

Lab 4 - Data Analytics

Introduction

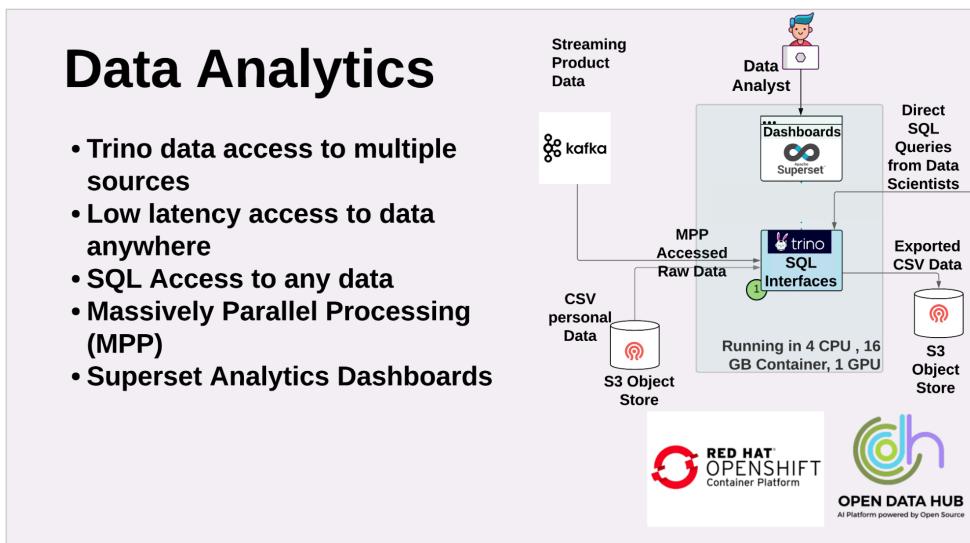
One of the first things your data analysts and data engineers will need to do is analyse the raw data, with a view to preparing and transforming it to a state that will be consumable by AI/ML model algorithms.

In this lab, we're going to use a powerful toolset combining

- an in-memory data analytics engine called Trino. Trino provides high speed access to many different on-premises and cloud based data sources. These include relational and no-SQL databases, object stores over S3 interfaces, Streaming data from Kafka and many more. Trino abstracts the actual underlying data store implementation and provides a uniform ANSI SQL interface, to access its many supported data stores.
- a visualization tool called Superset, which will use Trino as the backing data source.

The combination of these two tools will provide powerful data analytics capabilities, critical at this stage in the workflow.

This diagram illustrates what we're implementing:



You can see Trino is an SQL exposing abstraction in front of actual data located in Kafka and S3 Object storage. No data is moved – rather Trino provides a high speed parallel access mechanism for Kafka and S3 allowing Superset to easily display charts and dashboards.

To save time, the workshop administrators have already wired up the SQL exposing engine Trino to two backing datasets:

1. To a CSV file over an S3 interface. This CSV file is located in an underlying Object storage implementation called Minio. The file contains demographic type data on our customer data set, data such as gender, whether they have dependents and other demographic features.
2. To Streaming data located in an Apache Kafka store on the OpenShift cluster. This dataset contains product consumption for the same customers as are in the CSV file. Each record is labeled indicating whether that customer churned or not.

In Superset, using Trino, we've created two logical SQL tables corresponding underlying data sets as well as a query joining them on customer id.

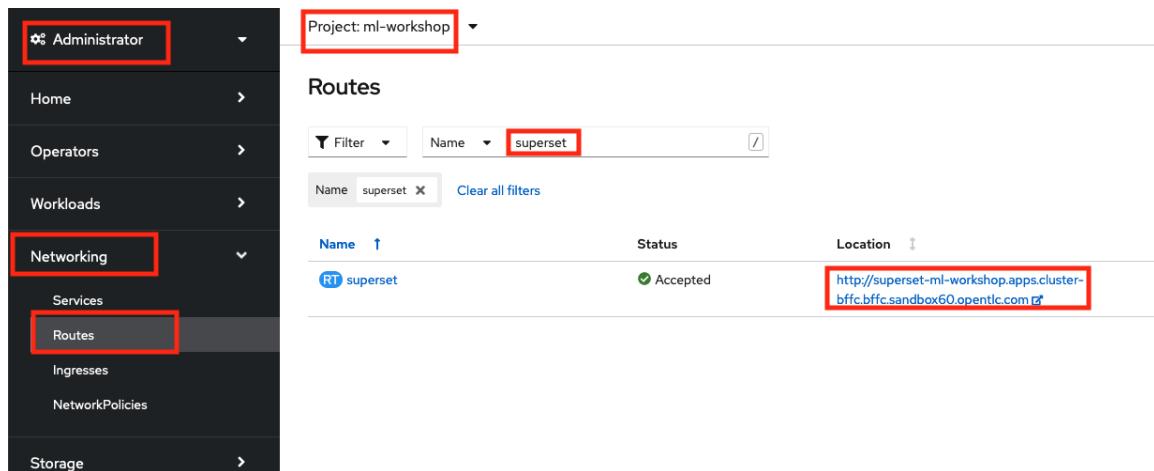
Instructions for the Trino backed Superset workshop

Login to OpenShift using the credentials your administrator gave you. Ensure your workshop project ml-workshop is selected.

The first thing you will do is login to Superset.

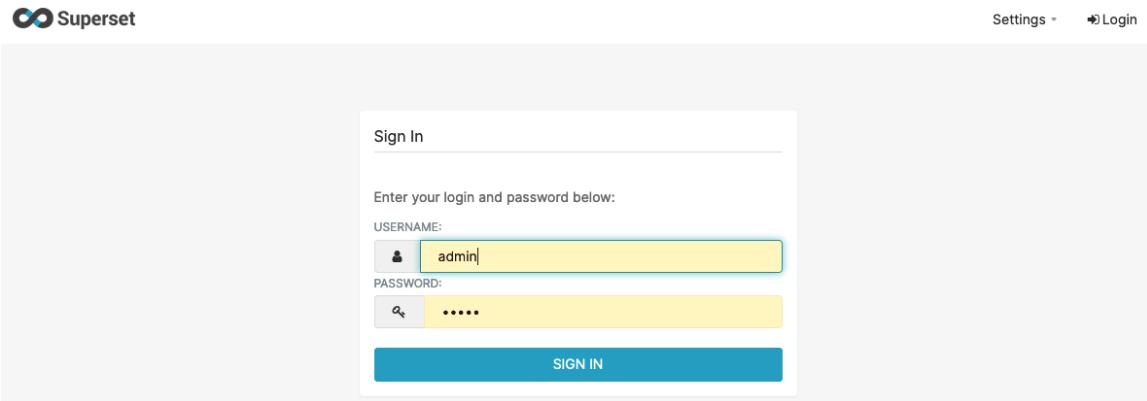
Choose the **Administration perspective**

1. Navigate to **Networking > Routes**.
2. Filter on the word Superset and open that route, by clicking on the URL as shown.



The screenshot shows the OpenShift Admin UI. On the left, there's a sidebar with 'Administrator' at the top, followed by 'Home', 'Operators', 'Workloads', 'Networking' (which is expanded), 'Services', and 'Routes'. Under 'Networking', 'Ingresses' and 'NetworkPolicies' are listed. On the right, the main area is titled 'Routes'. At the top, there's a filter bar with 'Name' dropdown set to 'superset' and a search input field also containing 'superset'. Below the filter are three columns: 'Name', 'Status', and 'Location'. A single row is visible, showing 'superset' in the Name column, 'Accepted' in the Status column, and a URL in the Location column: 'http://superset-ml-workshop.apps.cluster-bffc.bffc.sandbox60.opentlc.com'. A red box highlights this URL.

3. Enter credentials **admin / admin**.



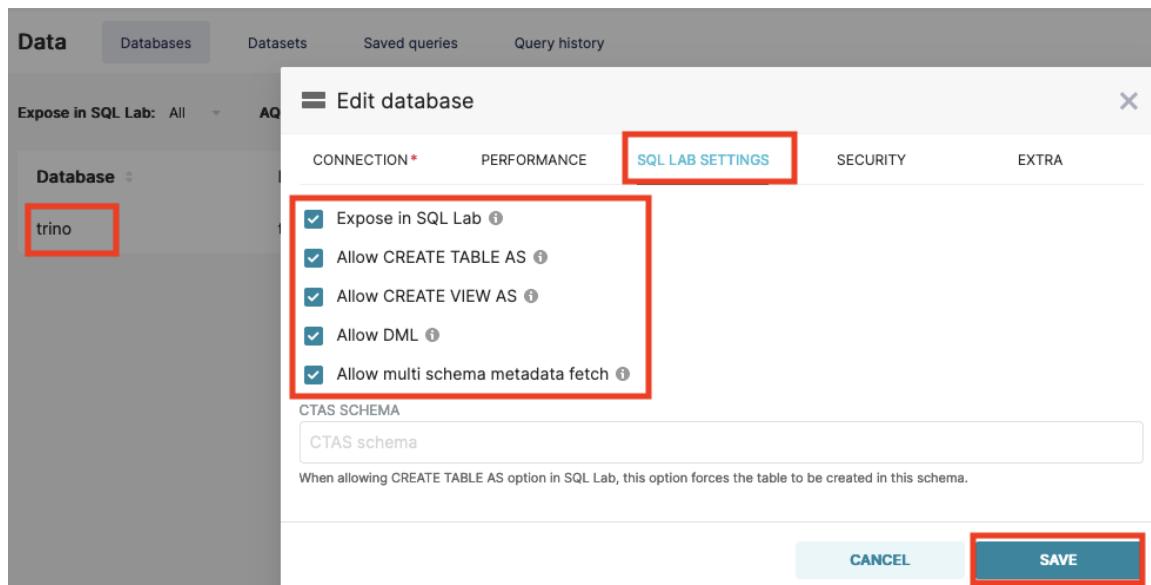
4. Choose menu **Data > Databases**. Edit the **trino** Database

As this is a shared service between all participants, and the setup has already been done by your instructor, we'll just show you the steps we took to connect Superset to Trino & from there to underlying data.

Database	Backend	AQE	DML	CSV upload	Expose in SQL Lab	Created by	Last modified	Actions
trino	trino	x	v	x	v	admin admin	27 minutes ago	 

Notice we simply added the URI **trino://admin@trino-service:8080/** to connect to Trino as shown. Test the Connection.

5. Move to the SQL LAB SETTINGS tab and notice we needed full access by selecting the checkboxes.

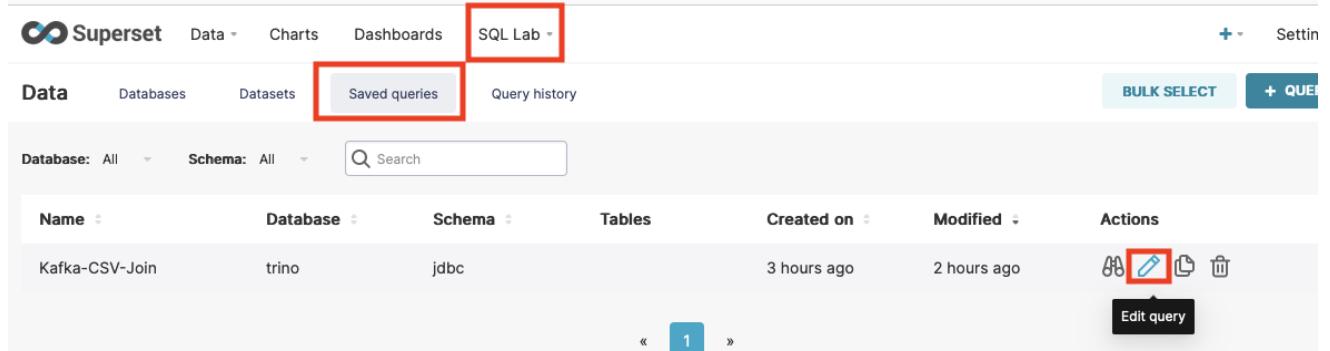


The screenshot shows the Red Hat Data interface. On the left, there's a sidebar with a 'Database' dropdown set to 'trino'. The main area has tabs for 'Data', 'Databases', 'Datasets', 'Saved queries', and 'Query history'. A modal window titled 'Edit database' is open over the main content. The modal has several tabs at the top: 'CONNECTION*', 'PERFORMANCE', 'SQL LAB SETTINGS' (which is highlighted with a red box), 'SECURITY', and 'EXTRA'. The 'SQL LAB SETTINGS' tab contains five checkboxes, all of which are checked and highlighted with a red box:

- Expose in SQL Lab ⓘ
- Allow CREATE TABLE AS ⓘ
- Allow CREATE VIEW AS ⓘ
- Allow DML ⓘ
- Allow multi schema metadata fetch ⓘ

Below these checkboxes is a section titled 'CTAS SCHEMA' with a dropdown menu set to 'CTAS schema'. A note below it says: 'When allowing CREATE TABLE AS option in SQL Lab, this option forces the table to be created in this schema.' At the bottom of the modal are two buttons: 'CANCEL' and 'SAVE' (which is also highlighted with a red box).

6. In Superset, choose **SQL LAB > Saved Queries**. Edit the query **Kafka-CSV-Join** as shown (though your query may be named differently)



Name	Database	Schema	Tables	Created on	Modified	Actions
Kafka-CSV-Join	trino	jdbc		3 hours ago	2 hours ago	

Earlier the workshop admin created a virtual '**table**' (hive.default.customer1) that uses the CSV data in our Minio S3 Object store as it's actual data - located in the bucket **rawdata**.

Earlier we also created a second virtual '**table**' backed by our Kafka streaming data. In our case this is the customer product consumption data.

Now Trino allows us to create a SQL Join across data that resides in S3 Object storage and Kafka as follows:

```
SELECT kafkaData.*, s3Data.*
from customerchurn.default.data kafkaData,
      hive.default.customer1 s3Data
where cast(kafkaData.customerId as VARCHAR) = s3Data.customerId
```

Very cool!



7. **Run** the Query. Notice the result set spanning data covering S3 and Kafka, joined on *customerId*. Click **Explore**

The screenshot shows a SQL Lab interface with a query editor at the top containing the following SQL code:

```
1 SELECT kafkaData.*, s3Data.*  
2 FROM customerchurn.default.kafkaData,  
3      hive.default.customer1 s3Data  
4 WHERE cast(kafkaData.customerId AS VARCHAR) = s3Data.customerId
```

Below the editor, there are buttons for **RUN**, **LIMIT: 1 000**, and **00:00:07:17**. To the right are **SAVE**, **COPY LINK**, and a **...** button. At the bottom, there are tabs for **RESULTS** and **QUERY HISTORY**. The **RESULTS** tab is active, displaying a table with 12 columns: **customerId**, **premium**, **relationshipmanager**, **primarychannel**, **hascreditcard**, **debitcard**, **incomeprotection**, **wealthmanagement**, **homeequityloans**, **moneymarketaccount**, and **creditrating**. The table contains 12 rows of data.

8. Select **Save As New**

Give it a name appending your username to **Kafka-CSV-Join**. In my case, with user29: **Kafka-CSV-Join-user29**. Then **Save & Explore**:

The dialog box is titled "Save or Overwrite Dataset". It contains the following fields:

- A label "Save this query as a virtual dataset to continue exploring"
- A radio button group for "Save as new" (selected) and "Overwrite existing". The "Save as new" option is highlighted with a red border.
- An input field for the dataset name, containing "Kafka-CSV-Join-user29".
- A second input field for "Select or type dataset name" with the placeholder "Select or type dataset name".
- A large blue "SAVE & EXPLORE" button at the bottom right, which is also highlighted with a red border.



You may see an error saving the dataset. Ignore it (this is a small bug) - it probably saved successfully.

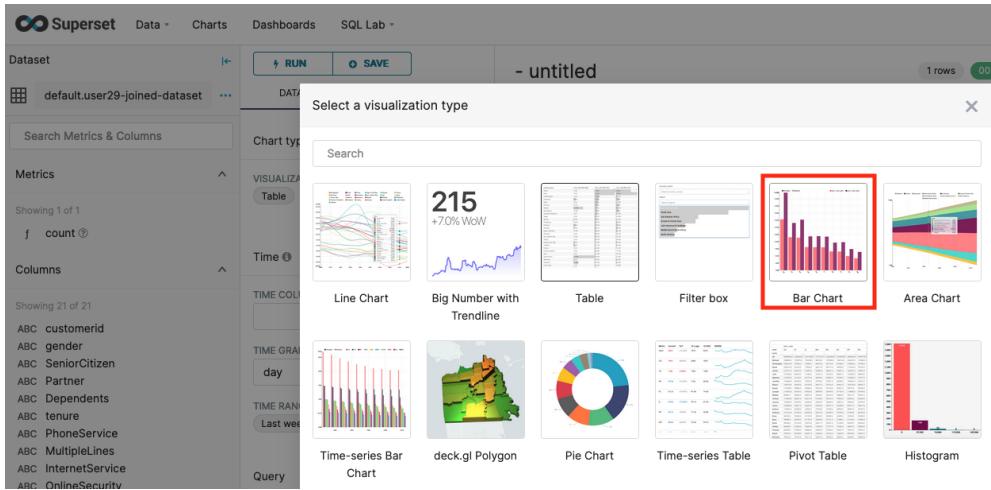
9. Move to **Data > DataSets**, find your new dataset - then click it to open it:

The screenshot shows the Superset interface with the 'Data' tab selected. Under the 'Datasets' tab, there is a list of datasets. The first dataset, 'user29-joined-dataset', is highlighted with a red box. The list includes columns for Name, Type, Source, Schema, Modified, Modified by, Owners, and Actions. At the bottom, there is a page number '1' and a note '1-1 of 1'.

10. By default **Table** Visualisation Type is selected

The screenshot shows the Superset interface for a specific dataset named 'default.user29-joined-dataset'. On the left, there are sections for Metrics and Columns. On the right, there are tabs for RUN and SAVE, and sections for Chart type, Visualization Type (with 'Table' selected), Time, and Query. The 'Table' visualization type is highlighted with a red box.

Click **Table**. You have a large number of Visualization Types to choose from. Click **Bar Chart**:



You can now start visualising and understanding your data. First we'll create a bar chart representation of the entire dataset, representing the count of the different categories of *Primary Channel*:

- *Branch*
- *Mobile and*
- *No primary channel.*



Click **Last Week** under **TIME RANGE**. On the popup, click the **Range Type** dropdown, select **No Filter** and then **Apply**.

Chart type: Bar Chart

Time: _timestamp

TIME COLUMN: _timestamp

TIME RANGE: Last week (highlighted)

Query: f(x) count

METRICS: f(x) count (highlighted)

FILTERS: + Add filter

SERIES: primarychannel

Edit time range

RANGE TYPE

- Last (highlighted)
- Previous
- Custom
- Advanced
- No filter (highlighted)
- last month
- last quarter
- last year

Actual time range: 2021-10-07 ≤ col < 2021-10-14

CANCEL APPLY (highlighted)

Select **Count** under **Metrics** and select **PrimaryChannel** under **Series**.

Name the chart **Count-PrimaryChannel-userXX** (in my case

Count-PrimaryChannel-user29) and click **Save**.

DATA CUSTOMIZE

RUN SAVE (highlighted)

Count-PrimaryChannel-user29

1 rows 00:00:03.35

DATA CUSTOMIZE

Chart type: Bar Chart

TIME COLUMN: _timestamp

TIME RANGE: No filter

Query

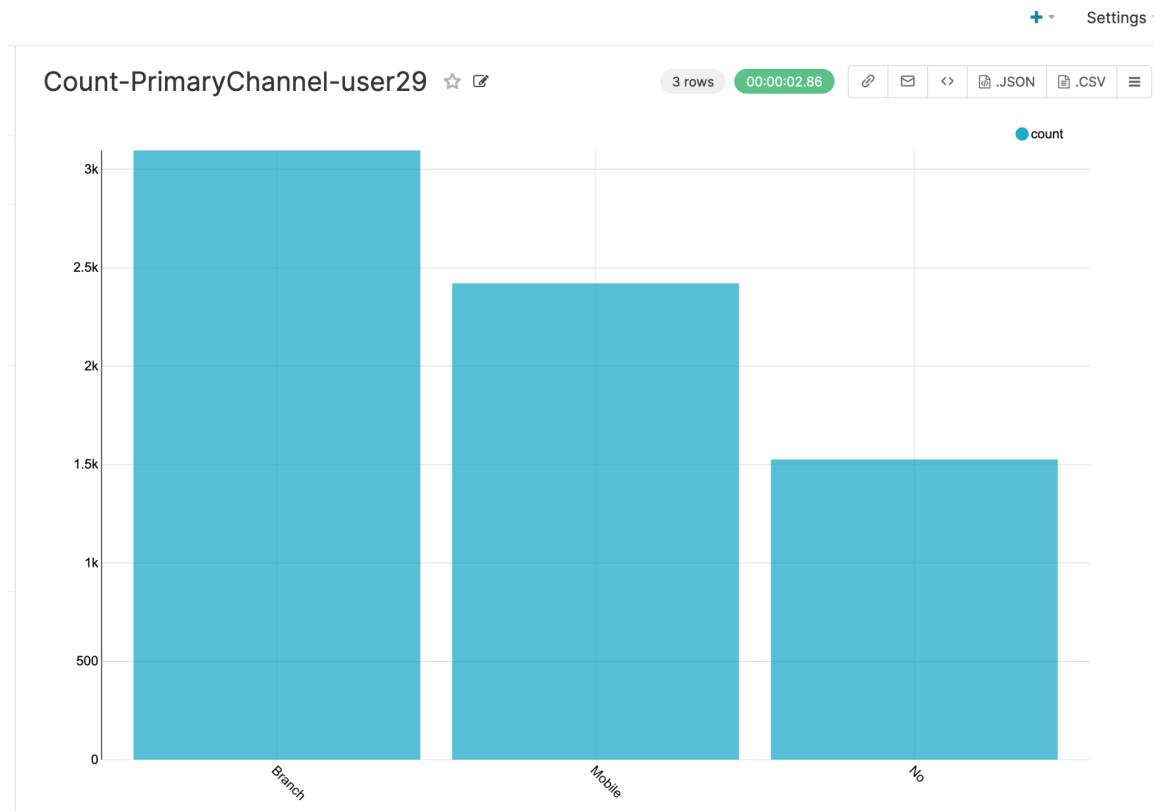
RUN QUERY

METRICS: f(x) count (highlighted)

FILTERS: + Add filter

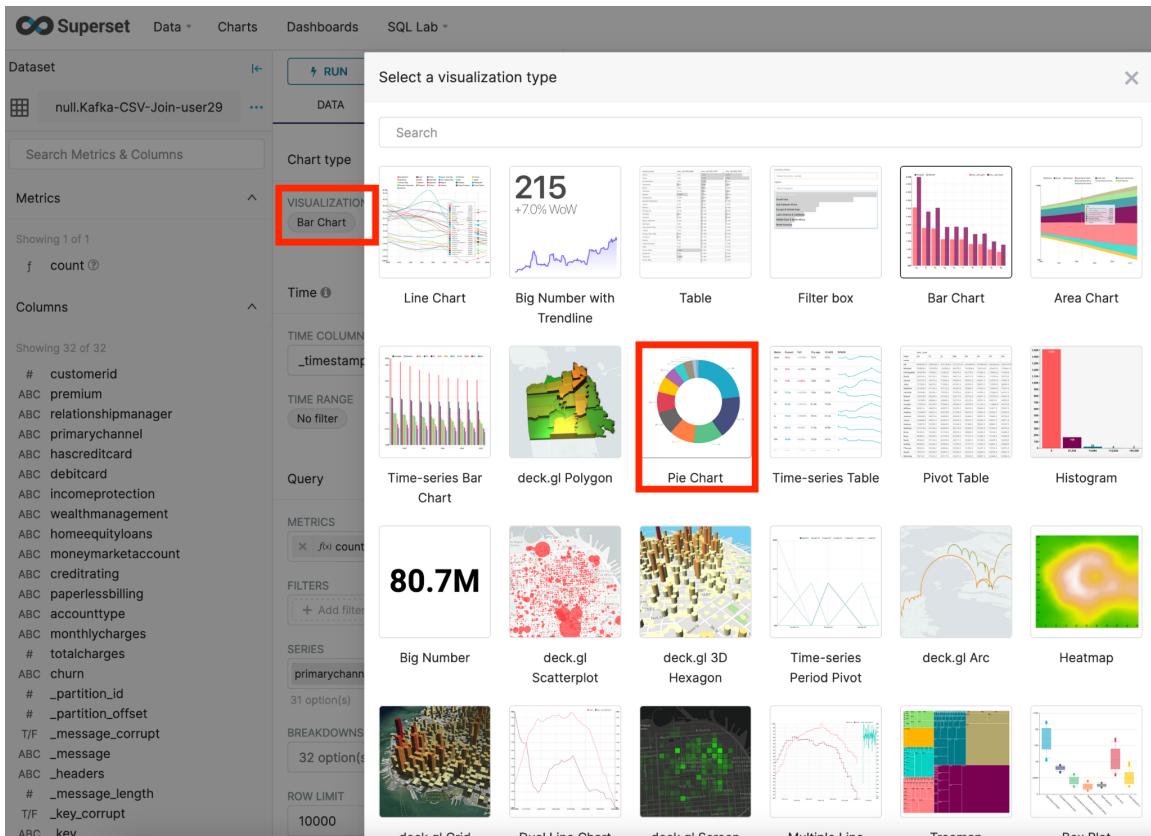
SERIES: primarychannel (highlighted)

This will cause the query to run - and you will be presented with this bar chart - showing the breakdown of the resultset by Primary Channel, Branch, Mobile or None.



11. Now we'll create a Pie Chart - and add another dimension - *account type*.

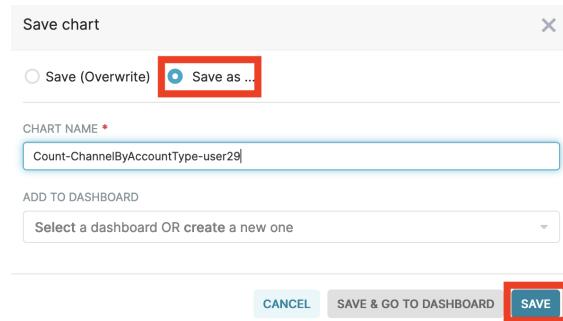
On your open Bar Chart screen, click **Bar Chart**. On the popup click **Pie Chart**.



The screenshot shows the Superset interface with the 'Bar Chart' visualization selected in the left sidebar. A red box highlights the 'Pie Chart' option in the central visualization grid.

Now make the following changes:

- On the **Group By** dropdown, add **accounttype**
- Name the chart **Count-ChannelByAccountType-userXX** (in my case **Count-ChannelByAccountType-user29**) and click **Save**.
- Go with the defaults on the popup. Click **Save As** then **Save**



Save chart

Save (Overwrite) Save as...

CHART NAME *

Count-ChannelByAccountType-user29

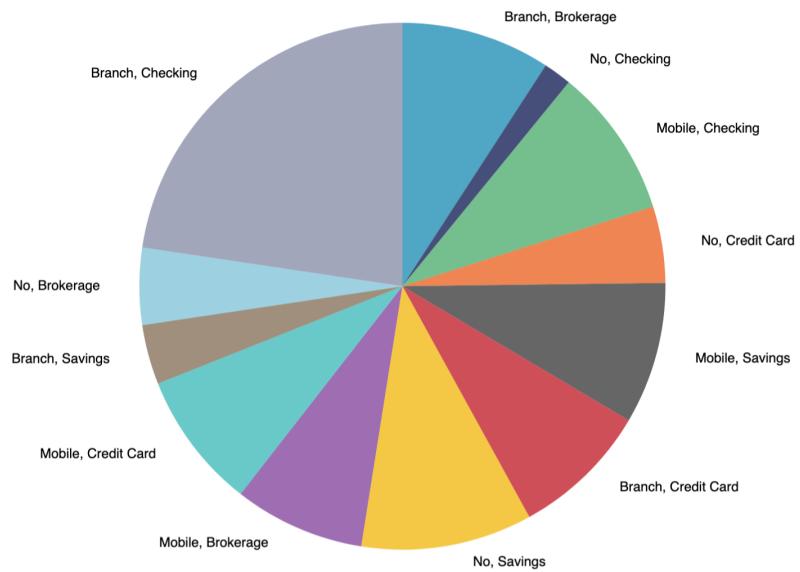
ADD TO DASHBOARD

Select a dashboard OR create a new one

CANCEL SAVE & GO TO DASHBOARD **SAVE**

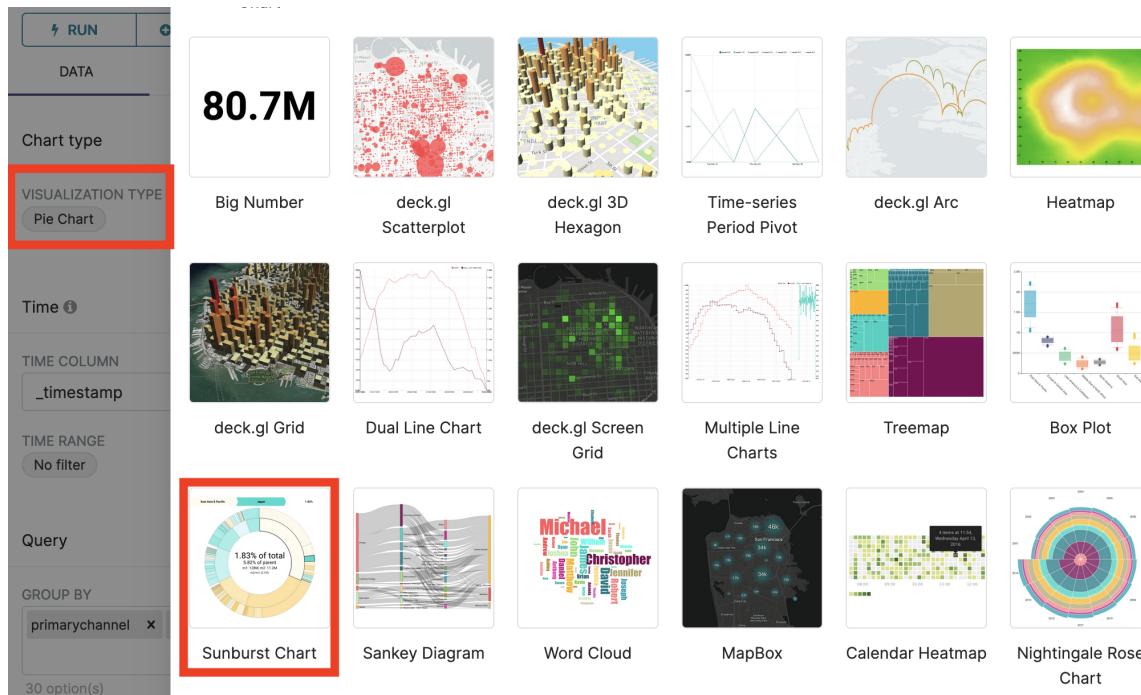
This will cause the following pie chart to display - grouping *Primary Channel* by *Account Type*

Count-ChannelByAccountType-user29 ⭐ ⓘ 12 rows 00:00:01.72 ⌂ ⌃ ⌄ .JSON ⌄ .CSV ⌄



12. Next, we're going to showcase another unusual though informative visualization – the Sunburst Chart. It allows you to visualise splits of your data with varying degrees of granularity – within the same chart.

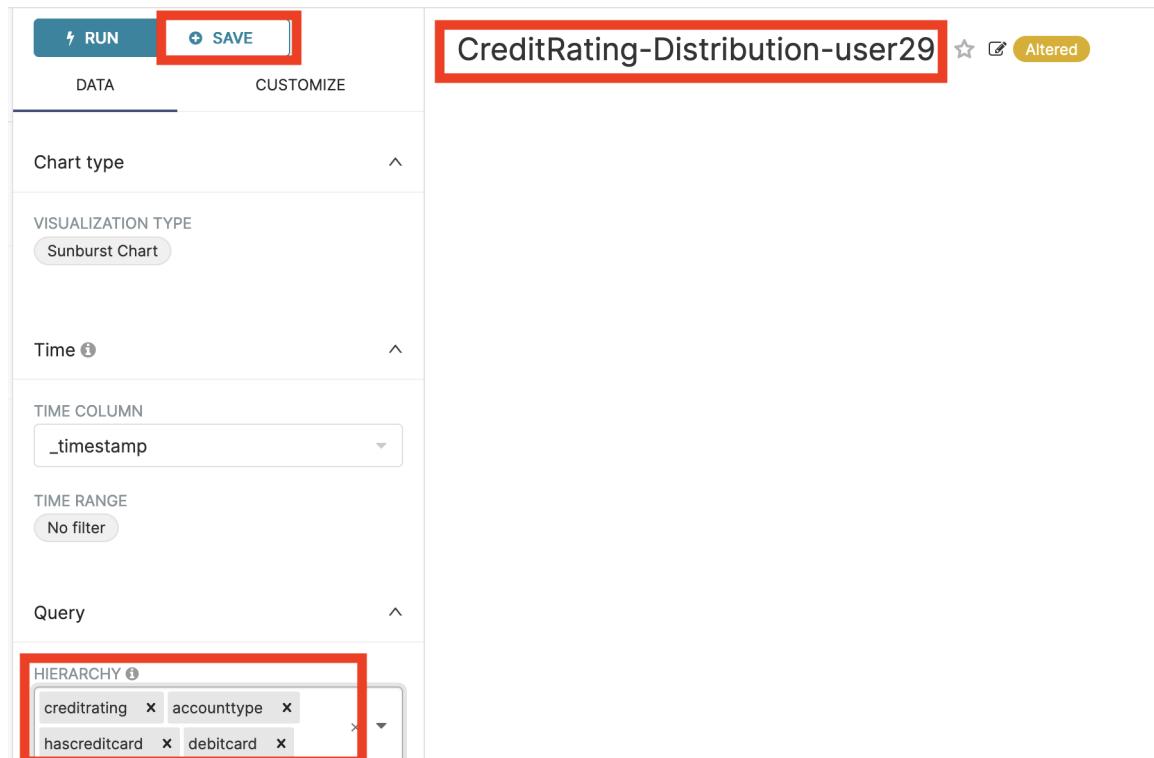
On your open Pie Chart screen, click **Pie Chart** then on the popup, scroll down and click **Sunburst Chart** as shown:



Make the following changes

- Name the chart **CreditRating-Distribution-userXX** (in my case **CreditRating-Distribution-user29**)
- On the **Hierarchy** dropdown, select
 - creditrating**
 - accounttype**
 - hascreditcard**
 - debitcard**
- click **Save**

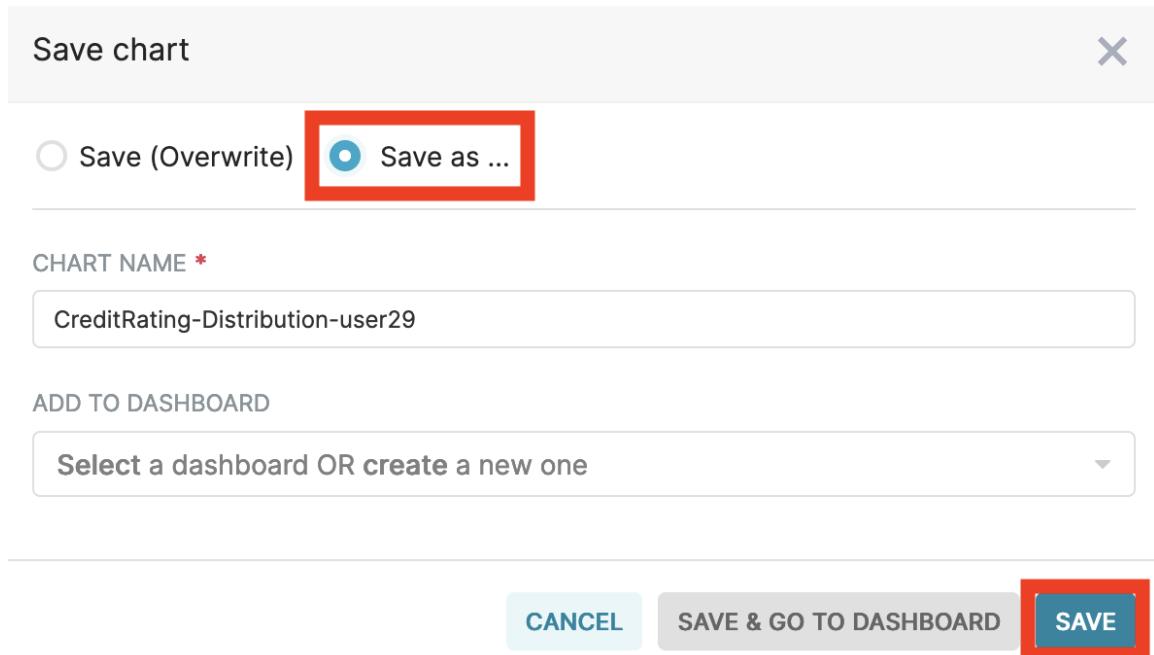
as shown:



The screenshot shows a data visualization interface with the following configuration:

- Top Bar:** Includes 'RUN' and 'SAVE' buttons, and 'DATA' and 'CUSTOMIZE' tabs.
- Title:** CreditRating-Distribution-user29 (highlighted with a red box).
- Chart Type:** Sunburst Chart.
- Time:** Set to '_timestamp'.
- Query Hierarchy:** Contains 'creditrating', 'accounttype', 'hascreditcard', and 'debitcard' (highlighted with a red box).

Confirm by clicking **Save As** then **Save**



Save chart

Save (Overwrite) Save as ... (highlighted with a red box)

CHART NAME *

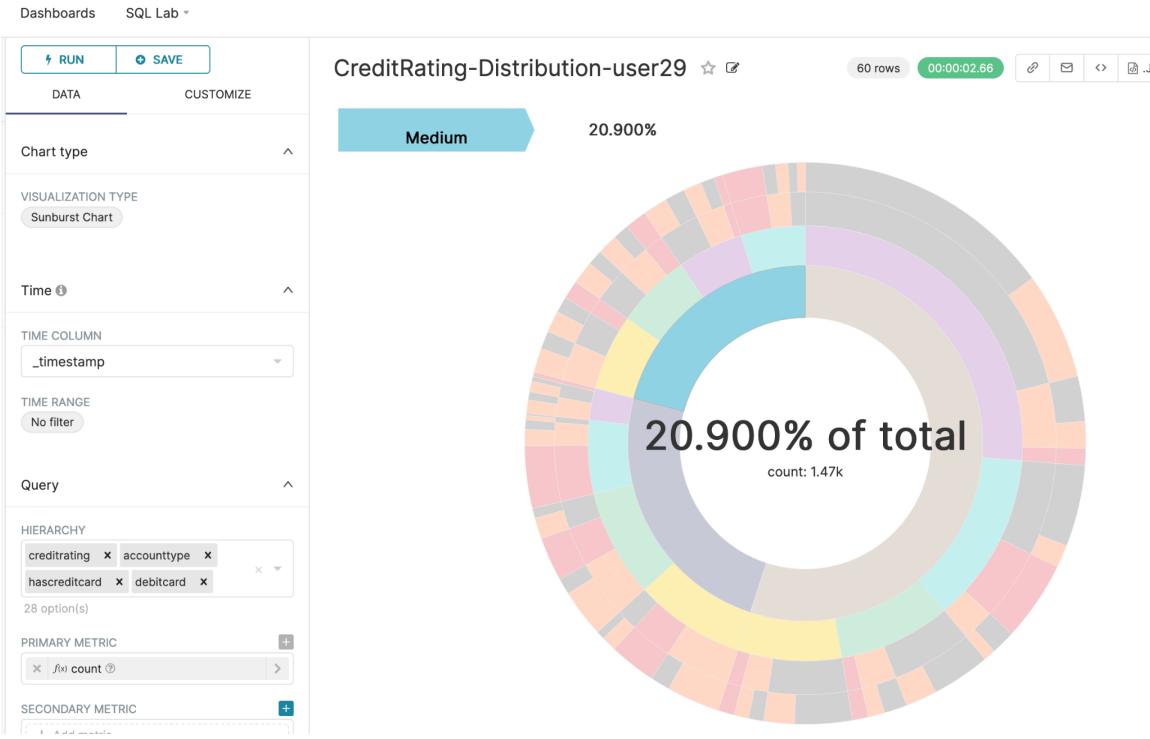
CreditRating-Distribution-user29

ADD TO DASHBOARD

Select a dashboard OR create a new one

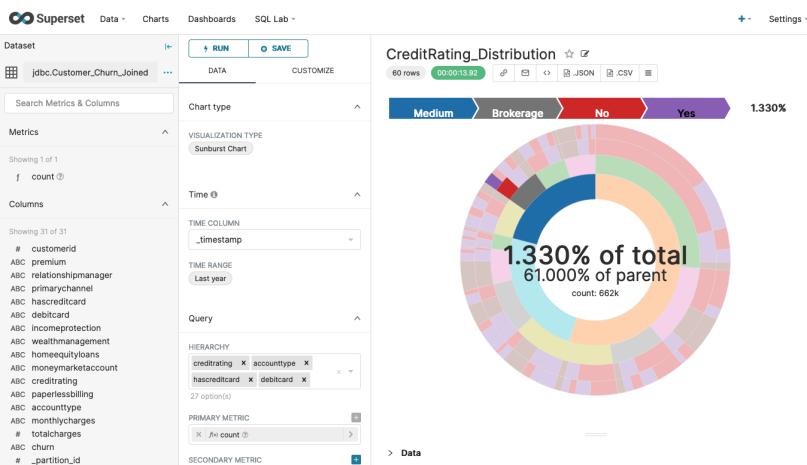
CANCEL **SAVE & GO TO DASHBOARD** **SAVE** (highlighted with a red box)

A chart similar to the following will appear.

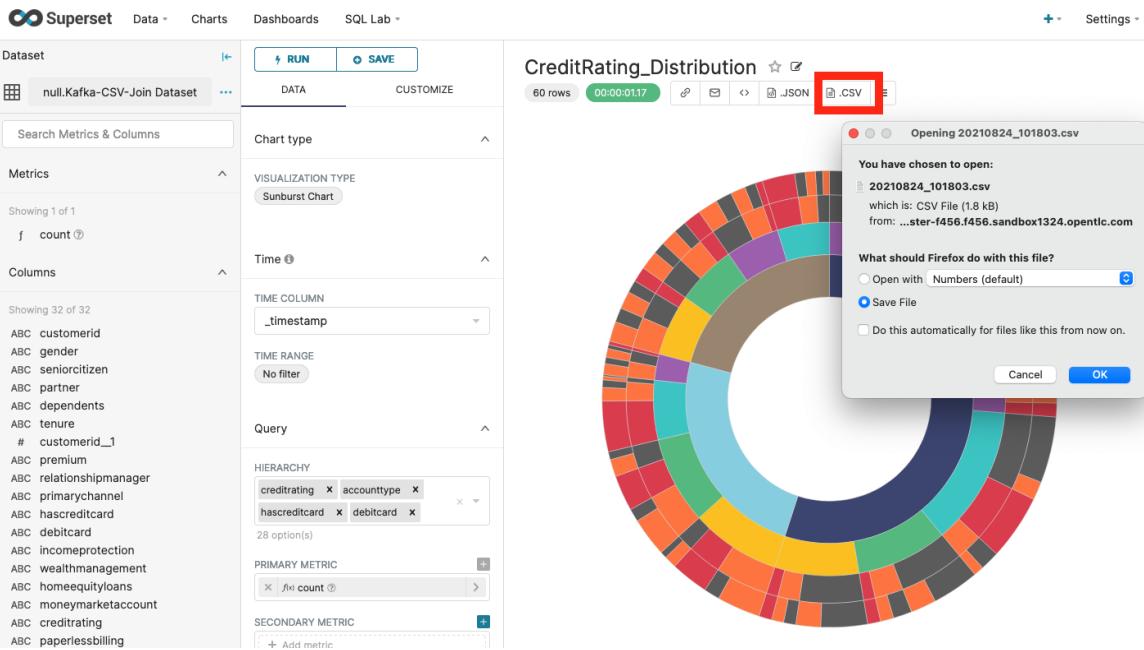


If you hover over the inner circle, the breakdown according to the first hierarchical element is shown, in my case *medium* credit rating.

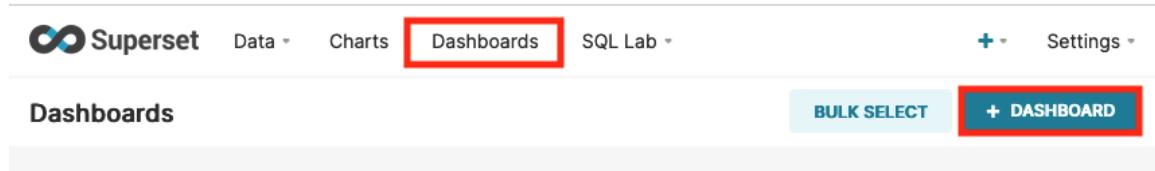
You can see, you can also further refine, by hovering out towards over the outer circle - giving a very fine grained breakdown - according to the 4 hierarchies:



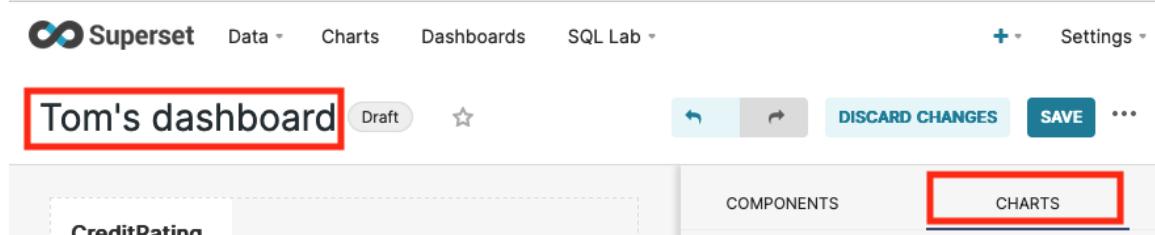
Any of these datasets can be easily exported to JSON or CSV – as shown below. Then fed for example to an AI model training use case.



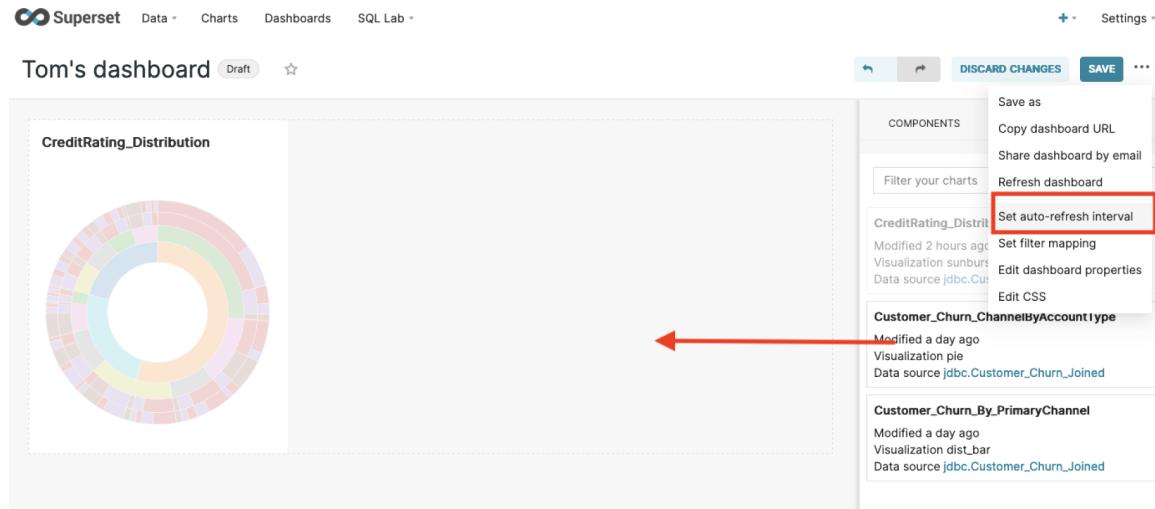
13. Finally we're going to show you how to create a dashboard - to which we can add previously saved charts. Choose Dashboards -> Add Dashboard as shown:



Name it and choose the Charts tab:



You can simply drag your charts from the right over to the display panel on the left as shown. You can also make them dynamic by choosing a refresh interval. Very cool!



Feel free to continue to experiment different ways of accessing and visualising the underlying data.