SQLwallet Documentation

SQLwallet Ltd

Jan 10, 2019

Contents

This documentation is still under development. The new articles are being added regularly.

CHAPTER 1

Installation and upgrade

1.1 Windows Server - Standalone Service

This article explains how to install and run SQLwallet as a standalone Windows Service.

Prerequisites: Windows Server 2008 R2 or later (Windows 7 SP1 or later). No IIS required.

1. Download SQLwallet installation executable from the Downloads page and run the downloaded file.
2. Follow the installation wizard instructions. When asked to enter the server hostname and port, specify the hostname and port you will be running SQLwallet instance. Make sure the port is not used by any other applications. We suggest port 88 as the default one.

   Note: You can modify these settings later in appsettings.json file.

3. Once the installation finishes, a browser window will open and SQLwallet will start. On the first run you will be prompted to enter the admin user password.

If the browser window does not open automatically, try to launch SQLwallet site manually by navigating to http://hostname:port in the browser, for example http://localhost:88.

In case SQLwallet website fails to start, please make sure that SQLwallet service is running. If the service fails to start, check the application and system logs under Windows Event Viewer and the logs folder under SQLwallet installation folder, that can help you with the troubleshooting.

To upgrade, run the setup file and follow the instructions. Make sure to always check the special upgrade instructions and information of any manual steps and breaking changes in the Release Notes.

1.2 Windows Server - IIS

This article explains how to install and run SQLwallet under Internet Information Services on Windows machines.

Prerequisites: Windows Server 2008 R2 or later (Windows 7 SP1 or later), with IIS installed.

1. Download and install .NET Core 2.x Runtime & Hosting bundle. If you already have this installed, skip this step.
2. Download SQLwallet ZIP archive from the Downloads page and extract its content into a local folder.

   Note: Makes sure you unblock the archive before unpacking it!

1. In IIS manager, create a new website pointing to the folder with SQLwallet files and change the Application Pool to "No Managed Code". For the details, please refer to the article on Microsoft website.
2. Grant Modify permissions to the app pool identity user ("IIS AppPool\<your_apppool_name>") for the SQLwallet root folder.
3. Navigate to the website URL to check that SQLwallet website is up and running.

Note: When running SQLwallet under IIS, leave ServerHostName and ServerPort parameters empty in appsettings.json file.

To upgrade, follow the step 2 above and copy the ZIP archive content over the existing files. If there are any locked files preventing the copying, stop the IIS service before the upgrading.

1.3 Azure App Service

This article explains how to install and run SQLwallet as an Azure App Service.

Prerequisite: you should have an active Azure subscription.

1. On Azure Portal create a new "Microsoft Web App" app service. We recommend to use at least S2 pricing tier (3.5 GB memory).
2. Download SQLwallet ZIP archive from the Downloads page and extract its content into a local folder on your computer. Makes sure you unblock the archive before unpacking it.
3. Deploy the content of the folder to your App Service using FTP (here is how).
4. Navigate to the web app URL to check that SQLwallet website is up and running. If the service does not start, use this troubleshooting guide.

Note: When running SQLwallet as an Azure App Service, leave ServerHostName and ServerPort parameters empty in appsettings.json file.

To upgrade, follow the steps 2 - 4 above and upload the ZIP archive content over the existing files. If there are any locked files preventing the copying, stop the Azure App Service service before the upgrading. Alternatively, you can use deployment slots technique.

CHAPTER 2

Creating reports

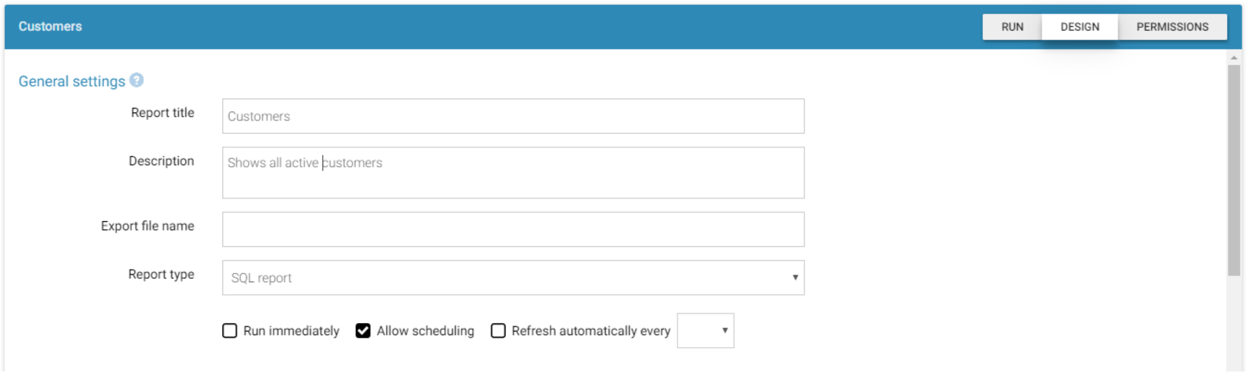
To create a new report, click “New Report” button when you are in the root or in any folder in the navigation tree (if you don’t see this button, that means you are not allowed to create new reports).

To modify a report, click “Design” button in the report header (if you don’t see this button, that means you are not allowed to change this report).

Design report sections:

2.1 General

This section describes usage of the general report settings.



Report title: Customers
Description: Shows all active customers
Export file name:
Report type: SQL report
☐ Run immediately ☑ Allow scheduling ☐ Refresh automatically every

- **Report title** - specifies how the report will appear in the navigation tree.
- **Description** - short report description that will appear above report parameters. Use this to provide instructions or clarify the report purpose for the users.
- **Export file name** - here you can specify the file name for export into Excel or CSV. You can use macro parameters (see below) in this field, to specify date of the export or other dynamic values. If not specified, the report tile will be used as the export file name.

- **Run immediately** - if enabled, the report will be executed on screen immediately, otherwise the user will need to click Run button to execute the report. Use this option if the report does not have mandatory parameters and if the report execution time is quick.
- **Report type - specifies type of the report:**
    - **SQL report** - the report is based on SQL query. This is the most common report type.
    - **External page** - the report is hosted on an external website and will be shown in IFRAME. You can send parameters from SQLwallet into the report webpage. Use this option if you want quickly add your legacy reports into SQLwallet portal.

Allow scheduling - if enabled, the authorized users will be able to schedule the report. * **Refresh automatically** - if enabled, the report on screen will be auto-refreshed on the specified intervals of time.

2.1.1 Examples of macro parameters that can be used in the Export file name

Macro	Meaning
${now}	Current date/time (at the moment of the export, e.g. 23/12/2018 17:45)
${yesterday}	Yesterday's date (e.g. 22/12/2018)
${today:yyyy-MM-dd}	Today's date formatted (e.g. 2018-12-23)
${param.DateFrom}	Value of report parameter "DateFrom" (e.g. 23/12/2018)
${param.DateFrom:yyyy-MM-dd}	Value of report parameter "DateFrom", formatted (e.g. 2018-12-23)

2.2 Parameters

2.2.1 Usage of parameters

Parameters (filters) are used to get capture user input into variables, that can be used in the query.

The following parameter types are supported:

Parameter Type	Multiple values?	How to use in the report SQL query
String	No	SELECT * FROM employees WHERE username = $filter OR full_name LIKE '%'+$FILTER+'%';
Number	No	SELECT * FROM employees WHERE age > $age;
DatePicker	No	SELECT * FROM employees WHERE startDate BETWEEN $dateFrom AND DATEADD(DAY,1,$dateTo);
DateTimePicker	No	SELECT * FROM timeLog WHERE entry_time > $timeFrom;
Checkbox	No	SELECT * FROM employees WHERE is_active = $isActive;
Radiobutton	No	SELECT * FROM employees WHERE employee_type = $type;
Textarea	Yes	SELECT * FROM employees WHERE ID IN ($ListOfIds);
CheckboxList	Yes	SELECT * FROM employees WHERE office_id IN ($office);
Lookup	Yes	SELECT * FROM employees WHERE office_id IN ($office);
Dropdown	Yes	SELECT * FROM employees WHERE office_id IN ($office);

Parameters can be used in the query in two ways, either as variables (which is the most common use), or as macros.

Variables

Variables are prefixed by $ symbol in your SQL query (you do not need specify $ sign in the parameter name when you create a parameter). When using a variable, SQLwallet will automatically converts its type, for example DatePicker parameter variable can be used as a regular DATETIME variable.

```
SELECT * FROM Orders
WHERE orderDate BETWEEN $DateFrom AND DATEADD(DAY,1,$DateTo);
```

Macros

You also can use the parameters as macros by prefixing a parameter name with $$. Macros give you an ability to dynamically construct virtually any SQL query. All macros will be substituted in the query text “as is”. The advantage of macros is that you can use them as any part of SQL, even if the native SQL does not allow use of variables.

Consider a case when you need to switch to the different schema or database dynamically, depending on user input. All you need to do is to create a drop-down parameter with name “database”, which contains names of the databases as the options, and use it as a macro:

```
USE $$database;
```

2.2.2 Optional parameters

If you specify a parameter as optional, you will need to add the corresponding logic into your SQL query.

Example 1. If the user does not enter the customer name (optional parameter $CustName), the report will return all customers, otherwise only the customers with this name.

```
SELECT * FROM Customers
WHERE ($CustName IS NULL OR Customers.Name = $CustName);
```

Example 2. If the user does not enter the start date (optional parameter $DateFrom), the report will return all invoices for the last 1 month by default.

```
SELECT * FROM Invoices
WHERE InvoiceDate >= ISNULL($DateFrom, DATE_ADD(MONTH, -1, GETDATE());
```

2.2.3 Multi-value parameters

There is a special case for parameters that allow selection of multiple values, they are supposed to be used with IN clause:

Please note that in order to use the syntax above, your query must have type “Query” and not “Stored procedure”. If you query type is “Stored procedure”, then multiple valued parameters will be passed as comma-delimited string. This is required if you want to pass multiple values into a stored procedure, as it is the only way in SQL dialects that do not support arrays.

Query usage:

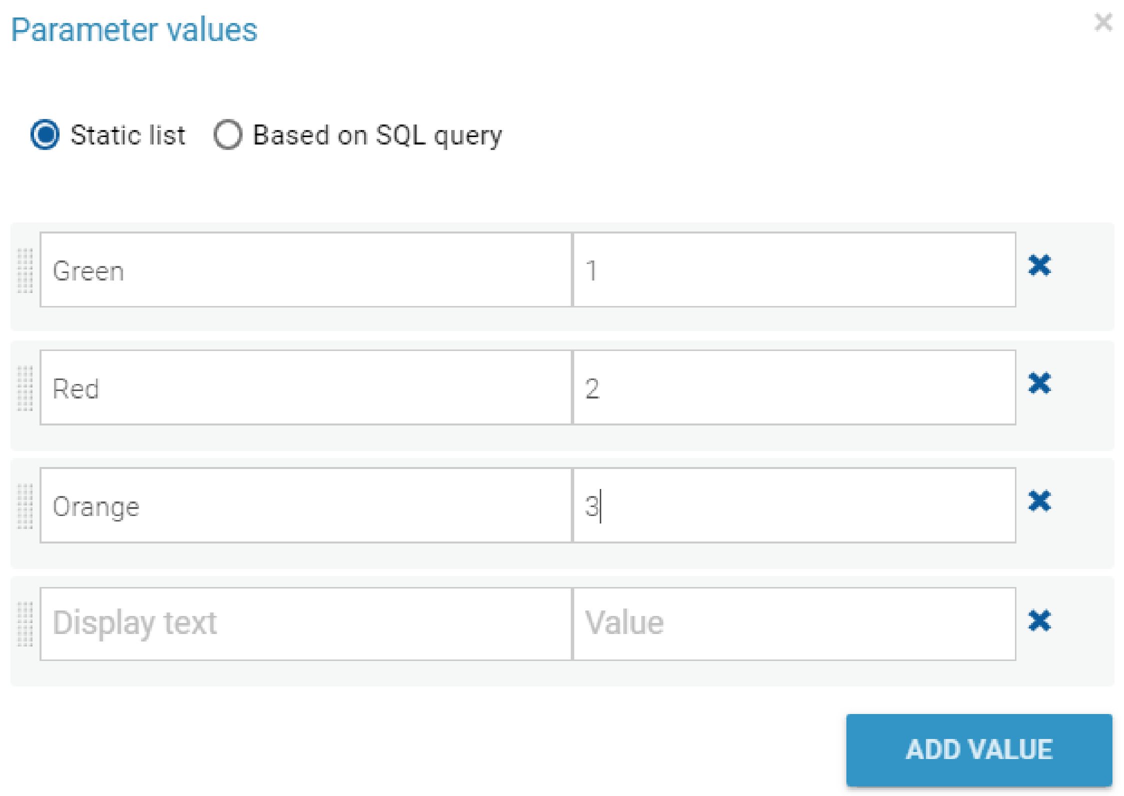
```
SELECT * FROM employees WHERE office_id IN ($office);
```

Stored procedure usage:

```
EXEC spGetEmployessFromOffices($office);
```

Specifying possible values

When you create a multivalue parameter, you need to specify list of possible values (except of Textarea parameter, which we'll disuss later). You can do it in two ways: either specifying static list of values, or providing an SQL query, that will populate the list of values from database.

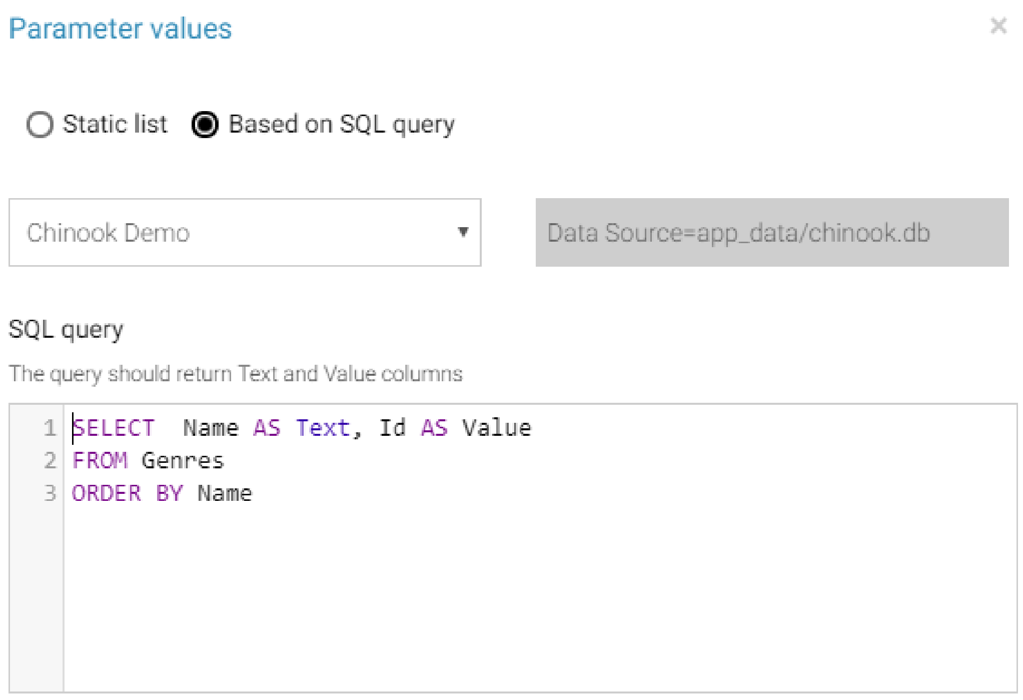


Parameter values

[✓ Static list] Based on SQL query

Green	1
Red	2
Orange	3
Display text	Value

If you are using an SQL query, make sure it returns two columns named "Text" and "Value". The Value will be passed into the parameter variable, and the Text will be shown to the user.



Parameter values

[ ] Static list [✓ Based on SQL query]

Chinook Demo

Data Source=app_data/chinook.db

SQL query

The query should return Text and Value columns

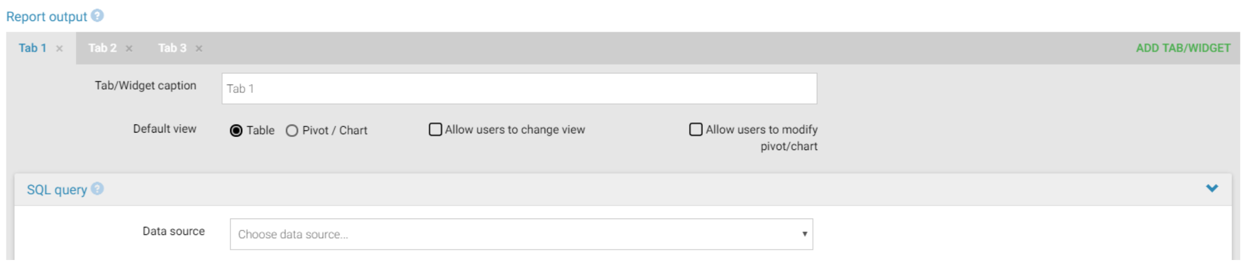
```
SELECT  Name AS Text, Id AS Value
FROM Genres
ORDER BY Name
```

The parameter query can have a different data source than the main report output query.

You can use macros in the parameter query, referring the other parameters. This is useful in scenarios, where the parameter's list of possible values depends on user selection of another parameter:

```
SELECT  Name AS Text, Id AS Value
FROM Genres
WHERE CategoryId = $Category
ORDER BY Name
```

2.3 Report output



Tab/Widget caption: Tab 1

Default view: [✓ Table] Pivot / Chart

[ ] Allow users to change view

[ ] Allow users to modify pivot/chart

SQL query

Data source: Choose data source...

The report output consists of one or more tabs/widgets. Each tab has its own data source SQL query, output columns and editing options. The SQL queries in all tabs has access to the same set of report parameters.

When users run the report they can rearrange the tabs to be shown alongside each other, like widgets in a dashboard.

When the report is exported as Excel, the tabs will be exported as separate sheet within the Excel file. When the report is exported as CSV, each tab will be exported a separate CSV file, and all the files will be packed into a ZIP archive.

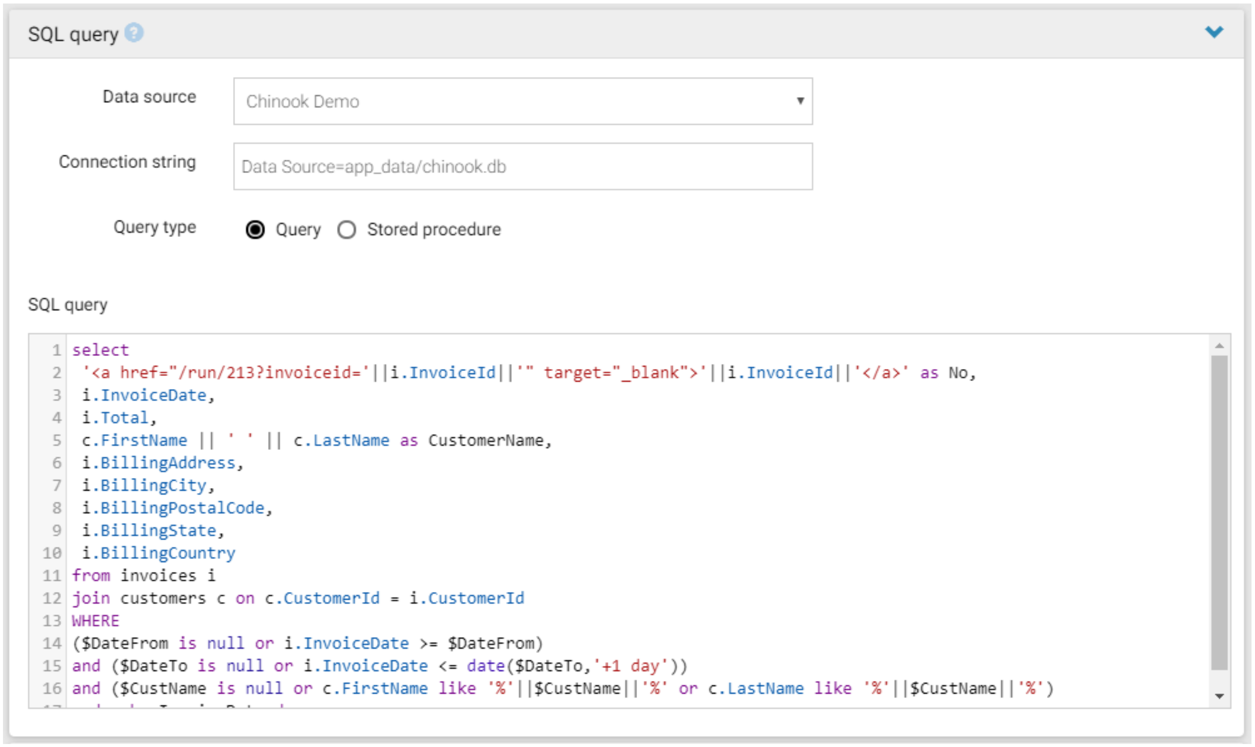
- **Tab/Widget caption** - the title of the tab/widget. If the report has only single tab, the title will not be shown.
- **Default view** - specifies the default view of the report, either flat table, or pivot/chart for data visualization.

- **Allow users to change view** - if enabled, the users will be able to switch between Table view and Pivot/Chart view.
- **Allow users to modify pivot/chart** - if enabled, the pivot/chart view will become interactive, allowing the users to manipulate the data aggregates and build their own data visualization.

Under each tab there are the following sections:

- *SQL query* - contains the data source and the SQL query, used to produce the report output.
- *Output columns* - contains definition of output columns (optional)
- *Data editing* - allows to enable data editing within the report output.

2.4 SQL query



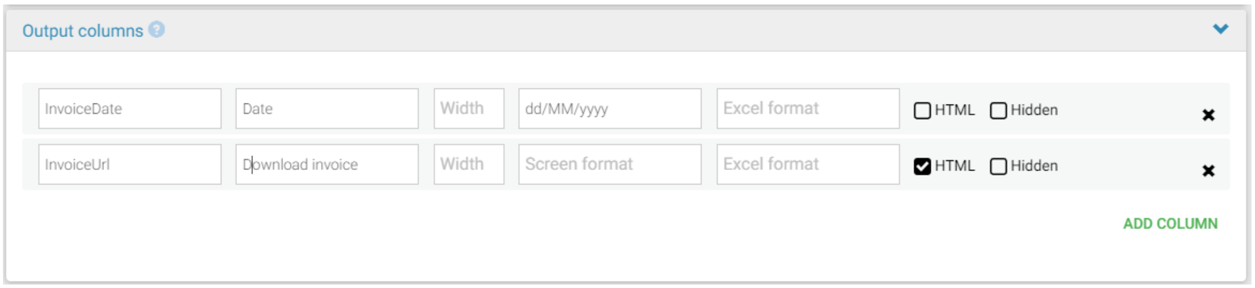
```sql
select
 '<a href="/run/213?invoiceid='||i.InvoiceId||'" target="_blank">'||i.InvoiceId||'</a>' as No,
 i.InvoiceDate,
 i.Total,
 c.FirstName || ' ' || c.LastName as CustomerName,
 i.BillingAddress,
 i.BillingCity,
 i.BillingPostalCode,
 i.BillingState,
 i.BillingCountry
from invoices i
join customers c on c.CustomerId = i.CustomerId
WHERE
($DateFrom is null or i.InvoiceDate >= $DateFrom)
and ($DateTo is null or i.InvoiceDate <= date($DateTo,'+1 day'))
and ($CustName is null or c.FirstName like '%'||$CustName||'%' or c.LastName like '%'||$CustName||'%')
```

- **Data source** - select the data_source (database connection) which will be used to run the SQL query. For your reference, the connection string for the selected data source will be shown below. The data sources are managed in the SQLwallet configuration section.
- **Query type** - select "Query" or "Stored procedure". See more about the difference between these two options in the Multi-value parameters section.
- **SQL query** - contains the SQL query used to retrieve the data from the database specified in the Data Source. You can use the report parameters variables and macros in the SQL query. For more details, please refer to the *Parameters* section.

Note: You can use multiple SQL statements, including declaration of variables, and T-SQL commands, such as IF, BEGIN/END, etc, if the SQL dialect supports it.

2.5 Output columns

By default SQLwallet will show all columns returned by the query, with default formatting. So you don't have to define output columns unless you want to override appearance of a specific column. You need only to add output columns for the columns you want to override.



Output column settings:

- **Column name** - this should match the exact name of the column as returned by the SQL query.
- **Display name** - use this to override the column name.
- **Width** - initial screen column width in percentage or pixels, for example 10%, or 200px.
- **Screen format** - use to format the output values on screen. See the available format patterns here.
- **Excel format** - use to format the output values when exporting into Excel/CSV, if different from the screen format. See the available format patterns here.
- **HTML** - enable if the column values contain HTML mark-up (for example hyperlinks) and you want it to be displayed as HTML.

  Note: Use HTML columns with caution, as HTML content can potentially be harmful. Make sure that the content you are displaying is safe.

- **Hidden** - enable this to hide the column from the output.
- **Editable** - enable this to allow users to edit the data in the column. Only visible is Data Editing is enabled (see *Data editing* section).
- **Importable** - enable this to allow users to import the data in the column from Excel or CSV. Only visible is Data Import is enabled (see *Data editing* section).

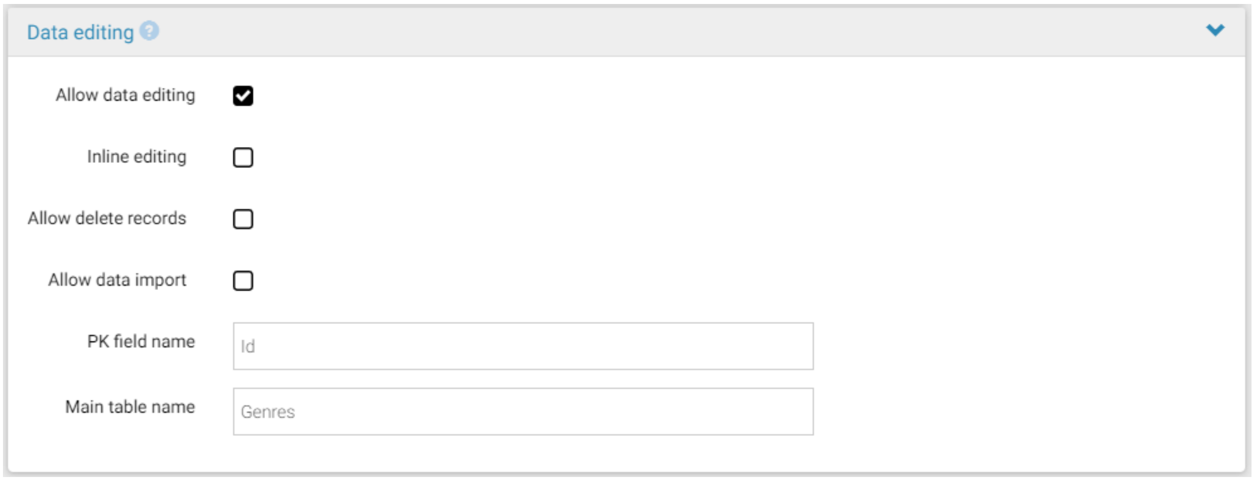
All output column fields may be left empty, except "Column name".

Note: You cannot change order of output columns in this section. If you need to change the order, you should modify the SQL query.

2.6 Data editing

2.6.1 Report settings

SQLwallet allows you to edit underlying data in the table reports, effectively turning a report into data editing app, that can be useful in scenarios when you need to quickly provide your customers with simple data editing pages or admin apps. The editing access can be limited to specific users or teams only in the *Permissions* section.



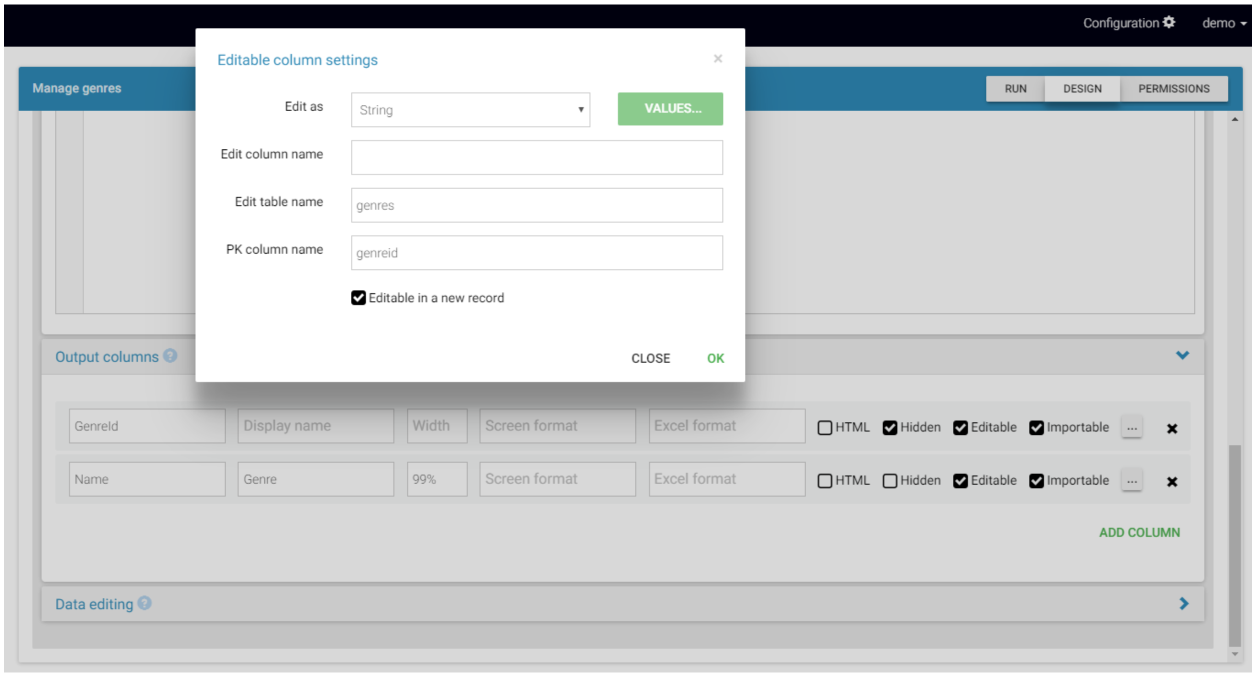
Data editing

Allow data editing	✓
Inline editing	☐
Allow delete records	☐
Allow data import	☐
PK field name	Id
Main table name	Genres

- **Allow data editing** - enables data editing for the report. After enabling it, you need to specify editable columns individually in *Column settings* section.
- **Inline editing** - if enabled, the users will be able to edit the values directly in table cells. Otherwise, a pencil icon will be displayed in each table row, and a popup window will be opened to edit the record.
- **Allow delete records** - if enabled, a Delete icon will be displayed in each table row. Deleting records can be limited to specific users or teams only in the *Permissions* section.
- **Allow data import** - if enabled, the users will be able to import data from Excel and CSV files to add new records or update existing records. Data importing can be limited to specific users or teams only in the *Permissions* section. Also you will need to specify which columns are allowed to be imported in Output Columns section.
- **PK field name** - specifies the primary/unique key column in the underlying table. This setting can also be set for each individual column. The column **must** be present in the SQL query.
- **Main table name** - specifies the name of the table in the database, that contains the data to be edited. This setting can also be set for each individual column.

2.6.2 Column settings

After you enable **Allow data editing** checkbox, you will need to define all editable columns under *Output columns* section.



- **Edit as** - specifies the input type (String, Dropdown, DatePicker, Checkbox, Textarea) for the column.
- **Values** - allows to specify list of possible values for dropdowns (see "Multi-value parameters" in the *Parameters* section).
- **Edit column name** - specifies the column name, which contains the data in the underlying table (if it is different from the column name returned by the report SQL query)
- **Edit table name** - specifies the name of the table in the database, that contains the data to be edited (if different from the **Main table name** field in the *Report settings*).
- **PK field name** - specifies the primary/unique key column in the underlying table (if different from the **PK field name** field in the *Report settings*). The column **must** be present in the SQL query.
- **Editable in a new record** - if enabled, then the column will be available for editing when the user creates a new record.

Note: You can hide primary key columns from the output using "Hidden" checkbox into Output Columns section.

2.7 Import and export

Sometimes you want to create a new report based on an existing one, or copy the report into another folder, or into another instance of SQLwallet.

Export

To export a report, switch into Design mode and click "Export" button at the bottom of the page. Then select a location on your computer to save the imported file.

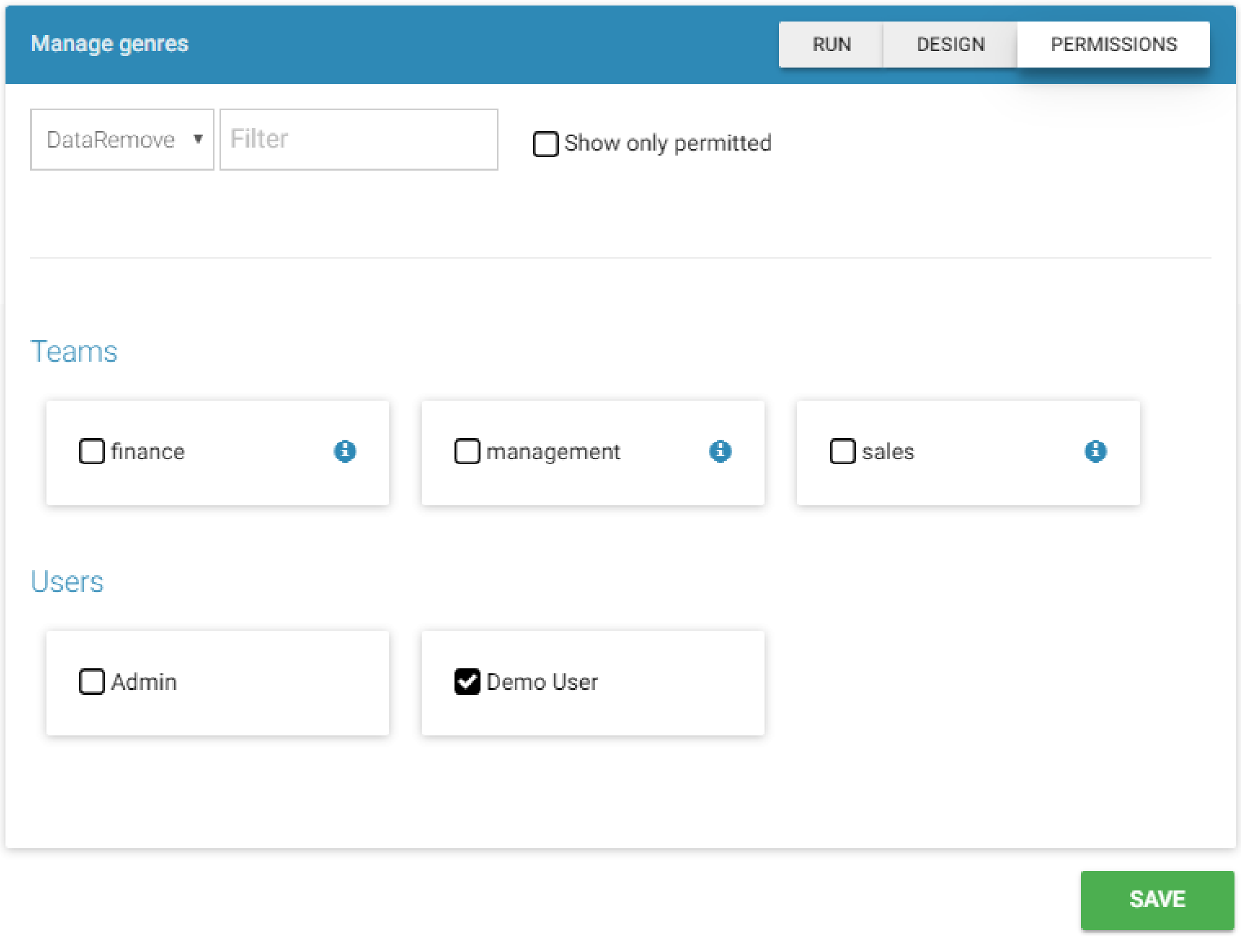
Import

To import a previously exported report:

1. In the navigation tree select the folder, where you want to place the new report, and click "New report button".
2. Click "Import from..." button at the bottom of the page and select the file to import.
3. Click "Confirm import" button.
4. Make any amendments if necessary, and click "Save".

Note: SQLwallet will try to match the data sources used in the exported report (in the main SQL query, or in multi-value parameters, or in editable columns), if such data sources exists. If data source with the same name does not exists, you will need to specify the new data sources before saving the exported report, otherwise you will receive an error message that the data source is missing.

2.8 Permissions



To grant specific permissions for a report to users or teams:

1. Click "Permissions" in the header of the report.

2. **Choose a type of permission in the upper left dropdown. The possible permissions are:**
    - **Run** - allows to execute the report.
    - **Design** - allows to modify the report.
    - **View** - allows to view the report definition (design view) but not to modify it.
    - **DataEdit** - for editable reports, allows to edit existing records.
    - **DataInsert** - for editable reports, allows to create new records.
    - **DataRemove** - for editable reports, allows to delete records (if data deletion is enabled in the report settings).
    - **DataImport** - for editable reports, allows to import records from Excel or CSV (if data import is enabled in the report settings).
    - **Grant** - allows user to manage permissions for this report. The users can only assign permissions that they have by themselves. For example, if a user has Run permission, they can grant the Run permission to other users, but cannot grant Design permission.
3. Select the users/teams you want to assign the permission. You can filter by name to quickly locate the user or team.
4. Repeat steps 2 - 3 for different permissions.
5. Click Save.

Note: You also can assign global permissions to users in the configuration section, such as Run All, View All etc. In this case you do not need to assign permissions to the user for each report separately.

Note: Use "i" icon next to the team name to quickly check the users, that are members of this team.

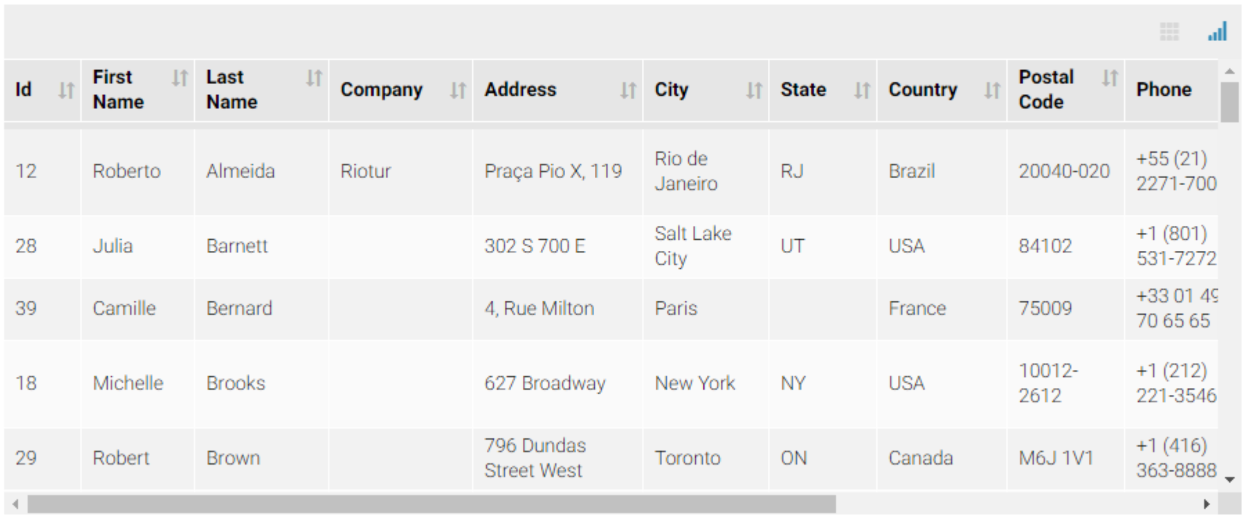
CHAPTER 3

Reports execution

3.1 Report views

The default view of the report output is specified in *Report output* section. Users can switch between views using the report toolbar icons, if the report allows it.

3.1.1 Table View

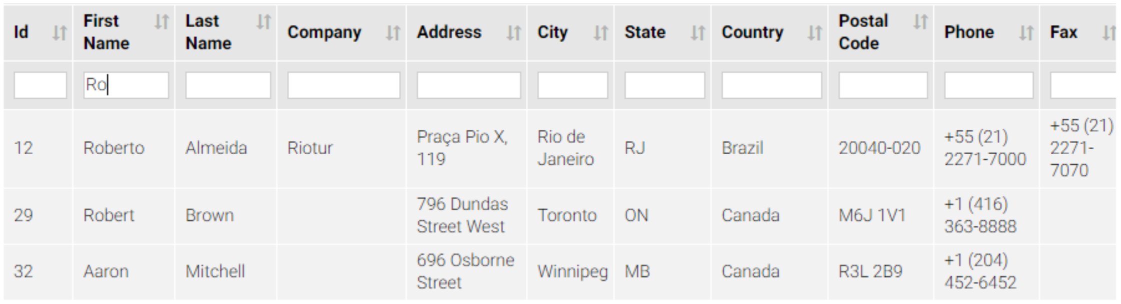


Id	First Name	Last Name	Company	Address	City	State	Country	Postal Code	Phone
12	Roberto	Almeida	Riotur	Praça Pio X, 119	Rio de Janeiro	RJ	Brazil	20040-020	+55 (21) 2271-700
28	Julia	Barnett		302 S 700 E	Salt Lake City	UT	USA	84102	+1 (801) 531-7272
39	Camille	Bernard		4, Rue Milton	Paris		France	75009	+33 01 49 70 65 65
18	Michelle	Brooks		627 Broadway	New York	NY	USA	10012-2612	+1 (212) 221-3546
29	Robert	Brown		796 Dundas Street West	Toronto	ON	Canada	M6J 1V1	+1 (416) 363-8888

Actions available in the Table view:

- **Sorting** - users can change the default order of rows by clicking ⇅ icon to sort by this column in ascending or descending order. It is possible to sort by multiple columns with holding Shift button. SQLwallet will remember the selected sorting order until the user resets the report layout by clicking ⊘ icon.

- **Resizing** - users can resize columns by dragging column borders in the table header. SQLwallet will remember the changed column widths until the user resets the report layout by clicking ⊘ icon.
- **Hiding columns** - user can hide columns by pointing mouse over the column header and clicking "x" icon. SQLwallet will remember the hidden columns until the user resets the report layout by clicking ⊘ icon.
- **Quick Filter** - in addition to the main report filters, users can filter by any column in the table. To open the quick filter panel, point the mouse between the table header and the first row.

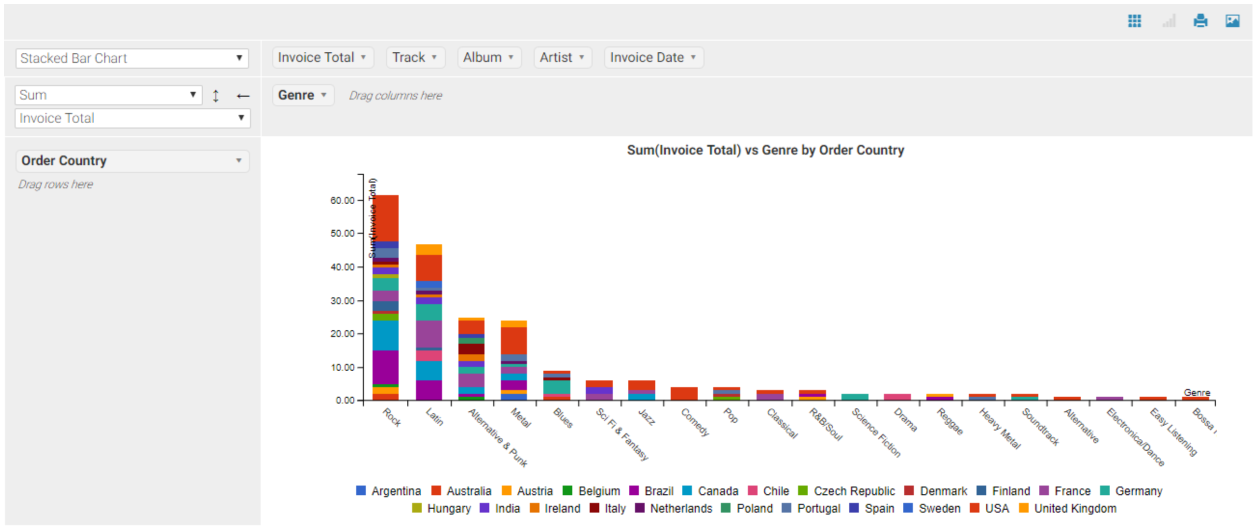


Id	First Name	Last Name	Company	Address	City	State	Country	Postal Code	Phone	Fax
	Ro									
12	Roberto	Almeida	Riotur	Praça Pio X, 119	Rio de Janeiro	RJ	Brazil	20040-020	+55 (21) 2271-7000	+55 (21) 2271-7070
29	Robert	Brown		796 Dundas Street West	Toronto	ON	Canada	M6J 1V1	+1 (416) 363-8888	
32	Aaron	Mitchell		696 Osborne Street	Winnipeg	MB	Canada	R3L 2B9	+1 (204) 452-6452	

Note: The table operations are performed in the browser. They do not cause the report to be executed again on the server, but they can take longer time if your dataset is big. If you want to implement sorting and filtering on server side, you should do so within SQL query and allow users to apply filters and sorting using the report parameters and macros.

By default the screen report output is limited by 2000 records. If the report produces more records, a user will get a warning message. The quick filtering and sorting is performed on the displayed records only.

3.1.2 Pivot/Chart View



In Pivot/Chat view users can select the data aggregation and visualization (this can be enabled or disabled in *Report output* section).

The following charts and visualizations are supported:

- **Table** - the data is aggregated by selected columns and displayed as a pivot table.

- **Table Bar Chart** - the data is aggregated by selected columns and displayed as a pivot table. Each sell contains a bar chart indicating the value weight in the row.
- **Table Heatmap** - the data is aggregated by selected columns and displayed as a pivot table. Each sell is highlighted according to the value weight in the whole table.
- **Table Heatmap** - the data is aggregated by selected columns and displayed as a pivot table. Each sell is highlighted according to the value weight in the row.
- **Column Heatmap**- the data is aggregated by selected columns and displayed as a pivot table. Each sell is highlighted according to the value weight in the column.
- **TSV Export** - the pivoted data in text format that can be copied and pasted into Excel.
- **Bar Chart**
- **Stacked Bar Chart**
- **Horizontal Bar Chart**
- **Horizontal Stacked Bar Chart**
- **Line Chart**
- **Area Chart**
- **Scatter Chart**

The user can select the metric column and the aggregate function (Sum, Average, Count, Unique Count, Min, Max etc). At the moment only single metric is supported per chart.

The user can select data series on both X and Y axes by dragging the column titles from the top area onto the rows/columns areas.

The following commands are available in the pivot/chart toolbar:

- 🖶 - prints the chart. Please note that for large charts the user may need to adjust the scale and page orientation in the Print dialog in order to fit the whole chart on the page.
- 🖼 - exports the chart as image file.
- 📌 - saves the current pivot/chart layout as a default. Only available for the report authors.
- ▦ - switches to the table view.

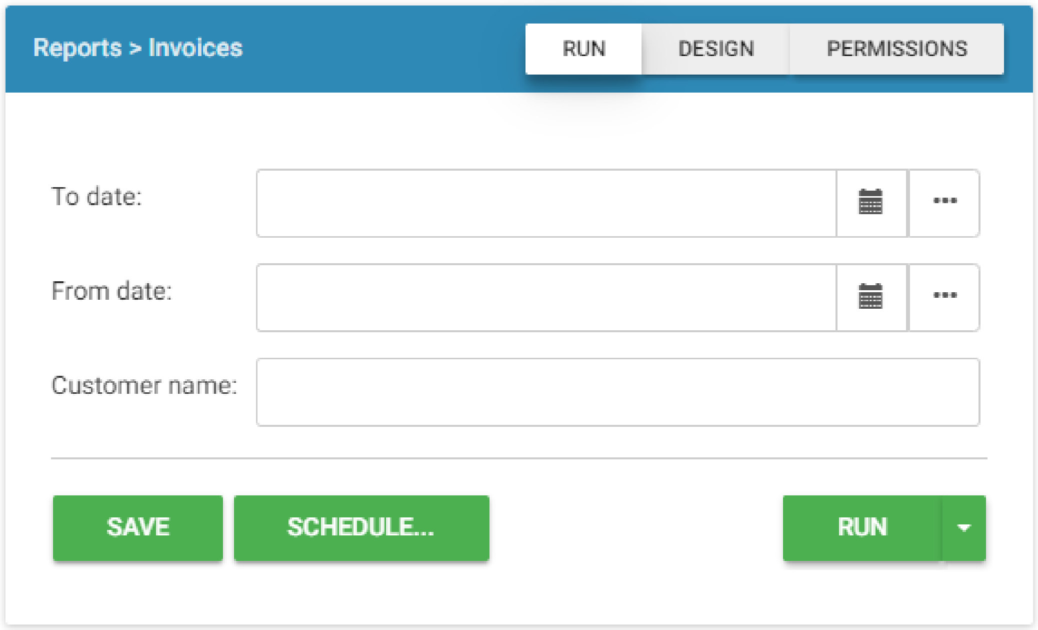
3.2 Scheduling reports

Scheduling allows to run reports automatically on specified time. The result of the report can be either emailed to users, or used to execute automation tasks, such as copying the report CSV file into a specific folder or uploading on FTP.

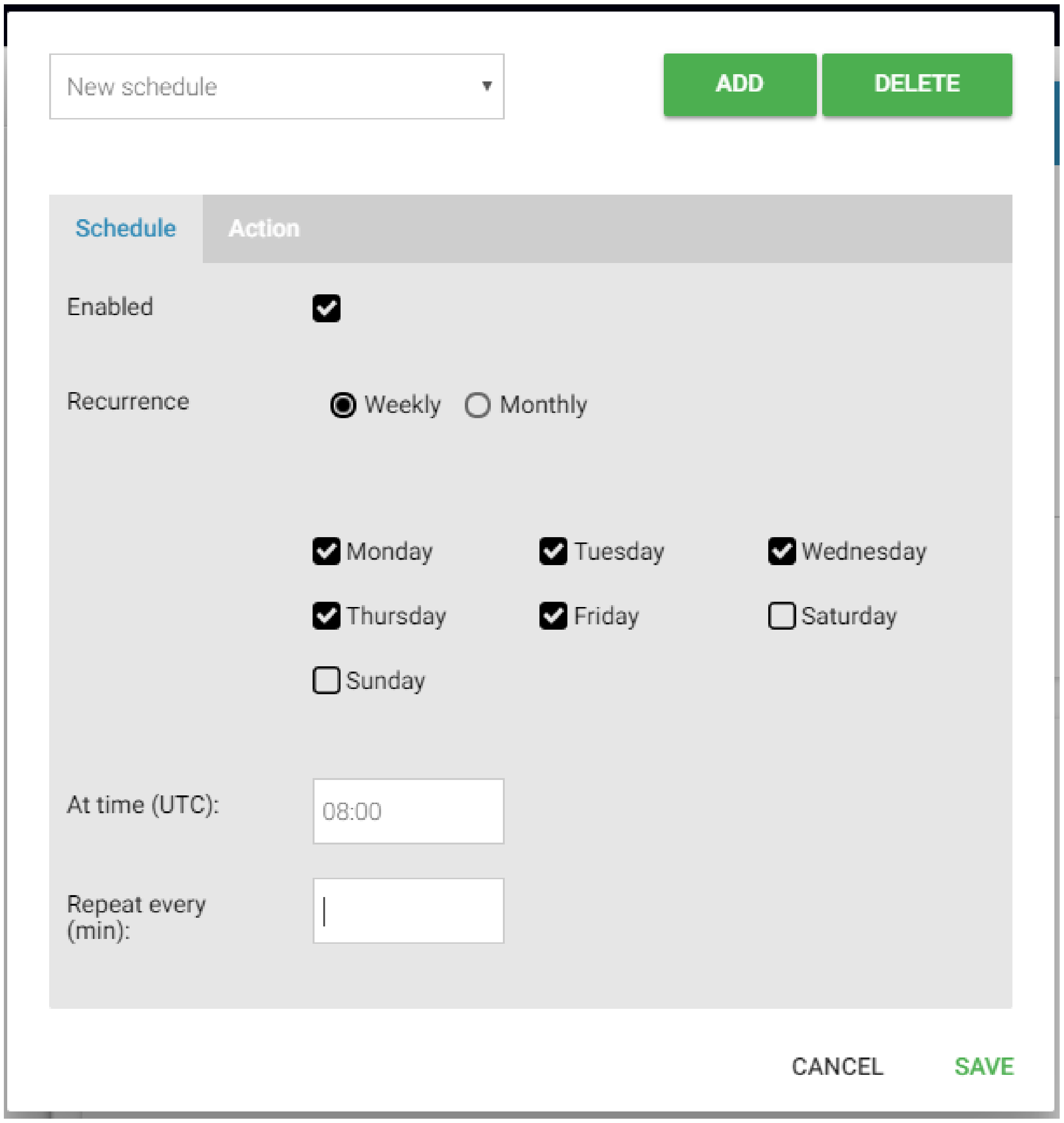
In order to schedule a report, the “Allow scheduling” checkbox should be enabled in the *General* section in the report Design View.

Once the scheduling is enabled, switch to Run view and click “Schedule” button.

Note: Scheduled reports will be executed only if all mandatory parameters are supplied. Therefore, If your report has mandatory parameters, you need to fill in the parameters, save the report, and then schedule the saved report.



When you click "Schedule" button, a popup dialog will appear. You can have more than one schedule for the report, by default a new empty schedule is created.



Enabled [✓]

Recurrence [✓ Weekly] Monthly

[✓ Monday] [✓ Tuesday] [✓ Wednesday]
[✓ Thursday] [✓ Friday] Saturday
Sunday

At time (UTC): 08:00

Repeat every (min):

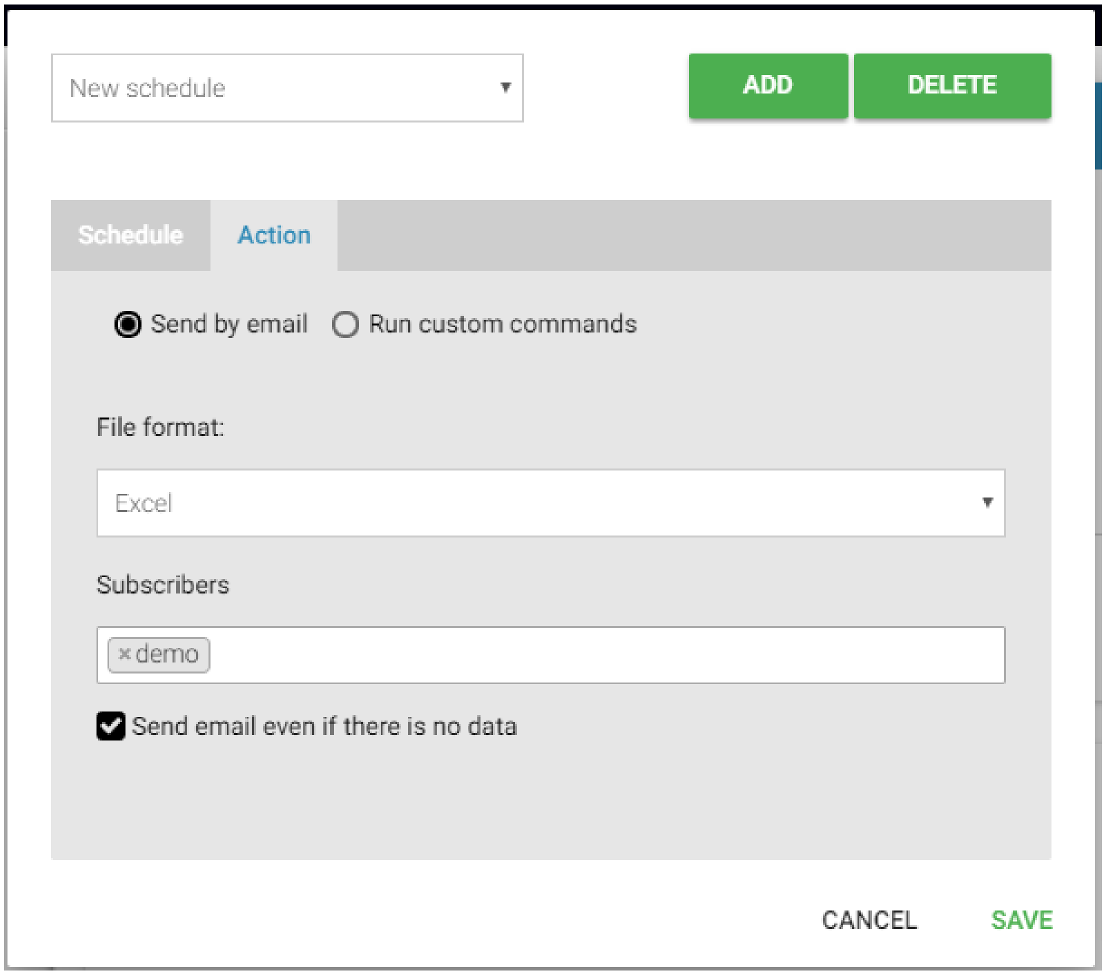
3.2.1 Schedule settings

- **Enabled** - enables or disables the schedule.
- **Recurrence** - select "Weekly" to run the report on specific days of week, or "Monthly" to run the report on specific days of month.
- **At time (UTC)** - specifies time when the report should be run on the selected days.
- **Repeat every (min)** - specifies interval in minutes, if the report should be run more often than once a day.

3.2.2 Action settings

Send by Email

Send an email with the report output to the selected recipients.



[● Send by email] ○ Run custom commands

File format: Excel

Subscribers: demo

[✓] Send email even if there is no data

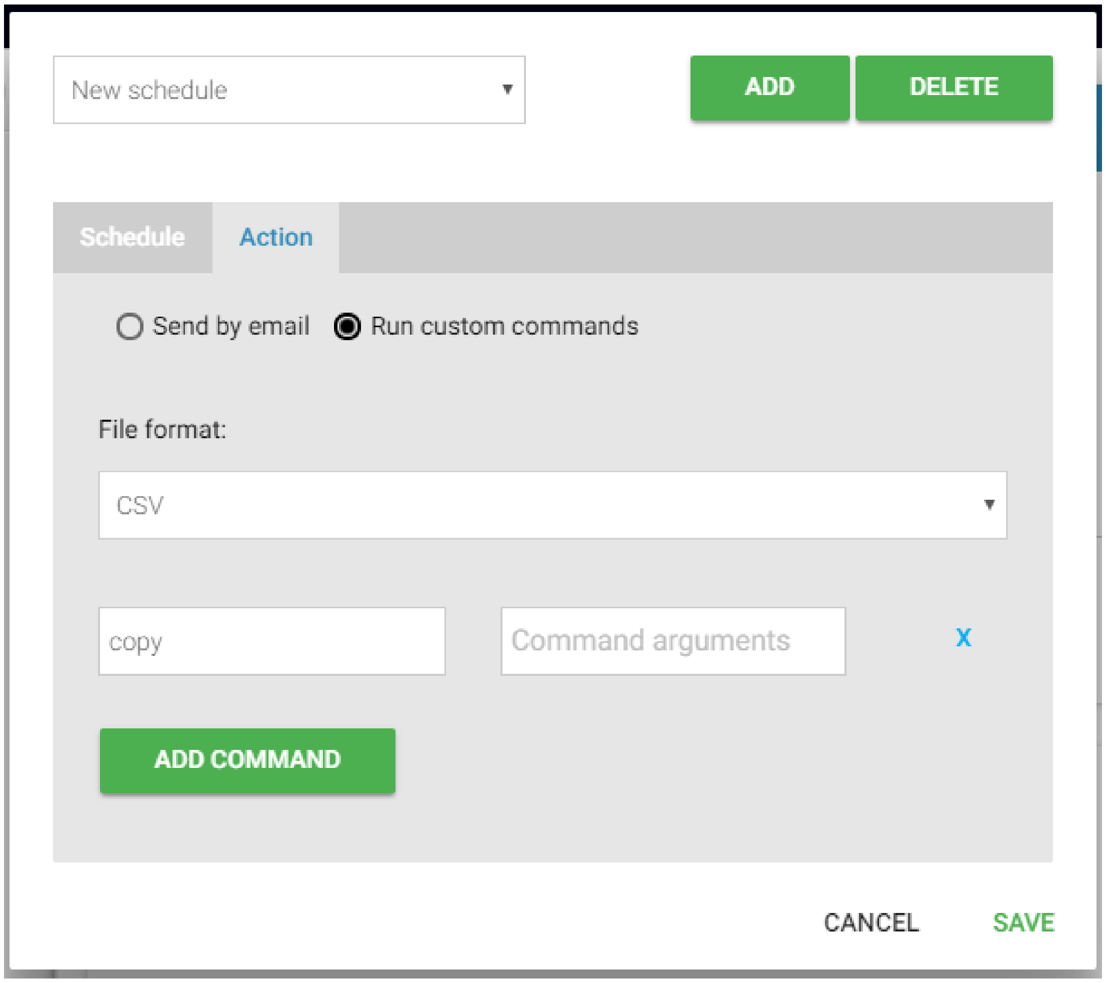
- **File format - specifies how the report will be sent:**
  - **Excel** - an Excel file attachment
  - **CSV** - a CSV file attachment (or ZIP with multiple csv files if the report has more than one output)
  - **InlineHTML** - the repot table will be in the email body, no attachment.
- **Subscribers** - allows to select recipients for the email. Only users with "Run" permission for the report can be selected.

Note: Teams cannot be selected as recipients. Each user should be selected individually. If you are intending to send the report to a distribution list, create a new user with the distribution list email address and select this user as a recipient.

- **Send email even if there is no data**
  - If checked, the email will be always sent, even if the report produced no output records.
  - If unchecked, the email will be only sent if the report returns output records. This is extremely useful when setting up alerts and notification.

Run custom commands

Executes one or multiple commands, passing the report file name as an argument.



Send by email [✓ Run custom commands]

File format: CSV

copy

Command arguments

- **File format - specifies how the report file will be generated:**
  - **Excel** - an Excel file attachment
  - **CSV** - a CSV file attachment (or ZIP with multiple csv files if the report has more than one output)
- **Command line** - the OS command to invoke.
- **Command arguments** - the command line to pass to the command. The report filename will always be appended at the end after all other parameters.

Custom command example

If you want to save the report file in c:\backup folder:

1. In the "Command line" field type *powershell*
2. In the "Command arguments" field, type *c:\scripts\copy.ps1 c:\backup*
3. Create file *c:\scripts\copy.ps1* with the following content:

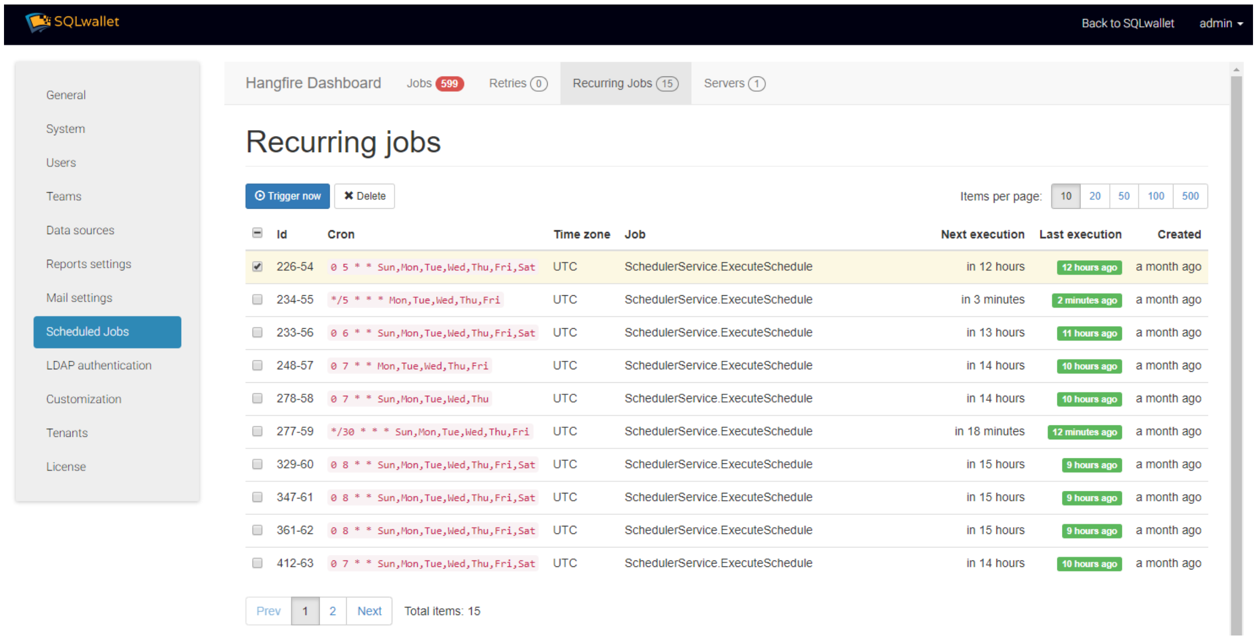
```
Copy-Item $args[1] -Destination $args[0] -Force -Recurse
```

Note: Make sure that PowerShell script execution is enabled for x86. See this post for details.

When the scheduler executes a custom command, it invokes the command line and passes the arguments. The last argument always will be the generated file name. Therefore within your script *$args[0]* will contain "*c:\backup*", and *$args[1]* will contain the generated report file name.

3.2.3 Testing the schedule

In order to test the scheduler without waiting for the scheduled time, go to configuration and select "Scheduled Jobs" from the menu. Then select your schedule under Recurring Jobs and click "Trigger now".



Recurring jobs

	Id	Cron	Time zone	Job	Next execution	Last execution	Created
✓	226-54	`0 5 * * Sun,Mon,Tue,Wed,Thu,Fri,Sat`	UTC	SchedulerService.ExecuteSchedule	in 12 hours	12 hours ago	a month ago
	234-55	`*/5 * * * Mon,Tue,Wed,Thu,Fri`	UTC	SchedulerService.ExecuteSchedule	in 3 minutes	2 minutes ago	a month ago
	233-56	`0 6 * * Sun,Mon,Tue,Wed,Thu,Fri,Sat`	UTC	SchedulerService.ExecuteSchedule	in 13 hours	11 hours ago	a month ago
	248-57	`0 7 * * Mon,Tue,Wed,Thu,Fri`	UTC	SchedulerService.ExecuteSchedule	in 14 hours	10 hours ago	a month ago
	278-58	`0 7 * * Sun,Mon,Tue,Wed,Thu`	UTC	SchedulerService.ExecuteSchedule	in 14 hours	10 hours ago	a month ago
	277-59	`*/30 * * * Sun,Mon,Tue,Wed,Thu,Fri`	UTC	SchedulerService.ExecuteSchedule	in 18 minutes	12 minutes ago	a month ago
	329-60	`0 8 * * Sun,Mon,Tue,Wed,Thu,Fri,Sat`	UTC	SchedulerService.ExecuteSchedule	in 15 hours	9 hours ago	a month ago
	347-61	`0 8 * * Sun,Mon,Tue,Wed,Thu,Fri,Sat`	UTC	SchedulerService.ExecuteSchedule	in 15 hours	9 hours ago	a month ago
	361-62	`0 8 * * Sun,Mon,Tue,Wed,Thu,Fri,Sat`	UTC	SchedulerService.ExecuteSchedule	in 15 hours	9 hours ago	a month ago
	412-63	`0 7 * * Sun,Mon,Tue,Wed,Thu,Fri,Sat`	UTC	SchedulerService.ExecuteSchedule	in 14 hours	10 hours ago	a month ago

Total items: 15

CHAPTER 4

Embedding reports

SQLwallet allows you to invoke reports directly from your application by adding a link to a specific report or by embedding the report page in IFRAME.

The URL for direct link to the report looks like:

```
http://yoursqlwalletdomain.com/run/<reportid>?access_token=<access token>
```

You can obtain the URL for the specific report using “Get direct link” button at the bottom of the Report Designer page.

4.1 Generating access token

To get access token, perform the following steps:

1. Invoke the token API request (server-to-server request) to obtain authorization token:

```
POST /connect/token HTTP/1.1
Host: yoursqlwalletdomain.com:88
Content-Type: application/x-www-form-urlencoded
Cache-Control: no-cache

client_secret=<secret>&grant_type=client_credentials&client_id=sqlwallet&
↪scope=sqlwallet
```

2. Invoke the API method (server-to-server request) to get the access token to run report on behalf of a specific user:

```
POST /auth/accesstoken/<username> HTTP/1.1
Host: yoursqlwalletdomain.com:88
Content-Type: application/json
Cache-Control: no-cache
Authorization: Bearer <authorization token>
```

(continues on next page)

(continued from previous page)

```
{"param1":"value1","param2":"value2",...}
```

Note:

- Authentication API calls shoudl be made using the primary domain SQLwallet is installed on (i.e. defined in appsettings.json)

Add report parameters in the request body as shown above. These parameters will be encrypted in the token, so users cannot change them in runtime.

4.2 Invoking the report

Invoke the report using the generated access token:

```
http://yoursqlwalletdomain.com/run/<reportid>?access_token=<access token>
```

If you want to be able to modify the parameters in the client side code, you can pass them directly in the report URL:

```
http://yoursqlwalletdomain.com/run/<reportid>?access_token=<access token>&
↪param1=value1&param2=value2...
```

Note:

- Parameters in the token will take precedence over parameters passed in query string.
- You can also pass access token to the report in the Authorization Bearer request header.

Alternatively, you can invoke the report with parameters panel visible, so users can modify report parameters:

```
http://yoursqlwalletdomain.com/embed?access_token=<access token>#<reportid>
```