Data Network Dashboards

This document is currently under construction

2020-11-07

Contents

Preface

Automated Characterization of Health Information at Large-scale Longitudinal Evidence Systems (ACHILLES) is a profiling tool developed by the OHDSI community to provide descriptive statistics of databases standardized to the OMOP Common Data Model. These characteristics are presented graphically in the ATLAS tool. However, this solution does not allow for database comparison across the data network. The Data Network Dashboards aggregates ACHILLES results files from databases in the network and displays the descriptive statistics through graphical dashboards. This tool is helpful to gain insight in the growth of the data network and is useful for the selection of databases for specific research questions. In the software demonstration we show a first version of this tool that will be further developed in EHDEN in close collaboration with all our stakeholders, including OHDSI.

Contributors

To develop this tool, EHDEN organized a hack-a-thon (Aveiro, December 2-3, 2019), where we defined and implemented a series of charts and dashboards containing the most relevant information about the OMOP CDM databases. The team involved in this task were composed by the following members:

- João Rafael Almeida[1]
- André Pedrosa[1]
- Peter R. Rijnbeek[2]
- Marcel de Wilde[2]
- Michel Van Speybroeck[3]
- Maxim Moinat[4]
- Pedro Freire[1]
- Alina Trifan[1]
- Sérgio Matos[1]
- José Luís Oliveira[1]

1 - Institute of Electronics and Informatics Engineering of Aveiro, Department of Electronics and Telecommunication, University of Aveiro, Aveiro, Portugal

2 - Erasmus MC, Rotterdam, Netherlands

3 - Janssen Pharmaceutica NV, Beerse, Belgium

4 - The Hyve, Utrecht, Netherlands

Considerations

This manual was written to be a guide for a clean installation of this system with all the dashboards that we defined during the project. The first chapter describes the goal of the system and the second how to install the system. The remaining chapters are dedicated to the dashboards, in which chapters describes one dashboard and all its charts. To simplify the representation of the dashboard's layout, we used similar schemas as it is presented in Figure 1. The white box is the dashboard and the inside boxes are charts. The colour changes in relation to the type of chart.

Figure 1: Example of a dashboards tool presenting the databases available in the network (simulated data)

License

The system is open-source and this manual was written in RMarkdown using the bookdown package.

Acknowledges

This work has been conducted in the context of EHDEN, a project that receives funding from the European Union's Horizon 2020 and EFPIA through IMI2 Joint Undertaking initiative, under grant agreement No 806968.

Chapter 1

Introduction

The OHDSI research network has been growing steadily which results in an increasing number of healthcare databases standardized to the OMOP CDM format. The OHDSI community created the ACHILLES tool (Automated Characterization of Health Information at Large-scale Longitudinal Exploration System) to characterize those databases. The results are available to the data custodian in their local ATLAS tool and helps them to gain insights in their data and helps in assessing the feasibility of a particular research questions.

ACHILLES was designed to extract the metadata from a single database, which by itself does not allow the comparison with the remaining databases in the network. However, we believe there is even more value in sharing this information with others to enable network research in a Data Network Dashboard.

1.1 Data Network Dashboard

The European Health Data and Evidence Network (EHDEN) project therefore designed a Data Network Dashboard tool, a web application to aggregate information from distributed OMOP CDM databases. It uses the ACHILLES results files to construct graphical dashboards and enables database comparison (Figure 1.1). The tool is built on Apache Superset, which is an open-source enterprise-ready business intelligence web application that can provide powerful and fully customizable graphical representations of data. Achilles results can be uploaded through the EHDEN Database Catalogue using the dashboards plugin but can also be directly uploaded in the tool. Figure 1. Example of a dashboards tool presenting age and gender distributions (simulated data).

In this tools, we defined and implemented a series of charts and dashboards containing the most relevant information about the databases, such as:

- **General**: dashboards that shows the databases types per country, the

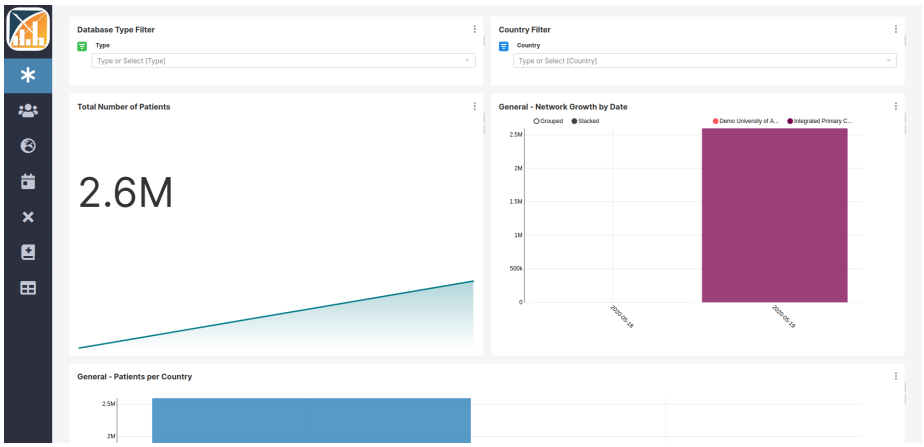

Figure 1.1: Example of a dashboards tool presenting the databases available in the network (simulated data)

distribution of data source types, the growth of the Network including the number of database and the number of patients in the databases over time;

- **Person**: representing the number of patients per country, age distribution at first observation, year of birth distribution and normalized gender distribution;
- **Population characteristics**: dashboard with the cumulative patient time, persons with continuous observation per month, and the start and end dates of those periods;
- **Visit**: chart to compare the number and type of visit occurrence records;
- **Death**: information about the number of death records by month, and the patient age at time of death;
- **Concepts**: bubble chart which shows the number of patients and records per concept over the databases;
- **Data domains**: heat map visualization of the major data domains in each database.

Chapter 2

Installation

Currently, we use docker to deploy our environment

2.1 First Steps

1. Clone the repository with the command `git clone --recurse-submodules https://github.com/EHDEN/NetworkDashboards`. If you already cloned the repository without the `--recurse-submodules` option, run `git submodule update --init` to fetch the superset submodule.

2. Create a `.env` file on the `docker` directory, using `.env-example` as a reference, setting all necessary environment variables (`SUPERSET\_MAPBOX\_API\_KEY` and `DASHBOARD\_VIEWER\_SECRET\_KEY`).

2.2 Dashboard Viewer setup

1. If you wish to expose the dashboard viewer app through a specific domain(s) you must add it/them to the `ALLOWED_HOSTS` list on file `dashboard_viewer/dashboard_viewer/settings.py` and remove the `'*'` entry.

2. Build containers' images: `docker-compose build`. This might take several minutes.

3. Set up the database and create an admin account for the dashboard viewer app: `docker-compose run --rm dashboard ./docker-init.sh`.

2.3 Insert Concepts

The concepts table is not in the repository due to its dimension, therefore we use directly the Postgres console to insert this table in the installation.

1. Get your concept csv file from Athena
2. Copy the file into postgres container

   ```
   docker cp concept.csv dashboard_viewer_postgres_1:/tmp/
   ```

3. Enter in the postgres container:

   ```
   docker exec -it dashboard_viewer_postgres_1 bash
   ```

4. Enter in the `achilles` database (value of the variable `POSTGRES_ACHILLES_DB` on the .env file) with the `root` user (value of the variable `POSTGRES_ROOT_USER` on the .env file):

   ```
   psql achilles root
   ```

5. Create the `concept` table

   ```
   CREATE TABLE concept (
     concept_id          INTEGER         NOT NULL,
     concept_name        VARCHAR(255)    NOT NULL,
     domain_id           VARCHAR(20)     NOT NULL,
     vocabulary_id       VARCHAR(20)     NOT NULL,
     concept_class_id    VARCHAR(20)     NOT NULL,
     standard_concept    VARCHAR(1)      NULL,
     concept_code        VARCHAR(50)     NOT NULL,
     valid_start_date    DATE            NOT NULL,
     valid_end_date      DATE            NOT NULL,
     invalid_reason      VARCHAR(1)      NULL
   );
   ```

6. Copy the CSV file content to the table (this could take a while)

   To get both ' (single quotes) and " (double quotes) on the `concept_name` column we use a workaround by setting the quote character to one that should never be in the text. Here we used `\b` (backslash).

   ```
   COPY public.concept FROM '/tmp/concept.csv' WITH CSV HEADER
     DELIMITER E'\t' QUOTE E'\b';
   ```

7. Create index in table (this could take a while):

   ```
   CREATE INDEX concept_concept_id_index ON concept (concept_id);
   CREATE INDEX concept_concept_name_index ON concept (concept_name);
   ```

8. Set the owner of the `concept` table to the `achilles` user (value of the variable `POSTGRES_ACHILLES_USER` on the .env file):

```
ALTER TABLE concept OWNER TO achiller
```

9. Bring up the containers: `docker-compose up -d`.
10. Run the command `docker-compose run --rm dashboard python manage.py generate_materialized_views` to create the materialized views on Postgres.

2.4 Superset setup

1. Make sure that the container `superset-init` has finished before continuing. It is creating the necessary tables on the database and creating permissions and roles.
2. Execute the script `./superset/one_time_run_scripts/superset-init.sh`. This will create an admin account and associate the `achilles` database to Superset. **Attention:** You must be in the docker directory to execute this script.
3. We have already built some dashboards so if you want to import them run the script `./superset/one_time_run_scripts/load_dashboards.sh`. **Attention:** You must be in the docker directory to execute this script.
4. If you used the default ports:
   - Go to `http://localhost` to access the dashboard viewer app.
   - Go to `http://localhost:8088` to access superset.
5. On release 0.37 of Superset, there is a bug related to the public role and because of that, we had to set `PUBLIC_ROLE_LIKE_GAMMA = True` on Superset settings. This leads the public role with permissions that he shouldn't have. To solve this, so any anonymous user can view dashboards, you should remove all its permissions and then add the following:
   - can explore JSON on Superset
   - can dashboard on Superset
   - all datasource access on all_datasource_access
   - can csrf token on Superset
   - can list on CssTemplateAsyncModelView

2.5 Dummy data

On a fresh installation, there are no achilles_results data so Superset's dashboards will display "No results". On the root of this repository, you can find the `demo` directory where we have an ACHILLES results file with synthetic data that you can upload to a data source on the uploader app of the dashboard viewer (localhost/uploader). If you wish to compare multiple data sources, on the `demo` directory the is also a python script that allows you to generate new

ACHILLES results files, where it generates random count values based on the ranges of values for each set of analysis_id and stratums present on a base ACHILLES results file. So, from the one ACHILLES results fill we provided, you can have multiple data sources with different data.

Chapter 3

General

3.1 CSS

To hide the dashboard header insert the following css code to the `CSS` field on the edit page:

```
.dashboard > div:not(.dashboard-content) {  /* dashboard header */
  display: none;
}
```

3.2 Database Type and Country Filter

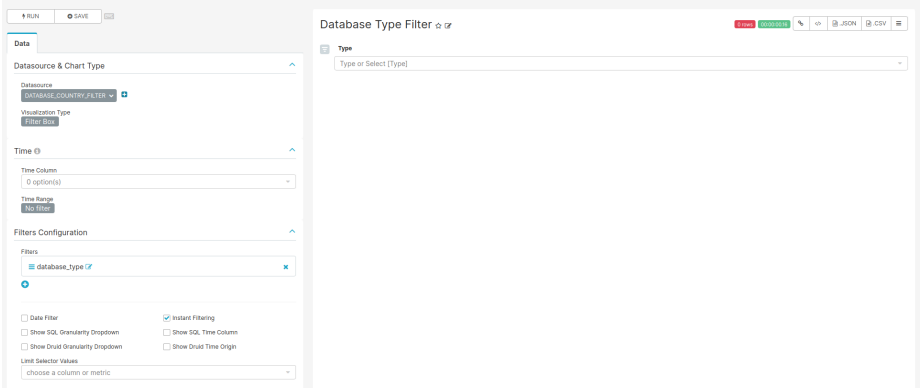

Figure 3.1: Settings for creating filters charts

Theses filter were designed to be used in the dashboard aiming the filtering of the data based on the field ‘’database_type” and “country” from the table

''data_source".

For the filters to work the name of the fields to filter should match in all tables used on the charts of this dashboard.

3.2.1 SQL query

```
SELECT source.name,
       country.country,
       source.database_type,
       source.acronym
FROM public.data_source AS source
INNER JOIN public.country AS country ON source.country_id=country.id
```

3.2.2 Chart settings

- Data Tab
  - Datasource & Chart Type
    - Visualization Type: Filter Box
  - Time
    - Time range: No filter
  - Filters Configuration
    - Filters:
      - database_type or country
    - Date Filter: off
    - Instant Filtering: on

3.3 Total Number of Patients

3.3.1 SQL query

```
SELECT
 country,
 database_type,
 release_date,
 SUM(count_value) OVER (ORDER BY release_date ASC)
FROM achilles_results
JOIN data_source ON data_source_id = data_source.id
JOIN country ON data_source.country_id = country.id
WHERE analysis_id = 1
```

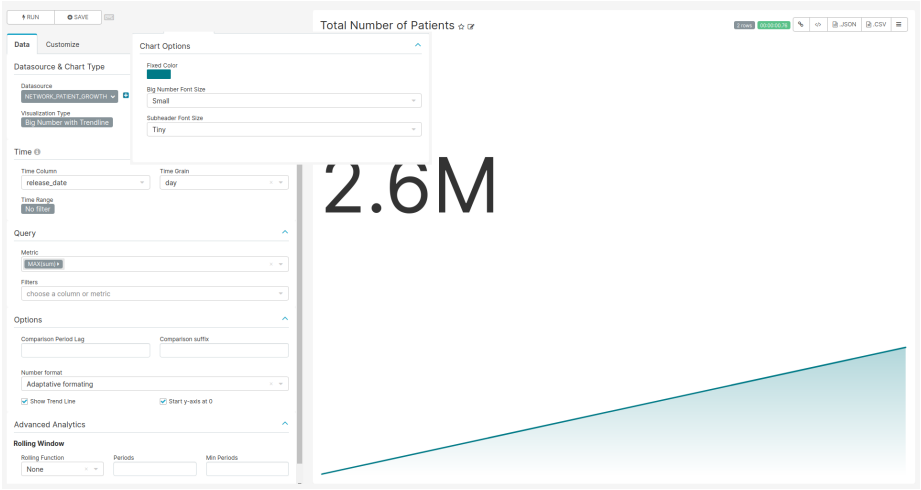

Figure 3.2: Settings for creating the Total Number of Patients chart

3.3.2 Chart settings

- Data Tab
  - Datasource & Chart Type
    - Visualization Type: Big Number with Trendline
  - Time
    - Time range: No filter
  - Query
    - Metrics: MAX(sum)
    - Series: release_date
    - Breakdowns: source
- Customize Tab
  - Chart Options
    - Big Number Font Size: Small
    - Subheader Font Size: Tiny

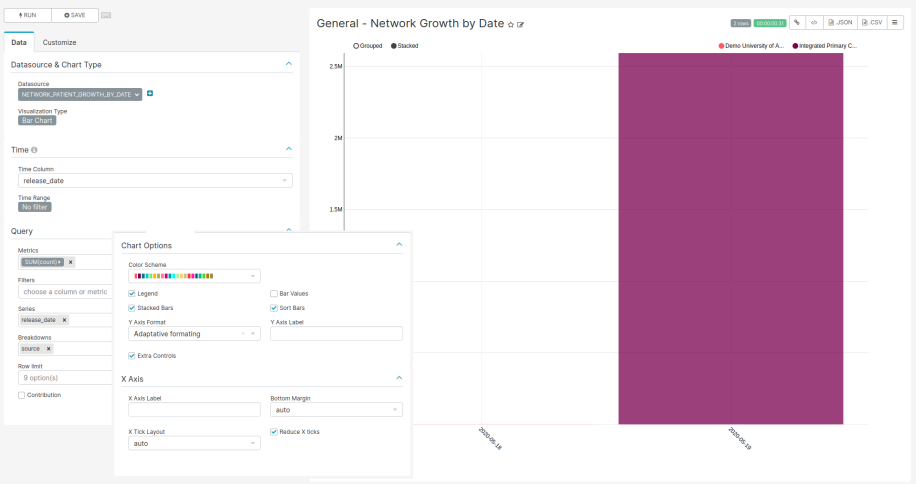

Figure 3.3: Settings for creating the Network Growth by Date chart

3.4 Network Growth by Date

3.4.1 SQL query

```
SELECT  source.name AS source,
        country.country,
        source.database_type,
        source.release_date,
        concepts.concept_name AS gender,
        achilles.count_value as count
FROM public.achilles_results AS achilles
INNER JOIN public.data_source AS source ON achilles.data_source_id=source.id
INNER JOIN public.country AS country ON source.country_id=country.id
JOIN (
  SELECT '8507' AS concept_id, 'Male' AS concept_name
  UNION
  SELECT '8532', 'Female'
) AS concepts ON achilles.stratum_1 = concept_id
WHERE analysis_id = 2
```

3.4.2 Chart settings

- Data Tab
  - Datasource & Chart Type
    - Visualization Type: Bar Chart

  - Time
    - Time range: No filter
  - Query
    - Metrics: SUM(count_value)
    - Series: release_date
    - Breakdowns: source

- Customize Tab
  - Chart Options
    - Stacked Bars: on
    - Sort Bars: on
    - Extra Controls: on
  - X Axis
    - Reduce X ticks: on

3.5 Patients per Country

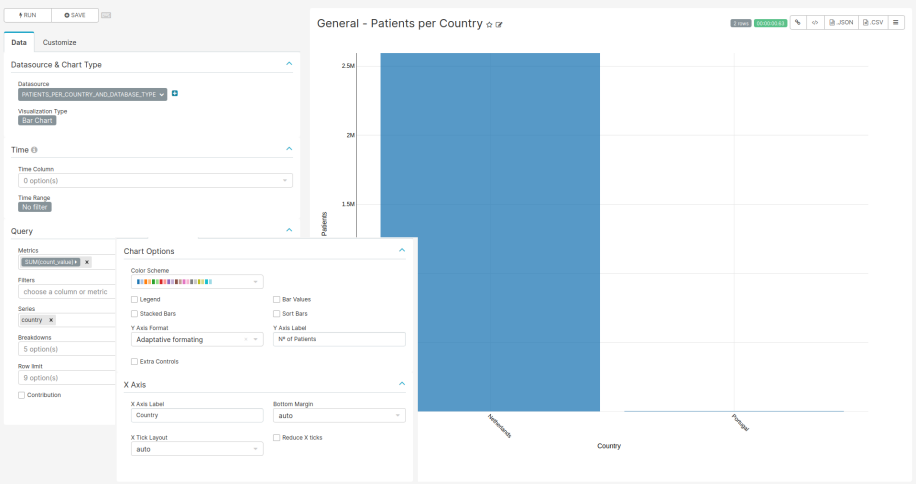

Figure 3.4: Settings for creating the Patients per Country chart

3.5.1 SQL query

```
SELECT country.country,
       source.database_type,
       count_value
FROM public.achilles_results AS achilles
INNER JOIN public.data_source AS source ON achilles.data_source_id=source.id
INNER JOIN public.country AS country ON source.country_id=country.id
WHERE analysis_id = 1
```

3.5.2 Chart settings

- Data Tab
  - Datasource & Chart Type
    - Visualization Type: Bar Chart
  - Time
    - Time range: No filter
  - Query
    - Metrics: SUM(count_value)
    - Series: country
- Customize Tab
  - Chart Options
    - Legend: off
    - Y Axis Label: Nº of Patients
  - X Axis
    - X Axis Label: Country

3.6 Database Types per Country

3.6.1 SQL query

Same as Patients per Country query

```
SELECT country.country,
       source.database_type,
       count_value
FROM public.achilles_results AS achilles
INNER JOIN public.data_source AS source ON achilles.data_source_id=source.id
INNER JOIN public.country AS country ON source.country_id=country.id
WHERE analysis_id = 1
```

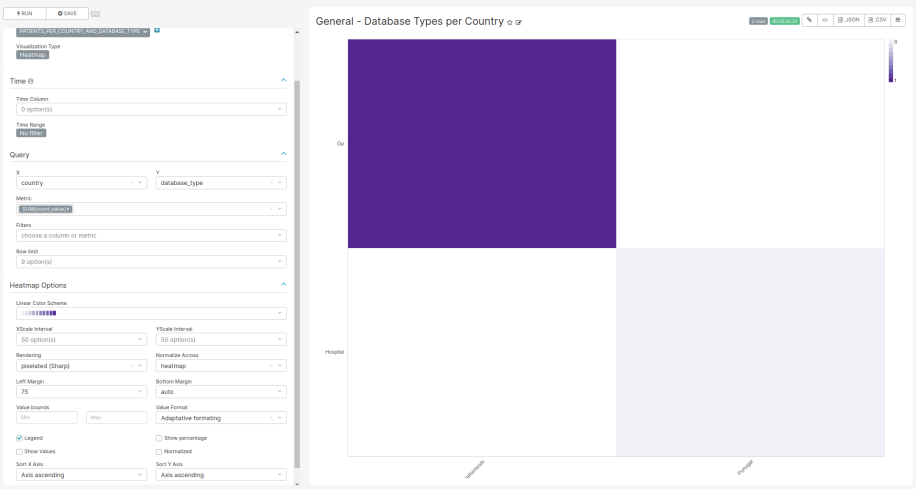

Figure 3.5: Settings for creating the Database Type per Country chart

3.6.2 Chart settings

- Data Tab
  - Datasource & Chart Type
    - Visualization Type: Heatmap
  - Time
    - Time range: No filter
  - Query
    - X: country
    - Y: database_type
    - Metric: SUM(countr_value)
  - Heatmap Options
    - Left Margin: 75
    - Show Percentage: off

3.7 World Map

3.7.1 SQL query

```
SELECT  name,
        acronym,
```

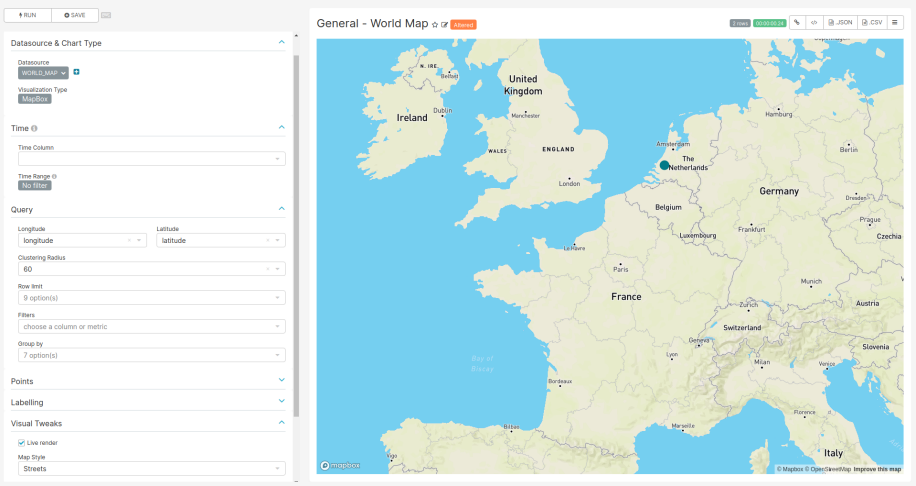

Figure 3.6: Settings for creating the World Map chart

```
        database_type,
        latitude,
        longitude,
        country
FROM public.data_source AS source
INNER JOIN public.country AS country ON source.country_id=country.id
```

3.7.2 Chart settings

- Data Tab
  - Datasource & Chart Type
    - Visualization Type: MapBox
  - Time
    - Time range: No filter
  - Query
    - Longitude: longitude
    - Latitude: latitude
  - Visual Tweaks
    - Map Style: Streets or Light or Outdoors

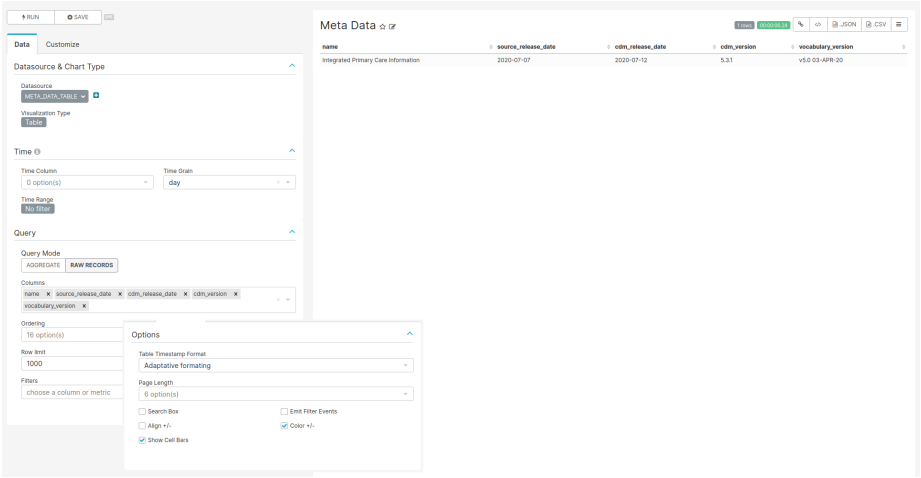

Figure 3.7: Settings for creating the Meta Data chart

3.8 Meta Data

3.8.1 SQL query

```sql
SELECT
  acronym,
  stratum_1 as "name",
  database_type,
  country,
  stratum_2 as "source_release_date",
  stratum_3 as "cdm_release_date",
  stratum_4 as "cdm_version",
  stratum_5 as "vocabulary_version"
FROM achilles_results
JOIN data_source ON achilles_results.data_source_id = data_source.id
JOIN country ON data_source.country_id = country.id
WHERE analysis_id=5000
```

3.8.2 Chart settings

- Data Tab
  - Datasource & Chart Type
    - Visualization Type: Table
  - Time
    - Time range: No filter

- Query
  - Query Mode: Raw Records
  - Columns: name, source_release_date, cdm_release_date, cdm_version, vocabulary_version

Chapter 4

Person

4.1 Label Colors

In order to obtain the colours blue and rose in the chart representing the gender distribution, add the following JSON entry to the JSON object of the `JSON Metadata` field on the edit dashboard page:

```
"label_colors": {
    "Male": "#3366FF",
    "Female": "#FF3399"
}
```

4.2 CSS

To hide the dashboard header insert the following css code to the `CSS` field on the edit page:

```
.dashboard > div:not(.dashboard-content) {  /* dashboard header */
  display: none;
}
```

4.3 Data Source Filter

For the filter to work the name of the fields to filter should match in all tables used on the charts of this dashboard.

4.3.1 SQL query

No SQL query, use the sql table `data_source` of the `achilles` database.

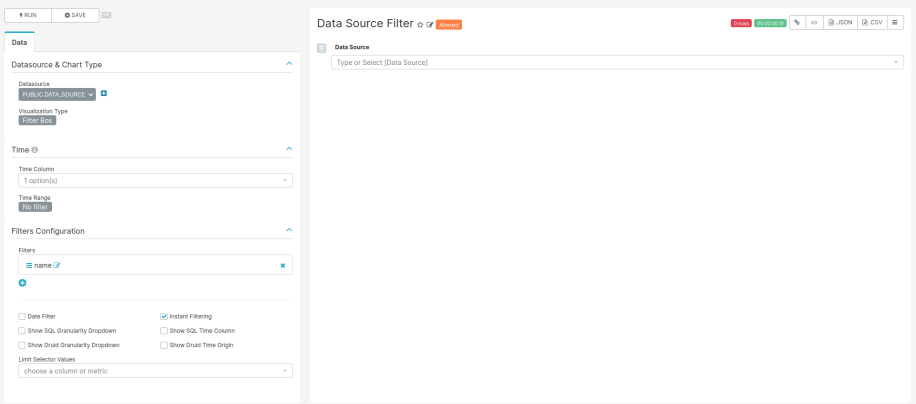

Figure 4.1: Settings for creating the Data Source filter chart

4.3.2 Chart settings

- Data Tab
  - Datasource & Chart Type
    - Visualization Type: Filter Box
  - Time
    - Time range: No filter
  - Filters Configuration
    - Filters:
      - name
    - Date Filter: off
    - Instant Filtering: on

4.4 Age at first observation - Table

4.4.1 SQL query

```
SELECT source.name,
       source.acronym,
       SUM(CASE WHEN CAST(stratum_2 AS INTEGER) < 10 THEN count_value END) AS "0-10",
       SUM(CASE WHEN CAST(stratum_2 AS INTEGER) >= 10 AND CAST(stratum_2 AS INTEGER) <
       SUM(CASE WHEN CAST(stratum_2 AS INTEGER) >= 20 AND CAST(stratum_2 AS INTEGER) <
       SUM(CASE WHEN CAST(stratum_2 AS INTEGER) >= 30 AND CAST(stratum_2 AS INTEGER) <
       SUM(CASE WHEN CAST(stratum_2 AS INTEGER) >= 40 AND CAST(stratum_2 AS INTEGER) <
```

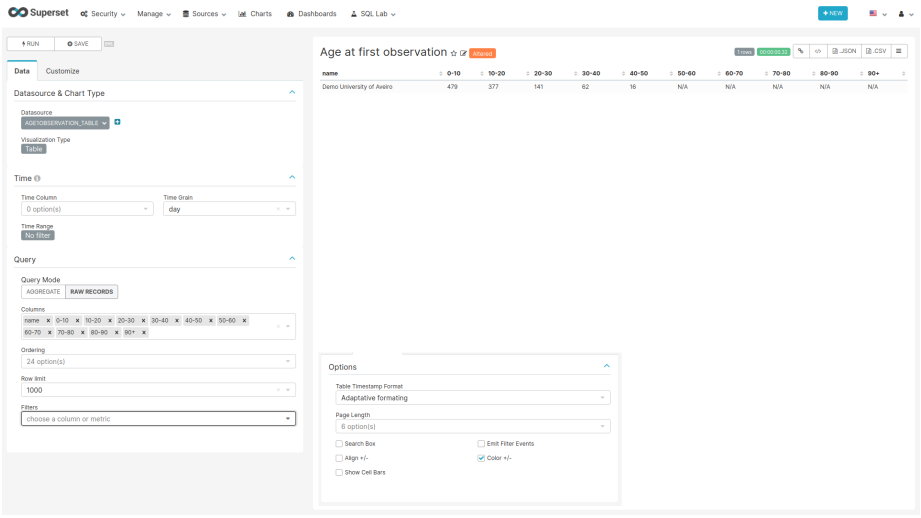

Figure 4.2: Settings for creating the Age at First Observation Table chart

```
        SUM(CASE WHEN CAST(stratum_2 AS INTEGER) >= 50 AND CAST(stratum_2 AS INTEGER) < 60 THEN co
        SUM(CASE WHEN CAST(stratum_2 AS INTEGER) >= 60 AND CAST(stratum_2 AS INTEGER) < 70 THEN co
        SUM(CASE WHEN CAST(stratum_2 AS INTEGER) >= 70 AND CAST(stratum_2 AS INTEGER) < 80 THEN co
        SUM(CASE WHEN CAST(stratum_2 AS INTEGER) >= 80 AND CAST(stratum_2 AS INTEGER) < 90 THEN co
        SUM(CASE WHEN CAST(stratum_2 AS INTEGER) >= 90 THEN count_value END) AS "90+"
FROM public.achilles_results AS achilles
INNER JOIN public.data_source AS source ON achilles.data_source_id=source.id
INNER JOIN public.concept ON stratum_1 = CAST(concept_id AS text)
WHERE analysis_id = 102
GROUP BY name, acronym
```

4.4.2 Chart settings

The main characteristics of this chart are presented in Figure **??**, being the following:

- Data Tab
  - Datasource & Chart Type
    - Visualization Type: Table
  - Time
    - Time range: No filter
  - Query
    - Query Mode: Raw Records

    * Columns: name, 0-10, 10-20, 20-30, 30-40, 40-50, 50-60, 60-70, 70-80, 80-90, 90+

- Customize Tab
  - Options
    * Show Cell Bars

4.5 Age at first observation - Bars

Figure 4.3: Settings for creating the Age at First Observation Bar chart

4.5.1 SQL query

```
SELECT source.name,
    cast(stratum_1 AS int) AS Age,
    count_value AS count,
    source.acronym
FROM public.achilles_results AS achilles
INNER JOIN public.data_source AS source ON achilles.data_source_id=source.id
WHERE analysis_id = 101
```

4.5.2 Chart settings

- Data Tab
  - Datasource & Chart Type

        * Visualization Type: Bar Chart
    - Time
        * Time range: No filter
    - Query
        * Metrics: MAX(count)
        * Series: age
        * Breakdowns: name
- Customize Tab
    - Chart Options
        * Stacked Bars: on
        * Sort Bars: on
        * Y Axis Label: Count
    - X Axis
        * X Axis Label: Age
        * Reduce X ticks: on

4.6 Year of Birth

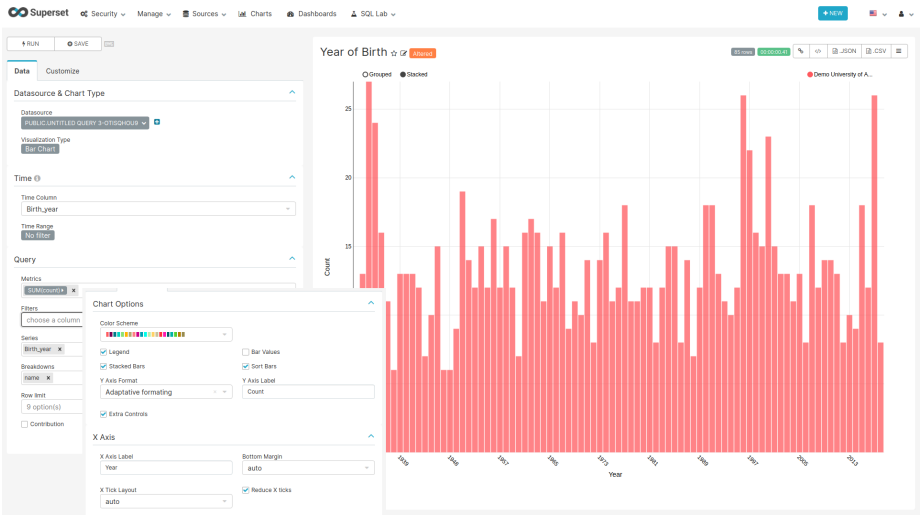

Figure 4.4: Settings for creating the Year of Birth chart

4.6.1 SQL query

```
SELECT source.name,
       source.acronym,
       stratum_1 AS "Birth_year",
       count_value AS count
FROM public.achilles_results AS achilles
INNER JOIN public.data_source AS source ON achilles.data_source_id=source.id
WHERE analysis_id = 3
```

4.6.2 Chart settings

- Data Tab
  - Datasource & Chart Type
    - Visualization Type: Bar Chart
  - Time
    - Time range: No filter
  - Query
    - Metrics: SUM(count)
    - Series: Birth_year
    - Breakdowns: name
- Customize Tab
  - Chart Options
    - Stacked Bars: on
    - Sort Bars: on
    - Y Axis Label: Count
    - Extra Controls: on
  - X Axis
    - X Axis Label: Year
    - Reduce X ticks: on

4.7 Gender

4.7.1 SQL query

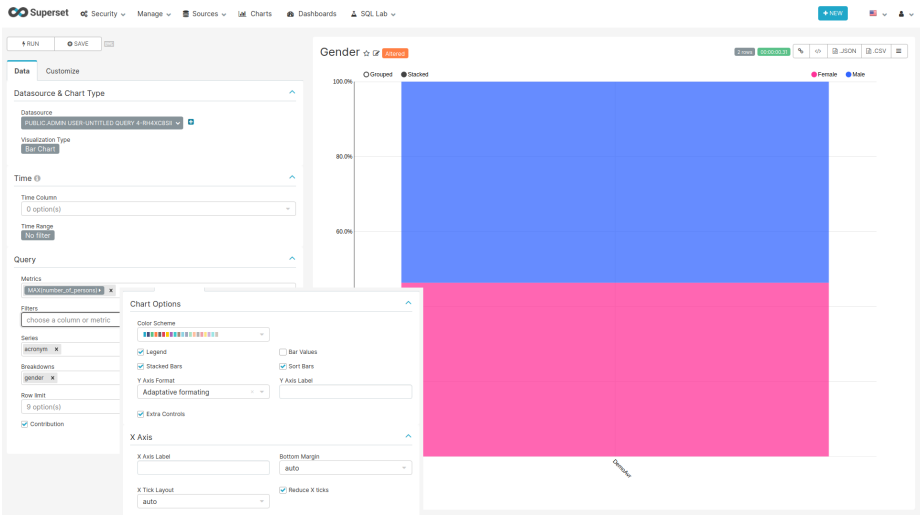

Figure 4.5: Settings for creating the Gender chart

```sql
SELECT source.name,
       concept_name AS Gender,
       count_value AS Number_of_persons,
       source.acronym
FROM public.achilles_results AS achilles
INNER JOIN public.data_source AS source ON achilles.data_source_id=source.id
JOIN (
  SELECT '8507' AS concept_id, 'Male' AS concept_name
  UNION
  SELECT '8532' AS concept_id, 'Female' AS concept_name
) AS concepts ON achilles.stratum_1 = concept_id
WHERE analysis_id = 2
```

4.7.2 Chart settings

- Data Tab
  - Datasource & Chart Type
    - Visualization Type: Bar Chart
  - Time
    - Time range: No filter
  - Query
    - Metrics: MAX(Number_of_persons)

        * Series: acronym
        * Breakdowns: gender
        * Contribution: on
- Customize Tab
    - Chart Options
        * Stacked Bars: on
        * Sort Bars: on
        * Extra Controls: on
    - X Axis
        * Reduce X ticks: on

Chapter 5

Population Characteristics

In this dashboard is present the ''Database Type Filter", that was detailed in the Chapter General.

5.1 Population characteristics - Patients in Observation Period per Month

Patients in Observation Period per month (whole month)

5.1.1 SQL query

```
-- 110    Population characteristics -
-- Patients in Observation Period per month (whole month)
SELECT source.name,
       to_date(stratum_1, 'YYYYMM') as Date,
       count_value as "Nr_patients",
       source.slug
FROM public.achilles_results AS achilles
    INNER JOIN public.data_source AS source ON
      achilles.data_source_id=source.id
WHERE analysis_id = 110;
```

5.1.2 Chart settings

The main characteristics of this chart are presented in Figure 5.1, being the following:

- **Data Tab**:
  - **Visualization Type**: Bar Chart

  - **Time range**: No filter
  - **Metrics**: MAX(Nr_patients) as “Num of Patients”
  - **Filters**: Empty
  - **Series**: date
  - **Breakdowns**: name
  - **Row limit**: Empty
  - **Contribution**: Not checked
- **Costumize Tab**:
  - **Y Axis Label**: Number of Patients
  - **X Axis Label**: Dates
  - **Legend**: Checked
  - **Stacked Bars**: Checked
  - **Bar Values**: Not checked
  - **Sort Bars**: Checked
  - **Extra Controls**: Not checked
  - **Reduce X ticks**: Checked

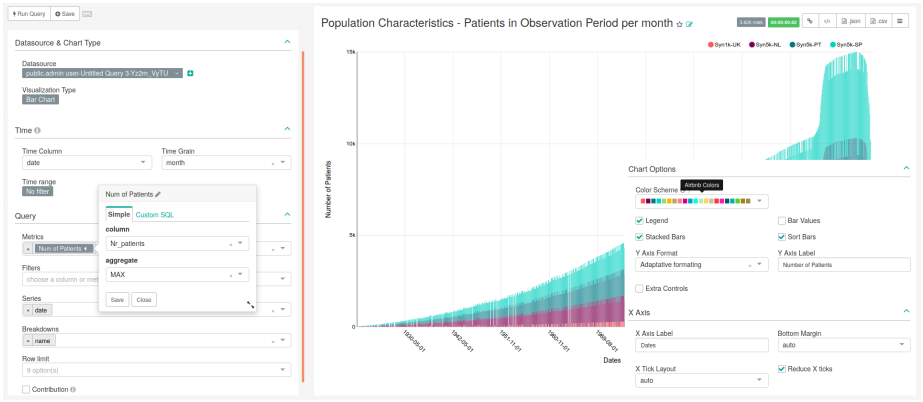

Figure 5.1: Settings for creating chart representing patient in observation per month (bar chart). Image changed to contain information hidden in the customize menu.

5.2 Population characteristics - Observation Period End Dates

5.2.1 SQL query

```
-- 112  Population characteristics -
--      Observation Period End Dates
SELECT source.name,
       to_date(stratum_1, 'YYYYMM') as year_month,
       count_value as patient_count
```

```
FROM public.achilles_results AS achilles
    INNER JOIN public.data_source AS source ON
      achilles.data_source_id=source.id
WHERE analysis_id = 112;
```

5.2.2 Chart settings

The main characteristics of this chart are presented in Figure 5.2, being the following:

- **Data Tab**:
  - **Visualization Type**: Bar Chart
  - **Time range**: No filter
  - **Metrics**: SUM(patient_count) as "Patients"
  - **Filters**: Empty
  - **Series**: year_month
  - **Breakdowns**: name
  - **Row limit**: Empty
  - **Contribution**: Not checked
- **Costumize Tab**:
  - **Y Axis Label**: Number of Patients
  - **X Axis Label**: Year
  - **Legend**: Checked
  - **Stacked Bars**: Checked
  - **Bar Values**: Not checked
  - **Sort Bars**: Checked
  - **Extra Controls**: Not checked
  - **Reduce X ticks**: Checked

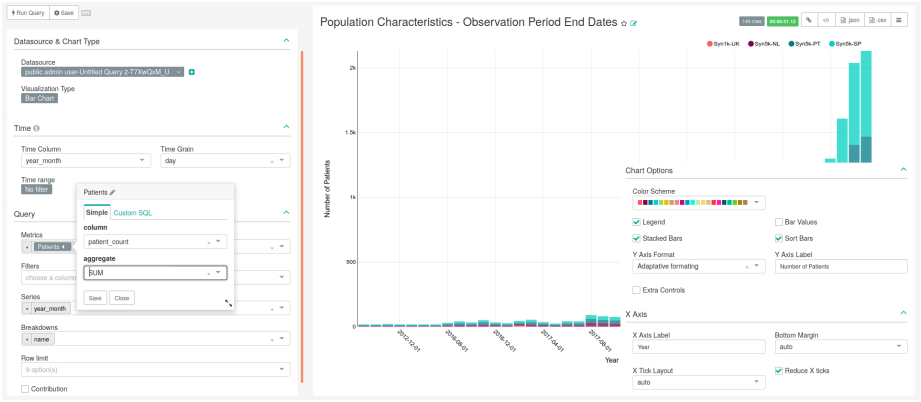

Figure 5.2: Settings for creating chart representing the number of patients at the end of their observation period (bar chart). Image changed to contain information hidden in the customize menu.

5.3 Population characteristics - Observation Period Start Dates

5.3.1 SQL query

```
-- 111  Population characteristics -
--      Observation Period Start Dates
SELECT source.name,
       to_date(stratum_1, 'YYYYMM') as year_month,
       count_value as patient_count
FROM public.achilles_results AS achilles
    INNER JOIN public.data_source AS source ON
      achilles.data_source_id=source.id
WHERE analysis_id = 111;
```

5.3.2 Chart settings

The main characteristics of this chart are presented in Figure 5.3, being the following:

- **Data Tab**:
  - **Visualization Type**: Bar Chart
  - **Time range**: No filter
  - **Metrics**: SUM(patient_count) as “Patients”
  - **Filters**: Empty
  - **Series**: year_month
  - **Breakdowns**: name
  - **Row limit**: Empty
  - **Contribution**: Not checked
- **Costumize Tab**:
  - **Y Axis Label**: Number of Patients
  - **X Axis Label**: Year
  - **Legend**: Checked
  - **Stacked Bars**: Checked
  - **Bar Values**: Not checked
  - **Sort Bars**: Checked
  - **Extra Controls**: Not checked
  - **Reduce X ticks**: Checked

Figure 5.3: Settings for creating chart representing the number of patients at the start of their observation period (bar chart). Image changed to contain information hidden in the customize menu.

Chapter 6

Visit

In this dashboard is present the ''Database Type Filter", that was detailed in the Chapter General.

6.1 Visit - Types

6.1.1 SQL query

```
-- 201  Visit - Types
SELECT source.name,
       concept_name AS "Observation",
       count_value AS "Nr_Observations",
       source.slug
FROM public.achilles_results AS achilles
    INNER JOIN public.data_source AS source ON
      achilles.data_source_id=source.id
    INNER JOIN public.concept ON
      stratum_1 = CAST(concept_id AS text)
WHERE analysis_id = 201;
```

6.1.2 Chart settings

The main characteristics of this chart are presented in Figure 6.1, being the following:

- **Data Tab**:
  - **Visualization Type**: Bar Chart
  - **Time range**: No filter
  - **Metrics**: MAX(Nr_Observations) as Observations
  - **Filters**: Empty

  - **Series**: name
  - **Breakdowns**: Observation
  - **Row limit**: Empty
  - **Contribution**: Not checked
- **Costumize Tab**:
  - **Y Axis Label**: Empty
  - **X Axis Label**: Databases
  - **Legend**: Checked
  - **Stacked Bars**: Checked
  - **Bar Values**: Not checked
  - **Sort Bars**: Checked
  - **Extra Controls**: Checked
  - **Reduce X ticks**: Checked

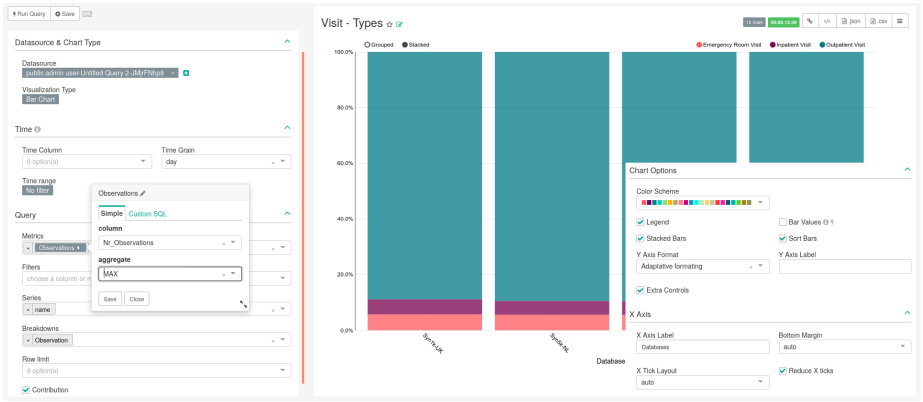

Figure 6.1: Settings for creating chart representing the types of visit in the databases (bar chart). Image changed to contain information hidden in the customize menu.

Chapter 7

Death

In this dashboard is present the ‘’Database Type Filter”, that was detailed in the Chapter General.

7.1 Death - By Month per Thousand People

7.1.1 SQL query

```
-- 502  Death - By Month per Thousand People
SELECT source.name,
       to_date(stratum_1, 'YYYYMM') as Date,
       count_value as count,
       source.slug
FROM public.achilles_results AS achilles
    INNER JOIN public.data_source AS source ON
      achilles.data_source_id=source.id
WHERE analysis_id = 502;
```

7.1.2 Chart settings

The main characteristics of this chart are presented in Figure 7.1, being the following:

- **Data Tab**:
  - **Visualization Type**: Bar Chart
  - **Time range**: No filter
  - **Metrics**: MAX(count) as “Count”
  - **Filters**: Empty
  - **Series**: date
  - **Breakdowns**: Empty

  - **Row limit**: Empty
  - **Contribution**: Not checked
- **Costumize Tab**:
  - **Y Axis Label**: Number of Patients (in thousands)
  - **X Axis Label**: Databases
  - **Legend**: Checked
  - **Stacked Bars**: Not checked
  - **Bar Values**: Not checked
  - **Sort Bars**: Checked
  - **Extra Controls**: Not checked
  - **Reduce X ticks**: Not checked

Figure 7.1: Settings for creating chart show in thousands the number of death patients in the network (bar chart). Image changed to contain information hidden in the customize menu.

7.2 Death - Number of Records

7.2.1 SQL query

```sql
-- 501  Death - Number of Records
SELECT source.name,
       count_value as count,
       source.slug
FROM public.achilles_results AS achilles
    INNER JOIN public.data_source AS source ON
      achilles.data_source_id=source.id
WHERE analysis_id = 501;
```

7.2.2 Chart settings

The main characteristics of this chart are presented in Figure 7.2, being the following:

- **Data Tab**:
  - **Visualization Type**: Bar Chart
  - **Time range**: No filter
  - **Metrics**: MAX(count) as "Count"
  - **Filters**: Empty
  - **Series**: name
  - **Breakdowns**: Empty
  - **Row limit**: Empty
  - **Contribution**: Not checked
- **Costumize Tab**:
  - **Y Axis Label**: Number of Patients
  - **X Axis Label**: Databases
  - **Legend**: Checked
  - **Stacked Bars**: Not checked
  - **Bar Values**: Not checked
  - **Sort Bars**: Not checked
  - **Extra Controls**: Not checked
  - **Reduce X ticks**: Checked

Figure 7.2: Settings for creating chart show the number of death patients in each database (bar chart). Image changed to contain information hidden in the customize menu.

Chapter 8

Concepts

This chapter follows a different organization due to the reuse of the same query in several charts.

8.1 Concepts General Tab

This tab uses an unique query and all the filters and charts were created using the same output. Therefore, this dashboard is composed by two filters, one big number, two bar charts and one table. The filters were splitted in order to apply the filtering of one over the other one. Figure 8.1 shows this dashboard's layout.

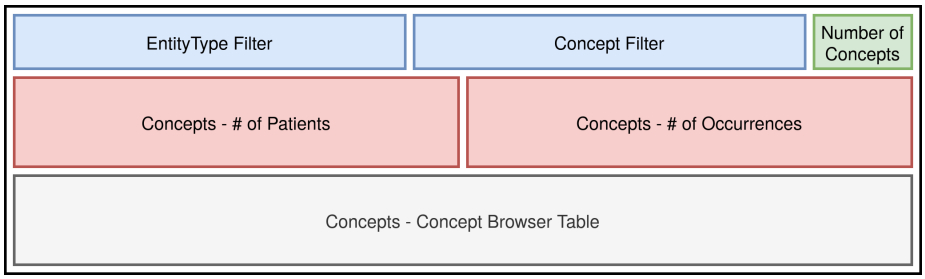

Figure 8.1: Distribution of charts on the dashboard.

8.1.1 SQL query

```
SELECT
    q1.concept_id AS concept_id,
    q1.concept_name AS concept_name,
    q1.domain_id,
    source.name,
```

```
    sum(q1.count_value) AS "Occurrence_count",
    sum(q1.count_person) AS "Person_count",
    CASE
        WHEN sum(q1.count_value)<=10 THEN '<=10'
        WHEN sum(q1.count_value)<=100 THEN '11-10^2'
        WHEN sum(q1.count_value)<=1000 THEN '10^2-10^3'
        WHEN sum(q1.count_value)<=10000 THEN '10^3-10^4'
        WHEN sum(q1.count_value)<=100000 THEN '10^4-10^5'
        WHEN sum(q1.count_value)<=1000000 THEN '10^5-10^6'
        ELSE '>10^6'
    END AS "magnitude_occurrences",
    CASE
        WHEN sum(q1.count_person)<=10 THEN '<=10'
        WHEN sum(q1.count_person)<=100 THEN '11-10^2'
        WHEN sum(q1.count_person)<=1000 THEN '10^2-10^3'
        WHEN sum(q1.count_person)<=10000 THEN '10^3-10^4'
        WHEN sum(q1.count_person)<=100000 THEN '10^4-10^5'
        WHEN sum(q1.count_person)<=1000000 THEN '10^5-10^6'
        ELSE '>10^6'
    END AS "magnitude_persons"
FROM (SELECT analysis_id,
             stratum_1 concept_id,
             data_source_id,
             concept_name,
             domain_id,
             count_value, 0 AS count_person
    FROM achilles_results
    JOIN concept ON cast(stratum_1 AS BIGINT)=concept_id
    WHERE analysis_id in (201, 301, 401, 601, 701, 801, 901, 1001,
        1801)
    UNION (SELECT  analysis_id,
                   stratum_1 concept_id,
                   data_source_id,
                   concept_name,
                   domain_id,
                   0 AS count_value,
                   sum(count_value) AS count_person
            FROM  achilles_results
            JOIN concept on cast(stratum_1 AS BIGINT)=concept_id
            WHERE analysis_id in (202, 401, 601, 701, 801, 901,
                1001, 1801)
            GROUP BY analysis_id, stratum_1, data_source_id,
                concept_name, domain_id) ) AS q1
    INNER JOIN public.data_source AS source ON
        q1.data_source_id=source.id
```

```
GROUP BY q1.concept_id, q1.concept_name, q1.domain_id,source.name;
```

8.1.2 Charts

8.1.2.1 Entity Type Filter

This filter, which is a type of chart in Superset, was designed to be used in the dashboard aiming the filtering of the data based on the field ''domain_id" from the table ''concept". The main characteristics of this filter are presented in Figure 8.2, being the following:

- **Data Tab**:
  - **Visualization Type**: Filter Box
  - **Time range**: No filter
  - **Metrics**:
  - **Filters - Column**: domain_id
  - **Filters - Label**: Entity Type
  - **Data Filter**: Not checked
  - **Row limit**: Empty
  - **Contribution**: Not checked
  - **Instant Filtering**: Checked
  - **Show SQL Granularity Dropdown**: Not checked
  - **Show Druid Granularity Dropdown**: Not checked
  - **Show SQL Time Column**: Not checked
  - **Show Druid Time Origin**: Not checked
  - **Limit Selector Values**: Empty

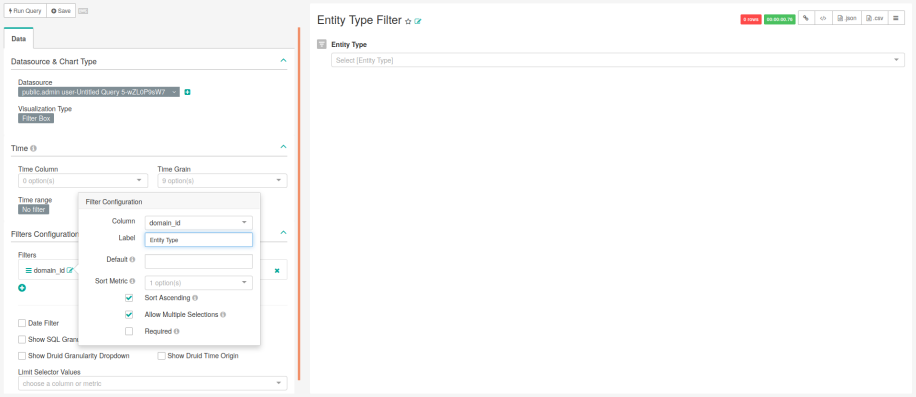

Figure 8.2: Settings for creating the database entity type filter.

8.1.2.2 Concept Filter

Similar to the previous filter, this was designed to be used in the dashboard aiming the filtering of the data based on the field ''concept_name" from the

table ''concept". The main characteristics of this filter are presented in Figure 8.3, being the following:

- **Data Tab**:
  - **Visualization Type**: Filter Box
  - **Time range**: No filter
  - **Metrics**:
  - **Filters - Column**: concept_name
  - **Filters - Label**: Concept
  - **Data Filter**: Not checked
  - **Row limit**: Empty
  - **Contribution**: Not checked
  - **Instant Filtering**: Checked
  - **Show SQL Granularity Dropdown**: Not checked
  - **Show Druid Granularity Dropdown**: Not checked
  - **Show SQL Time Column**: Not checked
  - **Show Druid Time Origin**: Not checked
  - **Limit Selector Values**: Empty

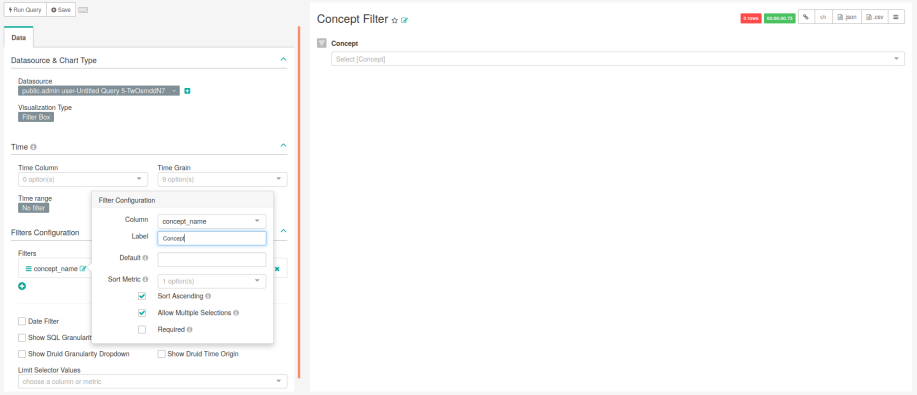

Figure 8.3: Settings for creating the concept filter.

8.1.2.3 Number of Concepts

Discuss what is important to see in this chart... TO DO The main characteristics of this chart are presented in Figure 6.1, being the following:

- **Data Tab**:
  - **Visualization Type**: Bar Chart
  - **Time range**: No filter
  - **Metrics**:
  - **Filters**: Empty
  - **Series**:
  - **Breakdowns**:

  - **Row limit**: Empty
  - **Contribution**: Not checked
- **Costumize Tab**:
  - **Y Axis Label**:
  - **X Axis Label**:
  - **Legend**: Checked
  - **Stacked Bars**:
  - **Bar Values**:
  - **Sort Bars**:
  - **Extra Controls**:
  - **Reduce X ticks**:

Figure 8.4: Settings for creating chart representing the number of patients at the start of their observation period (bar chart). Image changed to contain information hidden in the customize menu.

8.1.2.4 Concept Browser Table

The main characteristics of this chart are presented in Figure 6.1, being the following:

- **Data Tab**:
  - **Visualization Type**: Bar Chart
  - **Time range**: No filter
  - **Metrics**:
  - **Filters**: Empty
  - **Series**:
  - **Breakdowns**:
  - **Row limit**: Empty
  - **Contribution**: Not checked
- **Costumize Tab**:
  - **Y Axis Label**:
  - **X Axis Label**:

  - **Legend**: Checked
  - **Stacked Bars**:
  - **Bar Values**:
  - **Sort Bars**:
  - **Extra Controls**:
  - **Reduce X ticks**:

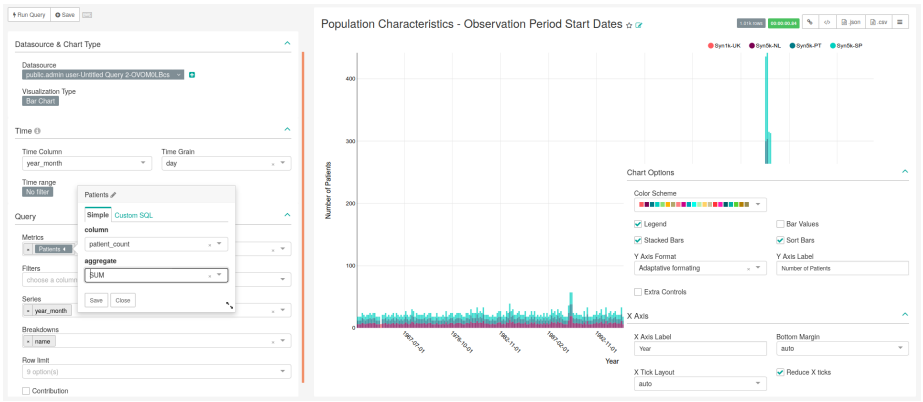

Figure 8.5: Settings for creating chart representing the number of patients at the start of their observation period (bar chart). Image changed to contain information hidden in the customize menu.

8.1.2.5 # of Occurrences

The main characteristics of this chart are presented in Figure 6.1, being the following:

- **Data Tab**:
  - **Visualization Type**: Bar Chart
  - **Time range**: No filter
  - **Metrics**:
  - **Filters**: Empty
  - **Series**:
  - **Breakdowns**:
  - **Row limit**: Empty
  - **Contribution**: Not checked
- **Costumize Tab**:
  - **Y Axis Label**:
  - **X Axis Label**:
  - **Legend**: Checked
  - **Stacked Bars**:
  - **Bar Values**:
  - **Sort Bars**:
  - **Extra Controls**:

  - **Reduce X ticks**:

Figure 8.6: Settings for creating chart representing the number of patients at the start of their observation period (bar chart). Image changed to contain information hidden in the customize menu.

8.1.2.6 # of Patients

The main characteristics of this chart are presented in Figure 6.1, being the following:

- **Data Tab**:
  - **Visualization Type**: Bar Chart
  - **Time range**: No filter
  - **Metrics**:
  - **Filters**: Empty
  - **Series**:
  - **Breakdowns**:
  - **Row limit**: Empty
  - **Contribution**: Not checked
- **Costumize Tab**:
  - **Y Axis Label**:
  - **X Axis Label**:
  - **Legend**: Checked
  - **Stacked Bars**:
  - **Bar Values**:
  - **Sort Bars**:
  - **Extra Controls**:
  - **Reduce X ticks**:

Figure 8.7: Settings for creating chart representing the number of patients at the start of their observation period (bar chart). Image changed to contain information hidden in the customize menu.

8.2 Concepts Domains Tab

This tab is composed of six bar charts that show the percentage of existent concepts in each database. These charts are similar but divided by the concept domains existent in the standard vocabularies, which are the conditions, procedures, drugs, observations, measurements and devices. Figure 8.8 shows this dashboard's layout.

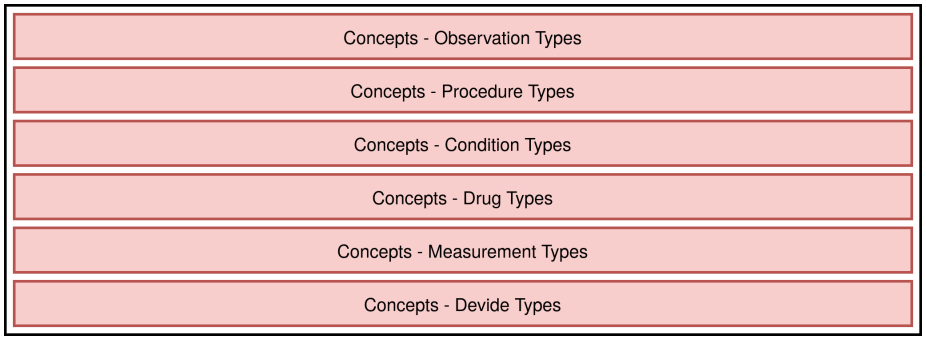

Figure 8.8: Distribution of charts on the dashboard.

8.2.1 SQL query

```
-- Concepts Domains
SELECT source.name,
    CASE WHEN analysis_id = 405 THEN 'Condition'
    WHEN analysis_id = 605 THEN 'Procedure'
```

```
    WHEN analysis_id = 705 THEN 'Drug'
    WHEN analysis_id = 805 THEN 'Observation'
    WHEN analysis_id = 1805 THEN 'Measurement'
    WHEN analysis_id = 2105 THEN 'Device'
    ELSE 'Other' END AS domain_name,
    concept_name, sum(count_value) AS num_records
FROM public.achilles_results AS achilles
    INNER JOIN public.data_source AS source ON
      achilles.data_source_id=source.id
    INNER JOIN public.concept AS c1 ON
      stratum_2 = CAST(concept_id AS text)
WHERE analysis_id IN (405, 605, 705, 805, 1805, 2105)
GROUP BY source.name, concept_name,
    CASE WHEN analysis_id = 405 THEN 'Condition'
    WHEN analysis_id = 605 THEN 'Procedure'
    WHEN analysis_id = 705 THEN 'Drug'
    WHEN analysis_id = 805 THEN 'Observation'
    WHEN analysis_id = 1805 THEN 'Measurement'
    WHEN analysis_id = 2105 THEN 'Device'
    ELSE 'Other' END
```

8.2.2 Chart settings

The difference between the six charts related to concept domains is the condition in the filter. Therefore, to create all the charts of this dashboard, it is necessary to follow the main characteristics presented in Figure 8.9 and the list of possible values for the filter. Those characteristics are the following:

- **Data Tab**:
  - **Visualization Type**: Bar Chart
  - **Time range**: No filter
  - **Metrics**: SUM(num_records) as "Nr Records"
  - **Filters**: domain = **See filter list**
  - **Series**: name
  - **Breakdowns**: concept_name
  - **Row limit**: Empty
  - **Contribution**: Checked
- **Costumize Tab**:
  - **Y Axis Label**:
  - **X Axis Label**:
  - **Legend**: Checked
  - **Stacked Bars**: Checked
  - **Bar Values**: Not checked
  - **Sort Bars**: Not checked
  - **Extra Controls**: Not checked
  - **Reduce X ticks**: Not checked

- **Filter List**:
  - **Concepts - Condition Types**: Condition
  - **Concepts - Procedure Types**: Procedure
  - **Concepts - Drug Types**: Drug
  - **Concepts - Observation Types**: Observation
  - **Concepts - Measurement Types**: Measurement
  - **Concepts - Device Types**: Device

Figure 8.9: Settings for creating all the charts related with the concept domains for each databases (bar chart). Image changed to contain information hidden in the customize menu.

Chapter 9

Data Domains

In this dashboard is present the ''Database Type Filter", that was detailed in the Chapter General.

9.1 Data Domains - Number of Records per Peson

9.1.1 SQL query

```
-- 201, 401, 501, 601, 701, 801, 1801, 2101, 2201
-- Data domains - Number of records per peson
SELECT
    source.name,
    CASE
      WHEN analysis_id = 201 THEN 'Visit'
      WHEN analysis_id = 401 THEN 'Condition'
      WHEN analysis_id = 501 THEN 'Death'
      WHEN analysis_id = 601 THEN 'Procedure'
      WHEN analysis_id = 701 THEN 'Drug Exposure'
      WHEN analysis_id = 801 THEN 'Observation'
      WHEN analysis_id = 1801 THEN 'Measurement'
      WHEN analysis_id = 2101 THEN 'Device'
      WHEN analysis_id = 2201 THEN 'Note'
    END AS Data_Domain,
    SUM(count_value) /AVG(num_persons) AS "Records per person",
    source.slug
FROM public.achilles_results AS achilles
    INNER JOIN public.data_source AS source ON
      achilles.data_source_id=source.id
```

```
    INNER JOIN (
        SELECT data_source_id , count_value AS num_persons
        FROM achilles_results
        WHERE analysis_id = 1
        ) counts ON
      achilles.data_source_id = counts.data_source_id
GROUP BY analysis_id, source.name, source.slug
HAVING analysis_id IN (201, 401, 501, 601, 701, 801, 1801, 2101,
    2201)
```

9.1.2 Chart settings

The main characteristics of this chart are presented in Figure 9.1, being the following:

- **Data Tab**:
  - **Visualization Type**: Heatmap
  - **Time range**: No filter
  - **X**: name
  - **Y**: data_domain
  - **Metric**: SUM(records_per_person) as "Sum of records per person"
  - **Filters**: Empty
  - **Row limit**: Empty
  - **Legend**: Checked
  - **Show percentage**: Checked
  - **Show Values**: Not checked
  - **Normalized**: Not checked

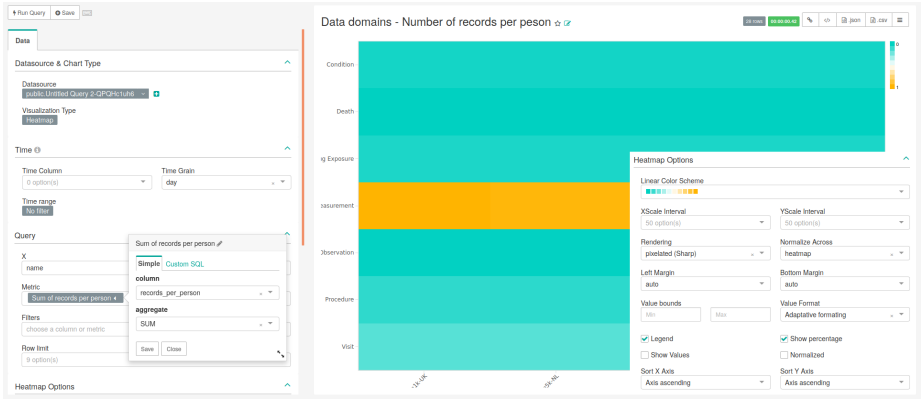

Figure 9.1: Settings for creating chart representing the number of records per patient in the different data domains (heatmap). Image changed to contain information hidden in the customize menu.