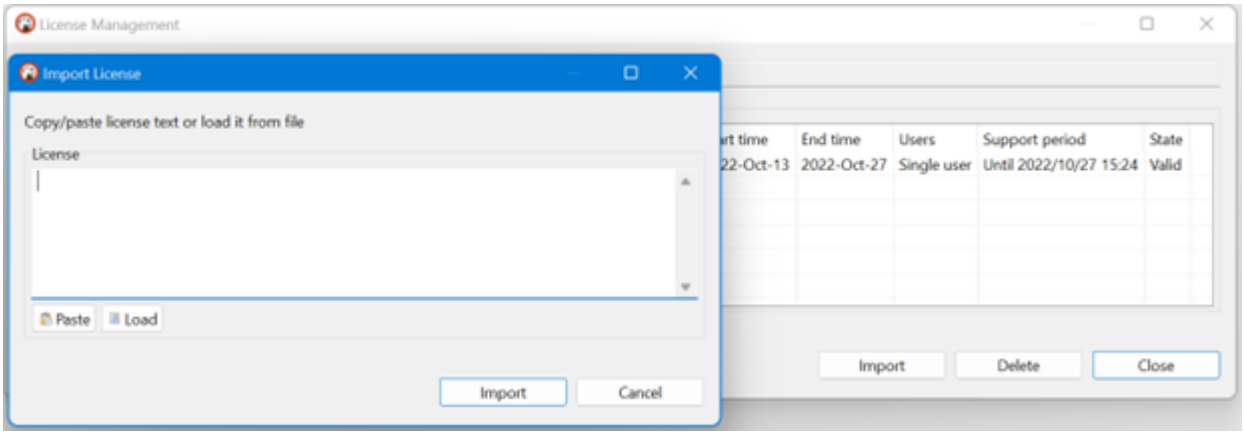


The License information window can look different depending on whether you already have a valid license or not.

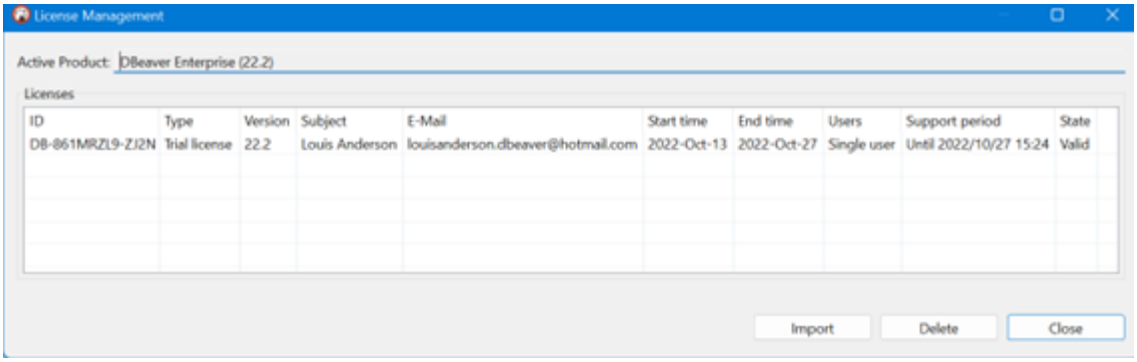




Then you open the License Manager and press the Import button to paste your License Key.



If you copied the License Key to the clipboard, press the Paste button and then Import. If you downloaded a .txt file with the License Key, press the Load button and then select the file from Downloads. The License Key will be pasted automatically. Then press the Import button and your license will be added to the License Manager. You have successfully imported your license.



You have successfully imported your license. Now you can close the License Manager and start using DBeaver.

Import of Subscription license

A subscription license requires internet access on the workstation for the first activation and each prolongation.

If you do not have an active internet connection or work behind a corporate firewall while importing the Subscription license, the following error can appear:

Invalid subscription

Can`t find the subscription information for license 'DB-841MRZHY-ZH54'.

Check your internet connection and/or firewall settings and restart application.

In this case you need to check that DBeaver has internet access or you will need to configure your firewall.

License extension

The standard DBeaver license is a perpetual license with a limited period of support (1 year or 2 years).

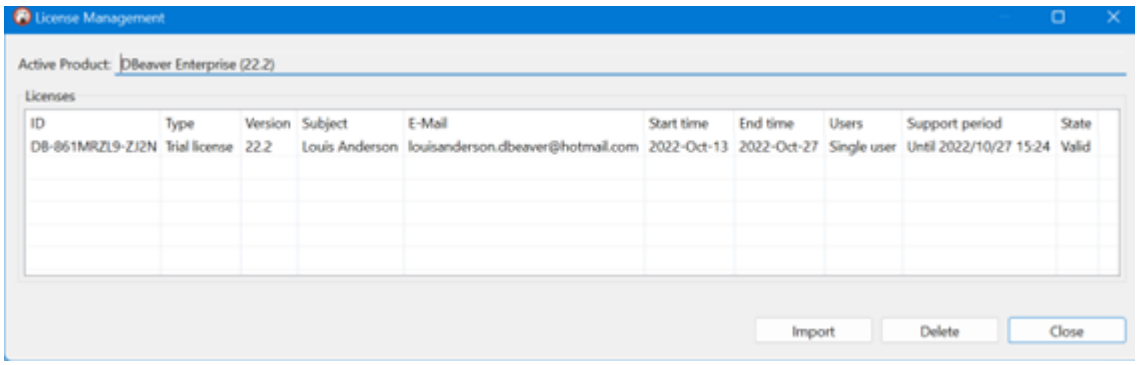
After the end of the selected support period you can continue to use commercial versions of DBeaver without support and updates or buy a license extension or a new license.

If you buy the DBeaver license extension and DBeaver has internet access, the license in DBeaver will be updated automatically. Otherwise, you have to import the license key from your personal account once again.

License Manager

License Manager provides you with the following information about your licenses:

- License ID e. g. DB-841MRZHY-ZH54;
- License type: Trial/Academic/Subscription/Standard;
- Version;
- License owner`s name and company name;
- License owner`s email;
- Start time is the date the license was received;
- End time is the date the license expires (standard perpetual licenses do not have this)
- Number of users: single user or multiuser for group licenses;
- Support period is the period you have access to the internal support system on the site and the possibility to download new product versions;
- State: valid or expired.



How to Reassign License

Note: This feature is available in Lite, Enterprise, Ultimate and Team editions only.

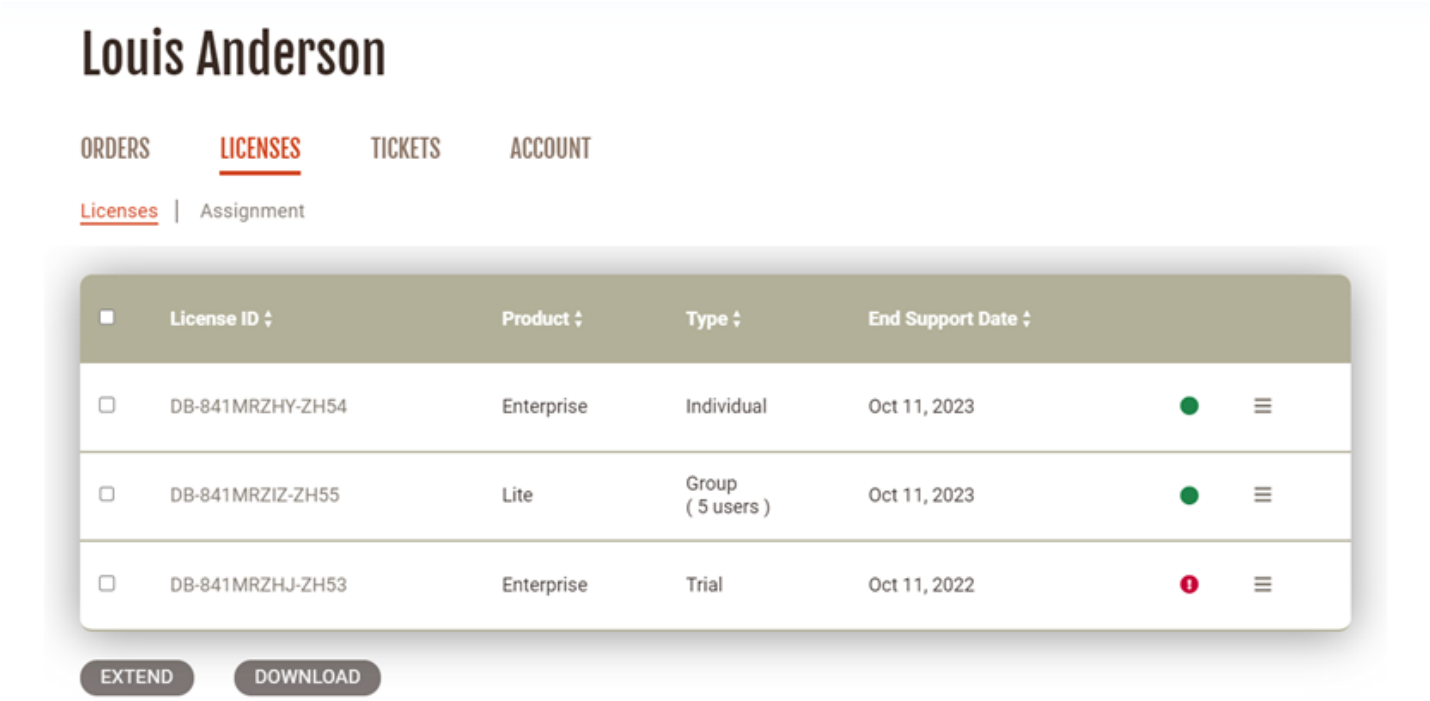
After purchasing a set of DBeaver licenses, you have to assign each license to an end user.

If an employee subsequently is leaving the company or the team that is using DBeaver, the license admin may need to reassign the license to another employee.

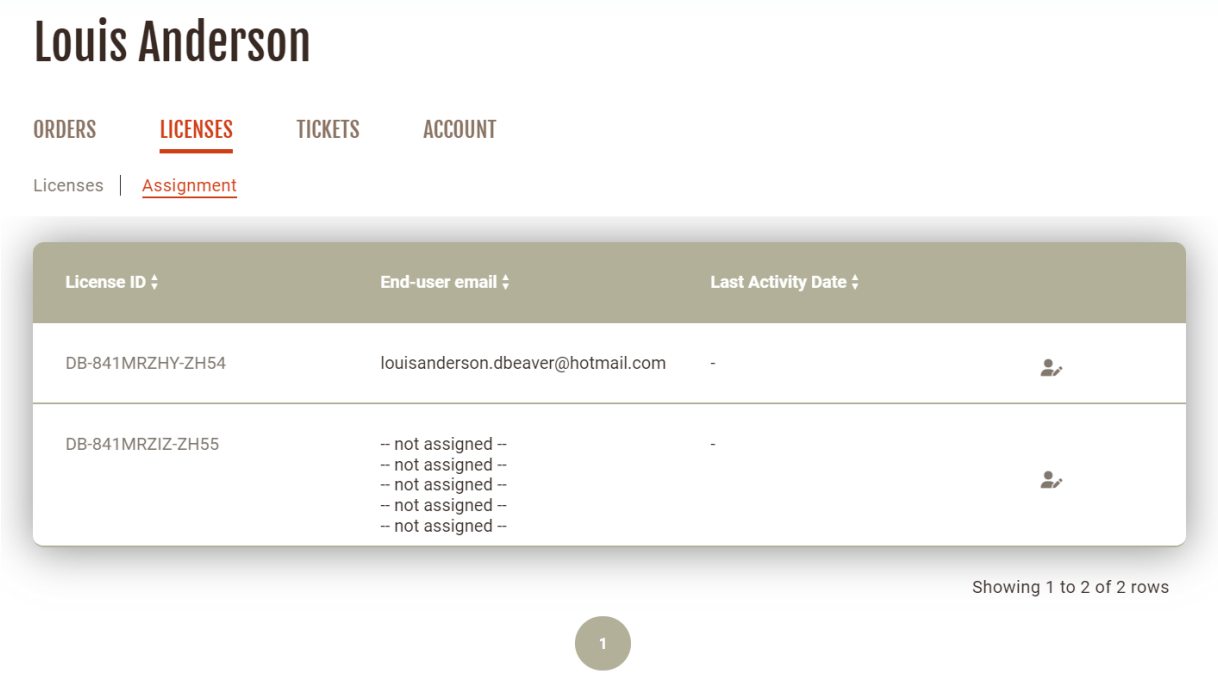
You can reassign the license to another user in your personal account.

Firstly, you need to Sign in.

Secondly, you should open the **Licenses tab**, where you can find all your licenses.

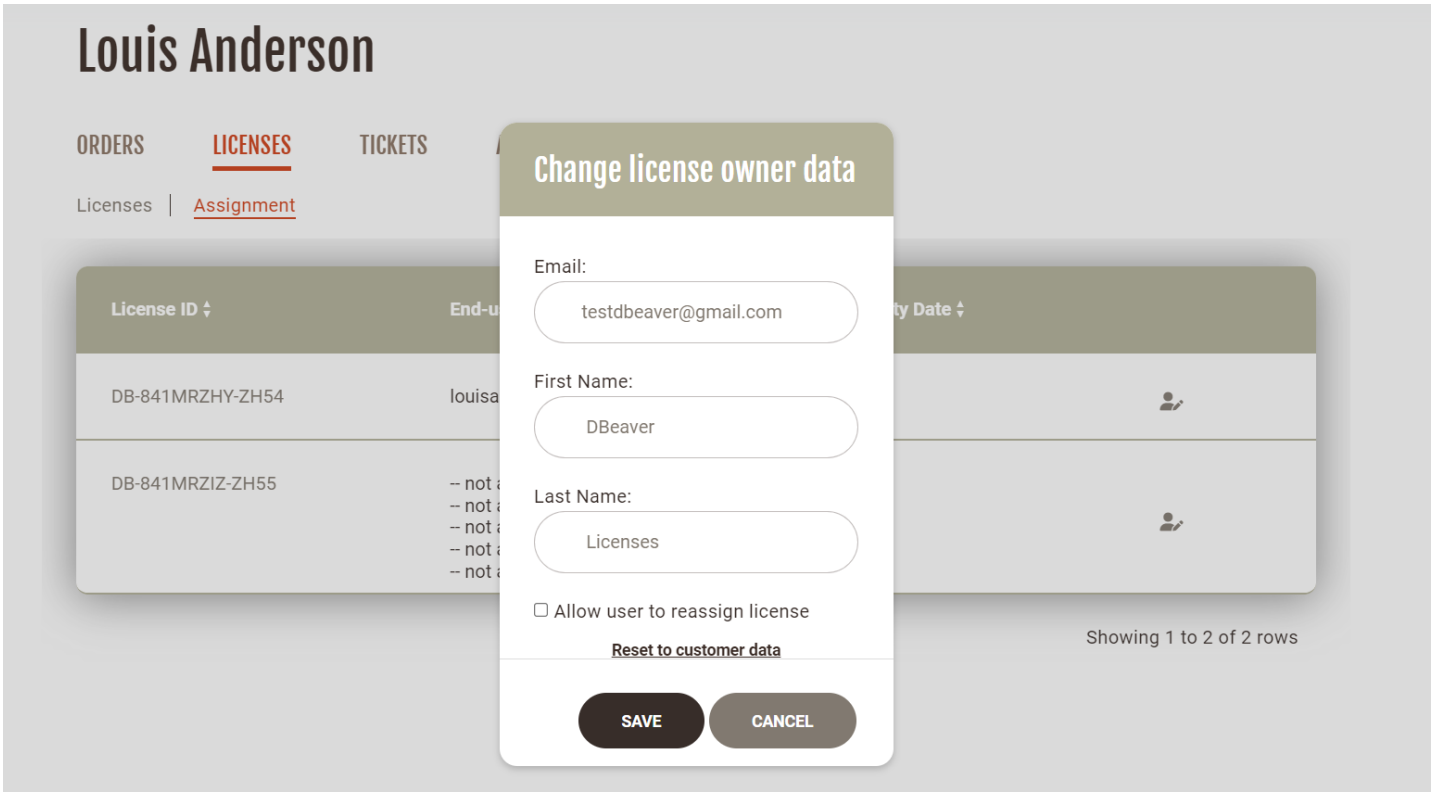


Secondly, you click the Assignment subtab. Here you can manage the license end-users and track the license usage.

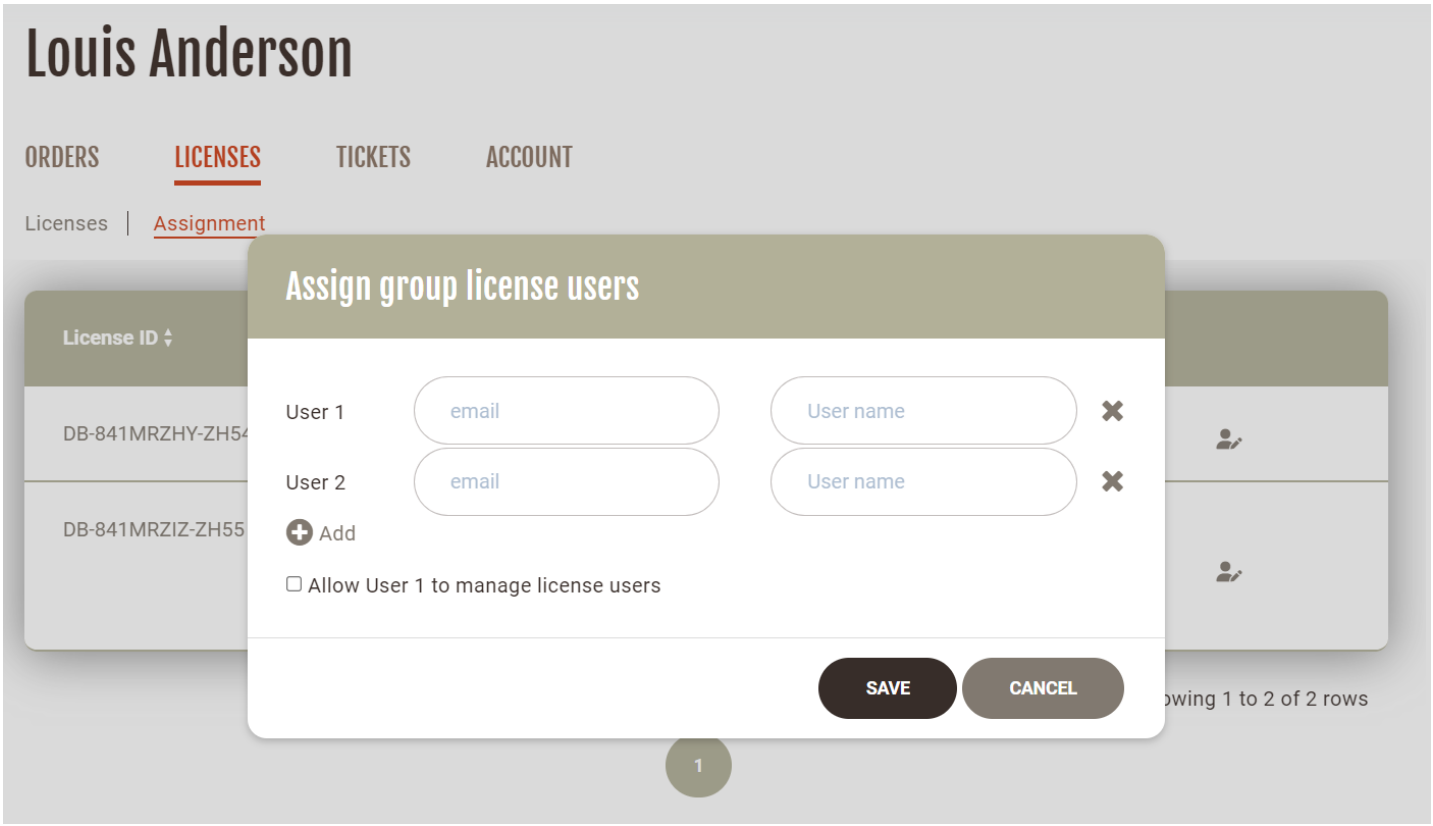


You select which license you need to reassign and press the icon in the right column.

A pop-up opens, and you can enter the new license end user's data: email; first and last name; and company. After filling in the form, you need to press the **SAVE** button and the license owner will be changed. The license key that contains the license end user's name and company name will be changed too.



If you have a group or unlimited license, it is also possible to assign users. In addition, if you click the checkbox 'Allow User 1 to manage license users', this user will become a license manager and will be able to assign users.



The license will remain in your personal account. The end user's email will be in the end-user column on the Assignment tab.

Also, the new license end user will be able to find the license in the personal account and import it but the end user cannot change the license data. If the new license end-user has not been signed up on our website, the account will be created automatically. The new user will receive a welcome email with a link to set up a password while the previous user will receive an email that the license has been withdrawn.

Connecting to Oracle Database using JDBC OCI driver

This article discusses how to establish connections to an Oracle database using JDBC OCI (Type II). Please take into consideration that the proposed way uses DBeaver's Generic driver. It means that you cannot get Oracle-specific functionality this way.

Prerequisites

JDBC OCI connections require Oracle Instant Client on the local machine. Please pay attention to the Instant Client and the JDBC driver versions, as they must be identical. DBeaver uses the 12.2.0.1 version by default at the moment, so we recommend using the 12.2.0.1 version of the Instant Client.

Install the Instant Client into some folder. We will refer to this folder as ORA_HOME for the rest of the article. Append ORA_HOME to the PATH variable and restart DBeaver before proceeding.

Configuration

1. Place your *tnsnames.ora* file into ORA_HOME/network/admin directory.
2. In DBeaver, click Window -> Driver Manager -> New. This opens Create new driver dialog.
3. In the Settings tab, add a *Driver name* of your liking. Set *Class Name* to 'oracle.jdbc.OracleDriver'. Set URL Template to 'jdbc:oracle:oci:@tnsAlias', where 'tnsAlias' is an alias from your *tnsnames.ora* file. Make sure that the Driver Type is set to Generic.
4. In the Libraries tab, you need to add Maven artifacts. To do that, click Add Artifact. Paste the following XML into the text field:

```
<dependencies>
    <dependency>
        <groupId>com.oracle.database.jdbc</groupId>
        <artifactId>ojdbc8</artifactId>
        <version>12.2.0.1</version>
    </dependency>
    <dependency>
        <groupId>com.oracle.database.nls</groupId>
        <artifactId>orai18n</artifactId>
        <version>12.2.0.1</version>
    </dependency>
    <dependency>

        <groupId>com.oracle.database.xml</groupId>
        <artifactId>xdb6</artifactId>
        <version>12.2.0.1</version>
    </dependency>
    <dependency>
        <groupId>com.oracle.database.xml</groupId>
        <artifactId>xmlparserv2</artifactId>
        <version>12.2.0.1</version>
    </dependency>
</dependencies>
```

NB: Replace the versions of the artifacts if you use a different version of the Instant Client.

5. In the Driver properties tab, make right-click -> Add new property.
6. Set the property name to 'protocol' (without quotes). Set the Value to 'oci' (without quotes).
7. Close the Driver manager.
8. Create a new connection using your newly configured driver.

Importing CA certificates from your local Java into DBeaver

Note: This article is relevant for you if you work behind the corporate firewall with a different root certificates set. In other cases, your issues can be caused by the incorrect SSL Configuration.

It's possible that your system administrator has installed a local Java and imported the required certificates to its keystore. We can use them to fix the issue.

Step 1: locate your Java

Windows

Press Windows + R to open the Run window. Type `cmd` in the prompt and press OK. It will open the command prompt. In the command prompt, type the following and press Enter:

```
where java
```

macOS

Open the Terminal and execute the following command:

```
/usr/libexec/java_home -V
```

Linux

Open a terminal and execute the following command:

```
readlink -e /usr/bin/java
```

Step 2: Find the JRE in DBeaver's installation

It's pretty easy. Just find the path where you installed DBeaver and open the `jre` folder there.

Step 3: Copy the cacerts

Open the folder with the Java you found in step 1. Locate the `cacerts` files under `/lib/security`, then copy-paste it into `<PATH_FROM_STEP_2>/lib/security`, replacing the old file. Restart DBeaver and you are ready to go.

SSL configuration

The `Pkix path building failed` error may occur due to incorrect SSL settings usage. As a general recommendation, we suggest **ignoring host name validation**. This checkbox is available in some databases' SSL tabs of the Connection settings.

New Table Creation

Overview

This guide provides detailed instructions on creating a new table in DBeaver. This is a beginner-friendly guide, meaning it is written for individuals who may not have previous experience with databases or DBeaver.

A database table, the focal point of our guide, is a systematic arrangement of data. It consists of rows and columns, forming cells where the data resides. While the number of columns is fixed for a table, it can encompass an unlimited number of rows. The maximum number of columns a table can have in a database depends on the database management system. DBeaver facilitates various operations, including data manipulation and data definition, with these tables.

Initially, the guide will cover the essential step of creating a table. After successfully creating a table, there are several enhancements you can add, such as columns, constraints, indexes, and triggers. The creation of a table is a fundamental operation in database management, and while there are many aspects to consider, not all of them are mandatory. The only mandatory step is the creation of the table itself and the definition of columns. The rest are optional and can be used to enhance the functionality and integrity of your database.

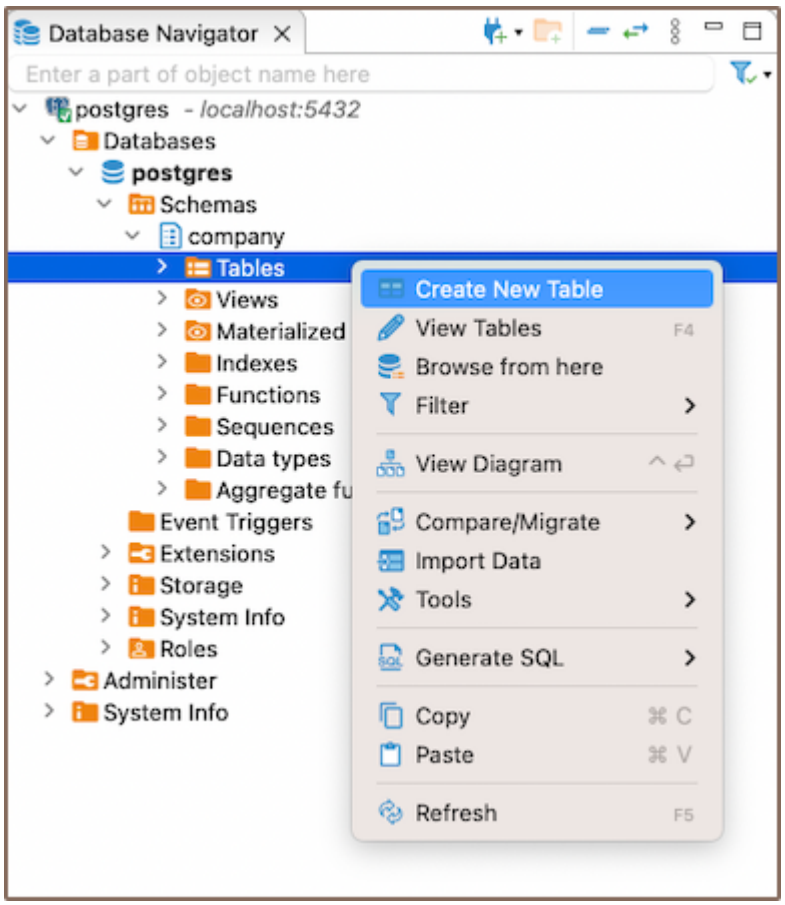
Table of contents:

- Table creation
- Creating columns
- Implementing constraints
- Utilizing foreign keys
- Creating indexes
- Incorporating triggers

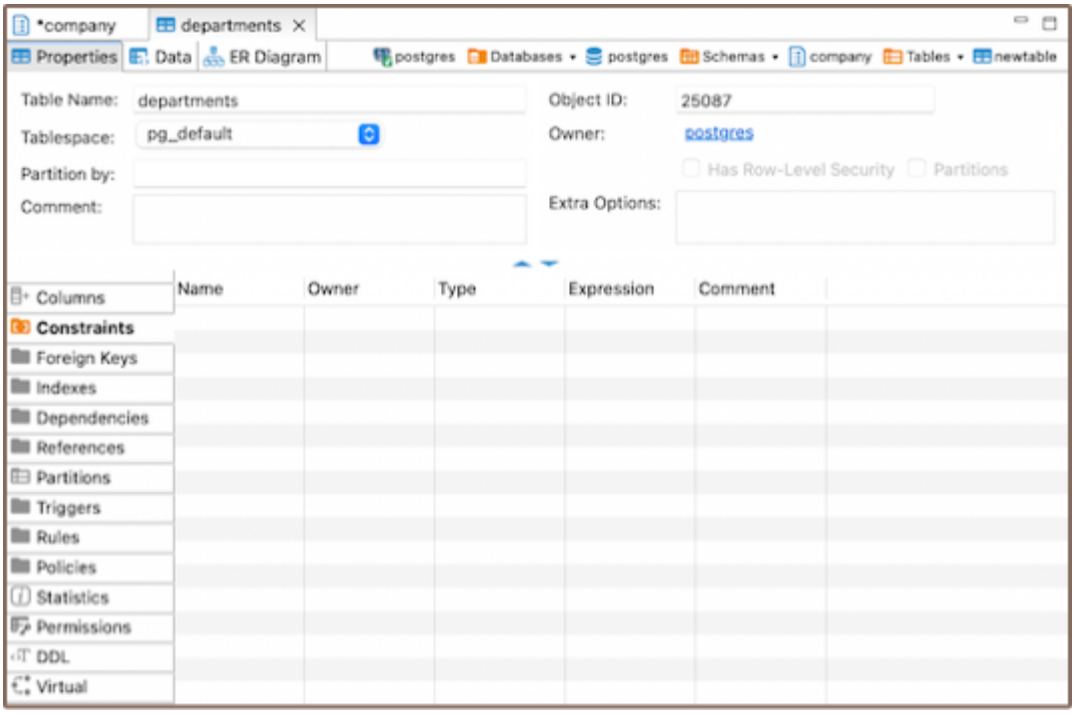
While this guide is designed around PostgreSQL within DBeaver, the processes can vary slightly with other database systems.

Create table

In DBeaver, creating a table is a straightforward process. Once you've connected to your database, you can initiate the table creation process by right-clicking on the Tables folder in the Database Navigator and selecting **Create New Table**.



Upon creating a new table in DBeaver, a window will open presenting the Properties editor. This is the dedicated interface where you can manage and manipulate your tables. The specific options available to you will depend on the capabilities of your chosen database system.



There are also various ways to create a table: from the **Properties editor** and from the **ER diagram**.

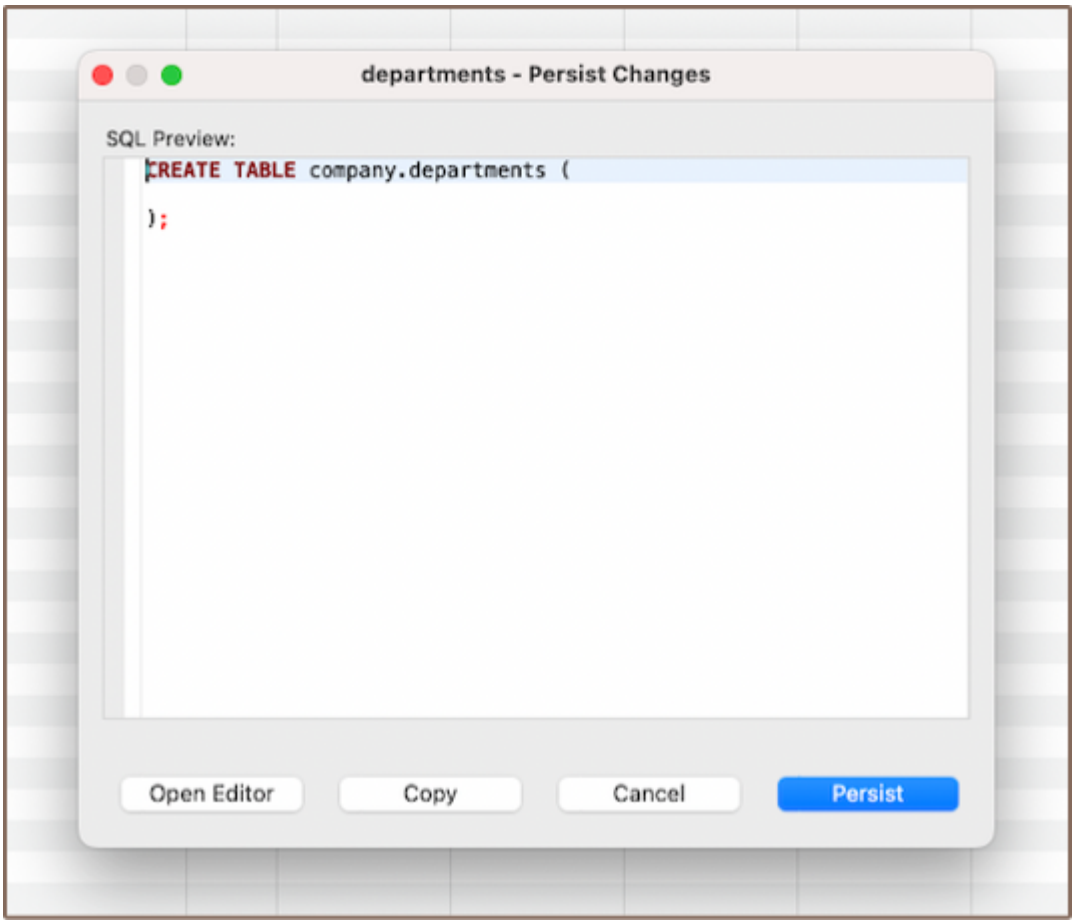
- From **Properties editor**: In the **Database Navigator**, double-click on the database name or table name, or select **View schema** from the context menu. In the window that opens, navigate to the **Properties** tab, select the **Tables** section, right-click on the window, and choose **Create New Table**.
- From ER Diagram: In the **Database Navigator**, double-click on the database name or table name. In the window that opens, navigate to the **ER Diagram** tab, right-click on the window, and choose **Create New Table**. Alternatively, in the **Database Navigator**, right-click on the database/table name and click **View Diagram**. In the window that opens, you can right-click and choose **Create New Table**.

Saving changes

Saving a table in DBeaver is a crucial yet straightforward process. Once you have defined the properties of the table, it's important to commit these changes to the database. Until you save your changes, your new table exists only within DBeaver and hasn't been created in the actual database. Here are the three options for saving the changes:

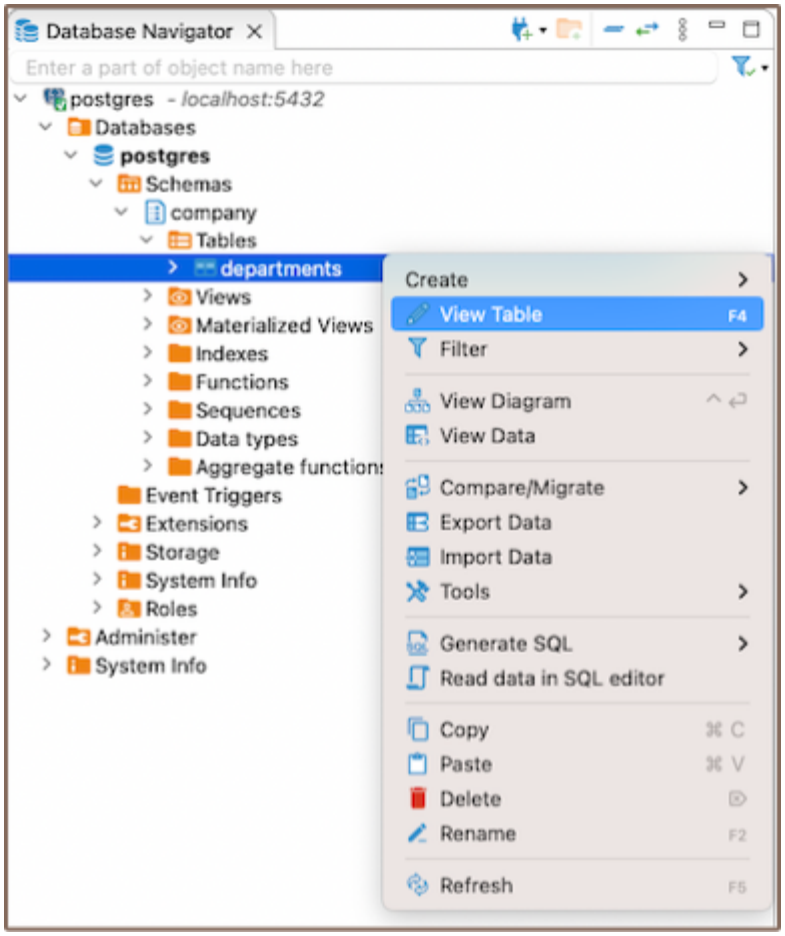
- Click on **File** -> **Save** -> **Persist**.
- Select the desired table in the **Database Navigator** and press `Ctrl+S` (or `CMD+S` for Mac OS), choose **Persist** to save the changes.
-

Utilize the **Save** button Save ... located at the bottom of the **Properties editor** and press **Persist** to save the changes.



Modify table

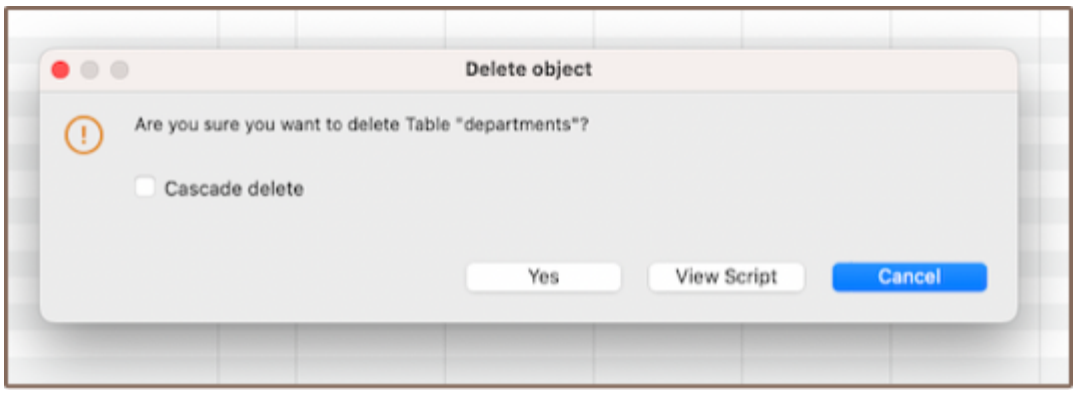
Modifying a table in DBeaver involves adjusting its structure or properties, such as adding, removing, or changing columns, adjusting constraints, or altering table settings. To make adjustments, navigate to the **Properties** editor. Access this by right-clicking the table's name in the **Database Navigator** menu and selecting **View Table**.



The specific options and capabilities available to you when modifying a table will depend on the database system you're working with. Always refer to the documentation of your specific database system for detailed information.

Delete table

Deleting a table in DBeaver involves a few simple steps. Firstly, navigate to the table you wish to delete in the **Database Navigator**. Once you've located the table, right-click on it and select the **Delete** option from the context menu. A confirmation dialog box will appear, allowing you to review the action before it's performed.



Upon confirming your decision, DBeaver will execute the necessary SQL command to remove the table from your PostgreSQL database. Please be aware that this action is permanent, and any data stored in the table will be irretrievably lost. Therefore, it's crucial to ensure that you've adequately backed up any important data before proceeding with the deletion.

Important: The confirmation dialog box, which appears when you are deleting a table in DBeaver, offers an important feature called the **Cascade delete** option. This option, when selected, automatically removes all related records in child tables, aligning with the deletion of a record in the parent table. While this can be useful for maintaining data integrity, it should be used with caution. If not handled properly, it can lead to unintended data loss by deleting records that you may not have intended to remove. Always ensure you understand the relationships and dependencies in your data before using the **Cascade delete** option.

Restrictions on table creation

DBeaver is a database management tool, and as such, it doesn't impose its own restrictions on table creation. Instead, the restrictions that apply when creating tables are determined by the specific database system you're using.

However, there are some general considerations to keep in mind when creating tables:

- **Naming Restrictions**: The name of the table must be unique within its schema. Most database systems also have restrictions on the length of the table name and the characters it can contain.
- **Storage Restrictions**: The total amount of data that can be stored in a table is determined by the storage capacity of the database system.
- **Performance Considerations**: While not a restriction per se, it's important to note that the structure of a table can have significant impacts on the performance of data retrieval and manipulation operations. For example, having a large number of columns, especially ones that are rarely used, can slow down query performance.

Remember that the specific restrictions and limitations can vary depending on the database system you're using. Always refer to the documentation of the specific database system for detailed information.

Further Reading:

New Table Creation | Creating columns | Implementing constraints | Utilizing foreign keys | Creating indexes | Incorporating triggers

Creating columns

As you delve deeper into the intricacies of table structures, understanding columns and their functionalities becomes crucial. In DBeaver, columns are fundamental components of tables, which in turn are housed within databases. Before you can create columns, you must first establish a database and a table.

Columns in DBeaver are versatile and customizable. You can create new columns, modify their settings, and even delete them when necessary. Each column in a table has a specific data type, dictating the kind of data it can store. You can also enforce rules on columns such as nullability, unique constraints, check constraints, and default values. These rules are essential to maintain data integrity, accuracy, and reliability in your database.

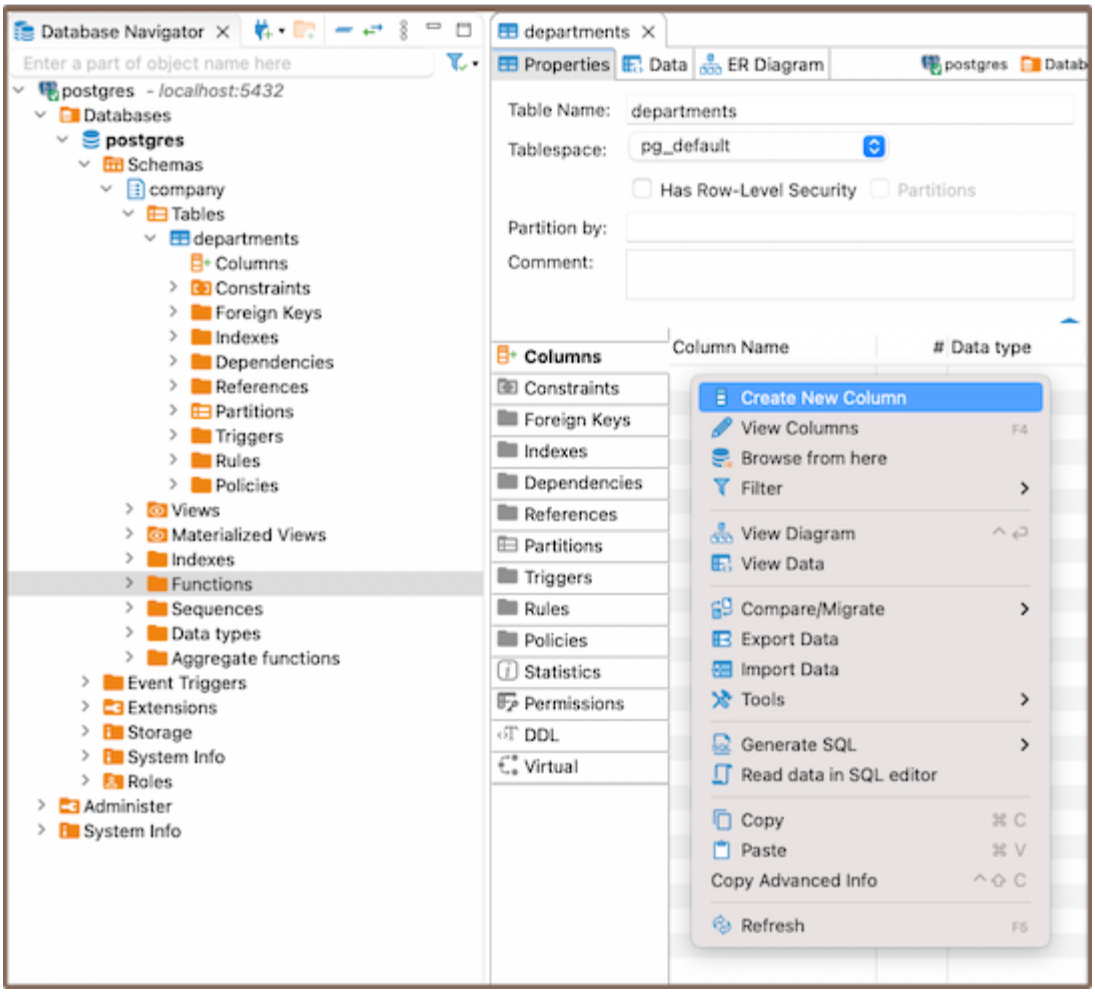
Beyond these basic operations, you can perform more advanced tasks with columns. You can rename columns, change their data types, adjust permissions, and add comments. All databases may not support some of these operations, so it's essential to be aware of the specific capabilities of your database system.

This guide will walk you through the process of creating, modifying, and deleting columns, as well as customizing their settings to suit your needs.

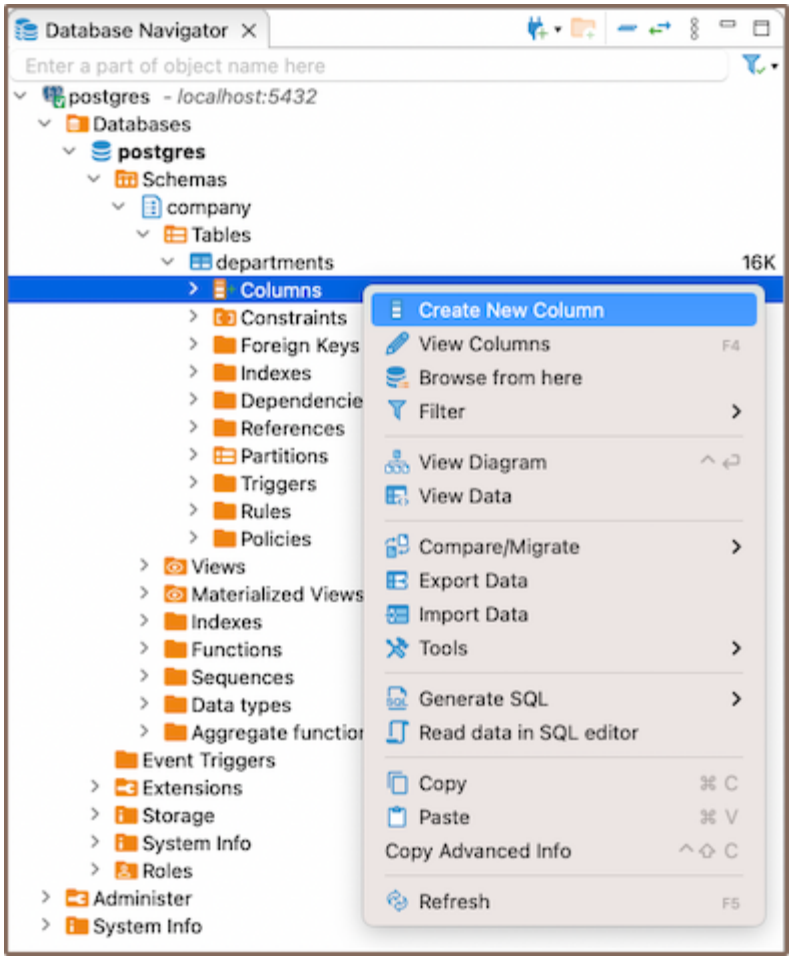
Create

Creating a new column in DBeaver involves a series of steps that guide you through the process. Here's how you can do it:

1) Go to the **Columns** tab in the Properties editor, right-click on the pane of the object's sub-entities, and select **Create New Column** from the context menu.

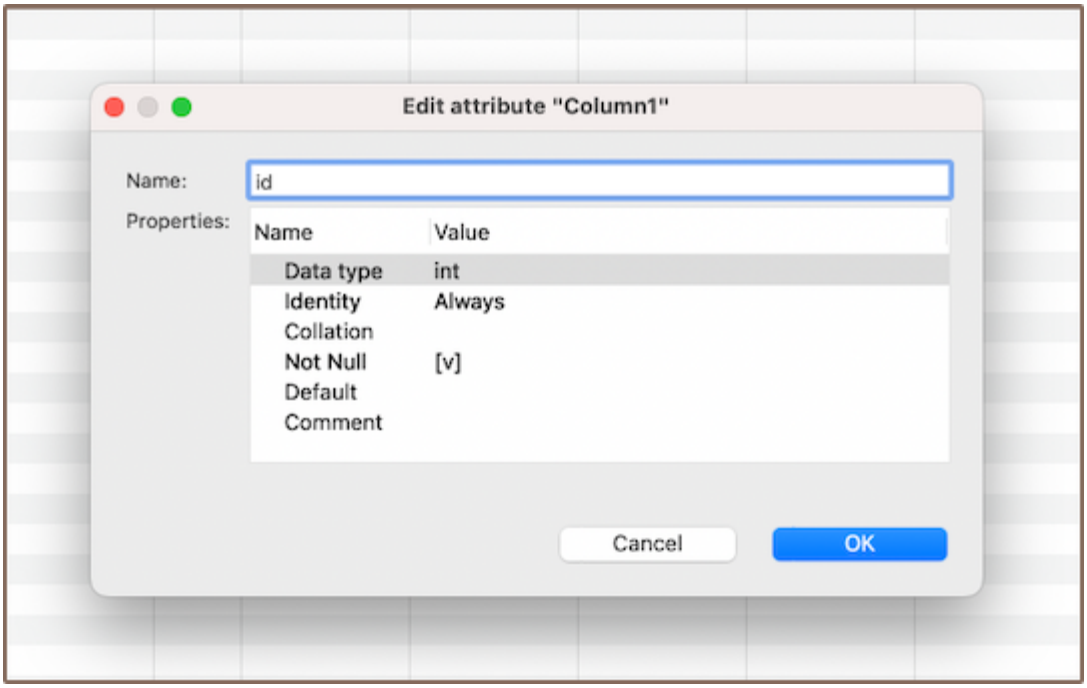


- Also you can also add a new column by expanding the table view in the Database Navigator and selecting **Create New Column**.



- Another method to create a new column is by utilizing the **Create New Column** button located at the bottom of the [Editor panel].

2) When creating a new column, you'll be presented with the **Edit Attribute** window. Here, you can customize the column's settings to suit your needs. You can adjust the **Name**, **Data type**, **Identity**, **Collation**, **Not null**, **Default**, and add a **Comment** as needed.



Field	Description
Name	Specify the name of the column.
Data type	Select the type of data the column will store.
Identity	Set the column as an identity column, which will auto-increment its value with each new row.
Collation	Set the collation for the column, which determines how data in the column is sorted and compared.
Not null	Enforce that the column must always contain a value.
Default value	Set a default value for the column, which will be used when no value is specified for the column during insertion.
Comment	Add a comment for the column, which can be helpful for documentation purposes.

Saving changes

Saving a column in DBeaver is an essential and simple operation. After you've configured the column properties, it's vital to persist these changes to the database. Until you save your modifications, your new column only exists within DBeaver and hasn't been added to the actual database table. Here are the three options for committing the changes:

- Click on **File** -> **Save** -> **Persist**.
- Select the desired table in the **Database Navigator** and press `Ctrl+S` (or `CMD+S` for Mac OS), choose **Persist** to save the changes.
- Utilize the **Save** button  located at the bottom of the Editor panel and press **Persist** to save the changes.

Modify

As your database evolves and your needs change, you may find it necessary to modify the properties of your columns. Whether it's renaming a column, changing its data type, adjusting its nullability, or updating its permissions, DBeaver provides you with the flexibility to adapt your columns to your changing requirements.

Modifying columns is a crucial aspect of database management, allowing you to ensure that your columns continue to serve their intended purpose effectively.

Here's how you can modify a column in DBeaver:

- Via the **Properties Editor**: You can double-click on the table name to open the **Properties Editor**. In this window, you can see all the columns of the table. Right-click on the column you want to modify and select **View Column**.
- Via the **Database Navigator**: Go to the **Database Navigator** menu, and choose the proper database and column to modify its settings. Right-click the column and select **View Column**.
- Via the **Bottom Menu**: In the bottom menu of the **Properties Editor** interface, there is a **View Column** button  . Clicking this button will open the properties of the currently selected column in the **Editor panel**, allowing you to modify its settings.

Note: Not all databases support all types of column modifications, so always check the specific capabilities of your database system.

Delete

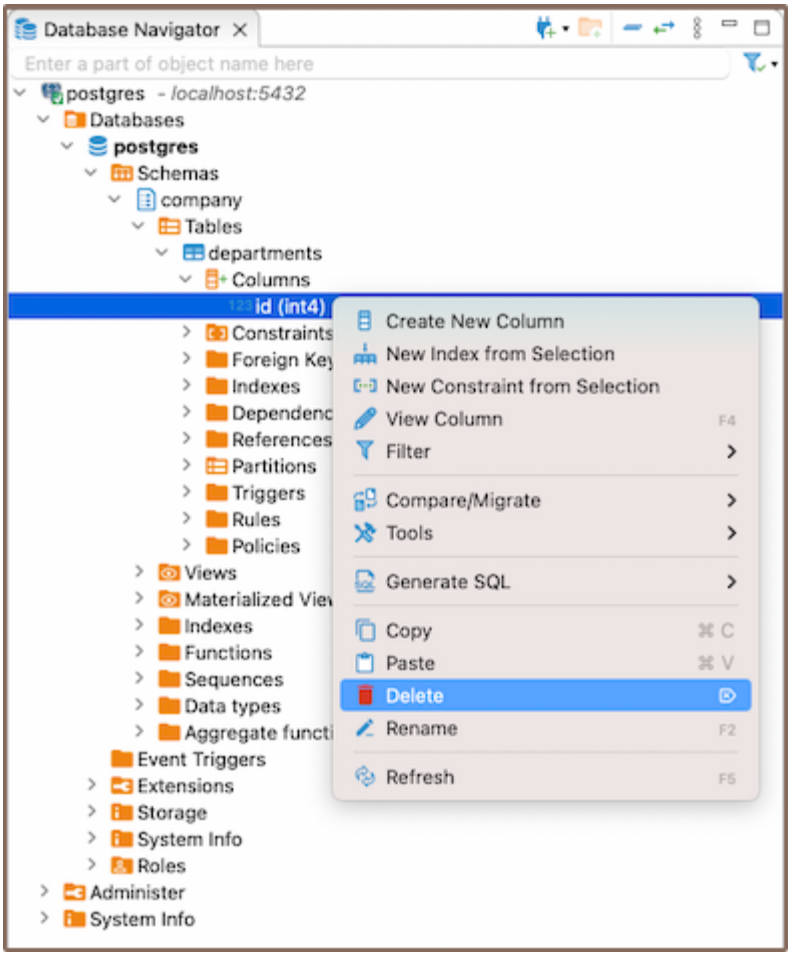
1) You can delete a column either through the **Properties Editor** or the **Database Navigator**:

- Using the **Properties Editor**: Open the Properties Editor, navigate to the **Constraints** tab of the corresponding table, and find the column you want to delete.
- Using the **Database Navigator**: Navigate to the database where the table with the column is located. Find and select the table, open the **Columns** folder, and find the column you want to delete.

2) To delete a column, right-click on the column's name and select **Delete**, or you can select the necessary column and press the `Delete` key, or **Delete button**  (in the **Bottom Menu** of the **Properties Editor**).

3) A confirmation dialog will appear, asking you to confirm the deletion. Before proceeding, ensure that you've selected the correct column for deletion.

4) After confirming the deletion, you'll need to persist the changes to apply them to the database.



Once a column is deleted, all the data stored in that column will be lost. Furthermore, there are several restrictions and considerations to keep in mind when deleting columns:

Limitations	Description
Data Loss	As mentioned, deleting a column will permanently remove all data stored in that column. Make sure to back up any important data before proceeding with the deletion.
Primary Key Columns	If the column you want to delete is a primary key column, you must first remove its primary key constraint within the table properties. Primary key columns are used to identify each row in the table uniquely, and deleting them without removing the constraint can cause issues.
Referential Integrity	If the column you want to delete is referenced by other tables in the database (i.e., it's a foreign key), you may not be able to delete the column without first deleting or modifying the referencing tables. This is to maintain referential integrity in the database.
Database Support	Not all databases support the ability to delete columns. Always check the specific capabilities of your database system.

Restrictions

When working with columns in DBeaver, it's important to understand the various restrictions that can be applied to them. These restrictions are not limitations but rather rules that help ensure the integrity, accuracy, and reliability of the data stored in your database. They dictate what data can be stored in a column, how it's treated, and how it interacts with other data in the database. Let's take a closer look at these restrictions:

- **Data Type**: Each column in a table has a specific data type. The type of data that can be stored in a column is dictated by its data type.
- **Nullability**: By default, columns can hold **NULL**. However, by specifying **NOT NULL** during column creation, you can enforce that every row must contain a value for that column.
- **Unique Constraint**: A unique constraint ensures that all values in a column are distinct. Duplicate entries are not permitted.
- **Check Constraint**: A check constraint enables you to specify a condition on a column. Every value added to the the column must satisfy this condition.
- **Default Value**: A column can be given a default value. If no value is specified during data insertion, the column uses the default.

Further Reading:

New Table Creation | Creating columns | Implementing constraints | Utilizing foreign keys | Creating indexes | Incorporating triggers

Implementing Constraints

As you continue to explore the intricacies of table structures, understanding the role and functionality of constraints becomes essential. Constraints are rules that we apply to data columns in a table when we want to limit the type of data that can go into a table. This can help ensure the accuracy and reliability of our data.

While creating constraints is not always necessary, it can be a powerful tool in the table creation process. Just like you create columns, you can also define constraints that uphold the integrity of your data. Constraints can be as simple as ensuring uniqueness in a column or as complex as a check expression that defines acceptable values for a column.

In this guide, you'll learn how to create, modify, and delete constraints, as well as how to customize their settings to suit your needs. You'll discover how to create primary keys, unique keys, and check expressions. You'll also learn about the restrictions and considerations you need to keep in mind when working with constraints.

Primary key

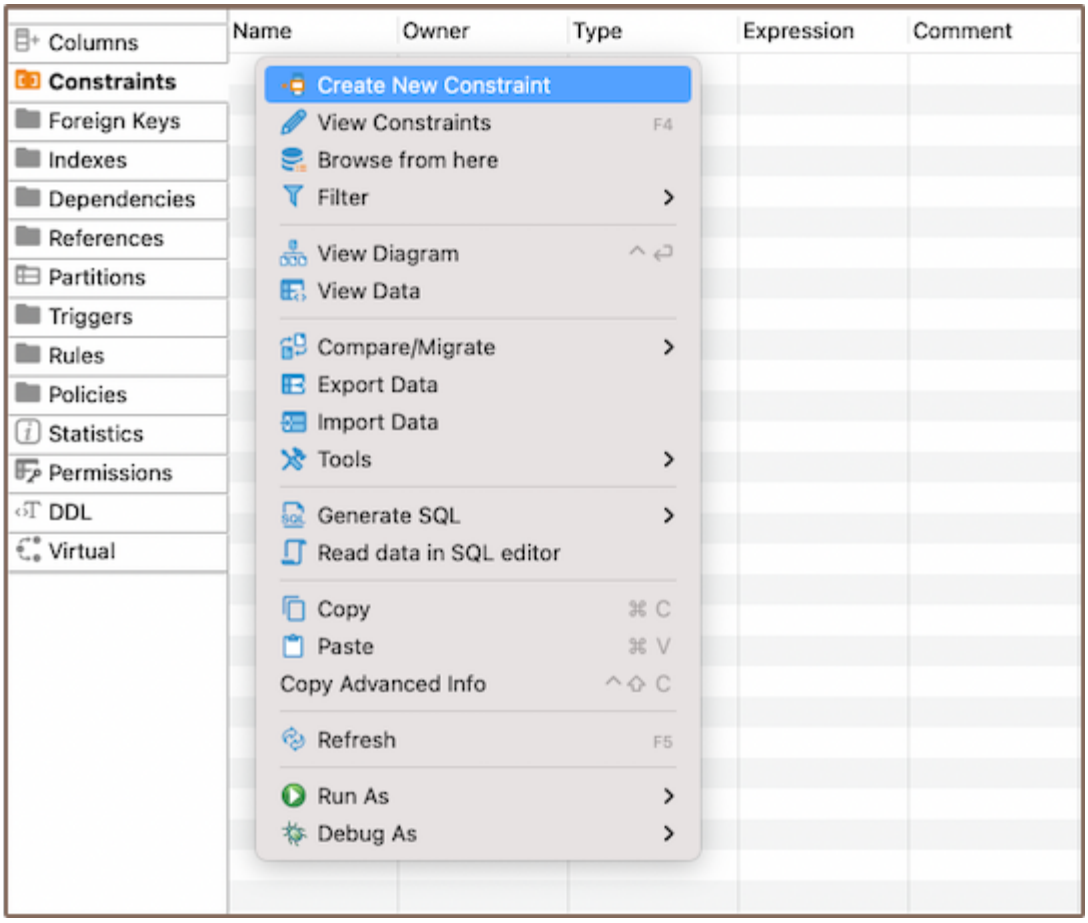
Primary keys are a crucial part of your database structure, uniquely identifying each record in a table. When you define a primary key, the database ensures no duplicate values exist in the specified column(s), maintaining data integrity and improving performance. DBeaver simplifies this process, providing an intuitive interface for creating, modifying, and deleting primary keys.

Create

Creating a new primary key in DBeaver involves a series of steps that guide you through the process. Here's how you can do it:

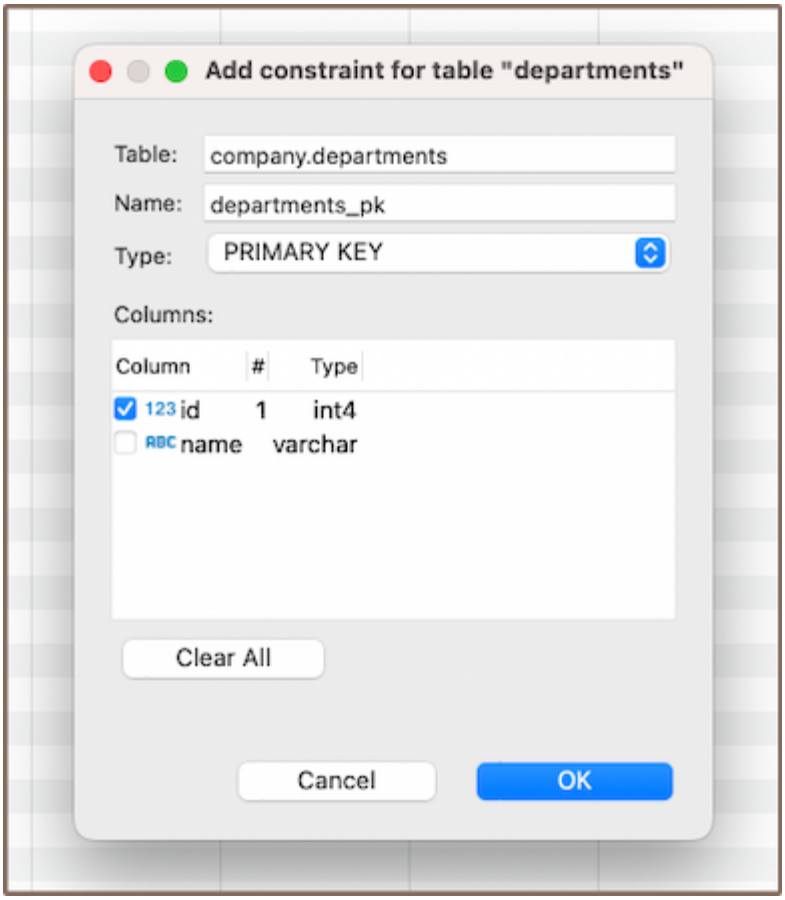
1) You can start the process of creating a new constraint in three ways:

- Using the Properties editor: Open the Properties Editor and navigate to the **Constraints** tab of the corresponding table. This is where you'll find all the existing constraints for the table and where you'll create new ones.



- Using the Database Navigator: Navigate to the database where the table you want to add a constraint to is located. Find and select the table for which you want to create a new constraint. Within the selected table, you'll find a folder named **Constraints**.
- You can also utilize the **Create New Constraint** button located at the bottom of the **Properties editor**.

2) Depending on the method you chose in step 1, either right-click on the window (in the **Properties Editor**) or within the **Constraints** folder (in the **Database Navigator**), and select **Create New Constraint**. If you're using the third method, simply click the **Create New Constraint** button. This will open a new window where you'll be able to define the details of your new constraint.



Field	Description
Name	Here, you can specify the name of the constraint. It's a good practice to choose a name that clearly indicates the purpose of the constraint.
Type	This is where you select the type of constraint. Options include Primary Key, Unique Key, Check. The type of constraint you choose will depend on the rule you want to enforce on the column.
Columns	In this section, you can select the column or columns that the constraint will apply to. If you're creating a composite constraint, you can select multiple columns.

3) After selecting the column, click **OK** to save your new constraint. A window will appear displaying the newly created primary key. This is a good time to double-check that everything is correct.

Saving changes

Once you have configured the constraint properties, it is essential to save these changes to the database. Until you commit your modifications, the new constraint will only exist within DBeaver and will not be added to the actual database table. Here are three options for committing the changes:

- Click on **File** -> **Save** -> **Persist**.

- Select the desired table in the **Database Navigator** and press `Ctrl+S` (or `CMD+S` for Mac OS), choose **Persist** to save the changes.
- Utilize the **Save** button  located at the bottom of the **Properties editor** and press **Persist** to save the changes.

Modify

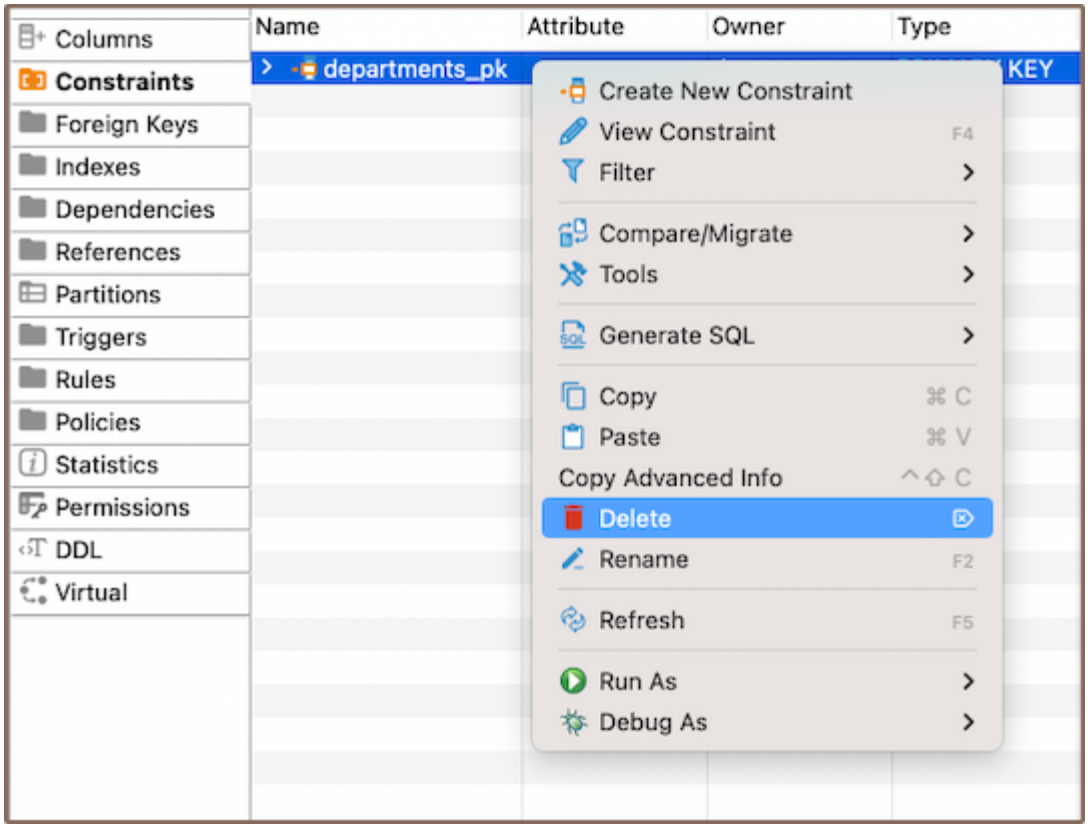
To modify an existing primary key, you need to first delete it and then add a new primary key with the updated parameters.

Note: Remember, modifying primary keys should be done with care. Primary keys are integral to maintaining the integrity of your data, and changing them can have significant effects on your database. Always double-check your changes and make sure they align with your data requirements.

Delete

1) You can delete a primary key either through the **Properties Editor** or the **Database Navigator**:

- Using the **Properties Editor**: Open the Properties Editor, navigate to the **Constraints** tab of the corresponding table, and find the constraint you want to delete.



- Using the **Database Navigator**: Navigate to the database where the table with the constraint is located. Find and select the constraint, open the **Constraints** folder, and find the constraint you want to delete.

2) To delete a primary key, right-click on the key's name and select **Delete**, or you can select the necessary primary key and press the `Delete` key, or **Delete button** (in the **Bottom Menu** of the **Properties Editor**).

3) A confirmation dialog will appear, asking you to confirm the deletion. Before proceeding, ensure that you've selected the correct constraint for deletion.

4) After confirming the deletion, you'll need to persist the changes to apply them to the database.

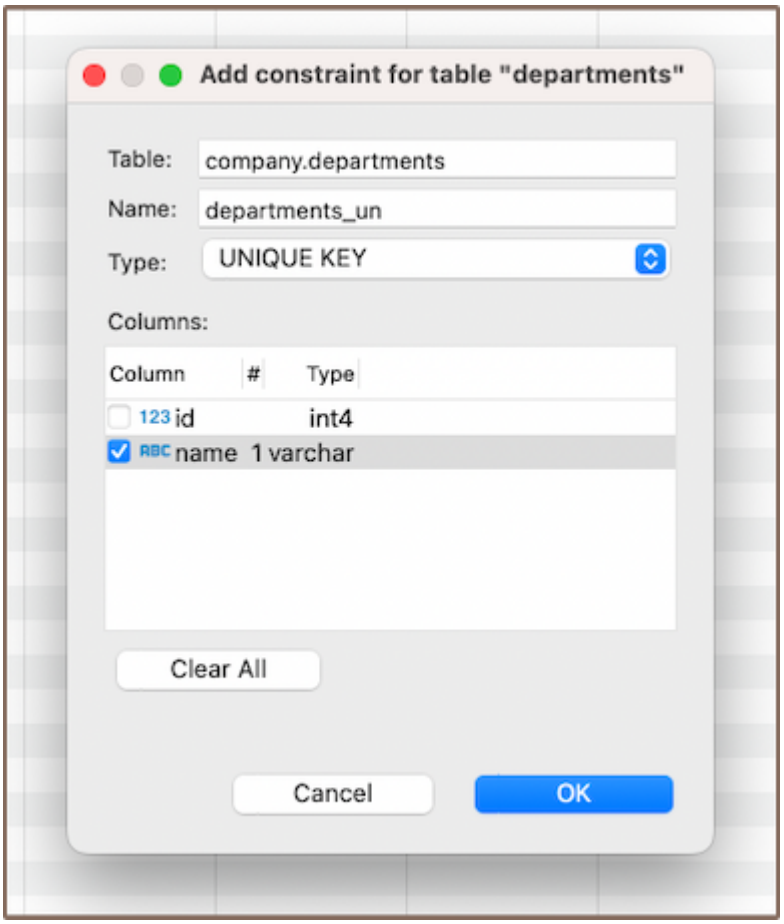
Important: Be particularly cautious if you're considering using the **Cascade delete** option. This option will not only delete the primary key but also any dependent records in your database. This can lead to significant data loss if not handled carefully.

Unique key

A unique key is a tool that ensures distinct values within a column or a group of columns in a table. While it can identify unique records, it doesn't have to be the primary key. You can create a unique key on a single column or

multiple columns, ensuring that each value is unique. This is crucial for maintaining data integrity within your table. If you attempt to insert a duplicate value into a column with a unique key constraint, you'll encounter an error.

The processes of creating, deleting and modifying a unique key are similar to those for a primary key.



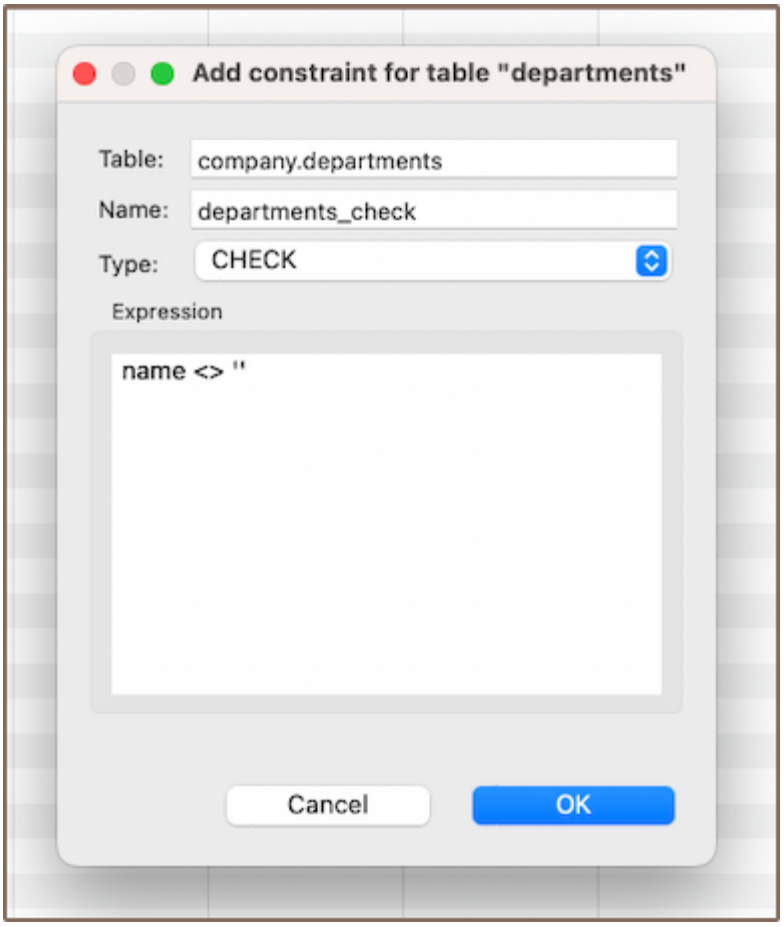
Check expression

A check expression in a database is also a constraint, it defines acceptable values for a column. It enforces specific conditions, ensuring that the data in the column complies with the defined rules. The processes of creating, deleting and modifying a check expression are similar to those for a primary key.

For instance, suppose you have a column named "name" and you want to ensure that its value is never an empty string. You could create a check expression as follows:

```
name <> ''
```

This would prevent any records from being inserted or updated with an empty string for the "name" column.



Note: Check expressions are a powerful tool for maintaining data integrity. They enforce specific conditions on your data, preventing invalid entries. Always consider the implications of your actions when working with check expressions.

Restrictions

Working with constraints comes with certain restrictions that you need to be aware of. These restrictions are in place to maintain the integrity of your data and prevent unwanted changes. Here's what you need to know:

- **Unique Values**: When creating a primary key or unique constraint, the column(s) you choose must contain unique values. If there are duplicate values in the column(s), the constraint creation will fail.
- **Null Values**: Primary keys cannot contain null values. If the column(s) you choose for a primary key contains null values, you will need to either remove or replace these before creating the constraint. Unique constraints, on the other hand, typically allow for multiple null values.

- **Data Type Compatibility**: The type of constraint you want to create may have specific data type requirements. For example, check constraints require a Boolean expression, so the column(s) involved must be able to participate in such an expression.
- **Existing Data Compliance**: When creating a check constraint, all existing data in the table must comply with the check condition. If any rows violate the condition, the constraint creation will fail.
- **Performance Impact**: Creating constraints, especially on large tables, can have a performance impact. This is because the database needs to check the constraint condition for all existing rows. This might take considerable time and resources for large tables.
- **Modifying Constraints**: Constraints cannot be directly modified. This is because constraints are defined with specific parameters that can't be changed directly.
- **Deleting Constraints with Dependencies**: You can't delete a primary key in a table with records unless you first remove all dependent foreign keys. This is because the primary key is likely being referenced by these foreign keys, and deleting it would break these references and potentially cause data integrity issues.

Further Reading:

New Table Creation | Creating columns | Implementing Constraints | Utilizing foreign keys | Creating indexes | Incorporating triggers

Utilizing Foreign-Keys

This article provides a comprehensive guide on the creation and management of Foreign Keys within DBeaver. While not always necessary, foreign keys can be used to establish relationships between tables, which can help maintain data consistency and integrity.

In DBeaver, the creation of a foreign key involves referencing a unique key in another table. This process forms a relationship that is essential for maintaining data integrity. Additionally, DBeaver allows you to define specific rules for the behavior of foreign keys when deleting or updating rows in the main table.

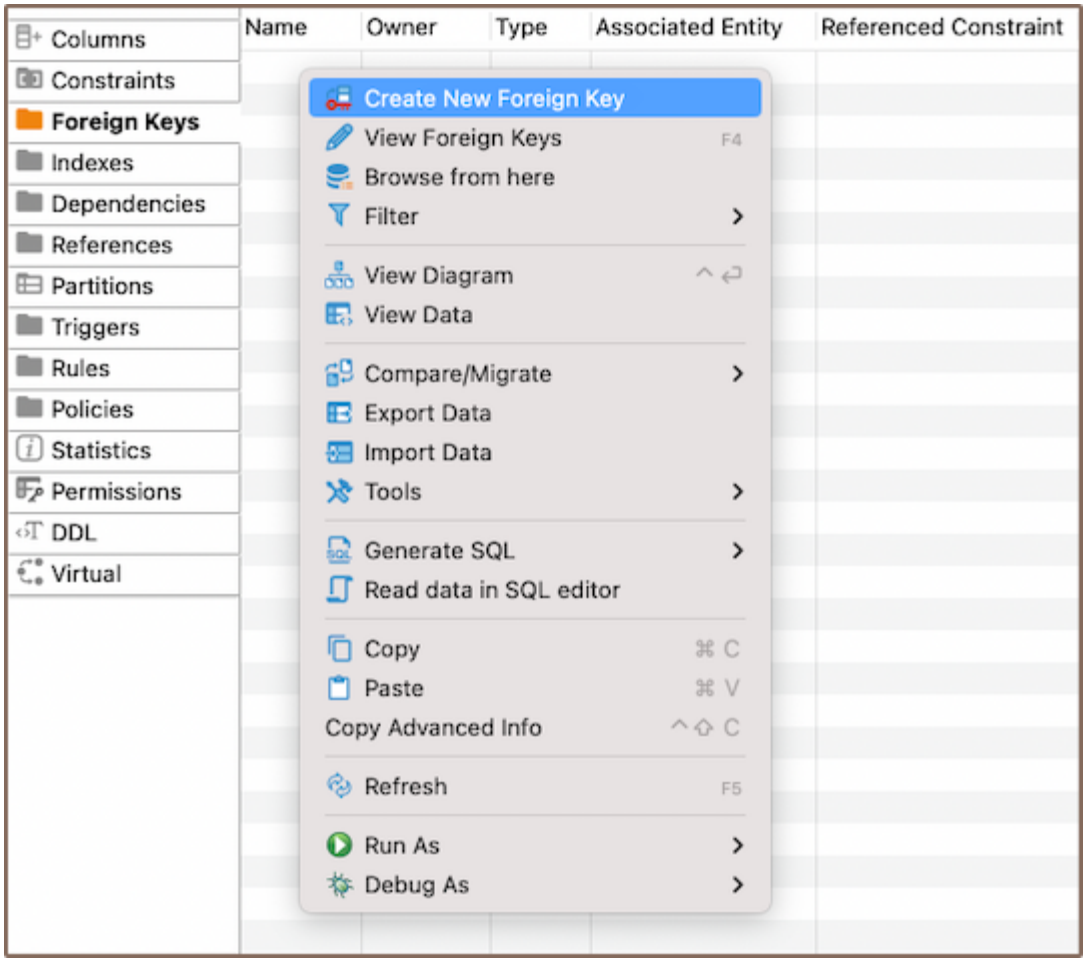
This guide will detail the steps involved in creating, modifying, and deleting foreign keys. It will also explain how to set up rules for their behavior. It is important to note that the management of foreign keys requires careful consideration due to their significant impact on data relationships.

Create

Creating a new foreign key in DBeaver involves a series of steps that guide you through the process.

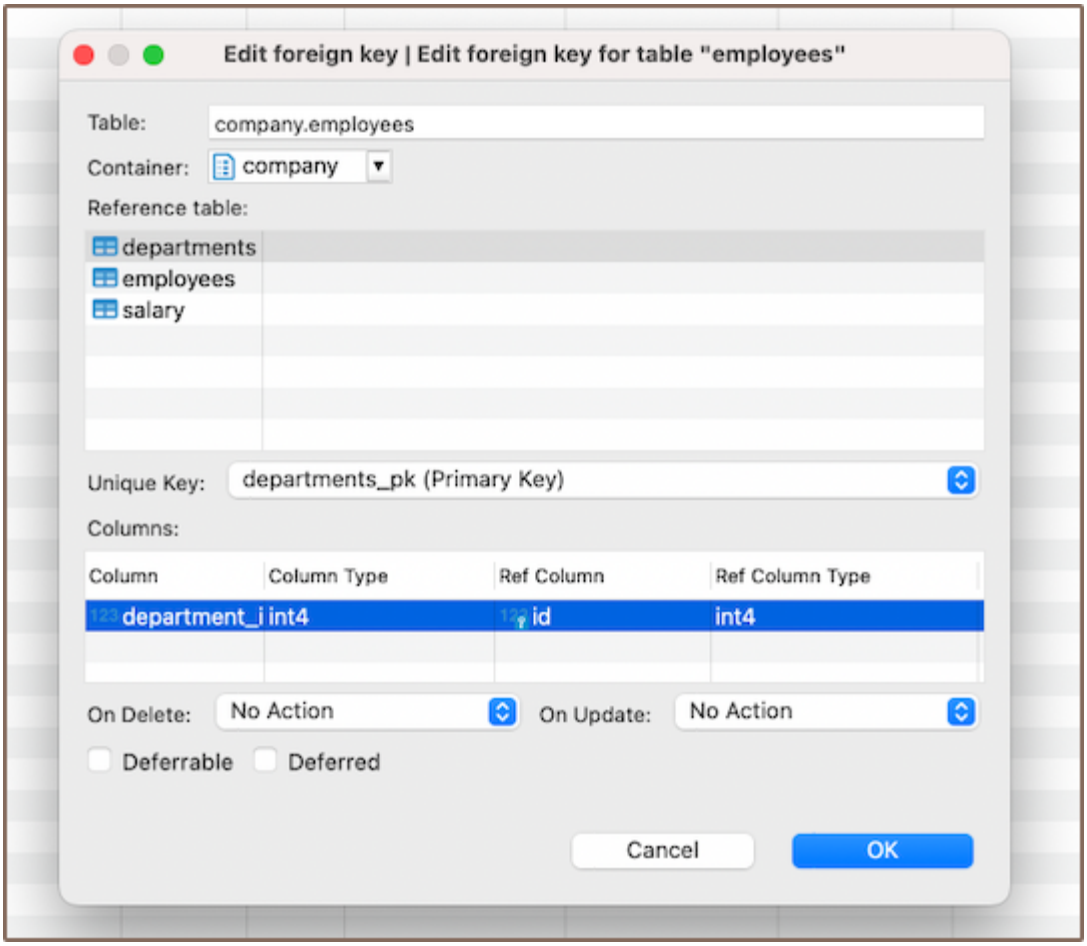
1) You can start the process of creating a new foreign key in three ways:

- Using the Properties editor: Open the Properties Editor and navigate to the **Foreign Keys** tab of the corresponding table.



- Using the Database Navigator: Navigate to the database where the table you want to add a foreign key to is located. Find and select the table for which you want to create a new foreign key. Within the selected table, you'll find a folder named **Foreign Keys**.
- Additionally, to create a new foreign key, you can utilize the **Create New Foreign Key** button located at the bottom of the **Properties editor**.

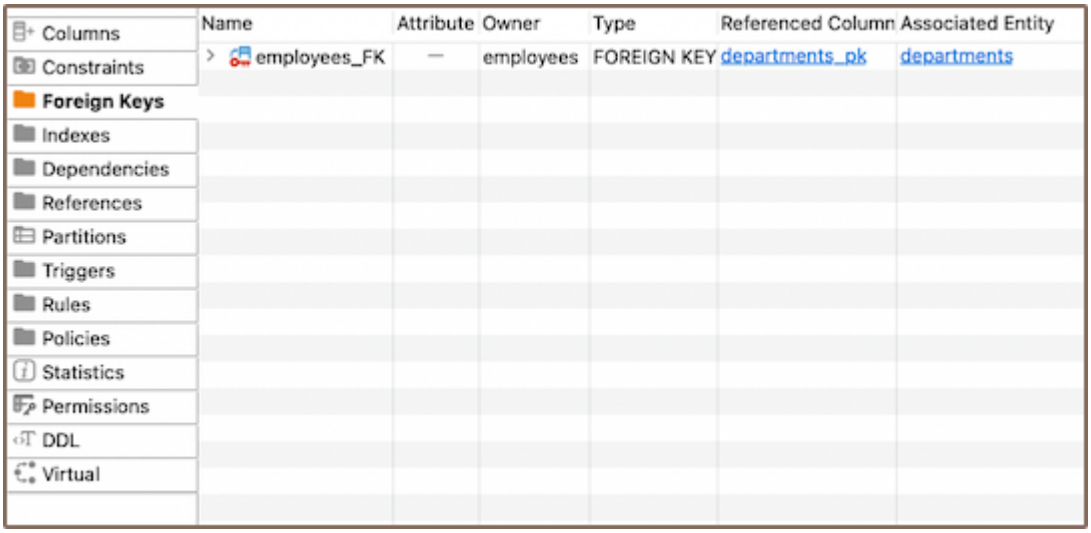
2) Depending on the method you chose in step 1, either right-click on the window (in the **Properties Editor**) or within the **Foreign Keys** folder (in the **Database Navigator**), and select **Create New Foreign Key**. If you're using the third method, simply click the **Create New Foreign Key** button. This will open a new window where you'll be able to customize the foreign key's settings to suit your needs.



Field	Description
Reference table	Select the table that the foreign key references.
Unique Key	Choose the unique key in the reference table that the foreign key will point to.
Columns	Select the columns in your table that will become the foreign key.
Deferrable	Specify whether the enforcement of the foreign key constraint can be deferred until the end of the transaction.
Deferred	If the foreign key is deferrable, this option sets whether the constraint is deferred by default.
On delete /on update:	Set the actions that will be taken when the referenced data is deleted or updated. The options are **No Action**, **Cascade**, **Restrict**, **Set NULL**, and **Set Default**.
- No Action	This option means that if the referenced data is deleted or updated, no action will be taken. However, if such a change would result in a violation of the foreign key constraint, the change will be disallowed.
- Cascade	If the referenced data is deleted or updated, the same action will be applied to all rows holding a foreign key that points to that data. For example, if a row in the referenced table is deleted, all rows in the table with the foreign

	key that point to that row will also be deleted.
- Restrict	This option disallows the deletion or updating of referenced data. If any foreign keys reference the data, the change will be disallowed.
- Set Null	If the referenced data is deleted or updated, all foreign keys that reference it will be set to `NULL`.
- Set Default	If the referenced data is deleted or updated, all foreign keys that reference it will be set to their default values. If no default value is specified, this option behaves the same as **No Action**.

3) After hitting **OK**, a window displaying the newly created foreign key will appear.



Saving changes

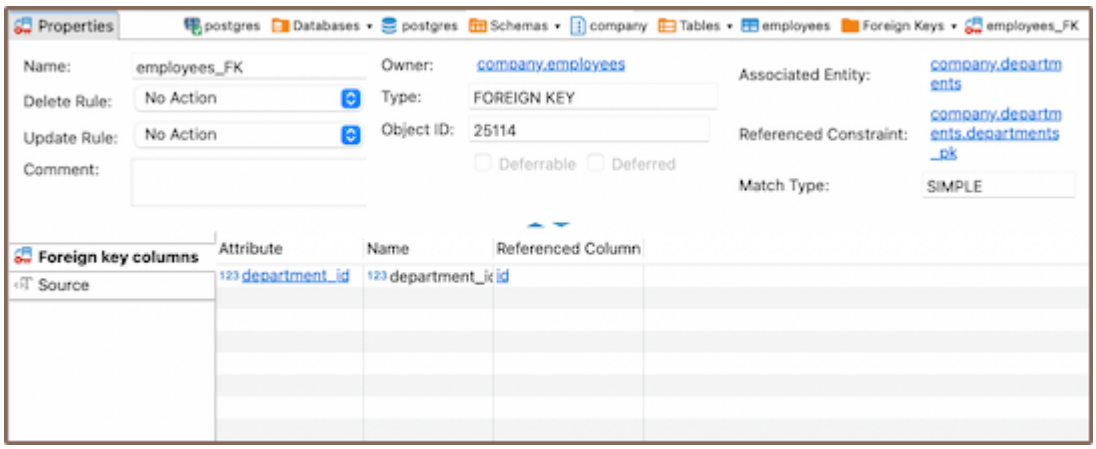
Once you have configured the foreign key properties, it is essential to save these changes to the database. Until you commit your modifications, the new foreign key will only exist within DBeaver and will not be added to the actual database table. Here are three options for committing the changes:

- Click on **File** -> **Save** -> **Persist**.
- Select the desired table in the **Database Navigator** and press `Ctrl+S` (or `CMD+S` for Mac OS), choose **Persist** to save the changes.
- Utilize the **Save** button Save ... located at the bottom of the **Properties editor** and press **Persist** to save the changes.

Modify

Over time, you may need to adjust the properties of your foreign keys. This could involve renaming a foreign key, changing its referenced table or unique key, adjusting its columns, or updating its rules for deletion or update. DBeaver allows you to make these modifications as needed. Here's how:

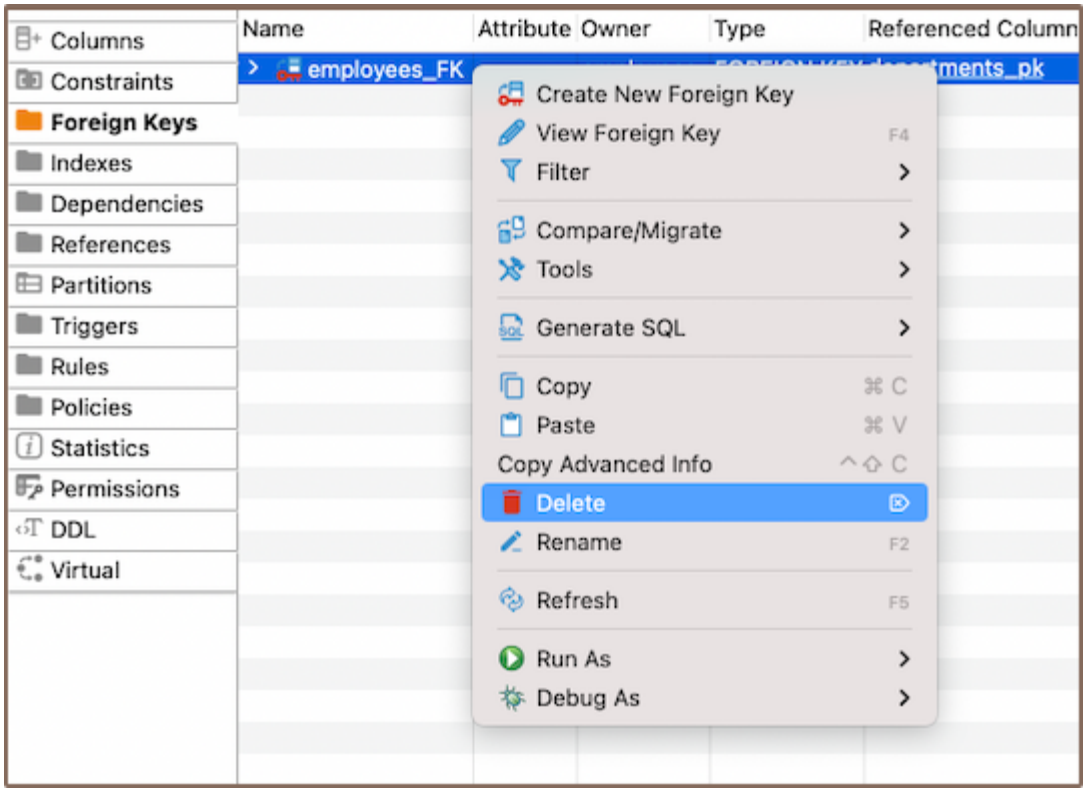
- **Properties Editor**: Open the Properties Editor by double-clicking on the table name. Within the selected table, you'll find a folder named **Foreign Keys**. Right-click on the foreign key you want to modify and select **View Foreign Key**.
- **Database Navigator**: In the Database Navigator, select the appropriate database and table. Within the selected table, you'll find a folder named **Foreign Keys**. Right-click the foreign key and select **View Foreign Key**.
- **Bottom Menu**: In the **Properties Editor** interface, use the **View Foreign Key** button  to open the properties of the selected foreign key in the **Editor panel**.



Delete

1) You can delete a foreign key either through the **Properties Editor** or the **Database Navigator**:

- Using the **Properties Editor**: Open the Properties Editor, navigate to the **Foreign Keys** tab of the corresponding table, and find the foreign key you want to delete.



- Using the **Database Navigator**: Navigate to the database where the table with the foreign key is located. Open the **Foreign Keys** folder, and find the foreign key you want to delete.

2) To delete a foreign key, right-click on the key's name and select **Delete**, or you can select the necessary foreign key and press the `Delete` key, or **Delete button** (in the **Bottom Menu** of the **Properties Editor**). 3) A confirmation dialog will appear, asking you to confirm the deletion. Before proceeding, ensure that you've selected the correct foreign key for deletion. 4) After confirming the deletion, you'll need to persist the changes to apply them to the database.

Restrictions

- **Referential Integrity**: Foreign keys enforce referential integrity. The values in the foreign key column must match those in the referenced primary key.
- **No Circular References**: Circular foreign key references are not permitted in most databases where two or more tables refer to each other.
- **Delete and Update Operations**: Deleting or updating rows from the referenced table can be complicated because of the need to preserve referential integrity.

- **Schema Modifications**: Changes to the schema of a table (like altering data types or column deletions) can be restricted if a foreign key references the table.

Further Reading:

New Table Creation | Creating columns | Implementing Constraints | Utilizing foreign keys | Creating indexes | Incorporating triggers

Creating Indexes

In the process of creating tables in a database, an integral part is the creation of indexes. Indexes are tools that can significantly enhance the performance of your database. They are designed to speed up the retrieval of data from the tables, making them particularly useful in large databases where without them, searches could be time-consuming.

Indexes work by providing a more direct path to the data, reducing the need for the database to scan the entire table. They can be created on one or more columns of a table, and when a query is made, the database can use the indexes to find the data more quickly.

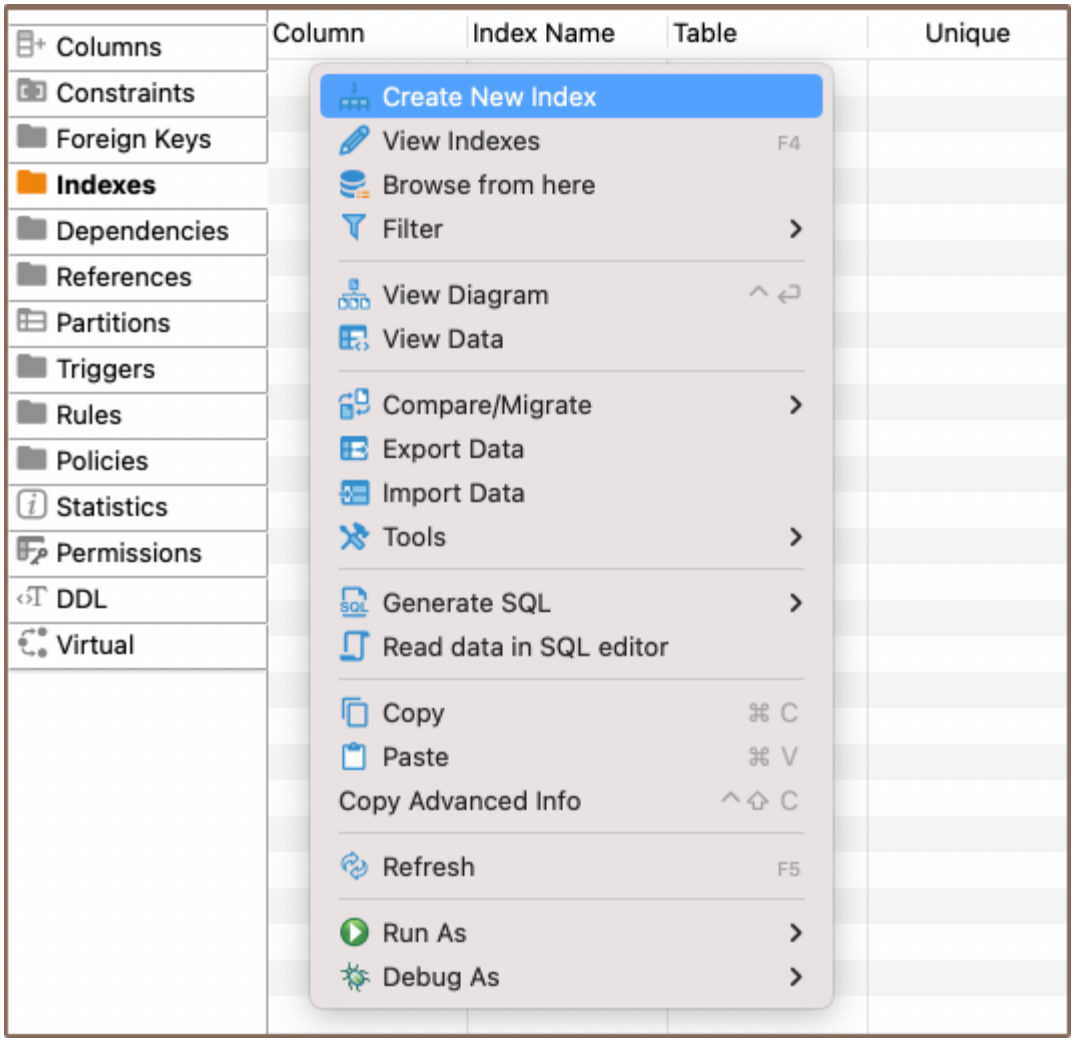
However, while indexes can improve the speed of data retrieval, they can also slow down data modification operations such as `INSERT`, `UPDATE`, and `DELETE`, as the index must also be updated. Therefore, it's crucial to strike a balance between the number of indexes and the frequency of data modification in your database.

This article will guide you through the process of creating, modifying, and deleting indexes using the DBeaver. It will also discuss the restrictions you need to be aware of when working with indexes, such as the impact on performance, storage considerations, and restrictions related to null values and certain data types.

Create

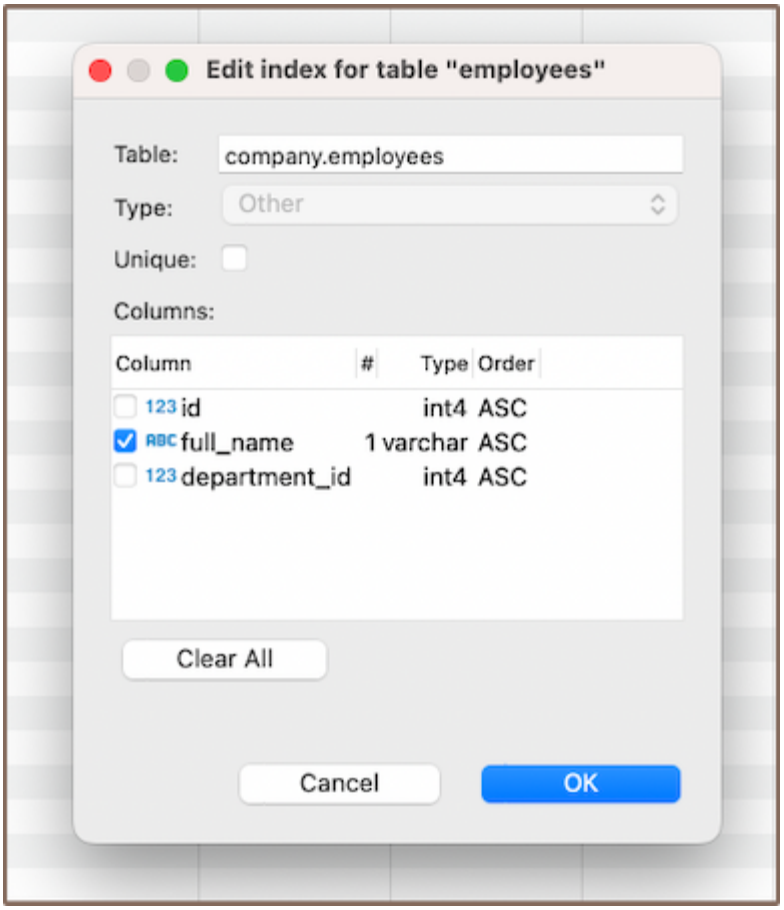
1) You can start the process of creating a new index in three ways:

- Using the Properties editor: Open the Properties Editor and navigate to the **Indexes** tab of the corresponding table.



- Using the Database Navigator: Navigate to the database where the table you want to add a index to is located. Find and select the table for which you want to create a new index. Within the selected table, you'll find a folder named **Indexes**.
- Additionally, to create a new index, you can utilize the **Create New Index** button located at the bottom of the **Properties editor**.

2) Depending on the method you chose in step 1, either right-click on the window (in the **Properties Editor**) or within the **Foreign Keys** folder (in the **Database Navigator**), and select **Create New Index**. If you're using the third method, simply click the **Create New Index** button. This will open a new window where you'll be able to customize the index's settings to suit your needs.



Field	Description
Unique	If set to unique, the index ensures that all entries in the indexed column(s) are distinct. This can limit data entry if values must be unique.
Columns	This setting allows you to select one or more columns for the index.

Note: In addition to creating a simple index on a single column, DBeaver also provides the option to create a composite index. A composite index is an index that is built on several table columns. This type of index can significantly boost query execution speed when dealing with multiple-column conditions. It's recommended to put unique fields at the end of the composite index to maximize its efficiency.

3) After you've chosen the column(s) for your index and configured any additional settings, you can finalize the creation of the index by clicking **OK**.

Saving changes

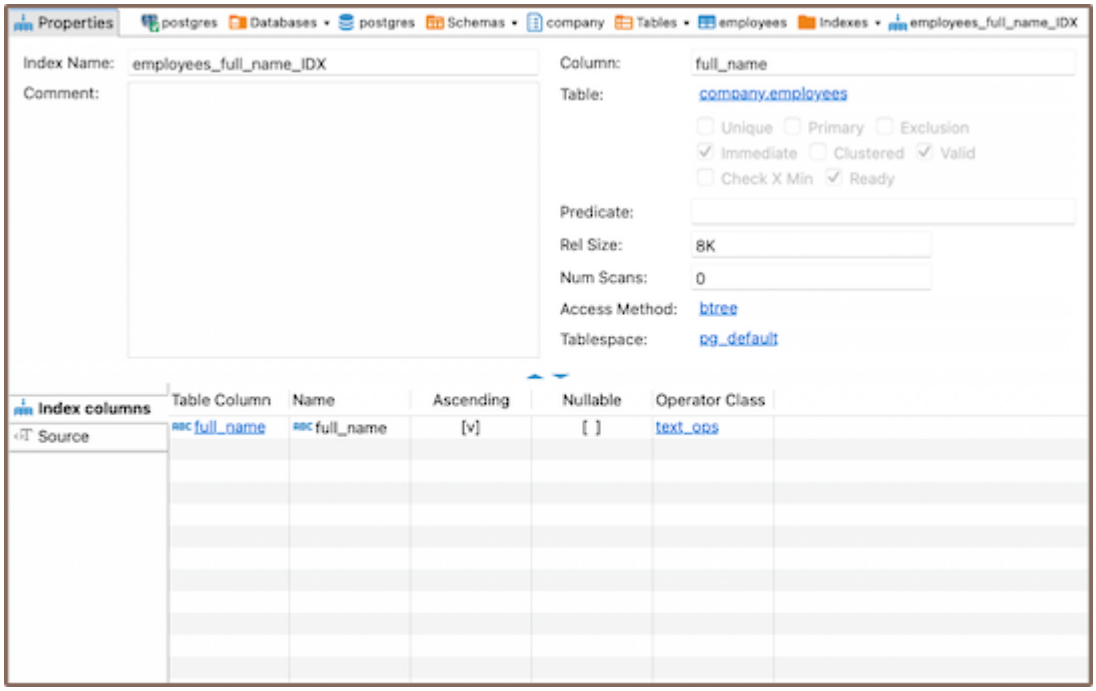
Once you have configured the index properties, it is essential to save these changes to the database. Until you commit your modifications, the new index will only exist within DBeaver and will not be added to the actual database table. Here are three options for committing the changes:

- Click on **File** -> **Save** -> **Persist**.
- Select the desired table in the **Database Navigator** and press `Ctrl+S` (or `CMD+S` for Mac OS), choose **Persist** to save the changes.
- Utilize the **Save** button Save ... located at the bottom of the **Properties editor** and press **Persist** to save the changes.

Modify

As your database evolves, there might be a need to modify the properties of your indexes. This could involve changing the indexed columns, adjusting the uniqueness constraint, or renaming the index. DBeaver provides the flexibility to make these modifications as required. Here's the process:

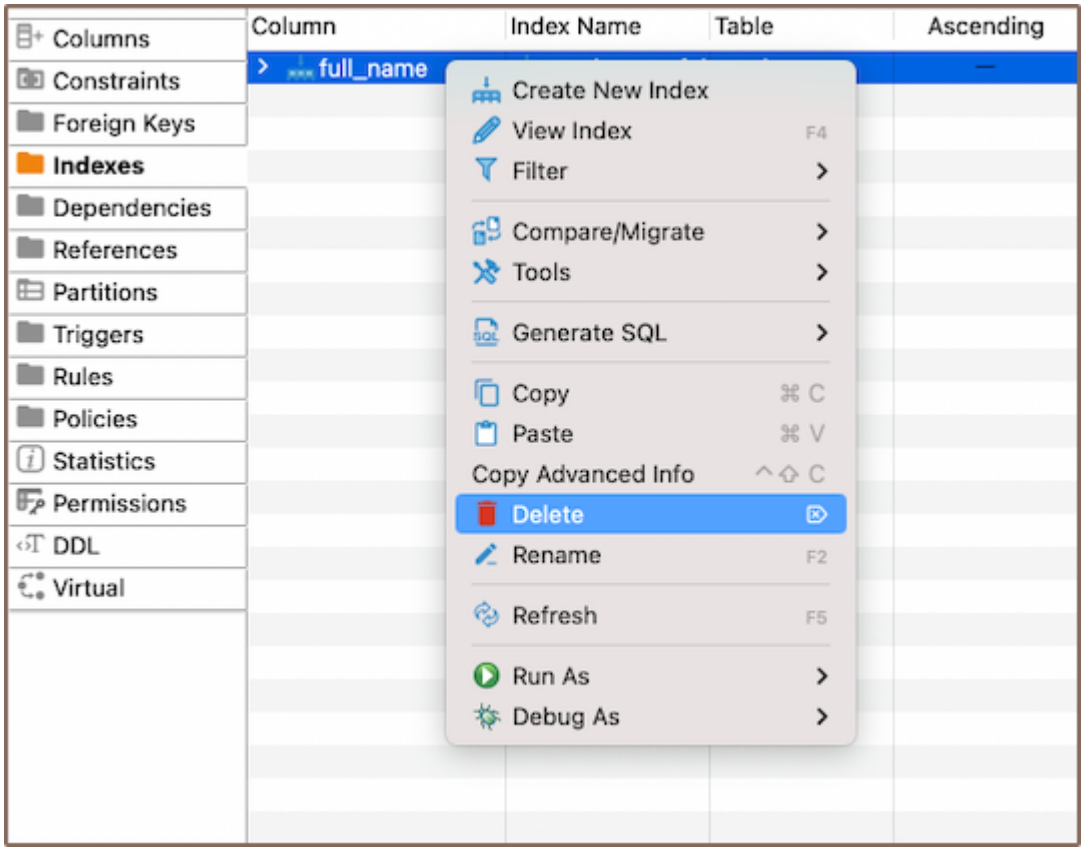
- **Properties Editor**: Open the **Properties Editor** by double-clicking on the table name. Within the selected table, you'll find a folder named **Indexes**. Right-click on the index you want to modify and select **View Index**.
- **Database Navigator**: In the Database Navigator, select the appropriate database and table. Within the selected table, you'll find a folder named **Indexes**. Right-click the index and select **View Index**.
- **Bottom Menu**: In the **Properties Editor** interface, use the **View Indexes** button  to open the properties of the selected index in the **Editor panel**.



Delete

1) You can delete an index either through the **Properties Editor** or the **Database Navigator**:

- Using the **Properties Editor**: Open the **Properties Editor**, navigate to the **Indexes** tab of the corresponding table, and find the index you want to delete.



- Using the **Database Navigator**: Navigate to the database where the table with the index is located. Open the **Indexes** folder, and find the index you want to delete.

2) To delete an index, right-click on the index's name and select **Delete**, or you can select the necessary index and press the `Delete` key, or **Delete button** (in the **Bottom Menu** of the **Properties Editor**). 3) A confirmation dialog will appear, asking you to confirm the deletion. Before proceeding, ensure that you've selected the correct index for deletion. 4) After confirming the deletion, you'll need to persist the changes to apply them to the database.

Restrictions

- Unique indexes created due to a primary key or unique constraint can't be deleted. Instead, remove the constraint. This action removes the uniqueness requirement for column values involved in the constraint expression and deletes the corresponding unique index.
- **Unique Indexes**: These enforce uniqueness on the indexed column(s), which can limit data entry if values must be distinct.
- **Performance Impact**: Indexes can improve query speed, but they slow down data modification operations (`INSERT` , `UPDATE` , `DELETE`) as the index must also be updated.
- **Storage**: Indexes consume disk space. This might become significant if there are many indexes or the table is large.
- **Null Values**: Depending on the database system, there may be restrictions related to indexing null values.
- **Data Types**: Certain data types might not be indexable or may require specific index types.

Further Reading:

New Table Creation | Creating columns | Implementing Constraints | Utilizing foreign keys | Creating indexes | Incorporating triggers

Incorporating Triggers

This article focuses on triggers, essential components in database management. Triggers are procedures that are automatically executed in response to certain events in a database. They are integral to the creation and manipulation of tables, enforcing constraints, and automating tasks.

The process involves creating a function, which is then used in a trigger. For example, a function can check if a new salary value is negative and raise an exception. This function can be used in a trigger that runs before any `INSERT` or `UPDATE` operations on a salary table.

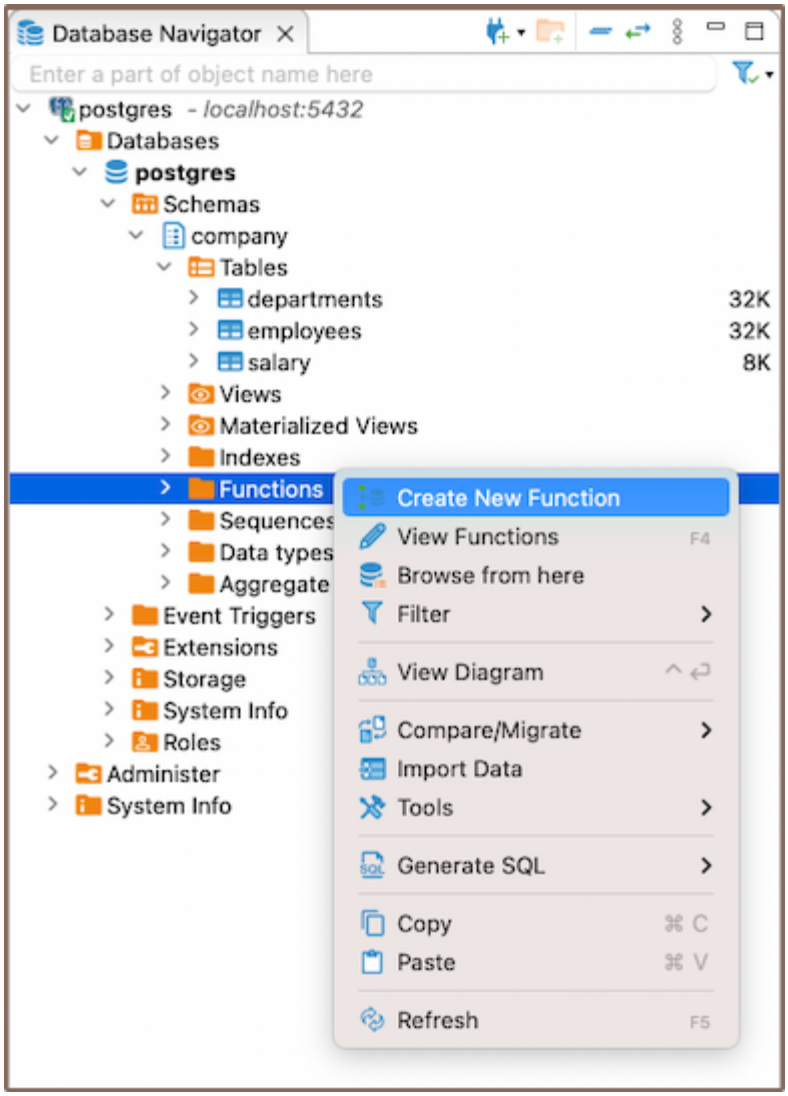
However, triggers come with restrictions and can impact performance, so their use should be optimized. The behavior of triggers can vary based on the database system in use, so always refer to the specific documentation for your system.

Create

Function

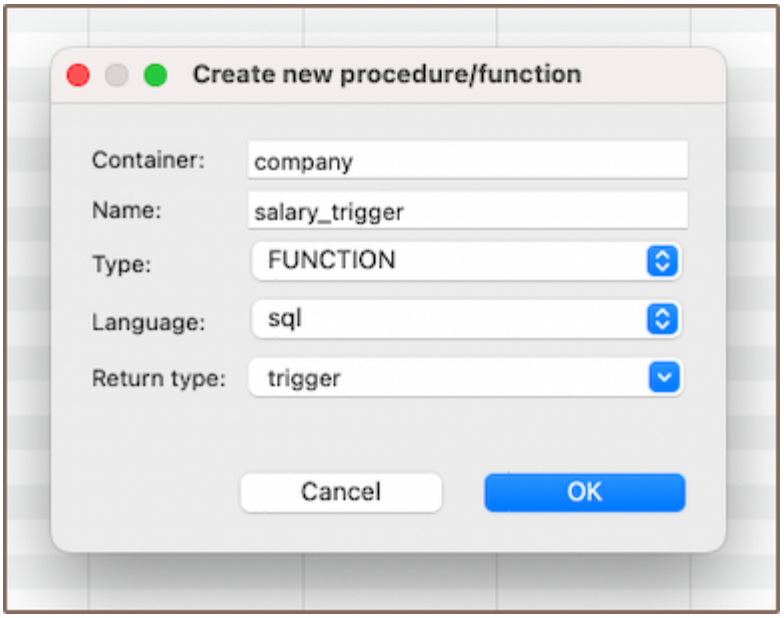
1) To create a trigger, start by creating a function. Here are three ways you can do it:

- Open the Properties editor of the database and navigate to the **Functions** tab of the corresponding database.
- Navigate to the database where you want to add a function in the Database Navigator. Within the selected database, you'll find a folder named **Functions**.



- Additionally, to create a new function, you can utilize the **Create New Function** button located at the bottom of the **Properties editor**.

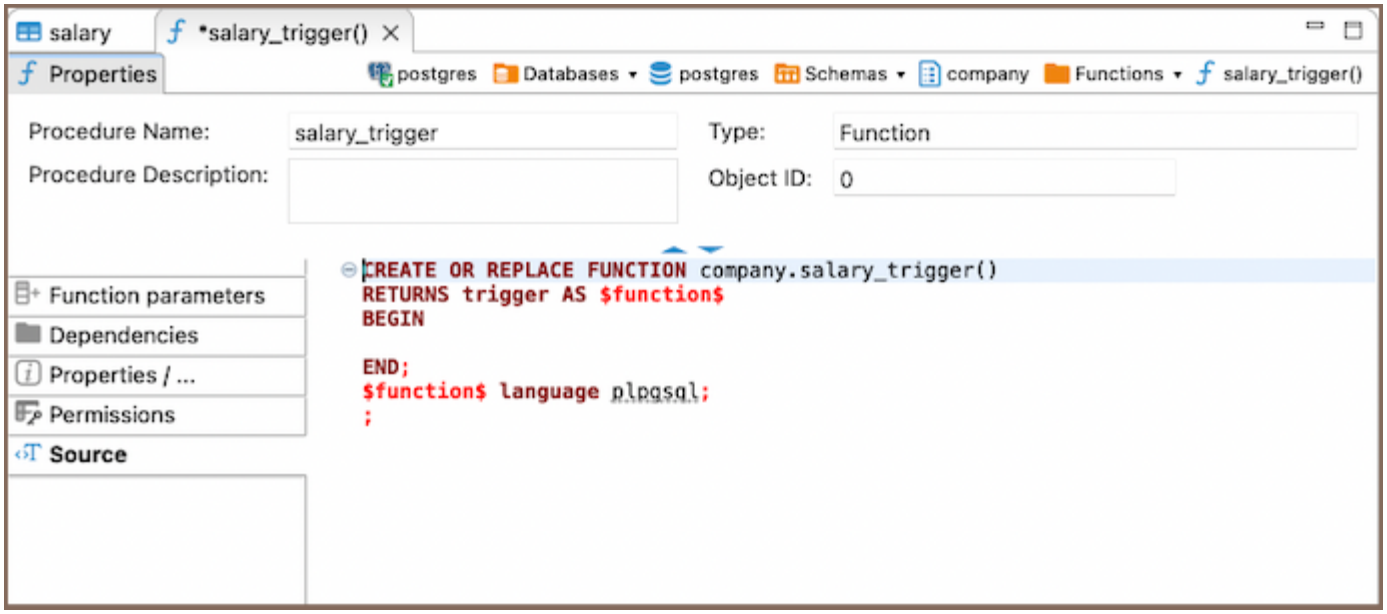
2) Depending on the method you chose in step 1, either right-click on the window (in the **Properties Editor**) or within the **Functions** folder (in the **Database Navigator**), and select **Create New Function**. If you're using the third method, simply click the **Create New Function** button. This will open a new window where you'll be able to customize the function's settings to suit your needs.



Field	Description
Function Name	This is the name you choose for the function. It should be unique within the schema where the function is being created.
Type	In this case, you would select **Function**.
Language	This is the programming language in which the function is written. For most database systems, this would be SQL or a variant of SQL, such as PL/pgSQL for PostgreSQL or PL/SQL for Oracle.
Return Type	This specifies the type of value the function will return. For a trigger function, this is usually specified as **trigger**.

Note: These settings can vary depending on the specific database system you're using. Always refer to the documentation for your database system for the most accurate information.

3) After you've chosen the needed settings for your function, you'll need to write the actual code for the function. This is done in the **Source** section.



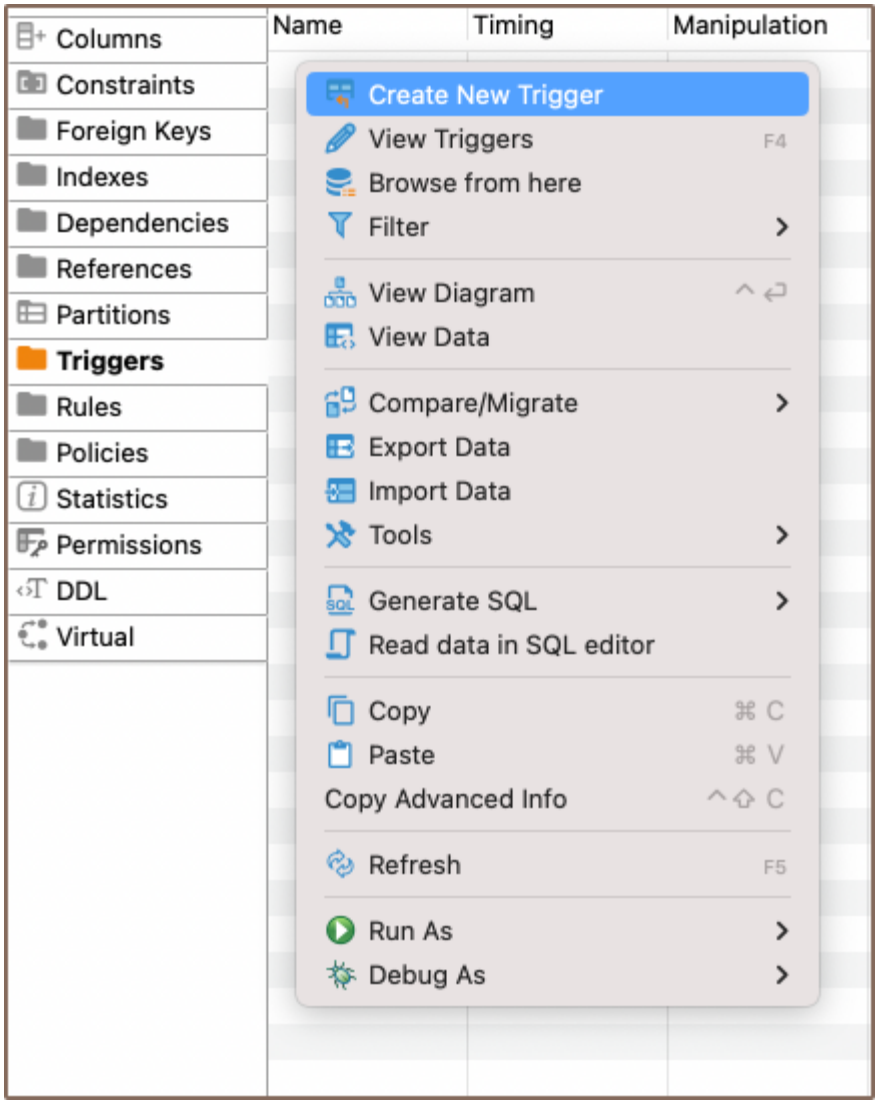
This function will be called when the trigger is activated. The function you write will depend on what you want the trigger to do. For example, if you want the trigger to check if a new salary value is negative and raise an exception if it is, you might write a function like this:

```
CREATE OR REPLACE FUNCTION company.salary_trigger()
RETURNS trigger AS $function$
BEGIN
    IF NEW.salary_amount < 0 THEN
        RAISE EXCEPTION 'Error: Salary cannot be negative';
    END IF;
    RETURN NEW;
END;
$function$ language plpgsql
```

Trigger

1) Creating a trigger is a process that mirrors the creation of a function. Once your function is defined, you can move on to create the trigger. This can be done through the **Properties Editor** or the **Database Navigator**.

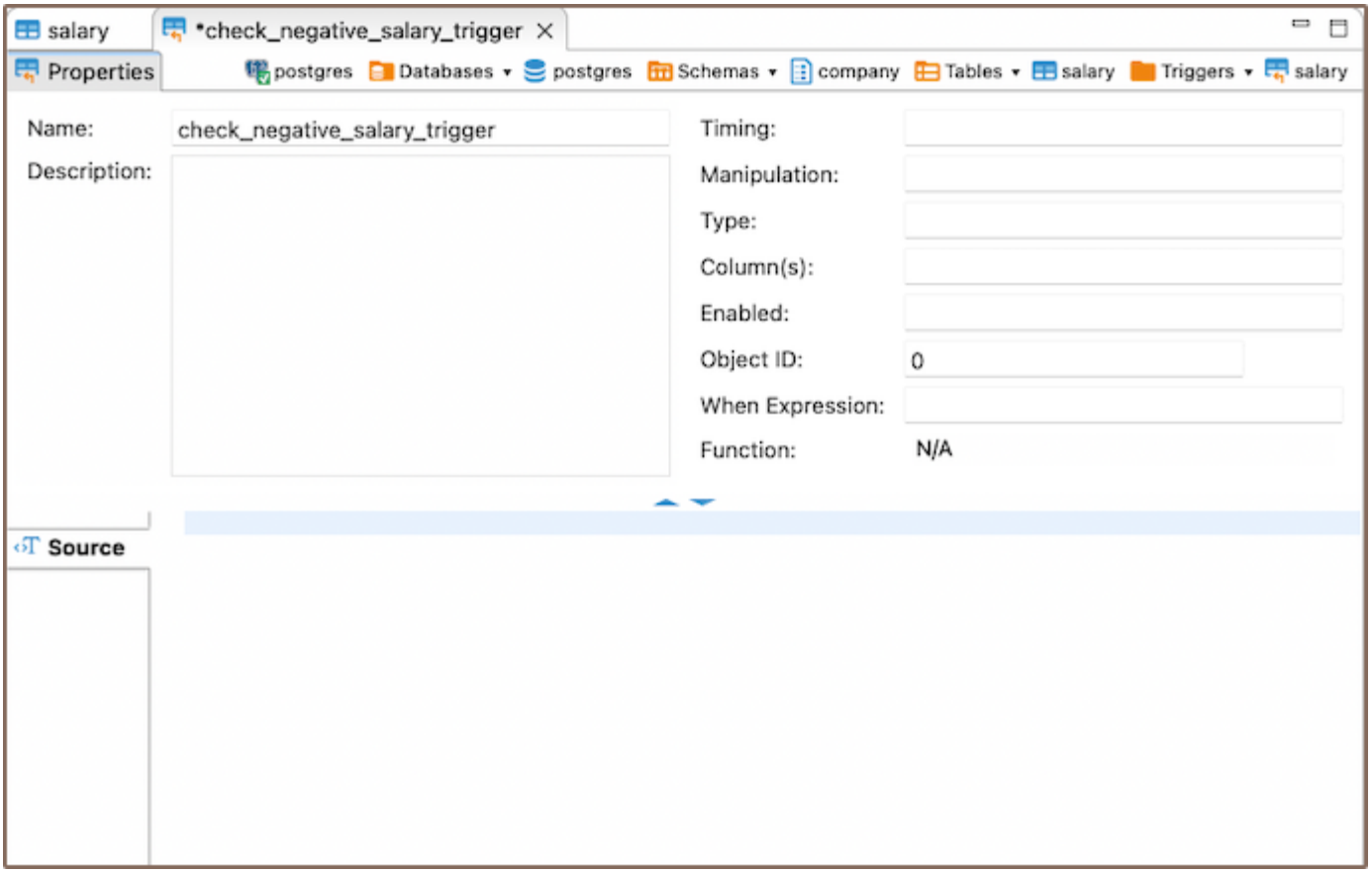
- In the **Properties Editor**, you can navigate to the **Triggers** section associated with the table for which you want to create the trigger, right-click on the window and select **Create New Trigger**. Here, you can also select **Create New Trigger** button  from the **Bottom menu**.



- Similarly, in the **Database Navigator**, you can right-click on the relevant table, select **Triggers**, and then choose **Create New Trigger**.

2) Next, you need to provide a name for the trigger and specify the function that it should call when it's activated. The trigger's function is typically written to perform a specific task when certain conditions are met. For example, if you've written a function to check if a new salary value is negative, you can create a trigger that calls this function before any `INSERT` operations on the salary table.

3) Finally, describe the code for the trigger.



For instance:

```
CREATE TRIGGER check_negative_salary_trigger
BEFORE INSERT
ON company.salary
FOR EACH ROW
EXECUTE function company.salary_trigger();
```

In this example, a `salary_trigger()` function is created that evaluates the new salary value (NEW.salary). If the salary is negative, it throws an exception with an error message. Then, a `check_negative_salary_trigger` is established to run this function before any `INSERT` operations on the "salary" table. Should the salary value be negative, an exception is triggered and the operation is stopped.

Saving changes

Once you have configured the trigger and function properties, it is essential to save these changes to the database. Until you commit your modifications, the new trigger and function will only exist within DBeaver and will not be added to the actual database. Here are three options for committing the changes:

- Click on **File** -> **Save** -> **Persist**.

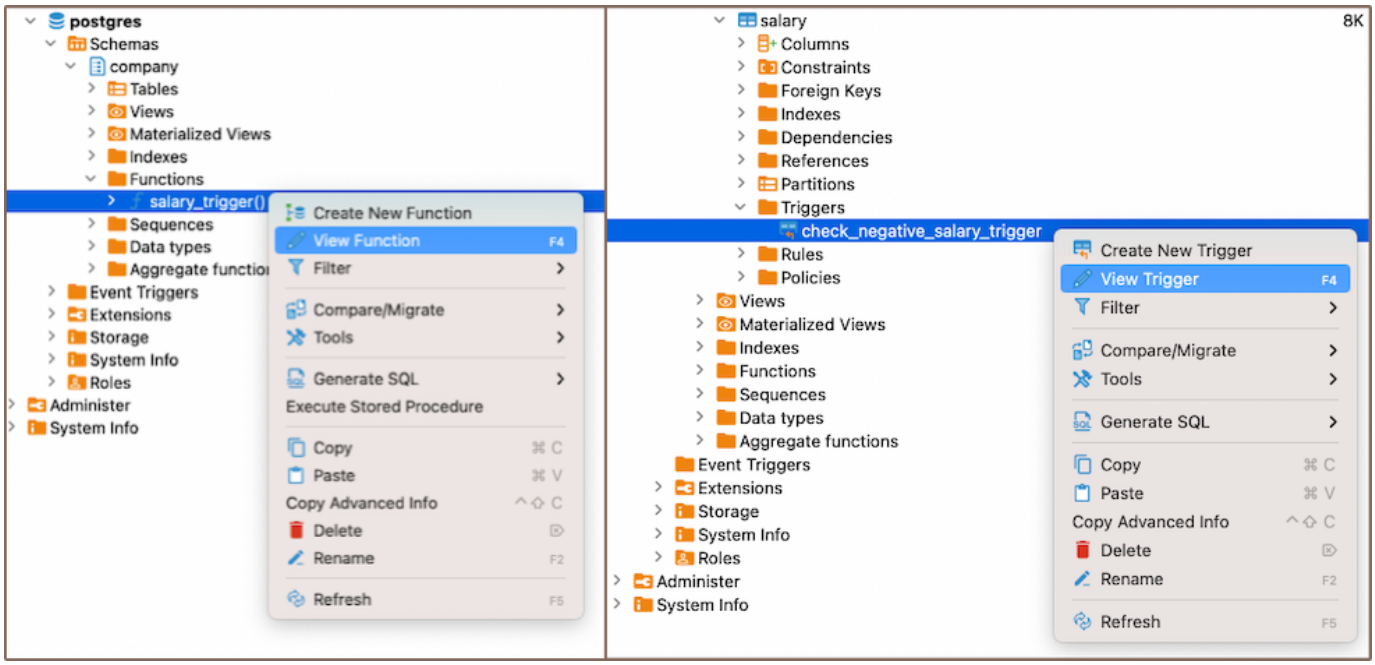
- Select the desired table in the **Database Navigator** and press `Ctrl+S` (or `CMD+S` for Mac OS), choose **Persist** to save the changes.
- Utilize the **Save** button Save ... located at the bottom of the **Properties editor** and press **Persist** to save the changes.

Modify

In the realm of database management, the need to modify existing functions or triggers is a common occurrence. Whether it's to correct an error, optimize performance, or adapt to new requirements, the ability to modify these elements is crucial. DBeaver offers intuitive and efficient ways to make these modifications.

The DBeaver interface provides multiple pathways to access and modify functions or trigger:

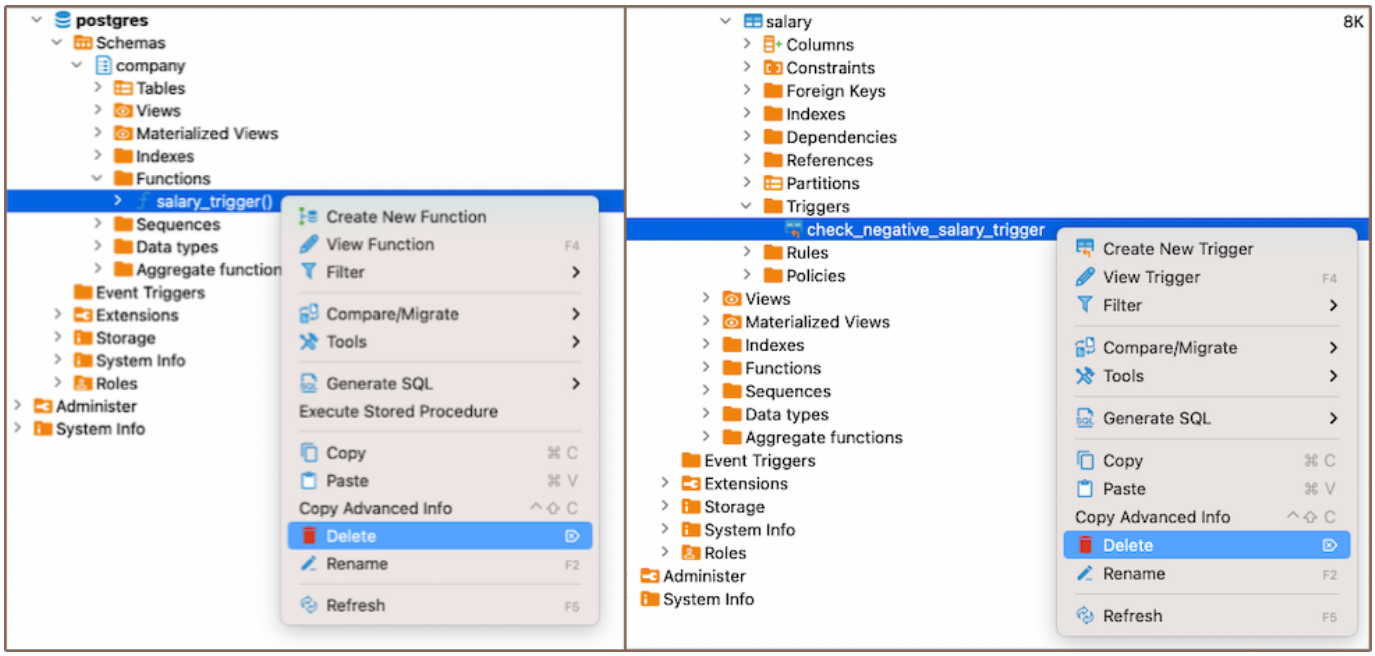
- **Properties Editor**: Open the **Properties Editor** by double-clicking on the database name (for functions) and table name (for triggers). Within the selected database or table, you'll find a folder named **Triggers** or **Functions**. Right-click on the trigger or function you want to modify and select **View Trigger** or **View Function**.
- **Bottom Menu**: In the **Properties Editor** interface, use the **View Indexes** or **View Function** button  .
- **Database Navigator**: In the **Database Navigator**, select the appropriate database or table. Within the selected database or table, you'll find a folder named **Triggers** or **Functions**. Right-click the trigger or function and select **View Index** or **View Function**.



Delete

You can delete an index either through the **Properties Editor** or the **Database Navigator**:

- Using the **Properties Editor**: Open the Properties Editor, navigate to the **Triggers** or **Functions** tab of the corresponding database (for functions) and table (for triggers), and find the trigger you want to delete.
- Using the **Database Navigator**: Navigate to the database where the table with the trigger or function is located. Open the **Triggers** or **Functions** folder, and find the trigger or function you want to delete.



2) To delete the trigger or function, right-click on the trigger's or function's name and select **Delete**, or you can select the necessary trigger or function and press the `Delete` key, or **Delete button** (in the **Bottom Menu** of the **Properties Editor**). 3) A confirmation dialog will appear, asking you to confirm the deletion. Before proceeding, ensure that you've selected the correct trigger or function for deletion. 4) After confirming the deletion, you'll need to persist the changes to apply them to the database.

Restrictions

- **Recursive Triggers**: Some databases allow recursive triggers, where a trigger can call itself. This can lead to infinite loops and excessive system load if not handled correctly.

- **Data Modification**: Triggers can't be used to modify a table that is already being used (for reading or modifying) in the statement that invoked the function or trigger.
- **Execution Order**: The order in which multiple triggers are fired isn't guaranteed. This can lead to unexpected results if triggers have interdependencies.
- **Performance**: Triggers can slow down data modification operations because they add extra processing. They should be used sparingly and optimized for performance.
- **Transaction Control Statements**: In many SQL databases, transaction control statements (like COMMIT and ROLLBACK) are not allowed within trigger code. However, exceptions apply depending on the database system.

Further Reading:

New Table Creation | Creating columns | Implementing Constraints | Utilizing foreign keys | Creating indexes | Incorporating triggers