EEA Discodata

WISE SOE - Waterbase

Basic user guide

version 1.0, July 2021

by Marek Staron, EEA

Introduction

The WISE SOE Waterbase data has been traditionally disseminated through the EEA Data and Maps portal as a series of downloadable files in various formats. The increase in the number of data, and the respective increase in the size of the files, however, makes this option less viable for some users.

For example, at the time of writing this guide, the latest WISE SOE Water Quality ICM (WISE6) dataset contains more than 51 millions of Disaggregated Data records. The users can download a .csv file containing all the data which is almost 13 GB in size once unzipped.

Consuming such an amount of data in a traditional spreadsheet software, like Microsoft Excel, is very difficult, or even impossible. It requires a use of a database software, which may not be a solution available to all users.

Some users may also be interested only in a very specific part of the dataset. For example, only records from a particular country, year, or of a specific substance. Downloading the whole file to use just a small part of it (and possibly struggle to do so due to its size) may be too inconvenient.

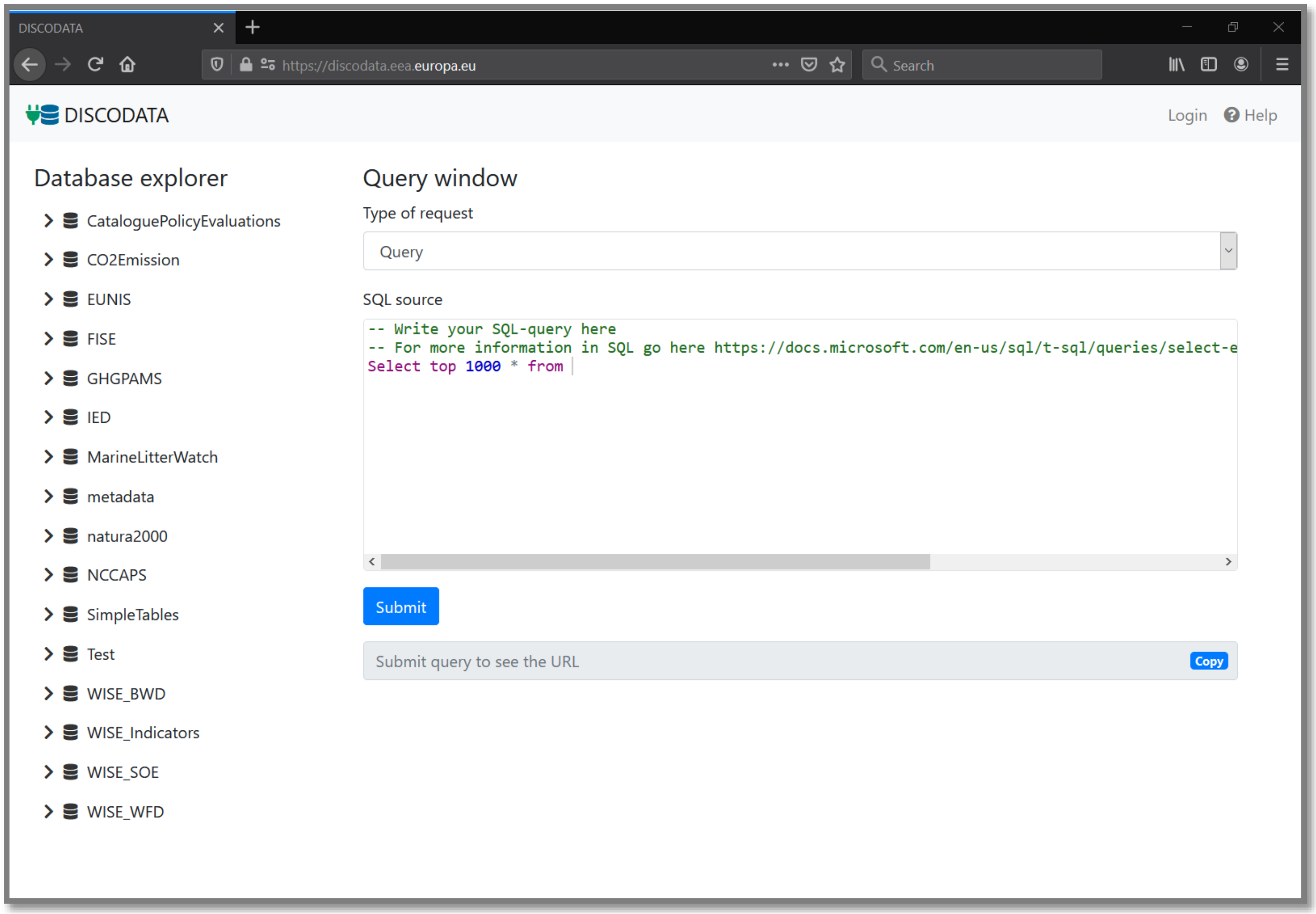
To tackle this issue, the EEA has developed a portal, named Discodata, which provides an endpoint to access and filter data directly in the database. The main use for the portal is to provide access to datasets for specifically developed applications. But it also provides a user interface that gives any user an option to access, filter and download the data.

This guide is to help the users with exploration, selection, and extractions of Waterbase data by introducing the layout and functions of the Discodata portal, with some examples of more advanced techniques.

Note: The Discodata portal is still in active development at the time of writing this guide. Some parts of this guide may therefore become obsolete, although we will try to update it when that happens. We will also try to extend the guide with additional examples, and potentially focusing on individual datasets and specific use of the Waterbase data.

The Discodata portal can be accessed through this URL:
https://discodata.eea.europa.eu/

When user opens the Discodata portal they will see this homepage:

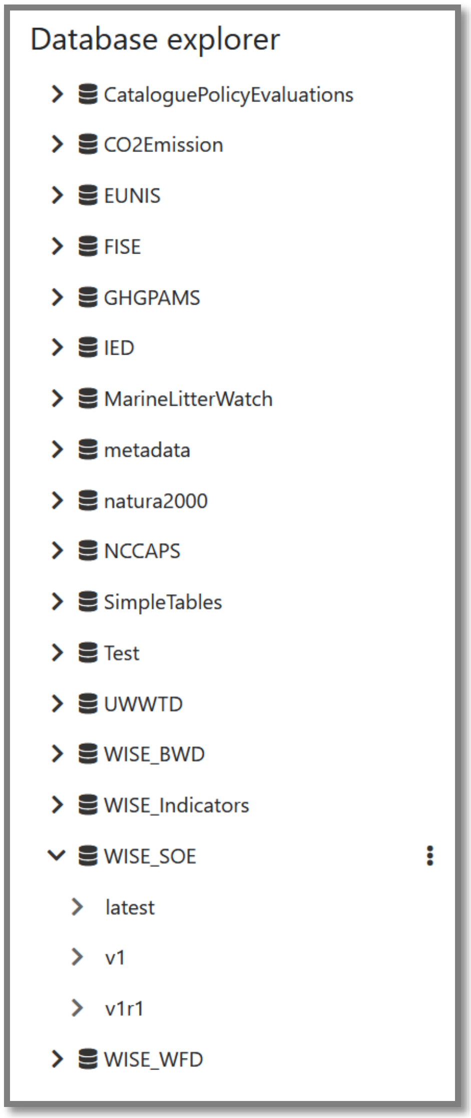


```
-- Write your SQL-query here
-- For more information in SQL go here https://docs.microsoft.com/en-us/sql/t-sql/queries/select-e
Select top 1000 * from
```

Database explorer

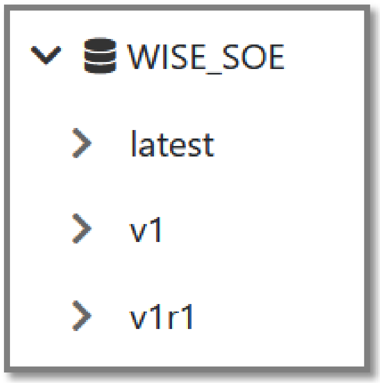
On the left side of the page, we see a list of databases that are currently accessible through the Discodata portal.

Clicking on the **WISE_SOE** database expands the list of the database versions, releases, and views.


Database explorer

- CataloguePolicyEvaluations
- CO2Emission
- EUNIS
- FISE
- GHGPAMS
- IED
- MarineLitterWatch
- metadata
- natura2000
- NCCAPS
- SimpleTables
- Test
- UWWTD
- WISE_BWD
- WISE_Indicators
- WISE_SOE
  - latest
  - v1
  - v1r1
- WISE_WFD

Database versioning


WISE_SOE

- latest
- v1
- v1r1

- **latest** – a folder that contains **views** pointing to the latest *releases* of the tables from the latest *version* of the database.
- **v1** – a folder that contains **views** pointing to the latest *releases* of tables from the database ***version*** **1**.
- **v1r1** – a folder with the actual **tables** from ***release*** **1** in the database ***version*** **1**.

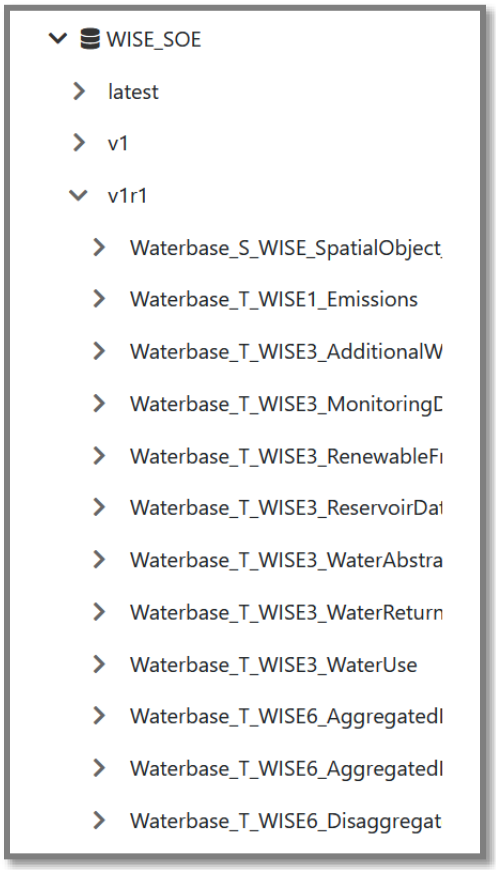
- *version* – a new version is created when the structure of the database tables is updated (a new field is added, an old field is removed, etc.).
- *release* – a new release is created when the content of the database tables is updated (e.g., with newly reported data).

Because the WISE_SOE database has currently only one version and release (v1r1), the views in both the **latest** and the **v1** folder point to the tables in the **v1r1** folder.

For the sake of simplicity this guide describes only the use of **v1r1** folder content.

Table list

Clicking on the **v1r1** folder will expand the list of tables in the WISE SOE v1r1 database.

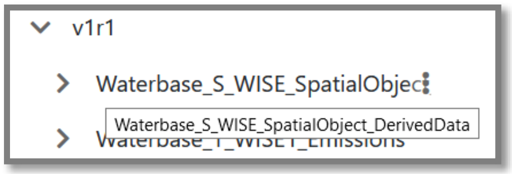


WISE_SOE
latest
v1
v1r1
Waterbase_S_WISE_SpatialObject
Waterbase_T_WISE1_Emissions
Waterbase_T_WISE3_AdditionalW
Waterbase_T_WISE3_MonitoringD
Waterbase_T_WISE3_RenewableFr
Waterbase_T_WISE3_ReservoirDat
Waterbase_T_WISE3_WaterAbstra
Waterbase_T_WISE3_WaterReturn
Waterbase_T_WISE3_WaterUse
Waterbase_T_WISE6_AggregatedI
Waterbase_T_WISE6_AggregatedI
Waterbase_T_WISE6_Disaggregat

The WISE_SOE database currently contains all dataset tables which were updated during the WISE SOE 2020 dataflow cycle and are published in the respective dataset in EEA Data and Maps portal.

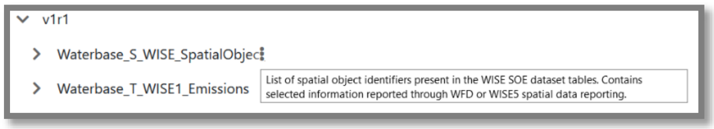
The table ‘Waterbase_S_WISE_SpatialObject_DerivedData’ contains information on all spatial objects referenced in the other tables. The information includes coordinates of the spatial object point or a centroid, if available. The remaining tables contain timeseries data (indicated by ‘_T_’ in the name).

Hovering over the name of a particular table brings a frame with the table’s full name, which may be useful if the name in the list is truncated when it’s too long:



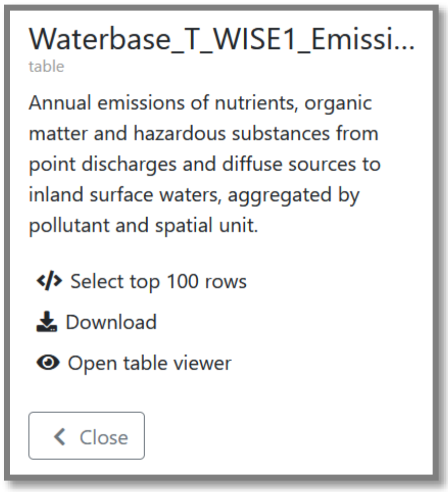
v1r1
Waterbase_S_WISE_SpatialObjec
Waterbase_S_WISE_SpatialObject_DerivedData
Waterbase_T_WISE1_Emissions

Hovering over the three vertical dots right of the table name brings a frame with the table descriptions.



v1r1
Waterbase_S_WISE_SpatialObjec
Waterbase_T_WISE1_Emissions
List of spatial object identifiers present in the WISE SOE dataset tables. Contains selected information reported through WFD or WISE5 spatial data reporting.

Right clicking on the table name, or clicking on the three dots, opens an info block between the Database explorer and the Query window. The info block contains description of the table and provides a few options:



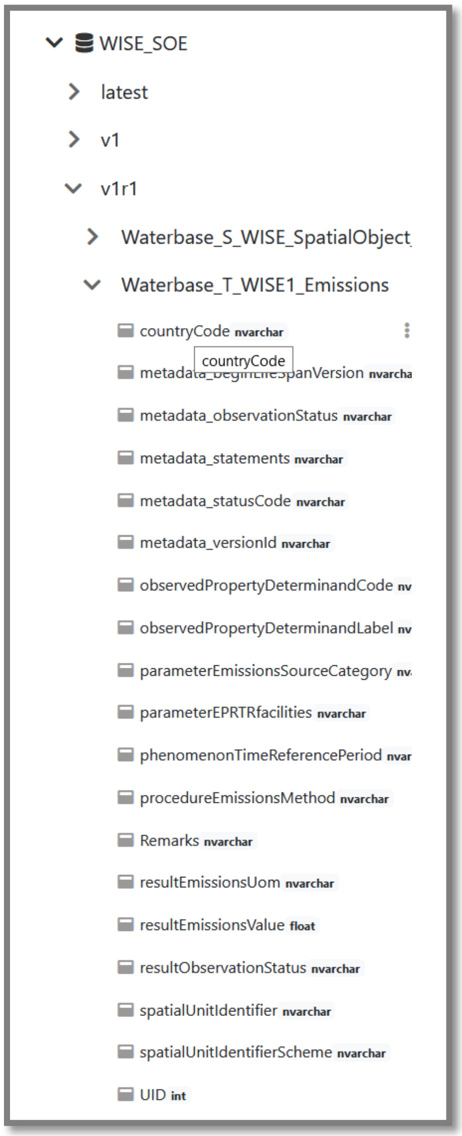
Waterbase_T_WISE1_Emissi...
table
Annual emissions of nutrients, organic matter and hazardous substances from point discharges and diffuse sources to inland surface waters, aggregated by pollutant and spatial unit.

- **Select top 100 rows** – clears the SQL source frame and puts in an SQL query statement which, when manually submitted, will select the first 100 records of the table, and show them in the query result frame (see the Query window for more information).
- **Download** – allows to download the whole table in zipped csv format. Please be aware, that downloading tables with large number of records may take some time and there is a risk of failure.
- **Go to table viewer** – opens a Table viewer page with option to explore, filter and download filtered table records. See Table viewer for more details on how to use the viewer.

Field list

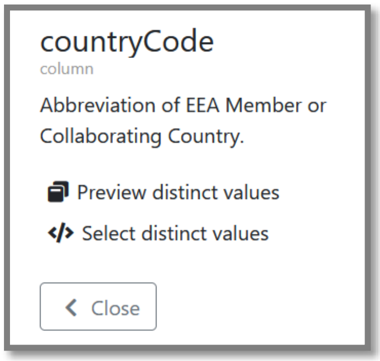
Clicking on a table name will expand a list of fields in the table. The fields are sorted alphabetically. The field's datatype is shown next after its name.

Hovering over the field's name brings a frame with the field's description. Hovering over the three vertical dots right of the field name brings a frame with the field's descriptions.:


WISE_SOE
latest
v1
v1r1
Waterbase_S_WISE_SpatialObject
Waterbase_T_WISE1_Emissions
countryCode nvarchar
countryCode
metadata_beginLifespanVersion nvarcha
metadata_observationStatus nvarchar
metadata_statements nvarchar
metadata_statusCode nvarchar
metadata_versionId nvarchar
observedPropertyDeterminandCode nv
observedPropertyDeterminandLabel nv
parameterEmissionsSourceCategory nv
parameterEPRTRfacilities nvarchar
phenomenonTimeReferencePeriod nvar
procedureEmissionsMethod nvarchar
Remarks nvarchar
resultEmissionsUom nvarchar
resultEmissionsValue float
resultObservationStatus nvarchar
spatialUnitIdentifier nvarchar
spatialUnitIdentifierScheme nvarchar
UID int

Note: Compared to datasets in EEA Data and Maps portal, most of the timeseries tables in Discodata contain additional fields, like country code or year, that can make filtering of specific data easier.

Right clicking on the field name, or clicking on the three dots, opens an info block between the Database explorer and the Query window. The info block contains description of the field and provides a few options:

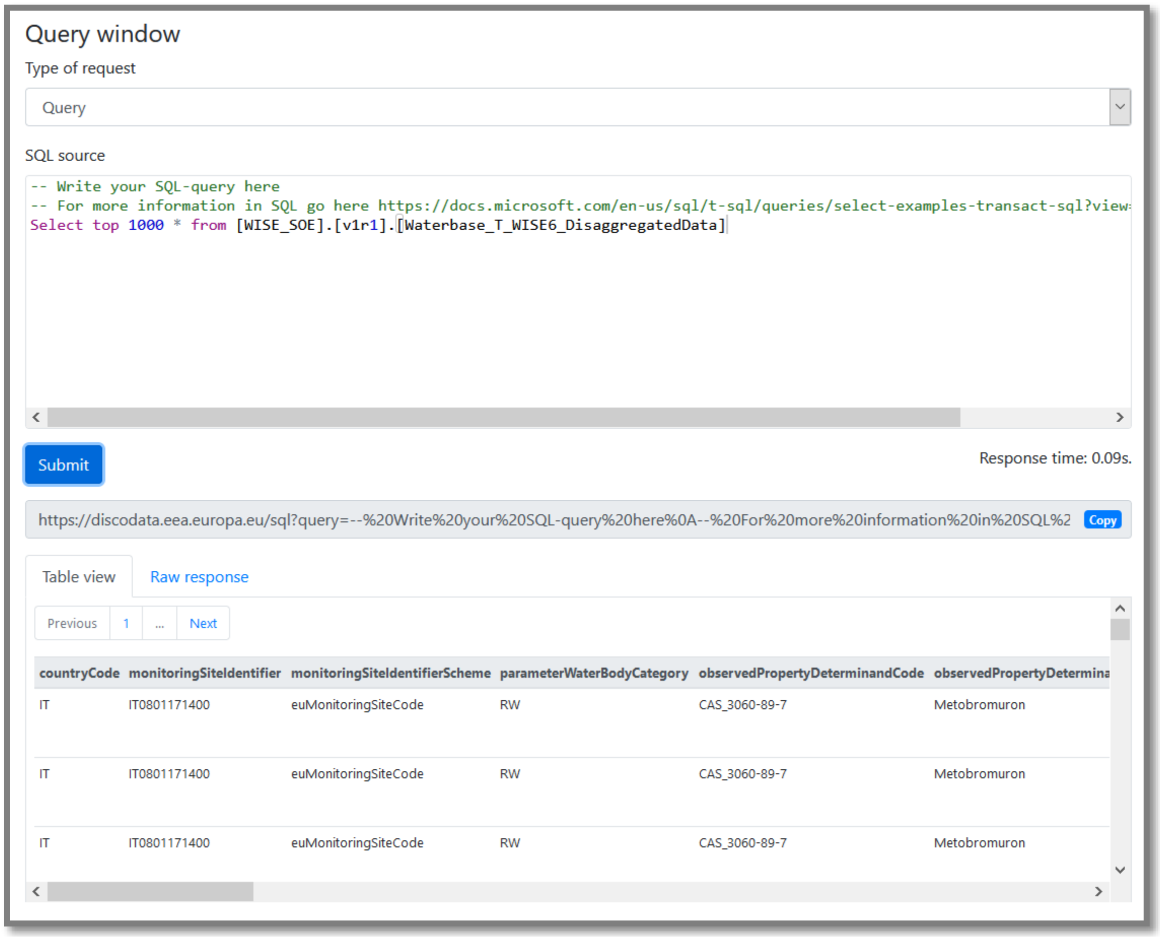

countryCode
column
Abbreviation of EEA Member or Collaborating Country.

- **Preview distinct values** – opens a window with alphabetically sorted list of all distinct values in this field.
- **Select distinct values** – clears the SQL source frame and puts in an SQL query statement which, when manually submitted, will select all distinct values in this field, and show them in the query result frame (see the Query window for more information).

Some of the fields contain alphanumeric codes instead of self-explanatory values. Explanation of these codes, or link to the respective reference, can be found in the table and dataset definition files available for download in the respective Waterbase dataset in the EEA Data and Maps portal, or in the respective Eionet data dictionary.

Query window

The Query window provides an option to create and execute SELECT type SQL queries over the Discodata databases. This allows the user to make customized selection of data and even to use more advanced SQL techniques, like table joins, aggregations, and others.



```
-- Write your SQL-query here
-- For more information in SQL go here https://docs.microsoft.com/en-us/sql/t-sql/queries/select-examples-transact-sql?view:
Select top 1000 * from [WISE_SOE].[v1r1].[Waterbase_T_WISE6_DisaggregatedData]
```

https://discodata.eea.europa.eu/sql?query=--%20Write%20your%20SQL-query%20here%0A--%20For%20more%20information%20in%20SQL%2

countryCode	monitoringSiteIdentifier	monitoringSiteIdentifierScheme	parameterWaterBodyCategory	observedPropertyDeterminandCode	observedPropertyDetermina
IT	IT0801171400	euMonitoringSiteCode	RW	CAS_3060-89-7	Metobromuron
IT	IT0801171400	euMonitoringSiteCode	RW	CAS_3060-89-7	Metobromuron
IT	IT0801171400	euMonitoringSiteCode	RW	CAS_3060-89-7	Metobromuron

Type of Request

The drop-dawn menu provides two option – **Query** and **Describe**. In this guide we will use only the **Query** option, as the Describe option is not sufficiently documented yet.

SQL Source

In this field the user can write, construct, and modify their **SQL query**, which they can then execute using the **Submit** button. See Writing a query for details on how to create a query. Examples of WISE SOE database queries can be found in the SQL query examples.

SQL query URL

The field under the Submit button, contains an URL version of the SQL query. If the URL is entered into a web browser, the result of the query is returned in JSON format. See the Discodata JSON for more information.

Result frame

The result of an SQL query is shown in two tabs underneath the URL field. The **Table view** shows the query result in a simple tabular structure, split to multiple pages. The **Raw response** tab contains the result on the current page of the Table view in JSON format. See Query result for more details.

Writing a query

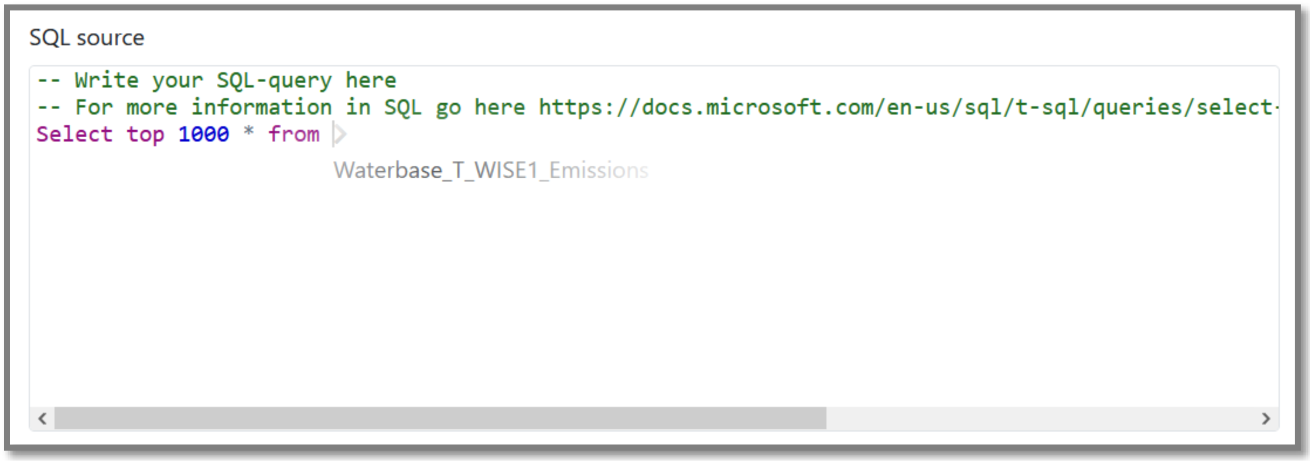
The Discodata users can use the SQL source field of the Query window to write a custom SQL query.

The Discodata SQL uses syntax and functions used in Microsoft SQL Server. As it's said in the green comment text of a default query, more information, including examples of certain SQL queries, can be found at https://docs.microsoft.com/en-us/sql/t-sql/queries/select-examples-transact-sql?view=sql-server-2017

Note: It is not a purpose of this guide to teach the SQL. Inexperienced users can, however, use the examples of various WISE SOE database queries in the SQL query examples to get an idea on how the SQL works.

A complete query can be written directly, by manually typing all the SQL statements, clauses, names of the tables and fields. To make it easier, however, it is possible to drag and drop the names of the items from the lists in the Database explorer into the SQL Source field.

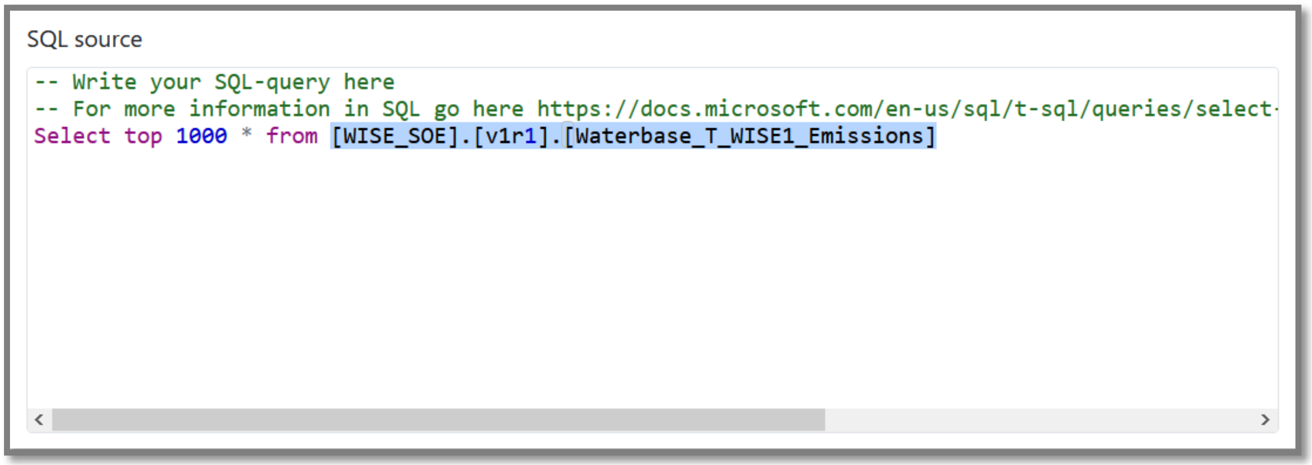
In this example we have grabbed (left click and hold) the name of the Waterbase_T_WISE1_Emissions **table** and dragged it to a proper place of the query:


SQL source

```
-- Write your SQL-query here
-- For more information in SQL go here https://docs.microsoft.com/en-us/sql/t-sql/queries/select·
Select top 1000 * from
```

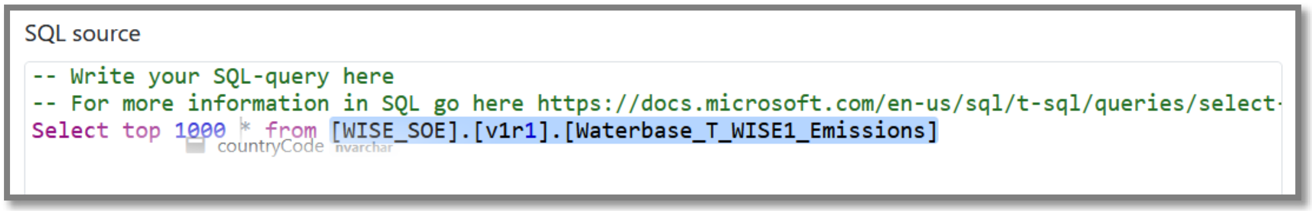
Note: The actual visual representation of the dragged item depends on the web browser you use. All screenshot in this guide were taken from the Firefox, but you should be able to use any modern web browser.

After releasing the mouse button, the table name, together with the full path, is pasted into the field:


SQL source

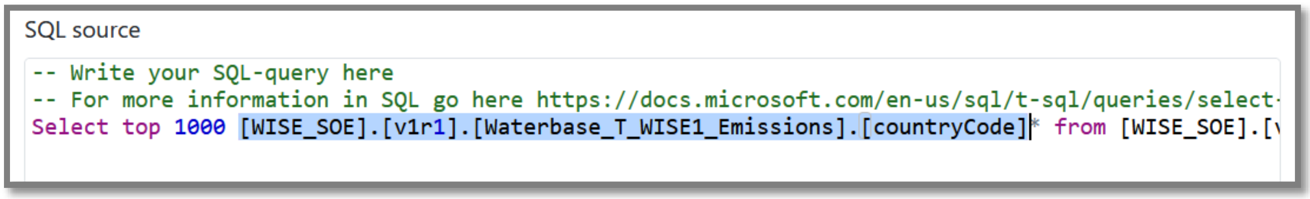
```
-- Write your SQL-query here
-- For more information in SQL go here https://docs.microsoft.com/en-us/sql/t-sql/queries/select·
Select top 1000 * from [WISE_SOE].[v1r1].[Waterbase_T_WISE1_Emissions]
```

The same drag and drop technique can be used for **fields**. Let's say that we want to select only the countryCode field values. We drag and drop the field countryCode from the table's field list into the SQL source field next to the asterisk:


SQL source

```
-- Write your SQL-query here
-- For more information in SQL go here https://docs.microsoft.com/en-us/sql/t-sql/queries/select-
Select top 1000 * from [WISE_SOE].[v1r1].[Waterbase_T_WISE1_Emissions]
```

This is what we will see after releasing the mouse button:

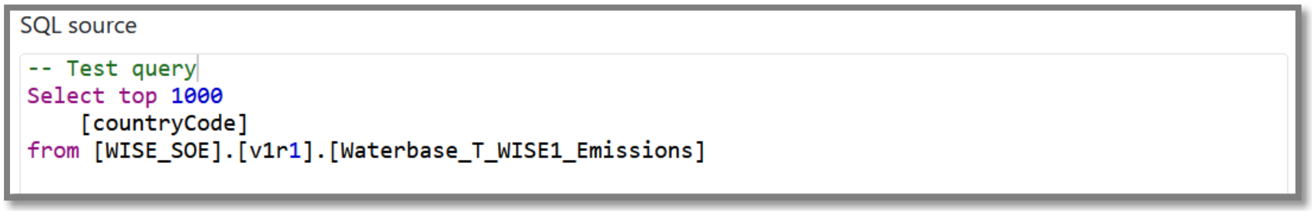

SQL source

```
-- Write your SQL-query here
-- For more information in SQL go here https://docs.microsoft.com/en-us/sql/t-sql/queries/select-
Select top 1000 [WISE_SOE].[v1r1].[Waterbase_T_WISE1_Emissions].[countryCode]* from [WISE_SOE].[v
```

We can see that the field notation is unnecessary verbose. In this example we do not need to identify the complete field path, so we can remove the notation of database, schema, and table.

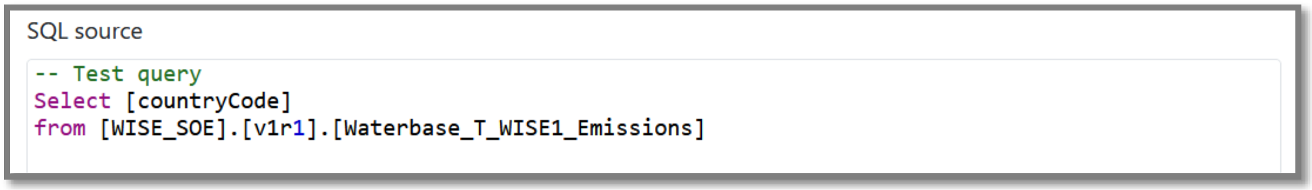
We remove the '*', as we want to select just values from one field, not from all fields.

Splitting the query into multiple lines is a good practice to make the query even more readable, especially on smaller screens. We can also remove the initial comment or replace it with our own.


SQL source

```
-- Test query
Select top 1000
    [countryCode]
from [WISE_SOE].[v1r1].[Waterbase_T_WISE1_Emissions]
```

When we are testing a query, and just want to see whether it works, it may be useful to limit the number of records the query returns. The most basic way is to use the **top** clause followed by a specific number. It can make the query finish faster. Once we decide to explore the whole result, we can remove it.


SQL source

```
-- Test query
Select [countryCode]
from [WISE_SOE].[v1r1].[Waterbase_T_WISE1_Emissions]
```

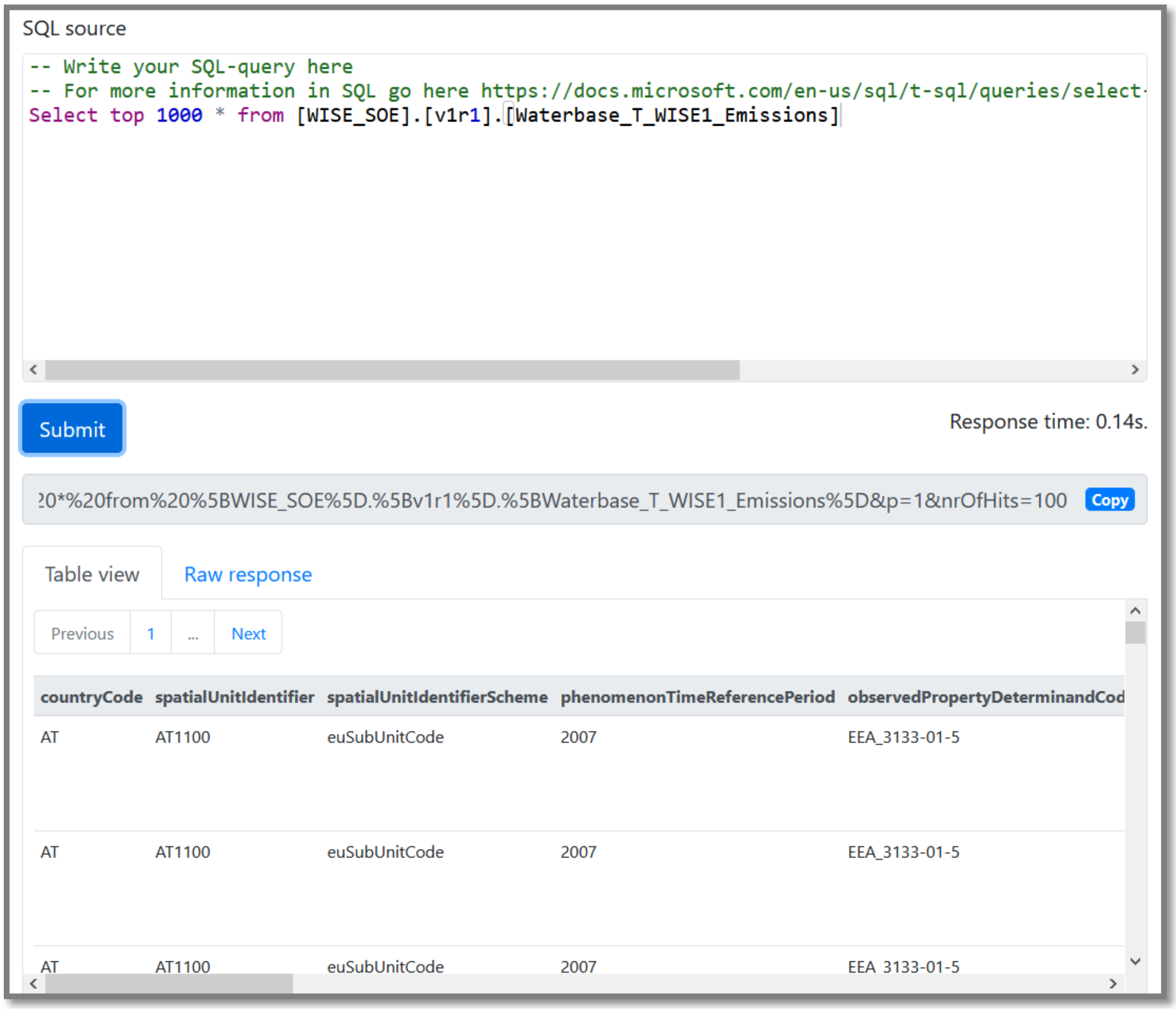
Query timeout limit

In the current Discodata configuration, if a query runs longer than 20 seconds, it ends in a timeout. This is to prevent overloading the system with many demanding queries running at the same time. The downside is, that it may not be possible to get a result of more complex queries, or queries accessing data in very large tables.

The responsible Discodata managers try to improve the query running times by adding various indexes into the database tables, but some advanced queries may still fail due to the timeout. In such case the recommended solution is trying to simplify the query or splitting the query operation to multiple steps, if possible.

Query result

A query is executed by pressing the Submit button. When the query finishes a new frame opens underneath the URL field, which contains the result of the query.



```
-- Write your SQL-query here
-- For more information in SQL go here https://docs.microsoft.com/en-us/sql/t-sql/queries/select-
Select top 1000 * from [WISE_SOE].[v1r1].[Waterbase_T_WISE1_Emissions]
```

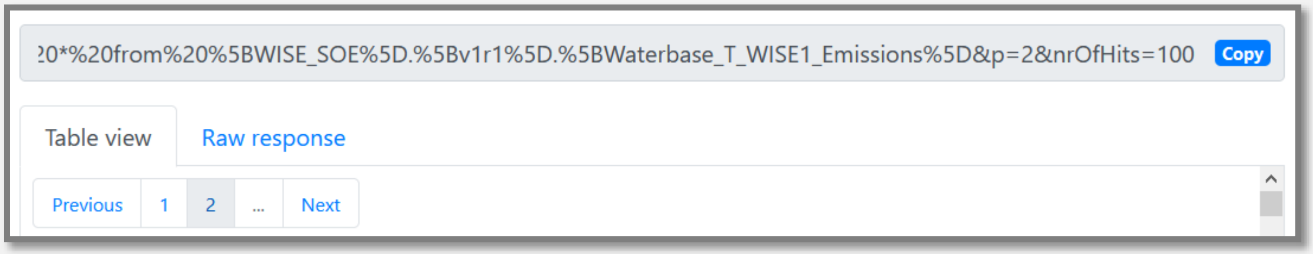
Submit
Response time: 0.14s.
20*%20from%20%5BWISE_SOE%5D.%5Bv1r1%5D.%5BWaterbase_T_WISE1_Emissions%5D&p=1&nrOfHits=100
Copy
Table view
Raw response
Previous
1
...
Next

countryCode	spatialUnitIdentifier	spatialUnitIdentifierScheme	phenomenonTimeReferencePeriod	observedPropertyDeterminandCod
AT	AT1100	euSubUnitCode	2007	EEA_3133-01-5
AT	AT1100	euSubUnitCode	2007	EEA_3133-01-5
AT	AT1100	euSubUnitCode	2007	EEA_3133-01-5

The result frame has two tabs. The **Table view** shows the query result in a simple tabular structure, split to multiple pages. The page navigation panel is at the top and the bottom of the tab. One page contains 100 records.

The **Raw response** tab contains the result in the current page of the Table view in JSON format.

Note: The SQL query URL changes depending on which result page is shown in the table view. See Discodata JSON for more details.

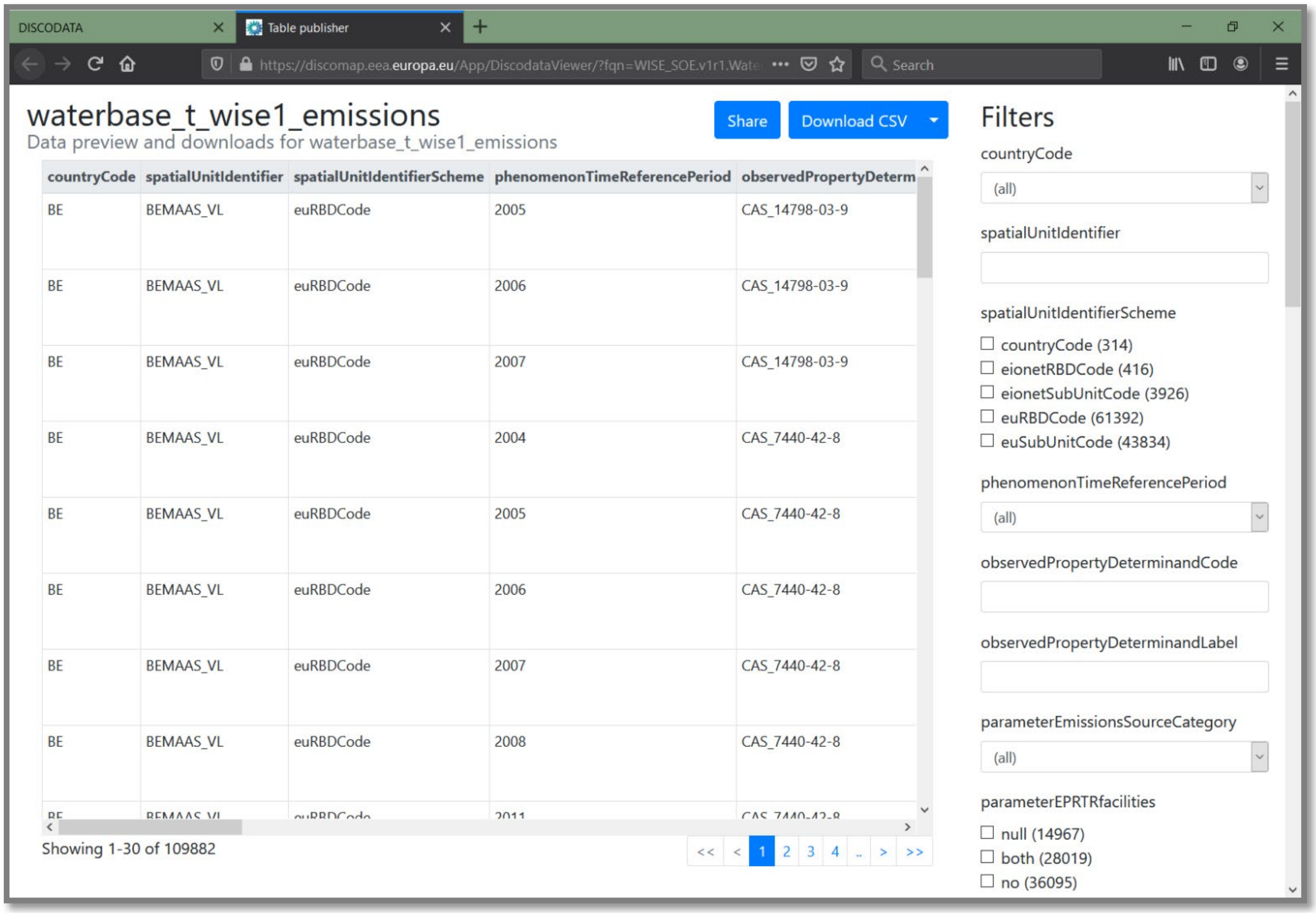

20*%20from%20%5BWISE_SOE%5D.%5Bv1r1%5D.%5BWaterbase_T_WISE1_Emissions%5D&p=2&nrOfHits=100
Copy
Table view
Raw response
Previous
1
2
...
Next

The option to download the SQL query result is currently not implemented. See Discodata JSON for a workaround solution, or the Table viewer for alternative way of selecting, filtering, and downloading data from a database table.

Table viewer

The table info blocked, opened by clicking on the three dots next to table name in the Database explorer, gives an option to open table in a table viewer. Please note that this option is available only for tables, not the views (see Database versioning for how to distinguish them).

The Table viewer opens in a new browser window or a tab.


DISCODATA
Table publisher
https://discomap.eea.europa.eu/App/DiscodataViewer/?fqn=WISE_SOE.v1r1.Wate
Search
Share
Download CSV

waterbase_t_wise1_emissions

Data preview and downloads for waterbase_t_wise1_emissions

countryCode	spatialUnitIdentifier	spatialUnitIdentifierScheme	phenomenonTimeReferencePeriod	observedPropertyDeterm
BE	BEMAAS_VL	euRBDCode	2005	CAS_14798-03-9
BE	BEMAAS_VL	euRBDCode	2006	CAS_14798-03-9
BE	BEMAAS_VL	euRBDCode	2007	CAS_14798-03-9
BE	BEMAAS_VL	euRBDCode	2004	CAS_7440-42-8
BE	BEMAAS_VL	euRBDCode	2005	CAS_7440-42-8
BE	BEMAAS_VL	euRBDCode	2006	CAS_7440-42-8
BE	BEMAAS_VL	euRBDCode	2007	CAS_7440-42-8
BE	BEMAAS_VL	euRBDCode	2008	CAS_7440-42-8
BE	BEMAAS_VL	euRBDCode	2011	CAS_7440-42-8

Showing 1-30 of 109882

<< < 1 2 3 4 .. > >>

Filters

countryCode
(all)

spatialUnitIdentifier

spatialUnitIdentifierScheme
- ☐ countryCode (314)
- ☐ eionetRBDCode (416)
- ☐ eionetSubUnitCode (3926)
- ☐ euRBDCode (61392)
- ☐ euSubUnitCode (43834)

phenomenonTimeReferencePeriod
(all)

observedPropertyDeterminandCode

observedPropertyDeterminandLabel

parameterEmissionsSourceCategory
(all)

parameterEPRTRfacilities
- ☐ null (14967)
- ☐ both (28019)
- ☐ no (36095)

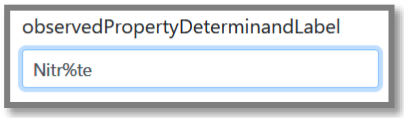
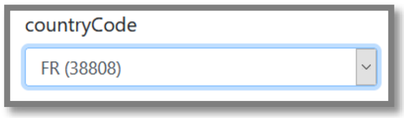
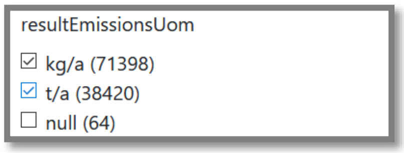
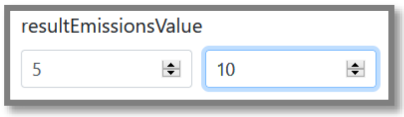
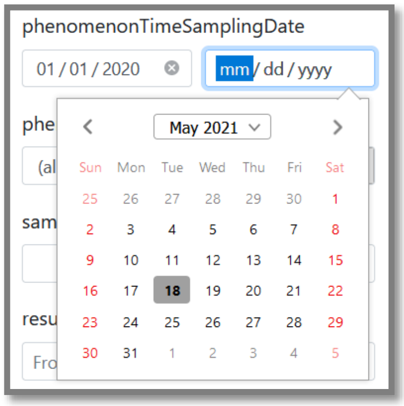
The main part of the window is taken by the data table. The table is split into pages, one page showing 30 records. The total number of the records can be seen at the bottom left, while page navigation panel is at the bottom right.

The **Share** button at the top gives user an option to copy URL of the current data view with all the filters applied, either to share with others, to reference, or to save for future use.

The **Download CSV** button allows user to download all data in the current view in CSV format. Clicking on the dropdown arrow opens additional options – download data in TSV (tab separated values) or in JSON format.

Filters

The right-hand side of the windows provides **filters** for each column in the table. The form of the filter depends on the respective column data type and number of distinct values it holds.

 observedPropertyDeterminandLabel Nitr%te	**Simple text filter.** The column contains too many distinct text values to put them into a dropdown filter. It is used also for date/time values. • The system returns all records with values that match the filter, whether fully or partially. The wildcard character '%' can be used for more advanced filtering, as shown in the example (returns both 'Nitrate' and 'Nitrite' records). • The filter is applied after pressing enter key or clicking outside the field. • Delete the filter text to reset the filter.
 countryCode FR (38808)	**Dropdown text filter.** The column contains up to 100 distinct values. • The number of records matching the current filters is shown in parentheses (disabled for most fields in large tables to improve performance). • Only one option can be selected at a time. The filters is applied after selecting a value from the dropdown menu. • Select the '(all)' option to reset the filter.
 resultEmissionsUom ☑ kg/a (71398) ☑ t/a (38420) ☐ null (64)	**Check box text filter.** The column contains up to 10 distinct values. • The number of records matching the current filters is shown in parentheses (disabled for most fields in large tables). • Multiple options can be selected at the same time. • The filter is applied when a checkbox is checked or unchecked. • Uncheck all options to reset the filter.
 resultEmissionsValue 5 10	**Numeric range filter.** The column contains numeric values. • Filter allows to define both upper and lower value limit, or just one of them. • The arrow/spinner buttons inside the fields will increase or decrease the value by 0.1 for decimals/floats or by 1 for integers. • The filter is applied after pressing enter key, clicking on a spinner button, or clicking outside the field. • Delete both filter values to reset the filter.
 phenomenonTimeSamplingDate 01 / 01 / 2020 mm / dd / yyyy May 2021 Sun Mon Tue Wed Thu Fri Sat 25 26 27 28 29 30 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 1 2 3 4 5	**Date range filter.** The column contains date values. • Filter allows to define both upper and lower value limit, or just one of them. • The date values can be entered either manually or selected using the calendar widget. It is recommended to use the widget, because manual entry, especially for year values, is sometimes difficult. Clicking on the 'month year' dropdown at the top of the widget switches to selection of year and month. • The filter is applied after selecting a day in the widget, changing a year, either in the widget or manually, after pressing enter key or clicking outside the field. • Press the circular reset buttons in both fields to reset the filter.

Application of a filter can take a significant time, depending on the number of records in the table.

The numbers of records in parentheses in the dropdown and check box filters are also recalculated every time a filter is applied. If no records with a given value match the current filters, the filter option for that value is hidden.

Discodata JSON

The SQL query result can be retrieved in JSON format. Copy the **SQL query URL** from the field under the Submit button and enter it into a web browser. This is the primary option how the Discodata datasets can be accessed by dedicated data viewer applications. It also can be used by users who are skilled in handling JSON data.

JSON to CSV conversion

One of the reasons for using the JSON data, is to work around the current lack of CSV download option for SQL query results. The user can create an SQL query, copy the resulting URL, modify it, and use it in a free JSON to CSV online converter that can be found on the Internet. Not all online converters can handle Discodata JSON though. Most of them will also have problem to convert large amount of data.

In our testing we have found that the following converter can convert JSON datasets with up to about 20000 records and provides a free download option.

https://www.convertcsv.com/json-to-csv.htm

Modifying URL

By default, the SQL result URL returns first 100 records from the query. This number can be modified.

This is an example of URL from the following SQL query:

```
-- all German and French WISE6 Aggregated data
Select *
from [WISE_SOE].[v1r1].[Waterbase_T_WISE6_AggregatedData]
where countryCode = 'DE' or countryCode = 'FR'
```

`https://discodata.eea.europa.eu/sql?query=--%20Write%20your%20SQL-query%20here%0A--%20For%20more%20information%20in%20SQL%20go%20here%20https%3A%2F%2Fdocs.microsoft.com%2Fen-us%2Fsql%2Ft-sql%2Fqueries%2Fselect-examples-transact-sql%3Fview%3Dsql-server-2017%0ASelect%20*%0AFROM%20%5BWISE_SOE%5D.%5Bv1r1%5D.%5BWaterbase_T_WISE6_AggregatedData%5D%0Awhere%20countryCode%20%3D%20'DE'%0A%20%20%20%20or%20countryCode%20%3D%20'FR'%0A&p=1&nrOfHits=100`

Please note the highlighted parameters at the end of the URL. The **p** parameter represents the page of the SQL query result, and the **nrOfHits** parameter defines number of records per page. By modifying the number after **nrOfHits** we can change the number of returned records. For example, like this:

…&p=1&nrOfHits=20000

Now the URL returns 20 000 records in JSON format.

If the query returns more records than we can safely convert using an online converter, we must split it in multiple parts by using the **p** parameter. So, to retrieve another set of 20 000 records, we modify the URL as follows:

…&p=2&nrOfHits=20000

It can certainly be a time-consuming exercise, but this way we can eventually get all required records to CSV.

As was said previously, this is a workaround solution. It is expected that CSV download of SQL query results will be eventually implemented inside the Discodata portal. In addition, EEA plans to develop a dedicated application for filtering and downloading the WISE SOE data, which will most likely include more advanced querying options.

SQL query examples

The following examples are there to give users, with no prior SQL experience, ideas on how to create Waterbase specific SQL queries. They are sorted by the level of complexity from simple to advanced. We tried to cover applications that the Waterbase users may find the most useful, but they do not focus on one particular Waterbase dataset.

Each example shows the text of the SQL query, which the users can copy and paste into the SQL source field on the Discodata portal, and the depiction of the query result they will see after pressing Submit button. Some examples may provide a short explanation or remark.

> **Note:** Discodata SQL is not case sensitive. The brackets around the object names are necessary only if the name contains a space, hyphen, or other special character (which is not the case in WISE SOE).

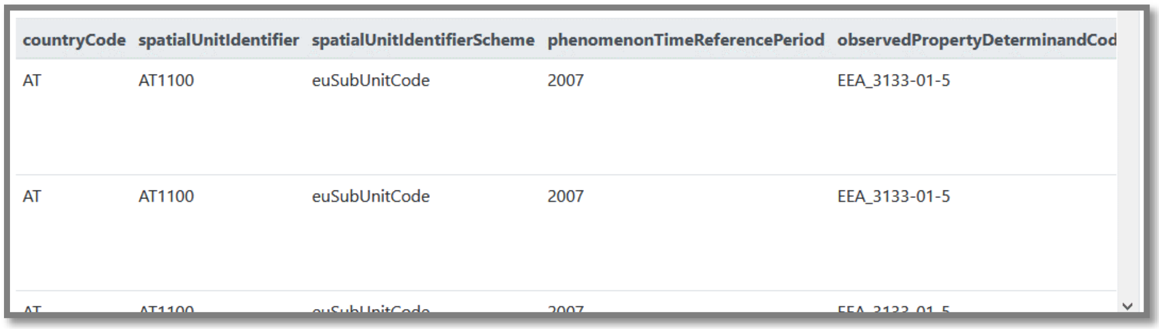
1. Basic selection

1.a Selection with result limit

SQL source

```
-- First 1000 WISE1 Emissions records
Select top 1000 * from [WISE_SOE].[v1r1].[Waterbase_T_WISE1_Emissions]
```

SQL result



countryCode	spatialUnitIdentifier	spatialUnitIdentifierScheme	phenomenonTimeReferencePeriod	observedPropertyDeterminandCod
AT	AT1100	euSubUnitCode	2007	EEA_3133-01-5
AT	AT1100	euSubUnitCode	2007	EEA_3133-01-5

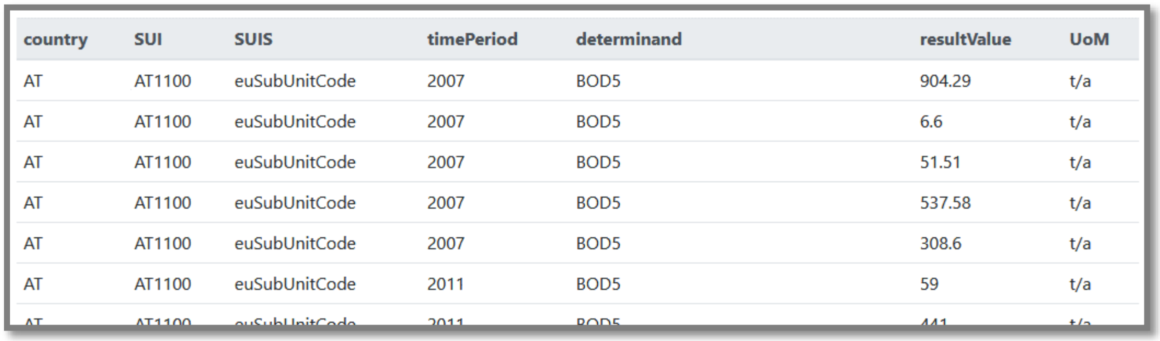
1.b Selection with specified result fields and field label aliases

Specifying, which fields we want in the result, and giving them an alias, will make the result more readable.

SQL source

```
-- Select all WISE1 Emissions records, specify fields and shorten their labels in the result
Select [countryCode] as country
    ,[spatialUnitIdentifier] as SUI
    ,[spatialUnitIdentifierScheme] as SUIS
    ,[phenomenonTimeReferencePeriod] as timePeriod
    ,[observedPropertyDeterminandLabel] as determinand
    ,[resultEmissionsValue] as resultValue
    ,[resultEmissionsUom] as UoM
from [WISE_SOE].[v1r1].[Waterbase_T_WISE1_Emissions]
```

SQL result



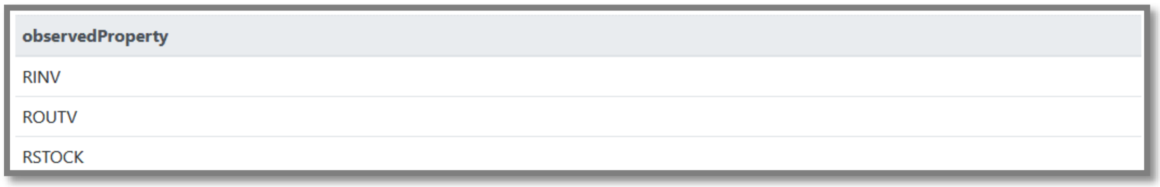
country	SUI	SUIS	timePeriod	determinand	resultValue	UoM
AT	AT1100	euSubUnitCode	2007	BOD5	904.29	t/a
AT	AT1100	euSubUnitCode	2007	BOD5	6.6	t/a
AT	AT1100	euSubUnitCode	2007	BOD5	51.51	t/a
AT	AT1100	euSubUnitCode	2007	BOD5	537.58	t/a
AT	AT1100	euSubUnitCode	2007	BOD5	308.6	t/a
AT	AT1100	euSubUnitCode	2011	BOD5	59	t/a

1.c Selection of unique values

SQL source

```sql
-- List of unique observed property values from WISE3 Reservoir data
Select Distinct [observedProperty]
from [WISE_SOE].[v1r1].[Waterbase_T_WISE3_ReservoirData]
```

SQL result



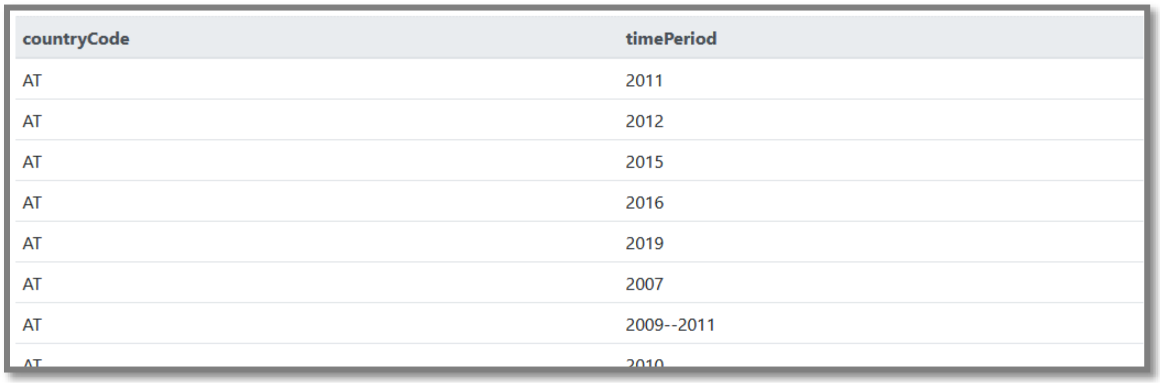
observedProperty
RINV
ROUTV
RSTOCK

1.d Selection of unique value combinations

SQL source

```sql
-- WISE1 Emissions - list of unique combinations of country and time period
Select Distinct [countryCode], [phenomenonTimeReferencePeriod] as timePeriod
from [WISE_SOE].[v1r1].[Waterbase_T_WISE1_Emissions]
```

SQL result



countryCode	timePeriod
AT	2011
AT	2012
AT	2015
AT	2016
AT	2019
AT	2007
AT	2009--2011
AT	2010

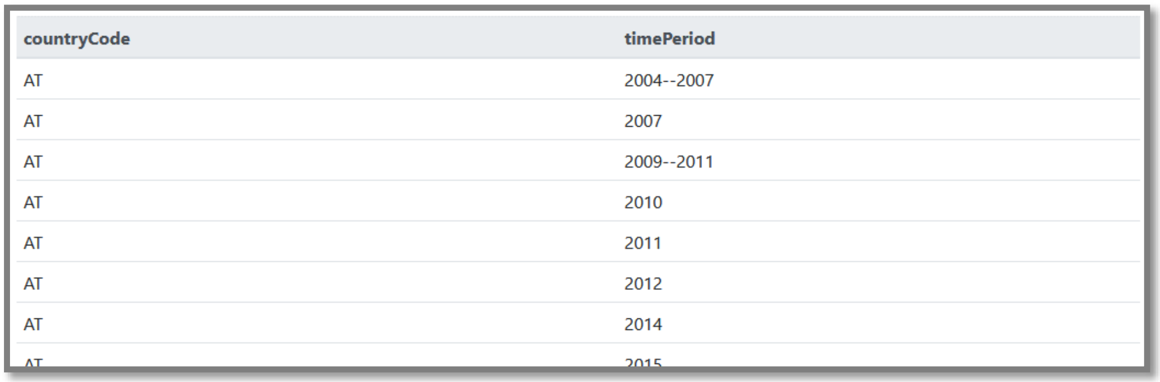
1.e Order by clause

Order by clause sorts the result by specified fields, making it more readable. The downside is, that the query will usually take longer to finish. Also, be aware that in Discodata, if you want to sort the result by a field with an alias, instead of the field name you must use the alias in the Order by clause. Let's order the result of 1.d.

SQL source

```sql
-- WISE1 Emissions – sorted list of unique combinations of country and time period
Select Distinct [countryCode], [phenomenonTimeReferencePeriod] as timePeriod
from [WISE_SOE].[v1r1].[Waterbase_T_WISE1_Emissions]
Order by [countryCode], timePeriod
```

SQL result



countryCode	timePeriod
AT	2004--2007
AT	2007
AT	2009--2011
AT	2010
AT	2011
AT	2012
AT	2014
AT	2015

2. Filtered selection

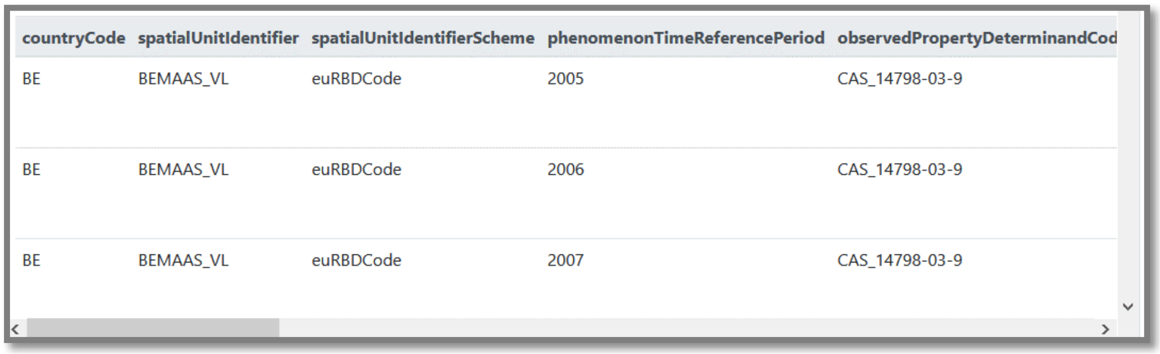
Selection of specific records is the main reason to use the SQL query.

2.a Selection with one filter

SQL source

```
-- All Belgian WISE1 Emissions records
Select *
from [WISE_SOE].[v1r1].[Waterbase_T_WISE1_Emissions]
where countryCode = 'BE'
```

SQL result



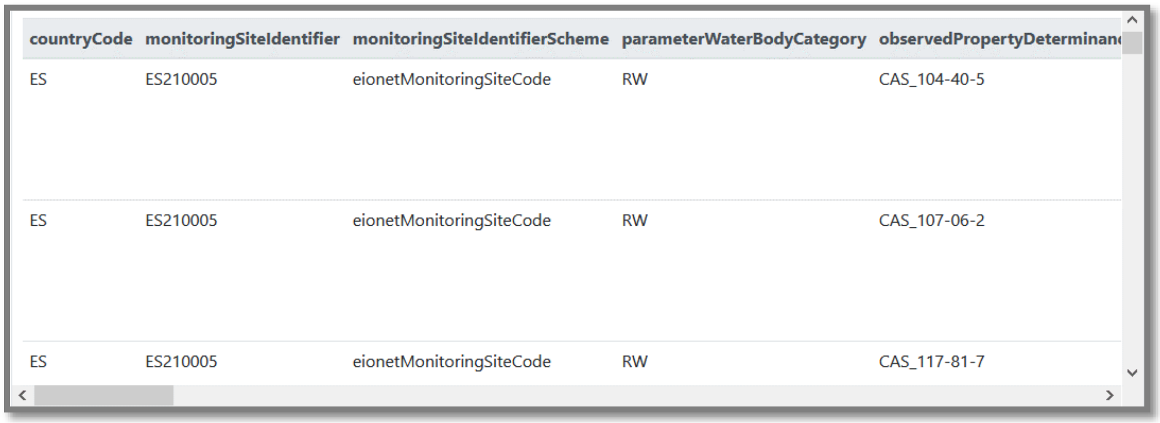
countryCode	spatialUnitIdentifier	spatialUnitIdentifierScheme	phenomenonTimeReferencePeriod	observedPropertyDeterminandCod
BE	BEMAAS_VL	euRBDCode	2005	CAS_14798-03-9
BE	BEMAAS_VL	euRBDCode	2006	CAS_14798-03-9
BE	BEMAAS_VL	euRBDCode	2007	CAS_14798-03-9

2.b Selection with multiple filters

SQL source

```
-- All Spanish and Portuguese WISE6 Aggregated Data records from between 2010 and 20015
Select *
from [WISE_SOE].[v1r1].[Waterbase_T_WISE6_AggregatedData]
where (countryCode = 'ES' or countryCode = 'PT')
    and [phenomenonTimeReferenceYear] >= 2010
    and [phenomenonTimeReferenceYear] <= 2015
```

SQL result



countryCode	monitoringSiteIdentifier	monitoringSiteIdentifierScheme	parameterWaterBodyCategory	observedPropertyDeterminan
ES	ES210005	eionetMonitoringSiteCode	RW	CAS_104-40-5
ES	ES210005	eionetMonitoringSiteCode	RW	CAS_107-06-2
ES	ES210005	eionetMonitoringSiteCode	RW	CAS_117-81-7

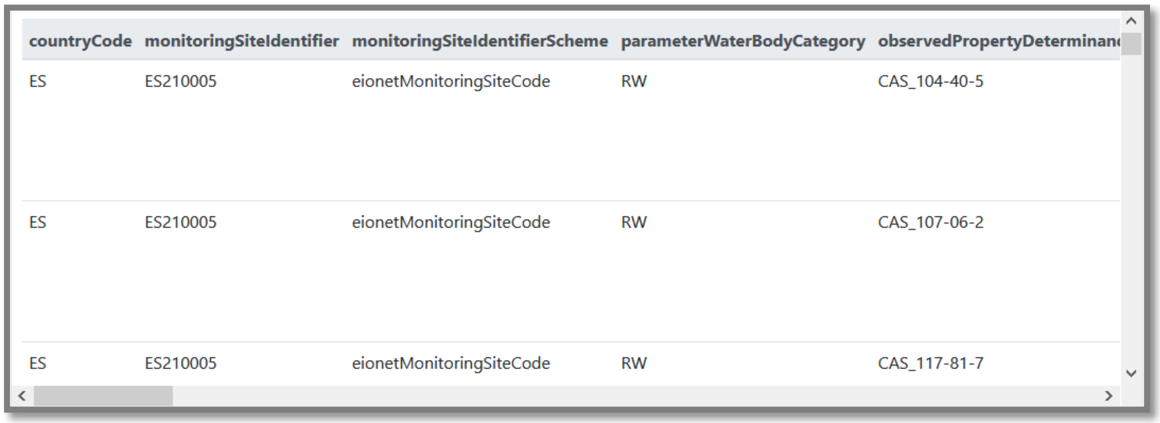
2.c Selection with Between operator filter

The previous query can be simplified using this operator, that selects values in range.

SQL source

```
-- All Spanish and Portuguese WISE6 Aggregated Data records from between 2010 and 20015
Select *
from [WISE_SOE].[v1r1].[Waterbase_T_WISE6_AggregatedData]
where (countryCode = 'ES' or countryCode = 'PT')
    and [phenomenonTimeReferenceYear] between 2010 and 2015
```

SQL result



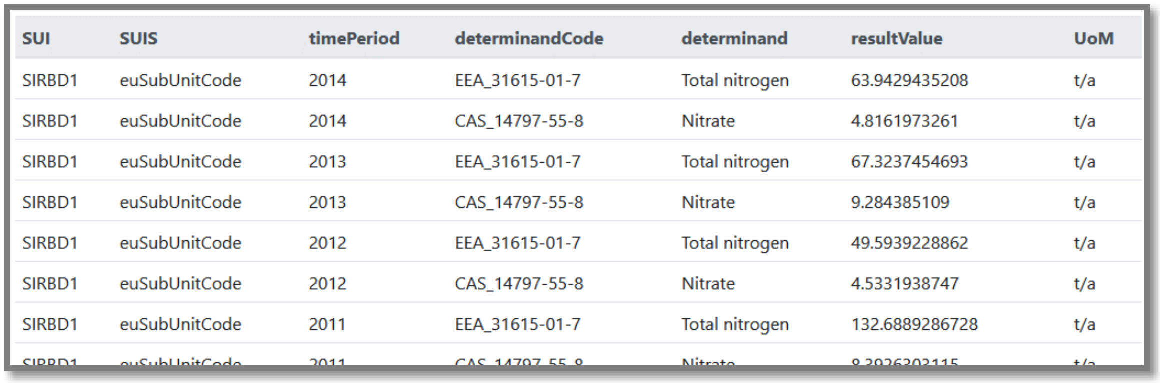
countryCode	monitoringSiteIdentifier	monitoringSiteIdentifierScheme	parameterWaterBodyCategory	observedPropertyDeterminan
ES	ES210005	eionetMonitoringSiteCode	RW	CAS_104-40-5
ES	ES210005	eionetMonitoringSiteCode	RW	CAS_107-06-2
ES	ES210005	eionetMonitoringSiteCode	RW	CAS_117-81-7

2.d Selection with a wildcard filter

SQL source

```
-- All Slovenian WISE1 Emissions data with a determinand label containing text 'nitr'
Select [spatialUnitIdentifier] as SUI
     ,[spatialUnitIdentifierScheme] as SUIS
     ,[phenomenonTimeReferencePeriod] as timePeriod
     ,[observedPropertyDeterminandCode] as determinandCode
     ,[observedPropertyDeterminandLabel] as determinand
     ,[resultEmissionsValue] as resultValue
     ,[resultEmissionsUom] as UoM
from [WISE_SOE].[v1r1].[Waterbase_T_WISE1_Emissions]
where (countryCode = 'SI')
    and observedPropertyDeterminandLabel like '%nitr%'
```

SQL result



SUI	SUIS	timePeriod	determinandCode	determinand	resultValue	UoM
SIRBD1	euSubUnitCode	2014	EEA_31615-01-7	Total nitrogen	63.9429435208	t/a
SIRBD1	euSubUnitCode	2014	CAS_14797-55-8	Nitrate	4.8161973261	t/a
SIRBD1	euSubUnitCode	2013	EEA_31615-01-7	Total nitrogen	67.3237454693	t/a
SIRBD1	euSubUnitCode	2013	CAS_14797-55-8	Nitrate	9.284385109	t/a
SIRBD1	euSubUnitCode	2012	EEA_31615-01-7	Total nitrogen	49.5939228862	t/a
SIRBD1	euSubUnitCode	2012	CAS_14797-55-8	Nitrate	4.5331938747	t/a
SIRBD1	euSubUnitCode	2011	EEA_31615-01-7	Total nitrogen	132.6889286728	t/a
SIRBD1	euSubUnitCode	2011	CAS_14797-55-8	Nitrate	9.3926303115	t/a

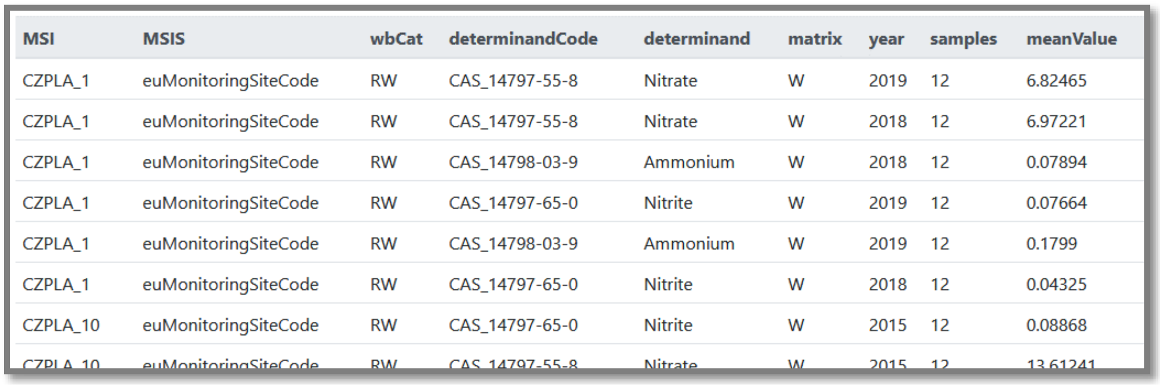
2.e Selection with IN operator filter

The **IN** clause, followed by a comma separated list of values enclosed in parentheses, simplifies filtering by multiple distinct values from one column.

SQL source

```
-- Select 2015 Nitrate, Nitrite and Ammonium WISE6 Aggregated Data from Czechia
Select [monitoringSiteIdentifier] as MSI
    ,[monitoringSiteIdentifierScheme] as MSIS
    ,[parameterWaterBodyCategory] as wbCat
    ,[observedPropertyDeterminandCode] as determinandCode
    ,[observedPropertyDeterminandLabel] as determinand
    ,[procedureAnalysedMatrix] as matrix
    ,[phenomenonTimeReferenceYear] as year
    ,[resultNumberOfSamples] as samples
    ,[resultMeanValue] as meanValue
from [WISE_SOE].[v1r1].[Waterbase_T_WISE6_AggregatedData]
where countryCode = 'CZ'
    and [phenomenonTimeReferenceYear] >= '2015'
    and [observedPropertyDeterminandCode] IN ('CAS_14798-03-9','CAS_14797-55-8','CAS_14797-65-0')
```

SQL result



MSI	MSIS	wbCat	determinandCode	determinand	matrix	year	samples	meanValue
CZPLA_1	euMonitoringSiteCode	RW	CAS_14797-55-8	Nitrate	W	2019	12	6.82465
CZPLA_1	euMonitoringSiteCode	RW	CAS_14797-55-8	Nitrate	W	2018	12	6.97221
CZPLA_1	euMonitoringSiteCode	RW	CAS_14798-03-9	Ammonium	W	2018	12	0.07894
CZPLA_1	euMonitoringSiteCode	RW	CAS_14797-65-0	Nitrite	W	2019	12	0.07664
CZPLA_1	euMonitoringSiteCode	RW	CAS_14798-03-9	Ammonium	W	2019	12	0.1799
CZPLA_1	euMonitoringSiteCode	RW	CAS_14797-65-0	Nitrite	W	2018	12	0.04325
CZPLA_10	euMonitoringSiteCode	RW	CAS_14797-65-0	Nitrite	W	2015	12	0.08868
CZPLA_10	euMonitoringSiteCode	RW	CAS_14797-55-8	Nitrate	W	2015	12	13.61241

3. Aggregation functions

3.a Count

Count is the most used aggregation function in SQL.

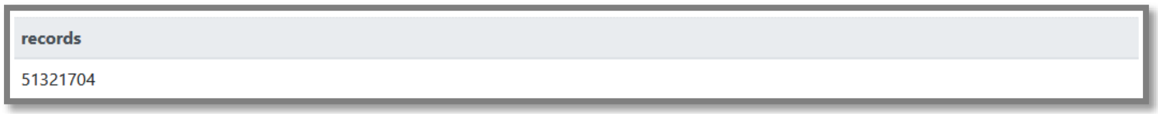
3.a.1 Basic record count

Knowing how many records is there in a specific table can provide a better idea on how challenging the handling will be.

SQL source

```
-- Count all WISE6 Disaggregated data records
Select count(*) as records
from [WISE_SOE].[v1r1].[Waterbase_T_WISE6_DisaggregatedData]
```

SQL result



records
51321704

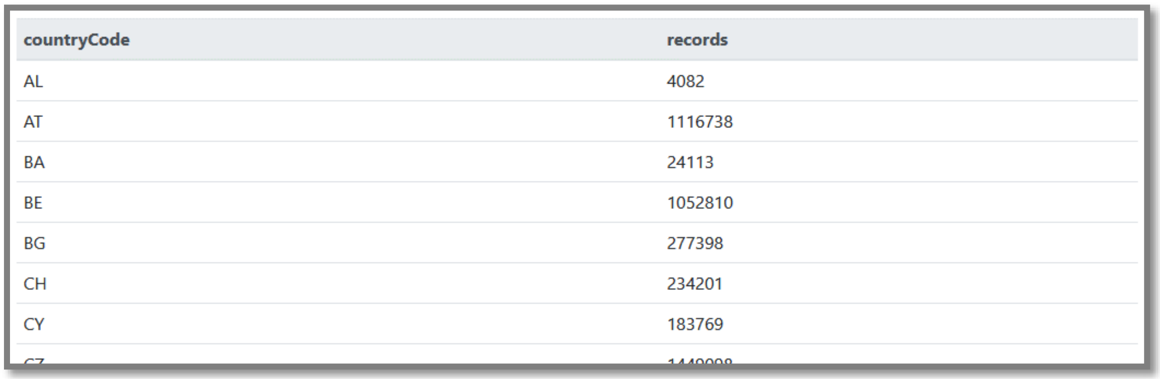
3.a.2 Record count grouped by one category

Group by clause allows us to subdivide aggregations into categories.

SQL source

```sql
-- Number of WISE6 Disaggregated data records per country
Select [countryCode], count(*) as records
from [WISE_SOE].[v1r1].[Waterbase_T_WISE6_DisaggregatedData]
group by [countryCode]
```

SQL result



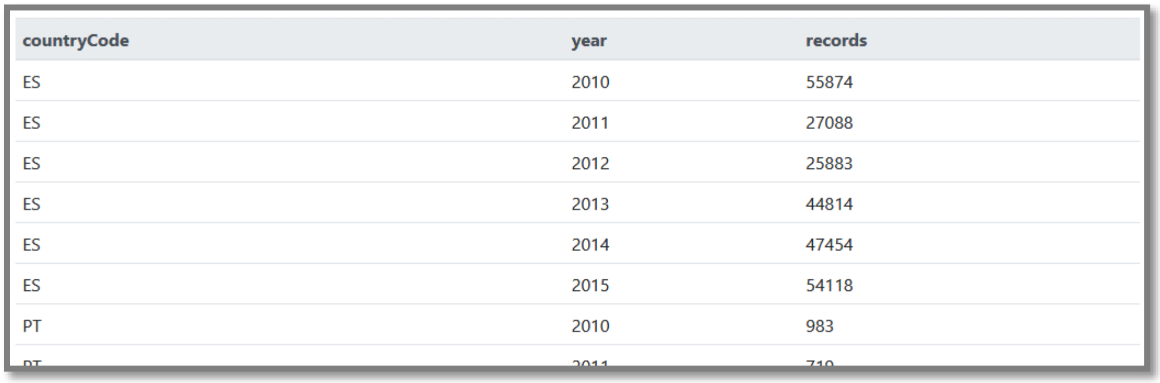
countryCode	records
AL	4082
AT	1116738
BA	24113
BE	1052810
BG	277398
CH	234201
CY	183769

3.a.3 Filtered record count grouped by multiple categories

SQL source

```sql
-- Yearly record count of all Spanish and Portuguese WISE6 Aggregated Data records from between 2010 and 20015
Select [countryCode], [phenomenonTimeReferenceYear] as year, count(*) as records
from [WISE_SOE].[v1r1].[Waterbase_T_WISE6_AggregatedData]
where (countryCode = 'ES' or countryCode = 'PT')
    and [phenomenonTimeReferenceYear] >= 2010
    and [phenomenonTimeReferenceYear] <= 2015
group by [countryCode], [phenomenonTimeReferenceYear]
order by [countryCode], year
```

SQL result



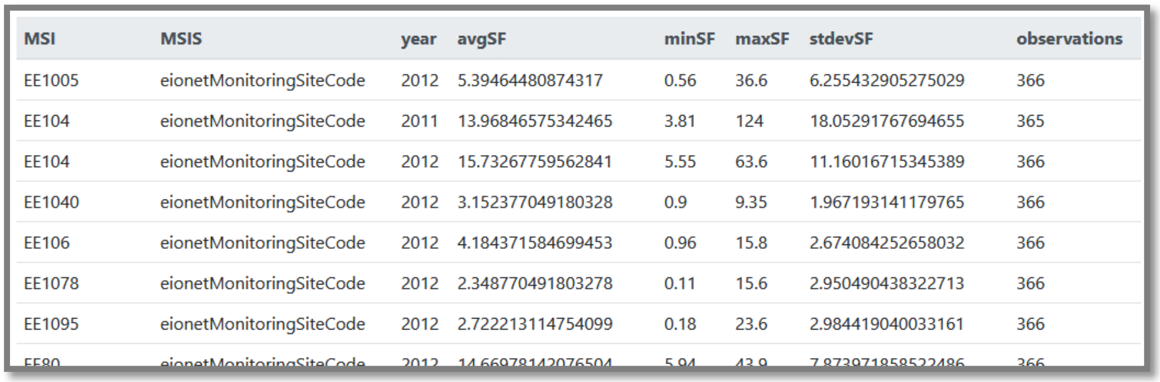
countryCode	year	records
ES	2010	55874
ES	2011	27088
ES	2012	25883
ES	2013	44814
ES	2014	47454
ES	2015	54118
PT	2010	983

3.b Average, Minimum, Maximum, Standard deviation

SQL source

```sql
-- Calculate yearly average, minimum, maximum, standard deviation values, and total number of streamflow observation in
all Estonian monitoring sites from WISE3 Monitoring data table
Select [monitoringSiteIdentifier] as MSI
    ,[monitoringSiteIdentifierScheme] as MSIS
    ,[phenomenonTimePeriod_year] as year
    ,avg([resultObservedValue]) as avgSF
    ,min([resultObservedValue]) as minSF
    ,max([resultObservedValue]) as maxSF
    ,stdev([resultObservedValue]) as stdevSF
    ,count(*) as observations
from [WISE_SOE].[v1r1].[Waterbase_T_WISE3_MonitoringData]
where countryCode = 'EE'
    and observedProperty = 'SF'
group by [monitoringSiteIdentifier],[monitoringSiteIdentifierScheme],[phenomenonTimePeriod_year]
order by MSI, MSIS, year
```

SQL result



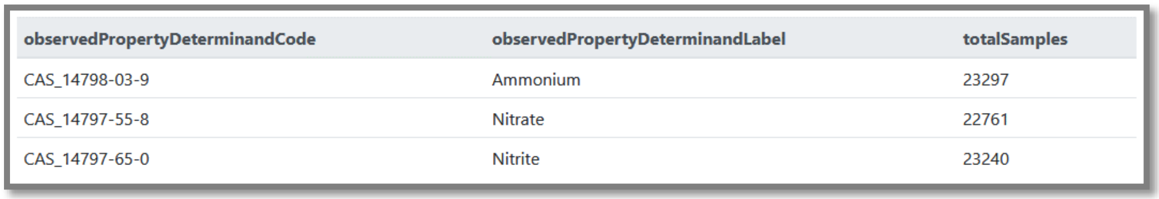
MSI	MSIS	year	avgSF	minSF	maxSF	stdevSF	observations
EE1005	eionetMonitoringSiteCode	2012	5.39464480874317	0.56	36.6	6.255432905275029	366
EE104	eionetMonitoringSiteCode	2011	13.96846575342465	3.81	124	18.05291767694655	365
EE104	eionetMonitoringSiteCode	2012	15.73267759562841	5.55	63.6	11.16016715345389	366
EE1040	eionetMonitoringSiteCode	2012	3.152377049180328	0.9	9.35	1.967193141179765	366
EE106	eionetMonitoringSiteCode	2012	4.184371584699453	0.96	15.8	2.674084252658032	366
EE1078	eionetMonitoringSiteCode	2012	2.348770491803278	0.11	15.6	2.950490438322713	366
EE1095	eionetMonitoringSiteCode	2012	2.722213114754099	0.18	23.6	2.984419040033161	366
EE80	eionetMonitoringSiteCode	2012	14.66978142076504	5.94	43.9	7.873971858522486	366

3.c Sum

SQL source

```sql
-- WISE6 Aggregated Data - total number of 2015 Nitrate, Nitrite and Ammonium samples from Czechia
Select [observedPropertyDeterminandCode]
    ,[observedPropertyDeterminandLabel]
    ,sum([resultNumberOfSamples]) as totalSamples
from [WISE_SOE].[v1r1].[Waterbase_T_WISE6_AggregatedData]
where countryCode = 'CZ'
    and [phenomenonTimeReferenceYear] >= '2015'
    and [observedPropertyDeterminandCode] in ('CAS_14798-03-9','CAS_14797-55-8','CAS_14797-65-0')
group by [observedPropertyDeterminandCode],[observedPropertyDeterminandLabel]
order by [observedPropertyDeterminandLabel]
```

SQL result



observedPropertyDeterminandCode	observedPropertyDeterminandLabel	totalSamples
CAS_14798-03-9	Ammonium	23297
CAS_14797-55-8	Nitrate	22761
CAS_14797-65-0	Nitrite	23240

4. Nested and Join queries

Nesting queries allows for results of one query to be used as a source for another. Joining multiple tables or query results with a use of common fields is a powerful tool for advanced data analysis.

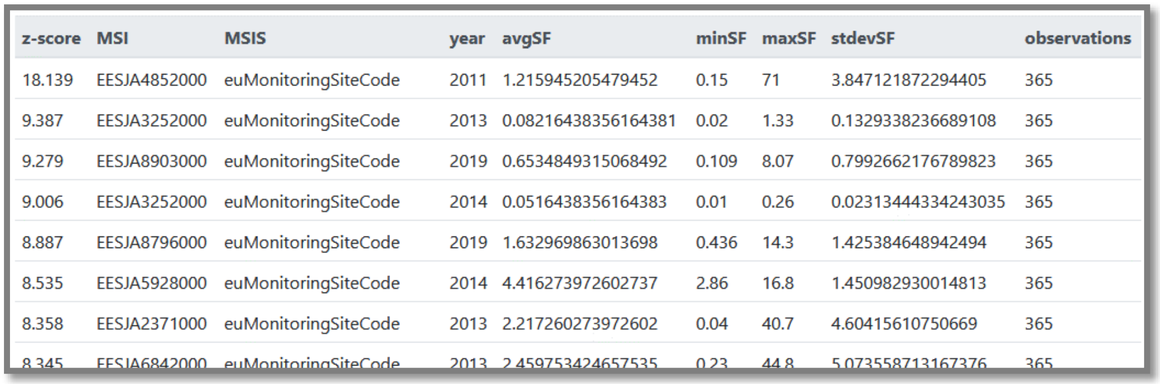
4.a Basic nested query – Z-score calculation

In this example we use the result from the example 3.b to calculate Z-score values and to find timeseries where it is more than 3 for the maximum value (may indicate outliers). Please note the use of **Round** function to reduce number of decimals, and the **Order by** modifier **DESC** to sort the result by the z-score value in descending order. Notice that the nested query is enclosed in parentheses and must be named by using an alias.

SQL source

```
-- WISE3 - Monitoring data
-- Estonian yearly streamflow aggregations where Z-score of the maximum value is > 3
Select round((maxSF - avgSF)/stdevSF,3) as [z-score], *
from (
    Select [monitoringSiteIdentifier] as MSI
        ,[monitoringSiteIdentifierScheme] as MSIS
        ,[phenomenonTimePeriod_year] as year
        ,avg([resultObservedValue]) as avgSF
        ,min([resultObservedValue]) as minSF
        ,max([resultObservedValue]) as maxSF
        ,stdev([resultObservedValue]) as stdevSF
        ,count(*) as observations
    from [WISE_SOE].[v1r1].[Waterbase_T_WISE3_MonitoringData]
    where countryCode = 'EE'
        and observedProperty = 'SF'
    group by [monitoringSiteIdentifier],[monitoringSiteIdentifierScheme],[phenomenonTimePeriod_year]
) as stats
where (maxSF - avgSF)/stdevSF > 3
order by [z-score] DESC
```

SQL result



z-score	MSI	MSIS	year	avgSF	minSF	maxSF	stdevSF	observations
18.139	EESJA4852000	euMonitoringSiteCode	2011	1.215945205479452	0.15	71	3.847121872294405	365
9.387	EESJA3252000	euMonitoringSiteCode	2013	0.08216438356164381	0.02	1.33	0.1329338236689108	365
9.279	EESJA8903000	euMonitoringSiteCode	2019	0.6534849315068492	0.109	8.07	0.7992662176789823	365
9.006	EESJA3252000	euMonitoringSiteCode	2014	0.0516438356164383	0.01	0.26	0.02313444334243035	365
8.887	EESJA8796000	euMonitoringSiteCode	2019	1.632969863013698	0.436	14.3	1.425384648942494	365
8.535	EESJA5928000	euMonitoringSiteCode	2014	4.416273972602737	2.86	16.8	1.450982930014813	365
8.358	EESJA2371000	euMonitoringSiteCode	2013	2.217260273972602	0.04	40.7	4.60415610750669	365
8.345	EESJA6842000	euMonitoringSiteCode	2013	2.459753424657535	0.23	44.8	5.073558713167376	365

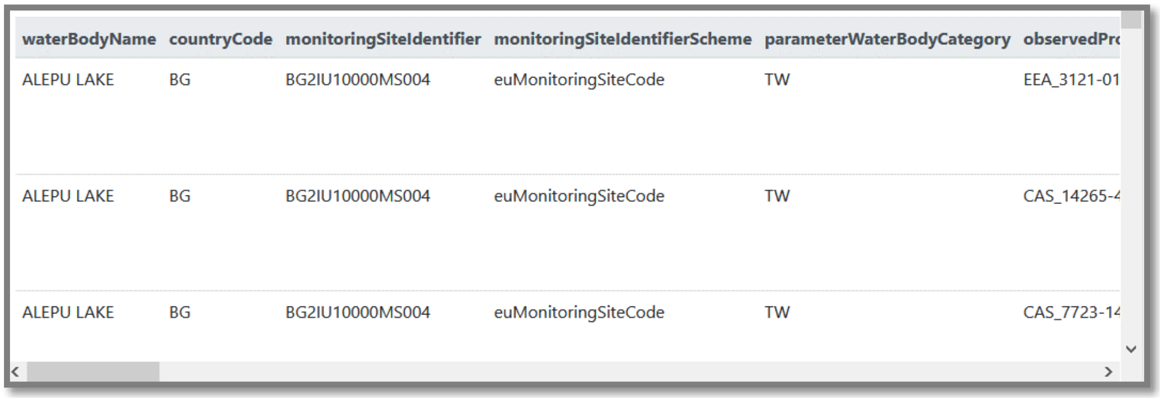
4.b Inner Join – join two data sources

Inner Join is the basic type of join. It returns records from both sides, whether it is tables or queries, that have a matching key or keys. The connection between the two is defined in the **ON** clause. In this example we join a timeseries table with the spatial metadata table, to retrieve a specific value from the spatial table.

SQL source

```
-- Select name of the respective water body with all Greek WISE6 Aggregated records where year >= 2010
Select sd.[waterBodyName], ad.*
from [WISE_SOE].[v1r1].[Waterbase_T_WISE6_AggregatedData] as ad
Inner Join [WISE_SOE].[v1r1].[Waterbase_S_WISE_SpatialObject_DerivedData] as sd
    ON ad.[monitoringSiteIdentifier] = sd.[monitoringSiteIdentifier]
        and ad.[monitoringSiteIdentifierScheme] = sd.[monitoringSiteIdentifierScheme]
where ad.countryCode = 'BG'
    and ad.phenomenonTimeReferenceYear >= 2010
```

SQL result



waterBodyName	countryCode	monitoringSiteIdentifier	monitoringSiteIdentifierScheme	parameterWaterBodyCategory	observedPr
ALEPU LAKE	BG	BG2IU10000MS004	euMonitoringSiteCode	TW	EEA_3121-01
ALEPU LAKE	BG	BG2IU10000MS004	euMonitoringSiteCode	TW	CAS_14265-4
ALEPU LAKE	BG	BG2IU10000MS004	euMonitoringSiteCode	TW	CAS_7723-14

4.c Left Join – find missing records

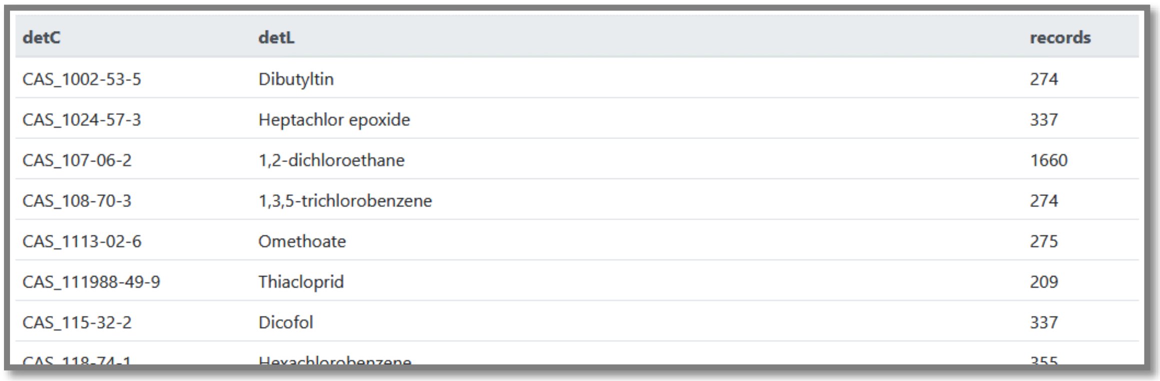
The **Left Join** (or its opposite Right Join) retrieves all records from the left (or right) source, independently whether there is a matching record on the other side. The left source is the one with keys of the left side of the ON clause. This join is useful, for example, to find records that do not have a counterpart on the other side.

In the example, see the use of nested queries querying the same table, and the **is null** filter operator.

SQL source

```
-- Find WISE6 DisaggregatedData determinands reported by Austria in period 2010-2018, but not later
Select olderData.detC, olderData.detL, olderData.records
from (
    Select [observedPropertyDeterminandCode] as detC
        ,[observedPropertyDeterminandLabel] as detL
        , count(*) as records
    from [WISE_SOE].[v1r1].[Waterbase_T_WISE6_DisaggregatedData]
    where countryCode = 'AT'
        and [phenomenonTimeSamplingDate_year] between 2010 and 2018
    group by [observedPropertyDeterminandCode], [observedPropertyDeterminandLabel]
) as olderData
Left Join (
    Select [observedPropertyDeterminandCode] as detC
    from [WISE_SOE].[v1r1].[Waterbase_T_WISE6_DisaggregatedData]
    where countryCode = 'AT'
        and [phenomenonTimeSamplingDate_year] > 2018
    group by [observedPropertyDeterminandCode]
) as latestData
    on olderData.detC = latestData.detC
where latestData.detC is null
```

SQL result



detC	detL	records
CAS_1002-53-5	Dibutyltin	274
CAS_1024-57-3	Heptachlor epoxide	337
CAS_107-06-2	1,2-dichloroethane	1660
CAS_108-70-3	1,3,5-trichlorobenzene	274
CAS_1113-02-6	Omethoate	275
CAS_111988-49-9	Thiacloprid	209
CAS_115-32-2	Dicofol	337
CAS_118-74-1	Hexachlorobenzene	355

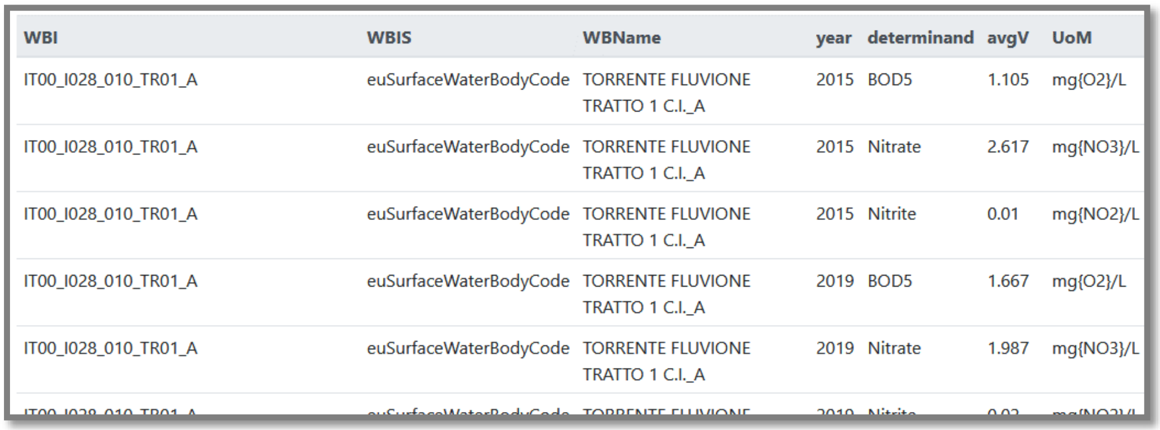
4.d Two step aggregation

This example joins results of two nested queries from timeseries and spatial table and aggregates them to calculate statistics on higher spatial level.

SQL source

```
-- From WISE6 Disaggregated Data calculate yearly mean observation values
-- for selected determinands from water matrix in Italian Water Bodies
Select so.WBI, so.WBIS, so.WBName, ts.year, ts.determinand, round(avg(ts.avgV),3) as avgV, ts.UoM
from (
        Select [monitoringSiteIdentifier] as MSI
                ,[monitoringSiteIdentifierScheme] as MSIS
                ,[phenomenonTimeSamplingDate_year] as year
                ,[observedPropertyDeterminandLabel] as determinand
                ,avg(resultObservedValue) as avgV
                ,[resultUom] as UoM
                from [WISE_SOE].[v1r1].[Waterbase_T_WISE6_DisaggregatedData]
        where [countryCode] = 'IT'
                and [phenomenonTimeSamplingDate_year] >= 2010
                and [procedureAnalysedMatrix] = 'W'
                and [observedPropertyDeterminandLabel] in ('Nitrate','Nitrite','BOD5')
        group by [monitoringSiteIdentifier],[monitoringSiteIdentifierScheme],[phenomenonTimeSamplingDate_year]
                ,[observedPropertyDeterminandLabel],[resultUom]
) as ts
Inner Join (
        Select [monitoringSiteIdentifier] as MSI
                ,[monitoringSiteIdentifierScheme] as MSIS
                ,[waterBodyIdentifier] as WBI
                ,[waterBodyIdentifierScheme] as WBIS
                ,[waterBodyName] as WBName
        from [WISE_SOE].[v1r1].[Waterbase_S_WISE_SpatialObject_DerivedData]
        where countryCode = 'IT'
                and [waterBodyIdentifier] is not null
) as so
        on ts.MSI = so.MSI and ts.MSIS = so.MSIS
group by so.WBI, so.WBIS, so.WBName, ts.year, ts.determinand, ts.UoM
order by WBI, WBIS, WBName, year, determinand
```

SQL result



WBI	WBIS	WBName	year	determinand	avgV	UoM
IT00_I028_010_TR01_A	euSurfaceWaterBodyCode	TORRENTE FLUVIONE TRATTO 1 C.I._A	2015	BOD5	1.105	mg{O2}/L
IT00_I028_010_TR01_A	euSurfaceWaterBodyCode	TORRENTE FLUVIONE TRATTO 1 C.I._A	2015	Nitrate	2.617	mg{NO3}/L
IT00_I028_010_TR01_A	euSurfaceWaterBodyCode	TORRENTE FLUVIONE TRATTO 1 C.I._A	2015	Nitrite	0.01	mg{NO2}/L
IT00_I028_010_TR01_A	euSurfaceWaterBodyCode	TORRENTE FLUVIONE TRATTO 1 C.I._A	2019	BOD5	1.667	mg{O2}/L
IT00_I028_010_TR01_A	euSurfaceWaterBodyCode	TORRENTE FLUVIONE TRATTO 1 C.I._A	2019	Nitrate	1.987	mg{NO3}/L
IT00_I028_010_TR01_A	euSurfaceWaterBodyCode	TORRENTE FLUVIONE	2019	Nitrite	0.02	mg{NO2}/L

5. Advanced queries

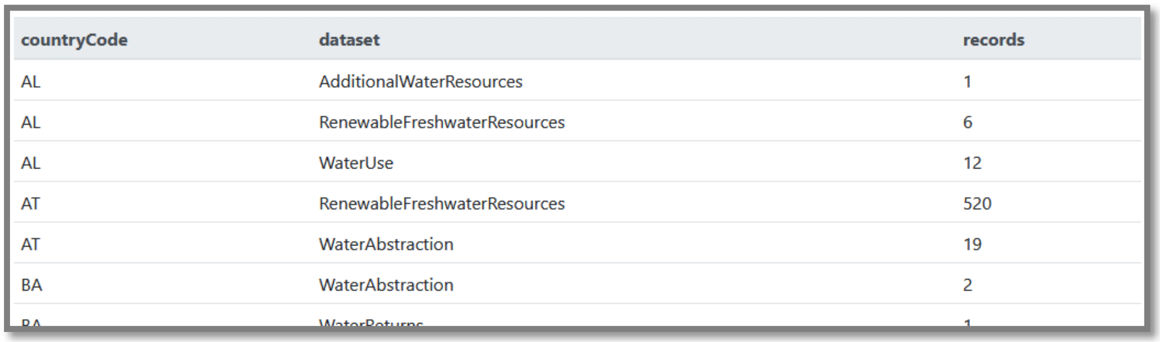
5.a Union

Union allows to merge results of multiple queries. Each query must return the same number of columns with the same datatypes. The column names do not need to match, although it is a good practice.

SQL source

```
-- WISE3 record count per country - Water Resources, Abstraction, Returns, Use
Select [countryCode],'AdditionalWaterResources' as dataset, count(*) as records
from [WISE_SOE].[v1r1].[Waterbase_T_WISE3_AdditionalWaterResources]
group by [countryCode]
UNION
Select [countryCode],'RenewableFreshwaterResources' as dataset, count(*) as records
from [WISE_SOE].[v1r1].[Waterbase_T_WISE3_RenewableFreshwaterResources]
group by [countryCode]
UNION
Select [countryCode],'WaterAbstraction' as dataset, count(*) as records
from [WISE_SOE].[v1r1].[Waterbase_T_WISE3_WaterAbstraction]
group by [countryCode]
UNION
Select [countryCode],'WaterReturns' as dataset, count(*) as records
from [WISE_SOE].[v1r1].[Waterbase_T_WISE3_WaterReturns]
group by [countryCode]
UNION
Select [countryCode],'WaterUse' as dataset, count(*) as records
from [WISE_SOE].[v1r1].[Waterbase_T_WISE3_WaterUse]
group by [countryCode]
```

SQL result



countryCode	dataset	records
AL	AdditionalWaterResources	1
AL	RenewableFreshwaterResources	6
AL	WaterUse	12
AT	RenewableFreshwaterResources	520
AT	WaterAbstraction	19
BA	WaterAbstraction	2
BA	WaterReturns	1

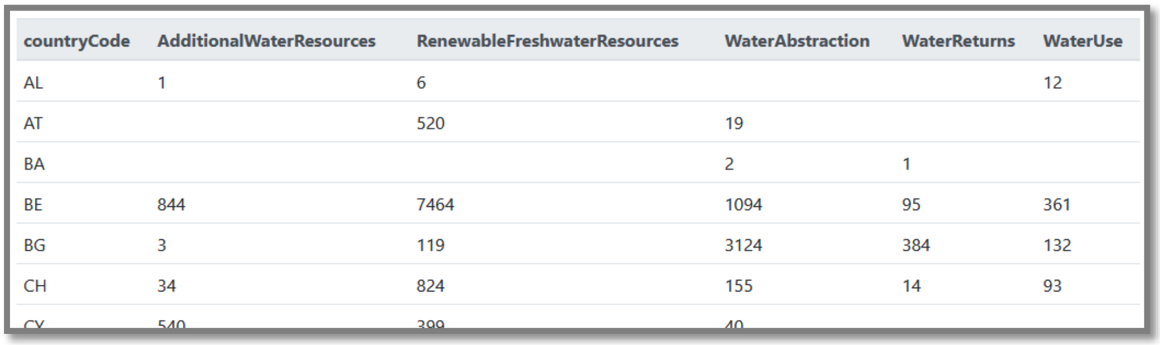
5.b Pivot

The Pivot is an advanced function for pivoting results of another query. Let's use the result of 5.a and make it more readable.

SQL source

```
-- WISE3 pivot record count per country - Water resources, abstraction, returns, use
Select countryCode
    ,[AdditionalWaterResources],[RenewableFreshwaterResources],[WaterAbstraction],[WaterReturns],[WaterUse]
from (
    Select [countryCode],'AdditionalWaterResources' as dataset, count(*) as records
    from [WISE_SOE].[v1r1].[Waterbase_T_WISE3_AdditionalWaterResources]
    group by [countryCode]
    UNION
    Select [countryCode],'RenewableFreshwaterResources' as dataset, count(*) as records
    from [WISE_SOE].[v1r1].[Waterbase_T_WISE3_RenewableFreshwaterResources]
    group by [countryCode]
    UNION
    Select [countryCode],'WaterAbstraction' as dataset, count(*) as records
    from [WISE_SOE].[v1r1].[Waterbase_T_WISE3_WaterAbstraction]
    group by [countryCode]
    UNION
    Select [countryCode],'WaterReturns' as dataset, count(*) as records
    from [WISE_SOE].[v1r1].[Waterbase_T_WISE3_WaterReturns]
    group by [countryCode]
    UNION
    Select [countryCode],'WaterUse' as dataset, count(*) as records
    from [WISE_SOE].[v1r1].[Waterbase_T_WISE3_WaterUse]
    group by [countryCode]
) as sourceTable
PIVOT (
    max(records)
    for dataset in
        ([AdditionalWaterResources],[RenewableFreshwaterResources],[WaterAbstraction],[WaterReturns],[WaterUse])
) as pivotTable
```

SQL result



countryCode	AdditionalWaterResources	RenewableFreshwaterResources	WaterAbstraction	WaterReturns	WaterUse
AL	1	6			12
AT		520	19		
BA			2	1	
BE	844	7464	1094	95	361
BG	3	119	3124	384	132
CH	34	824	155	14	93
CY	540	399	40		