xeus-sql

Mariana Meireles

Dec 29, 2020

GETTING STARTED

1 Licensing 3

`xeus-sql` is a Jupyter kernel for general SQL implementations based on the native implementation of the Jupyter protocol xeus and SOCI, a database access library for C++.

CHAPTER

ONE

LICENSING

We use a shared copyright model that enables all contributors to maintain the copyright on their contributions.

This software is licensed under the BSD-3-Clause license. See the LICENSE file for details.

1.1 Installation

1.1.1 With Conda or Mamba

`xeus-sql` has been packaged for the conda package manager.

To ensure that the installation works, it is preferable to install `xeus-sql` in a fresh conda/mamba environment. It is also needed to use a miniconda installation because with the full anaconda you may have a conflict with the `zeromq` library which is already installed in the anaconda distribution.

The safest usage is to create an environment named `xeus-sql` with your miniconda installation

```
conda create -n xeus-sql
conda activate xeus-sql # Or `source activate xeus-sql` for conda < 4.6
```

```
mamba create -n xeus-sql
mamba activate xeus-sql
```

Then you can install in this freshly created environment `xeus-sql` and its dependencies

```
conda install xeus-sql notebook -c conda-forge
```

```
mamba install xeus-sql notebook -c conda-forge
```

or, if you prefer to use JupyterLab

```
conda install xeus-sql jupyterlab -c conda-forge
```

```
mamba install xeus-sql jupyterlab -c conda-forge
```

Conda forge offers packaged versions for MySQL, PostgreSQL and SQLite and you can download them with: *soci-mysql*, *soci-postresql* or *soci-sqlite*.

`xeus-sql` includes `soci-core` only. Which consists on the SOCI package with no DB extension attached.

1.1.2 From Source

You can install `xeus-sql` from source with `cmake`. This requires that you have all the dependencies installed in the same prefix.

```
conda install cmake nlohmann_json xtl cppzmq xeus mysql sqlite postgresql cpp-
↪tabulate=1.3 xvega xvega-bindings xproperty jupyterlab compilers -c conda-forge
```

```
mamba install cmake nlohmann_json xtl cppzmq xeus mysql sqlite postgresql cpp-
↪tabulate=1.3 xvega xvega-bindings xproperty jupyterlab compilers -c conda-forge
```

```
mkdir build
cd build
cmake -DCMAKE_INSTALL_PREFIX=/path/to/prefix ..
make install
```

On Windows platforms, from the source directory:

```
mkdir build
cd build
cmake -G "NMake Makefiles" -DCMAKE_INSTALL_PREFIX=/path/to/prefix ..
nmake
nmake install
```

1.2 DB2

We're missing this part of the docs. If you've used `xeus-sql` in this context and would like to help, please feel free to open a PR! :)

1.3 MySQL

1.3.1 Linux

Installation

You can get MySQL from `conda-forge`

```
conda install xeus-sql soci-mysql jupyterlab -c conda-forge
```

Usage

To use MySQL from inside a notebook you need to first edit your MySQL configuration file that can usually be found at `/etc/mysql/my.cnf`:

```
[server]
socket=/tmp/mysql.sock

[client]
socket=/tmp/mysql.sock
```

To run the example notebook you'll need to load the `dbname` database or create one for yourself:

```
CREATE DATABASE dbname;
CREATE TABLE example ( id smallint unsigned not null auto_increment, name varchar(20)␣
↪not null, constraint pk_example primary key (id) );
```

You will also need to create a new user and grant access to it to the newly create database. You can achieve this with the following lines in your MySQL console:

```
CREATE USER 'user1'@'localhost' IDENTIFIED BY 'Password123#@!';
GRANT ALL PRIVILEGES ON dbname.* TO 'user1'@'localhost';
```

Note that it might be necessary to install MySQL from different sources other than conda if you intend to manipulate users and creation of databases.

More information about Firebird can be found in the SOCI documentation.

1.4 ODBC

We're missing this part of the docs. If you've used `xeus-sql` in this context and would like to help, please feel free to open a PR! :)

1.5 Oracle

We're missing this part of the docs. If you've used `xeus-sql` in this context and would like to help, please feel free to open a PR! :)

1.6 PostgreSQL

1.6.1 Linux

If you're running `xeus-sql` with a conda-forge PostgreSQL build, you will have to create a symbolic link to `/tmp/`:

```
sudo ln -s /var/run/postgresql/.s.PGSQL.5432 /tmp/.s.PGSQL.5432
```

Create an user:

```
sudo -u postgres createuser --superuser $USER_NAME
```

Start the service:

```
sudo service postgresql start sudo -u $USER_NAME psql
```

Create a database using the example available on this repository:

```
pg_restore --dbname=newdvdrental -U $USER_NAME pathto/xeus-sql/examples/dvdrental.tar
```

You can now run the example contained in `examples/PostgreSQL.ipynb`.

More information about PostgreSQL can be found in the SOCI documentation.

1.7 SQLite

To run the SQLite backend simply install it with:

```
mamba install soci-sqlite xeus-sql jupyterlab -c conda-forge
```

Or any other method desired.

No other preparation is needed to run the notebook example. You can head to `examples/SQLite.ipynb` and run the code.

More information about SQLite can be found in the SOCI documentation.

1.8 API

This API refers to the magics implemented to execute operations that are not SQL code. There's a distinction between `xvega` magics that implements magics from the `xvega` library and allows the user to create graphic visualizations of `xeus-sql` outputs and the `xeus-sql` magics that allows file operations.

1.8.1 SQL magics

Magics that allow you to operate on the database.

LOAD

`%LOAD database_type name_of_database`

To see how to use this command in depth, please refer to the specific page of the database.

1.8.2 XVega magics

Magics that allow you to create graph visualizations using XVega an implementation of vega-light to C++.

X_FIELD

`%X_FIELD name_of_column`
Represents the X axis of the graph. The name of the axis should be the same as the name of the SQLite column (or result of SQLite query).

TYPE

`%TYPE type_of_field`
Sub-attribute of **X_FIELD**.

Bellow there's list of the types supported by *xeus-sqlite*. If you want to learn more about types please refer to vega lite type official documentation.

- QUANTITATIVE
- NOMINAL

- ORDINAL
- TEMPORAL

BIN

%BIN type_of_field
Sub-attribute of **X_FIELD**.

Binning discretizes numeric values into a set of bins. If bin is true, default binning parameters are used.

To customize binning parameters, you can set bin to a bin definition object, which can have the following properties:

If you want to learn more about bin please refer to vega lite bin official documentation.

ANCHOR

%ANCHOR bin_position
Sub-attribute of **BIN**.

A value in the binned domain at which to anchor the bins, shifting the bin boundaries if necessary to ensure that a boundary aligns with the anchor value.

BASE

%BASE number
Sub-attribute of **BIN**.

The number base to use for automatic bin determination (default is base 10).

BINNED

%BASE boolean
Sub-attribute of **BIN**.

MAXBINS

%MAXBINS number
Sub-attribute of **BIN**.

Maximum number of bins.

MINSTEP

%MINSTEP number
Sub-attribute of **BIN**.

A minimum allowable step size (particularly useful for integer values).

NICE

%NICE bool
Sub-attribute of **BIN**.

If true, attempts to make the bin boundaries use human-friendly boundaries, such as multiples of ten.

STEP

%STEP number
Sub-attribute of **BIN**.

An exact step size to use between bins.

AGGREGATE

%AGGREGATE type_of_aggregation
Sub-attribute of **X_FIELD**.

The aggregate property of a field definition can be used to compute aggregate summary statistics (e.g., median, min, max) over groups of data.

Bellow there's list of the aggregations supported by *xeus-sqlite*. If you want to learn more about aggregations please refer to vega lite aggregate official documentation.

- COUNT
- VALID
- MISSING
- DISTINCT
- SUM
- PRODUCT
- MEAN
- AVERAGE
- VARIANCE
- VARIANCEP
- STDEV
- STEDEVP
- STEDERR
- MEDIAN
- Q1

- Q3
- CI0
- CI1
- MIN
- MAX
- ARGMIN
- ARGMAX

TIME_UNIT

%TIME_UNIT time
Sub-attribute of **X_FIELD**.

Time unit is used to discretize time.

Bellow there's list of the time units supported by *xeus-sqlite*. If you want to learn more about time units please refer to vega lite time unit official documentation.

- YEAR
- QUARTER
- MONTH
- DAY
- DATE
- HOURS
- MINUTES
- SECONDS
- MILISECONDS

Y_FIELD

%Y_FIELD name_of_column
Represents the Y axis of the graph. The name of the axis should be the same as the name of the SQLite column (or result of SQLite query).

TYPE

%TYPE type_of_field
Sub-attribute of **Y_FIELD**.

Bellow there's list of the types supported by *xeus-sqlite*. If you want to learn more about types please refer to vega lite type official documentation.

- QUANTITATIVE
- NOMINAL
- ORDINAL

- TEMPORAL

BIN

%BIN type_of_field
Sub-attribute of **Y_FIELD**.

Binning discretizes numeric values into a set of bins. If bin is true, default binning parameters are used.

To customize binning parameters, you can set bin to a bin definition object, which can have the following properties:

If you want to learn more about bin please refer to vega lite bin official documentation.

ANCHOR

%ANCHOR bin_position
Sub-attribute of **BIN**.

A value in the binned domain at which to anchor the bins, shifting the bin boundaries if necessary to ensure that a boundary aligns with the anchor value.

BASE

%BASE number
Sub-attribute of **BIN**.

The number base to use for automatic bin determination (default is base 10).

BINNED

%BASE boolean
Sub-attribute of **BIN**.

MAXBINS

%MAXBINS number
Sub-attribute of **BIN**.

Maximum number of bins.

MINSTEP

%MINSTEP number
Sub-attribute of **BIN**.

A minimum allowable step size (particularly useful for integer values).

NICE

%NICE bool
Sub-attribute of **BIN**.

If true, attempts to make the bin boundaries use human-friendly boundaries, such as multiples of ten.

STEP

%STEP number
Sub-attribute of **BIN**.

An exact step size to use between bins.

AGGREGATE

%AGGREGATE type_of_aggregation
Sub-attribute of **Y_FIELD**.

The aggregate property of a field definition can be used to compute aggregate summary statistics (e.g., median, min, max) over groups of data.

Bellow there's list of the aggregations supported by *xeus-sqlite*. If you want to learn more about aggregations please refer to vega lite aggregate official documentation.

- COUNT
- VALID
- MISSING
- DISTINCT
- SUM
- PRODUCT
- MEAN
- AVERAGE
- VARIANCE
- VARIANCEP
- STDEV
- STEDEVP
- STEDERR
- MEDIAN
- Q1
- Q3
- CI0
- CI1
- MIN
- MAX

- ARGMIN
- ARGMAX

TIME_UNIT

%TIME_UNIT time
Sub-attribute of **Y_FIELD**.

Time unit is used to discretize time.

Bellow there's list of the time units supported by *xeus-sqlite*. If you want to learn more about time units please refer to vega lite time unit official documentation.

- YEAR
- QUARTER
- MONTH
- DAY
- DATE
- HOURS
- MINUTES
- SECONDS
- MILISECONDS

WIDTH

%WIDTH number
Width of the graph in pixels.

HEIGHT

%HEIGHT number
Height of the graph in pixels.

MARK

%MARK mark
Marcs can be one of the following:

- ARC
- AREA
- BAR
- CIRCLE
- LINE
- POINT
- RECT

- RULE
- SQUARE
- TICK
- TRAIL

COLOR

%COLOR color
Sub-attribute of **MARK**.

Sets the color of a mark. The color can be one of the valid CSS color string.

GRID

%HEIGHT boolean
Enable or disable grid view on graph.