Published in HashmapInc

Senthil Anandarangan Follow
Aug 14, 2020 · 8 min read · Listen

Save

Using Starburst Presto to Federate SQL Queries Across Multiple Data Sources

by Senthil Anandarangan and Ganesh Hegde

The data virtualization world as we know it is full of ifs and buts. Many organizations are constantly experimenting on their data management architecture and developing scalable solutions to custom fit their existing and future needs.

Although doing ETL (or ELT) to centralize large data for reporting and analytics is the preferred approach for OLAP queries, it's not always the right approach that fits all business needs. Allowing real-time access to multiple source systems' data by hiding the technical details and physical location of data and combining them into one single federated SQL query view is the challenge that Data Virtualization tools take on!

In this blog post, we want to share our experiences with trying out data virtualization capabilities of the Starburst Enterprise Presto SQL query engine querying directly on the source data of **multiple** data sources.

From One to Many

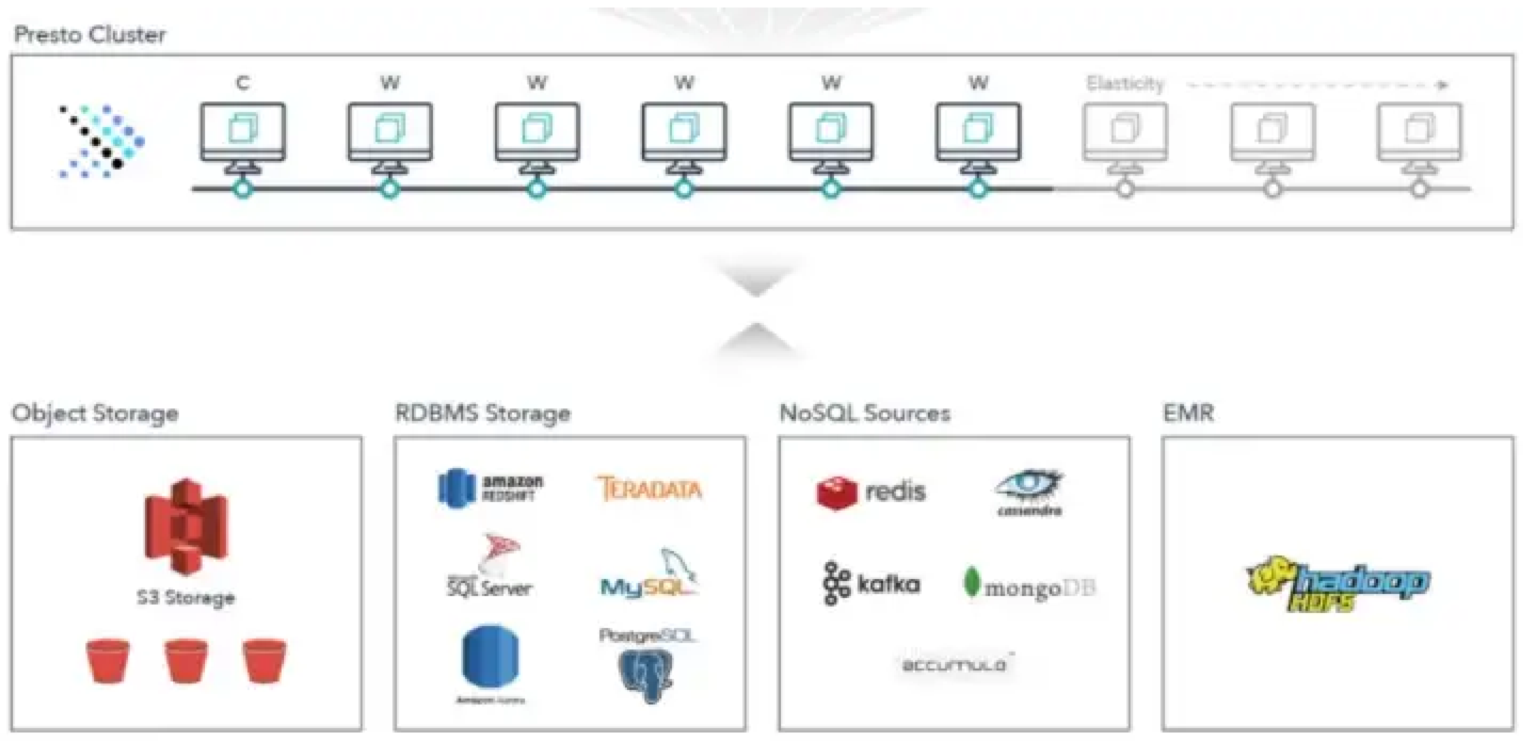
In our last blog post, we covered how to set up and run Starburst Enterprise Presto to run SQL queries directly over S3 data sources as well as how to connect Starburst Presto with Looker to create quick visualizations over the source data and analyze data patterns. Here it is in case you missed it:

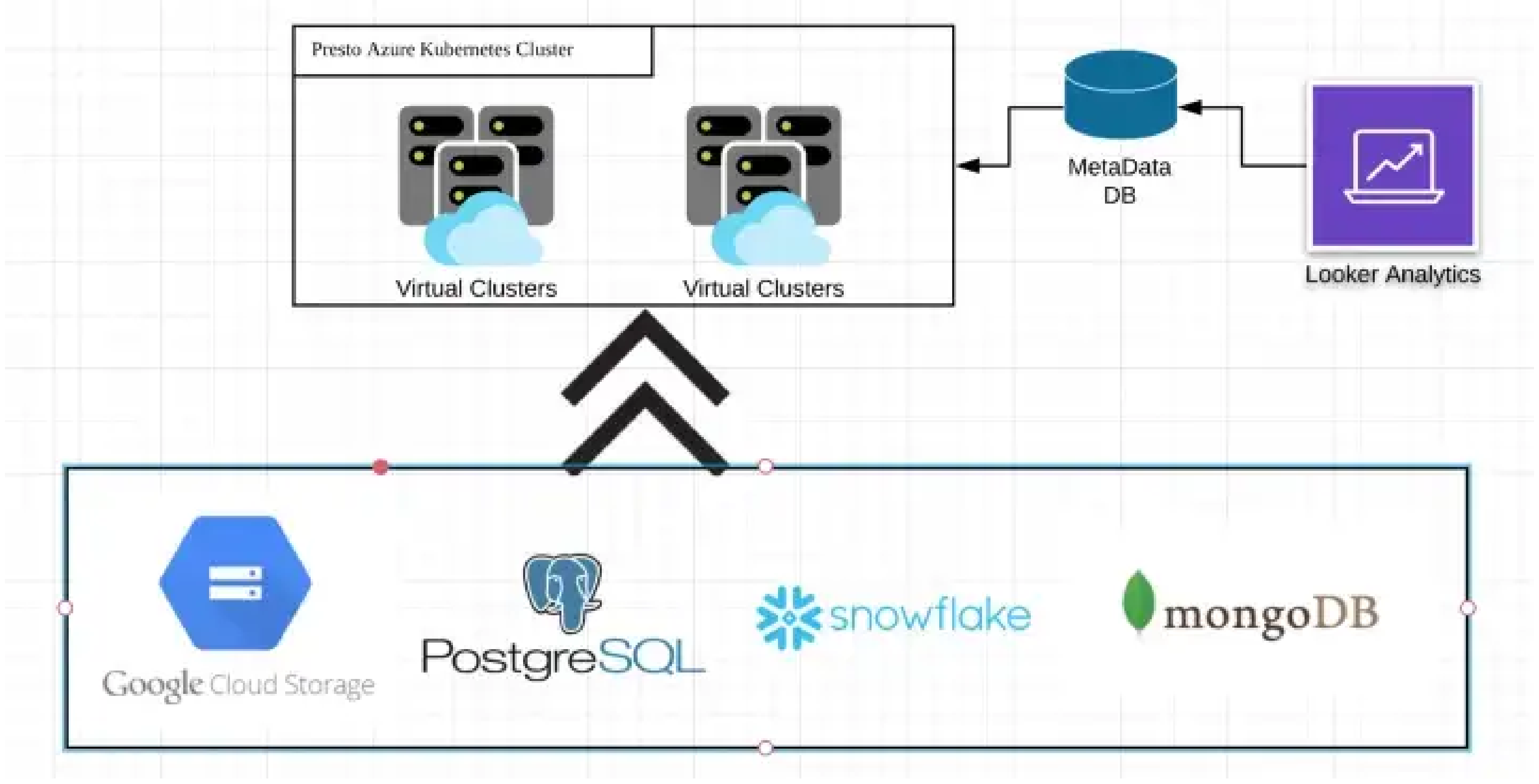
How to Connect S3 Data to Looker using Starburst Presto

In this era of increasing adoption to Data-driven solutions, faster Data accessibility becomes more and more critical…

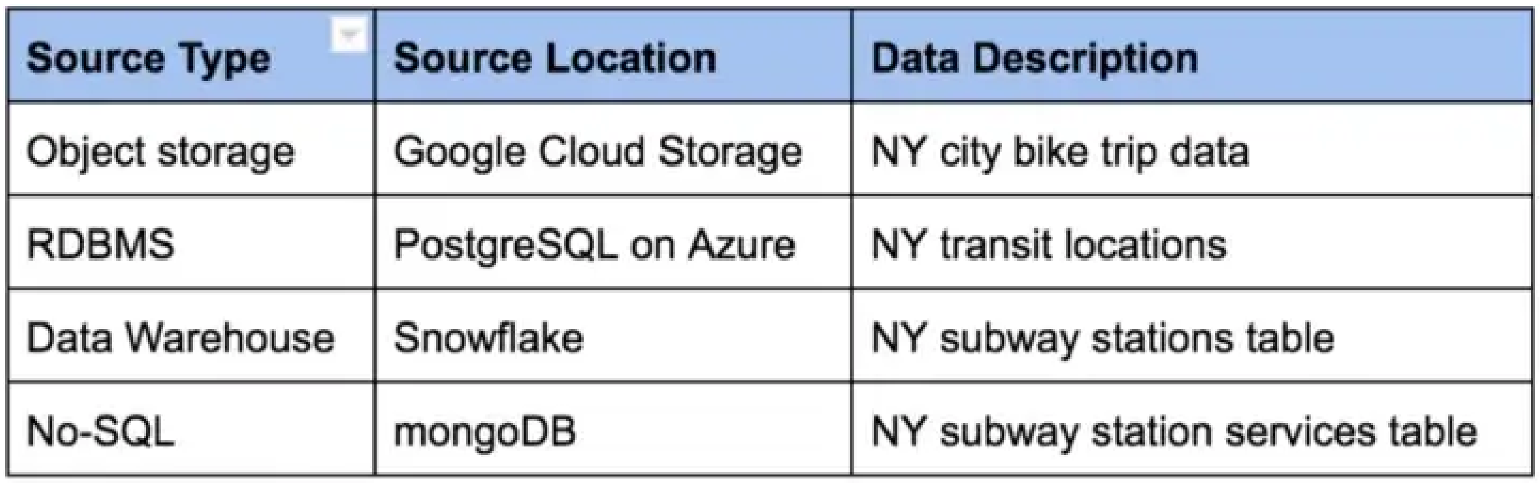
medium.com

A single Presto query can combine data from multiple sources. Presto offers connectors to data sources including files in HDFS, AWS S3, Azure Blob/ADLS, Google Cloud Storage, MySQL, PostgresSQL, SQLServer, Oracle, AWS Redshift, Snowflake, BigQuery, Cassandra, MongoDB, Redis and many more. Refer to the Presto documentation on various data sources and connectors here.





From the list of available data connectors, we picked four, created a Looker analytic view of New York subway stations, and plotted the most used stations for city bike trips.



Source Type	Source Location	Data Description
Object storage	Google Cloud Storage	NY city bike trip data
RDBMS	PostgreSQL on Azure	NY transit locations
Data Warehouse	Snowflake	NY subway stations table
No-SQL	mongoDB	NY subway station services table

Test Data source JSON:

http://gbfs.citibikenyc.com/gbfs/gbfs.json

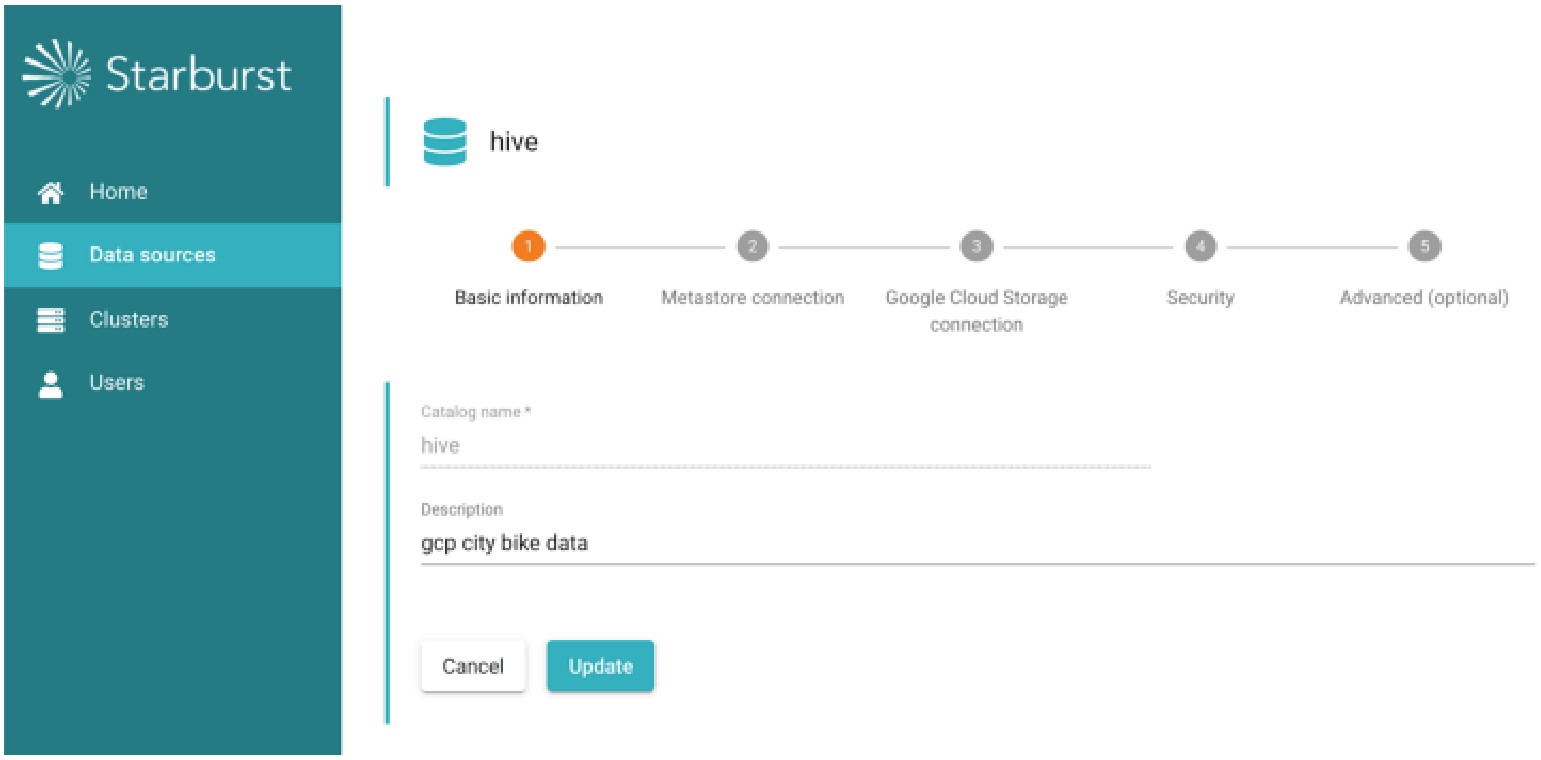
https://www.citibikenyc.com/system-data

Assuming, you have a running Presto Cluster setup (as described in our last blog post).

Start connecting your data sources using Mission Control UI

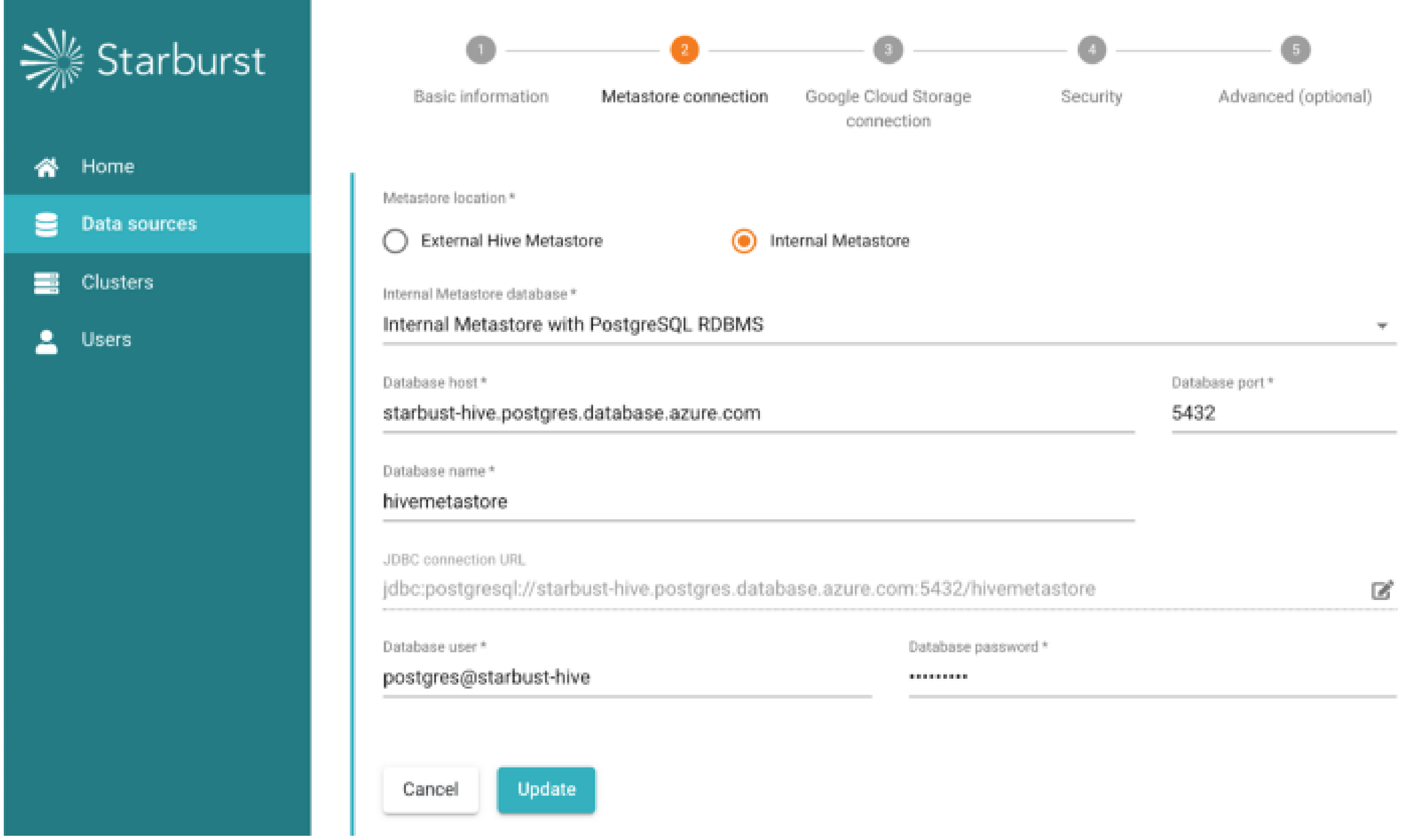
Steps to Connect Google Cloud Storage

1. Add + Data sources in the Mission Control and choose Google Cloud storage.
2. Note the catalog name is defaulted to 'hive' as it is enforced for object storage connections using Internal Metastore. Although it is a little confusing, this is the way Presto understands to differentiate Metastore connections in case of multiple object-store queries.
3. Choose Internal Metastore configuration and provide all details as shown.
4. Attach Google storage access key JSON for the storage location and secure storage access.



Catalog name *
hive

Description
gcp city bike data



Metastore location *

External Hive Metastore [✓ Internal Metastore]

Internal Metastore database *
Internal Metastore with PostgreSQL RDBMS

Database host *
starbust-hive.postgres.database.azure.com

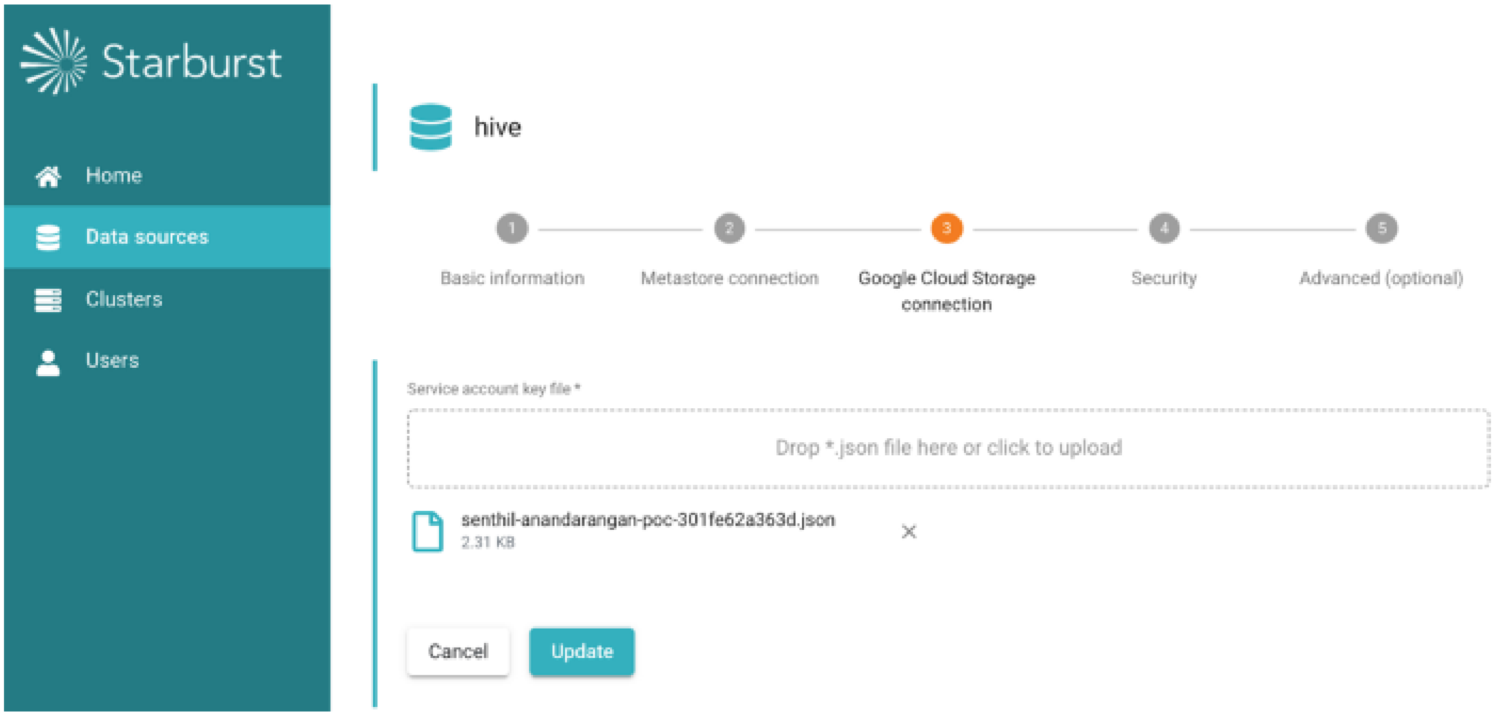
Database port *
5432

Database name *
hivemetastore

JDBC connection URL
jdbc:postgresql://starbust-hive.postgres.database.azure.com:5432/hivemetastore

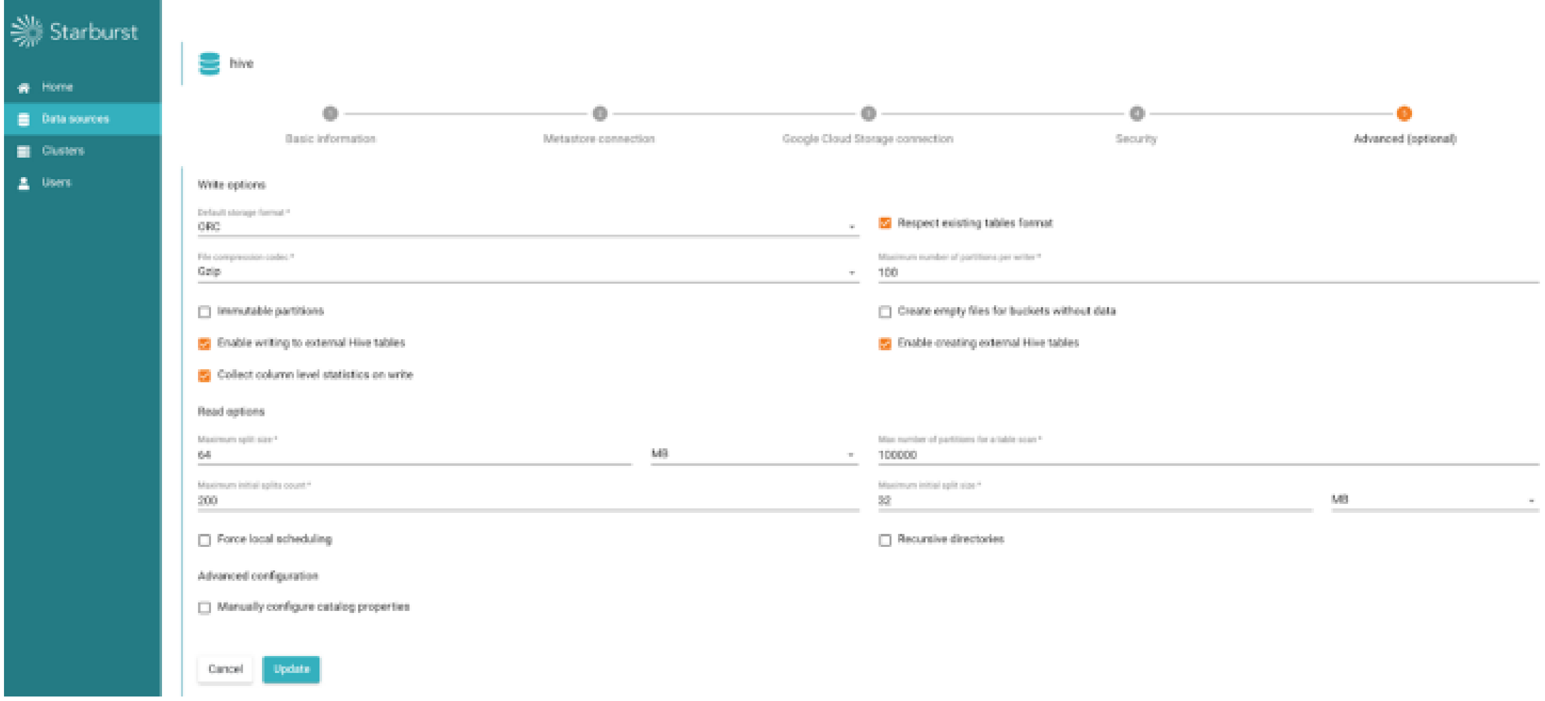
Database user *
postgres@starbust-hive

Database password *
••••••••••



Service account key file *

senthil-anandarangan-poc-301fe62a363d.json
2.31 KB


Starburst

Home
Data sources
Clusters
Users

hive

Basic information — Metastore connection — Google Cloud Storage connection — Security — Advanced (optional)

Write options

Default storage format *: ORC
☑ Respect existing tables format

File compression codec *: Gzip
Maximum number of partitions per writer *: 100

☐ Immutable partitions
☐ Create empty files for buckets without data
☑ Enable writing to external Hive tables
☑ Enable creating external Hive tables
☑ Collect column level statistics on write

Read options

Maximum split size *: 64 MB
Max number of partitions for a table scan *: 100000
Maximum initial splits count *: 200
Maximum initial split size *: 32 MB

☐ Force local scheduling
☐ Recursive directories

Advanced configuration

☐ Manually configure catalog properties

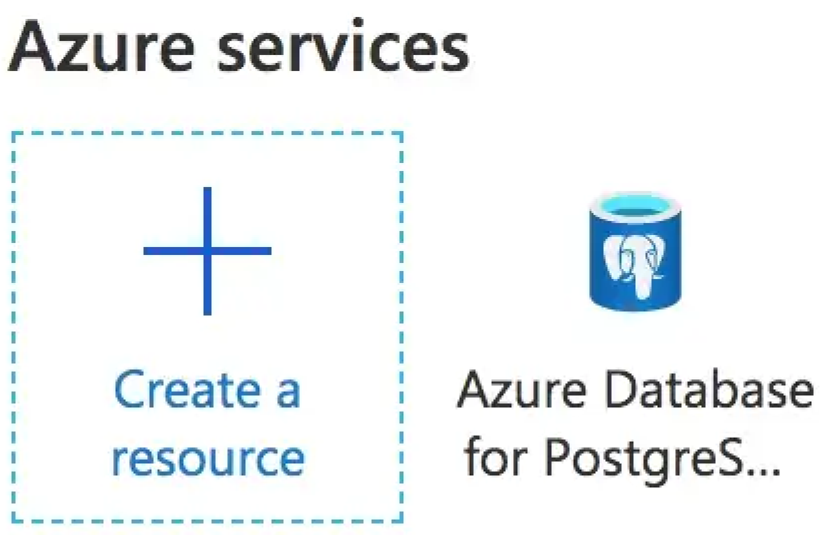
Cancel | Update

After the update, open cluster configuration in Mission Control and add/include Google Cloud storage data source.

That's it! We have configured the Google Cloud storage dataset. In order to start querying, it requires a restart of the Mission Control Clusters, however, we will hold on to restart until we connect the other three data sources.

Steps to Connect PostgreSQL on Azure Cloud

1. Add + Data sources in the Mission Control and choose PostgreSQL connector.
2. The configuration is very straight forward here; only the DB connection string parameters that you would provide here to establish a connection.



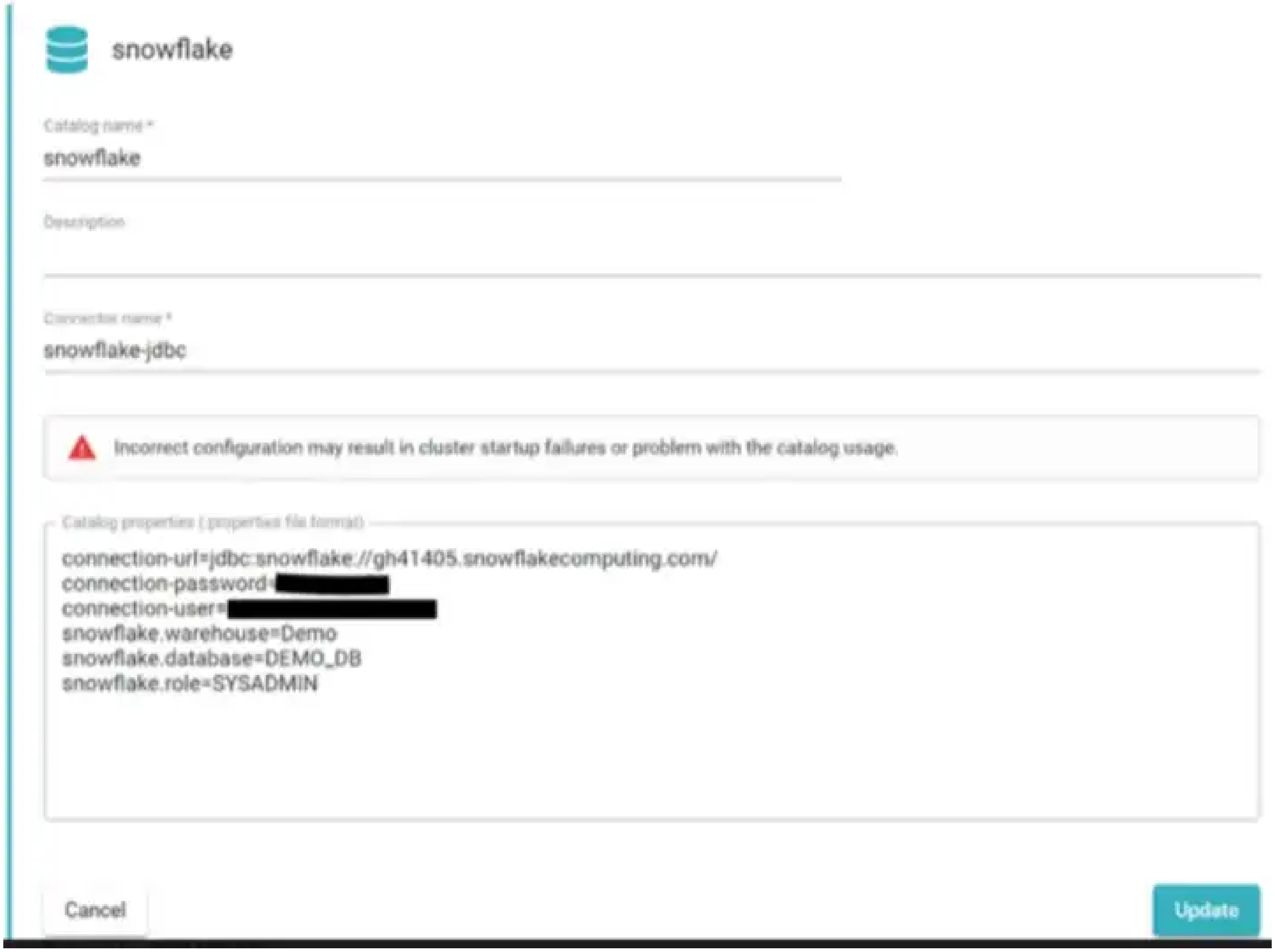
Steps to Connect to MongoDB no-SQL database

1. Add + Data sources in the Mission Control and choose MongoDB connector.
2. Provide the Mongo seed string to establish a connection.

Steps to Connect to Snowflake Data Warehouse

Although the Snowflake connector is available and supported by Presto, my Starburst version is not the latest and it was not listed in the available data sources to Add. There are 2 alternate ways to add unlisted connectors in Mission Control (of course, only if they are supported by Presto!).

1. Check if you find a link on the data sources screen that says 'Add one manually'. Once you add a manual data source, it will look like the example below in Snowflake:



Catalog name *
snowflake

Description

Connector name *
snowflake-jdbc

Incorrect configuration may result in cluster startup failures or problem with the catalog usage.

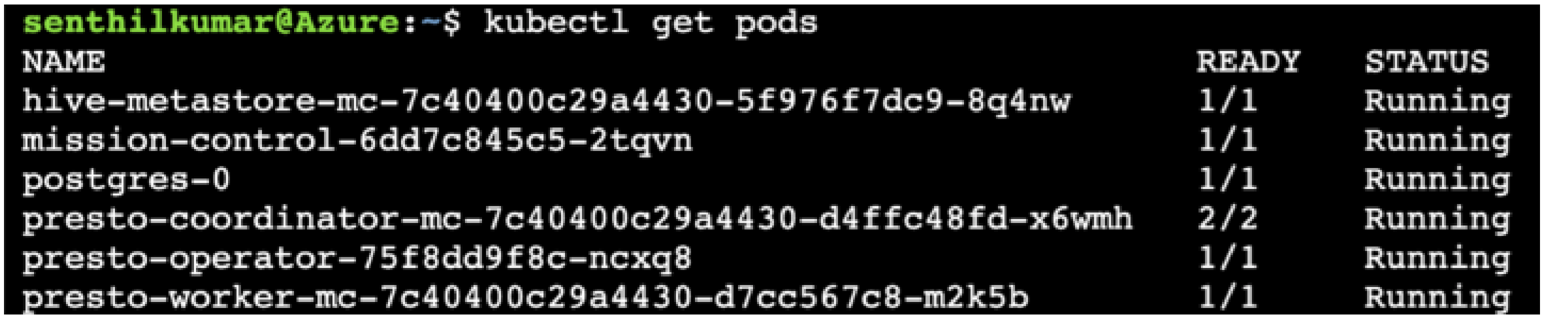
Catalog properties (properties file format)

```
connection-url=jdbc:snowflake://gh41405.snowflakecomputing.com/
connection-password=[redacted]
connection-user=[redacted]
snowflake.warehouse=Demo
snowflake.database=DEMO_DB
snowflake.role=SYSADMIN
```

2. If you do not see a Manual Add option, then you are probably not on the latest release of Mission Control. The alternative is to get .22 version installed where you can create manual

connectors. Since we do not want to upgrade at this point, here is the other workaround that added Snowflake as bootstrap entry:

To add a connector that isn't supported by Mission Control, you will need to create a bootstrap file that adds this file to the /etc/Presto/catalog directory of the Co-ordinator POD:



```
senthilkumar@Azure:~$ kubectl get pods
NAME                                                    READY   STATUS
hive-metastore-mc-7c40400c29a4430-5f976f7dc9-8q4nw      1/1     Running
mission-control-6dd7c845c5-2tqvn                        1/1     Running
postgres-0                                              1/1     Running
presto-coordinator-mc-7c40400c29a4430-d4ffc48fd-x6wmh   2/2     Running
presto-operator-75f8dd9f8c-ncxq8                        1/1     Running
presto-worker-mc-7c40400c29a4430-d7cc567c8-m2k5b        1/1     Running
```

Using a bootstrap:

1. Create yaml file below (shown as an example, any connector would work in here):

```
$ more snowflake.yaml
kind: ConfigMap
apiVersion: v1
metadata:
name: snowflake-connector
data:
bootstrap.sh: |-
tee << EOF > /etc/presto/catalog/snowflake.properties
connector.name=snowflake-jdbc
connection-url=jdbc:snowflake://xxxxxxxxxxxxx.snowflakecomputing.com/
connection-password=xxxxxxxxxxxxxxxxxx
connection-user=SNOWFLAKE_DEMO
snowflake.warehouse=Demo
snowflake.database=DEMO_DB
snowflake.role=SYSADMIN
EOF
```

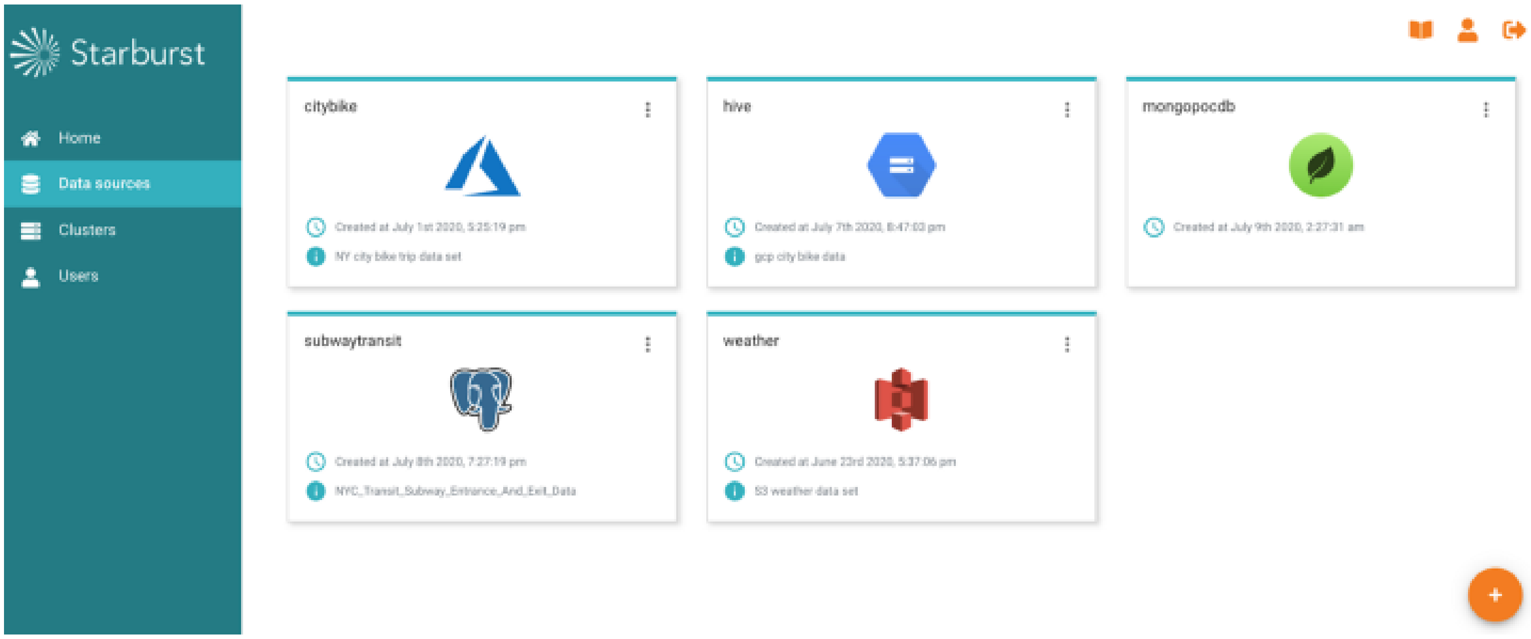
2. kubectl apply -f snowflake.yaml

3. Under the General Configuration click on the "Add Bootstrap script volume" and fill "snowflake-connector" as the name of the ConfigMap.

4. Save and restart the cluster.

Multiple Data Sources Connected View

After you connected all the above data sources, the view should look like below:


Starburst
Home
Data sources
Clusters
Users
citybike
Created at July 1st 2020, 5:25:19 pm
NY city bike trip data set
hive
Created at July 7th 2020, 8:47:03 pm
gcp city bike data
mongopocdb
Created at July 9th 2020, 2:27:31 am
subwaytransit
Created at July 8th 2020, 7:27:19 pm
NYC_Transit_Subway_Entrance_And_Exit_Data
weather
Created at June 23rd 2020, 5:37:06 pm
S3 weather data set
+

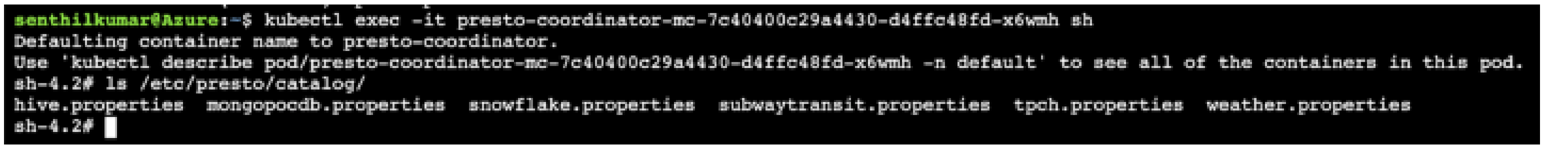
There are a few important things to note here:

1. Azure ADLS and AWS S3 are also present from my other project and are not used for this demo. We can have as many data sources connected and can selectively configure [add/remove] them in the Run Cluster configuration.

2. Keep in mind, Presto mandates a hive metastore connection for object storage connections. You can configure a connection to multiple object stores (ADLS, S3, etc.) and use the same Presto cluster to query them. For the 2nd connection (ADLS), configure it to use an external hive metastore and point it to the same one being used by the 1st (S3) connection. In terms of configuration, suppose if you have S3 and ADLS and are trying to query from both in the same SQL query, then the S3 connection will have Internal Metastore configured to the Hive Metastore DB connection parameters. ADLS will have the External Hive Metastore configured and pointing to the Internal Metastore connection hostname which should look like 'hive-metastore-mc-0xxxxxxx.default.svc.cluster.local' (if you are in doubt, this hostname can be located in the hive.properties file in /etc/presto/catalog of Co-ordinator POD).

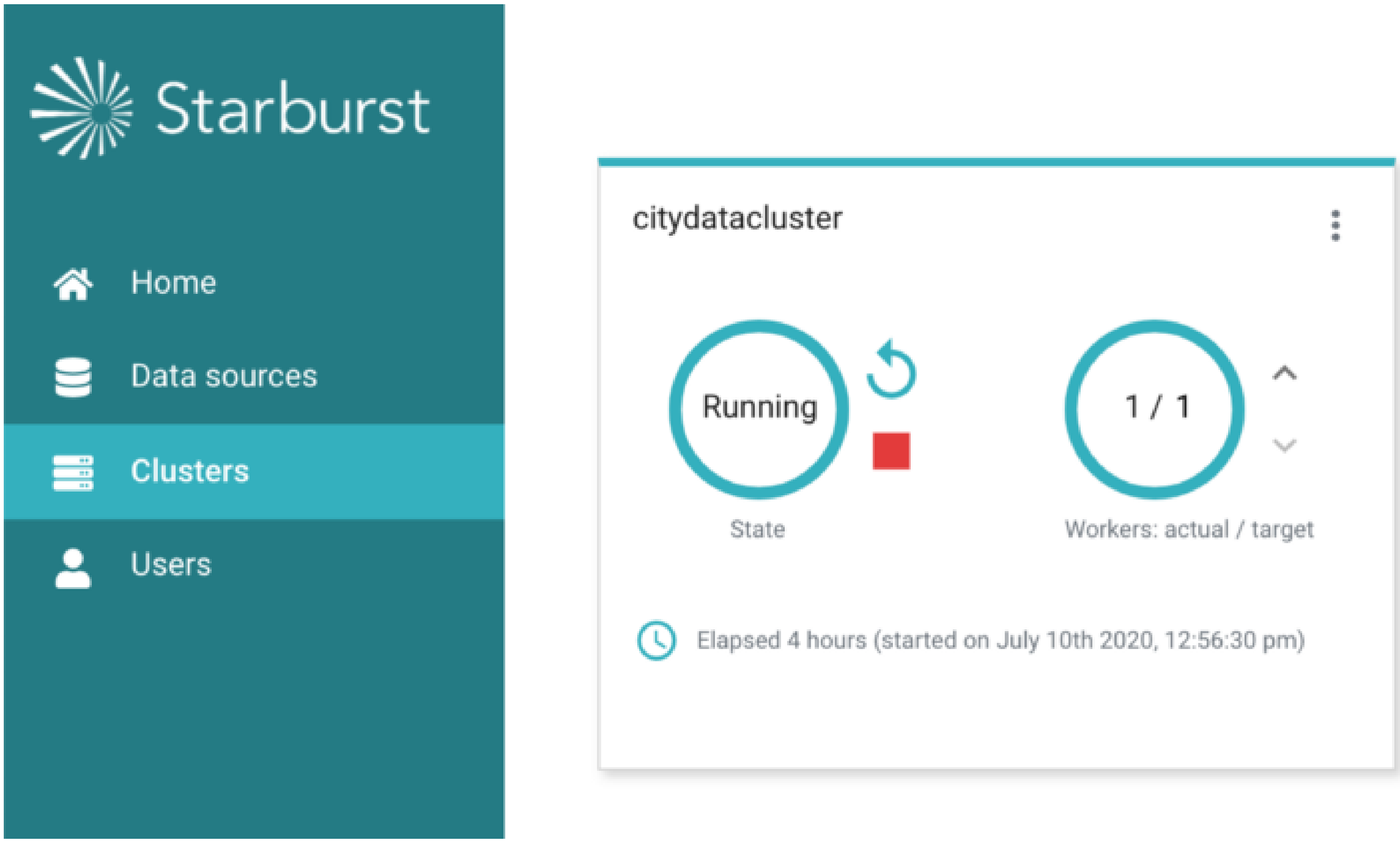
3. Snowflake is missing in the list. This is because we have manually configured the Snowflake connector as it is not automatically available in my version of Starburst Demo. Though it doesn't show up here, Snowflake tables can be still be queried (from Looker or another tool).

If you want to double-check this type of connector installation, you may do so by looking at the Co-ordinator POD directory. The '/etc/presto/catalog' directory should list all connected data sources. Look for 'snowflake.properties' here, like for any other source .properties file.



```
senthilkumar@Azure:~$ kubectl exec -it presto-coordinator-mc-7c40400c29a4430-d4ffc48fd-x6wmh sh
Defaulting container name to presto-coordinator.
Use 'kubectl describe pod/presto-coordinator-mc-7c40400c29a4430-d4ffc48fd-x6wmh -n default' to see all of the containers in this pod.
sh-4.2# ls /etc/presto/catalog/
hive.properties  mongopocdb.properties  snowflake.properties  subwaytransit.properties  tpch.properties  weather.properties
sh-4.2# █
```

It's time to Restart and Run the Presto Cluster!



Verify the cluster shows 'Running' state.

Connecting Looker to Starburst Presto and Multiple Databases

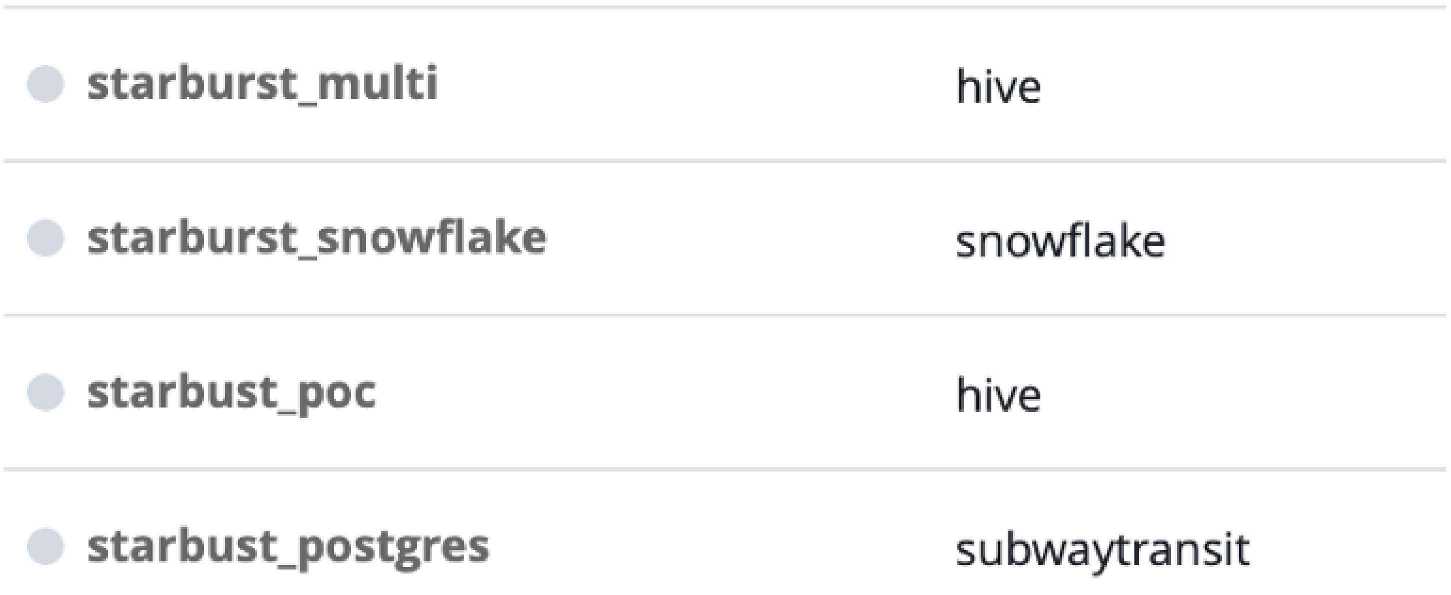
Refer to our previous blog post outlining details on connecting Looker to Presto.

How to Connect S3 Data to Looker using Starburst Presto

In this era of increasing adoption to Data-driven solutions, faster Data accessibility becomes more and more critical…

medium.com

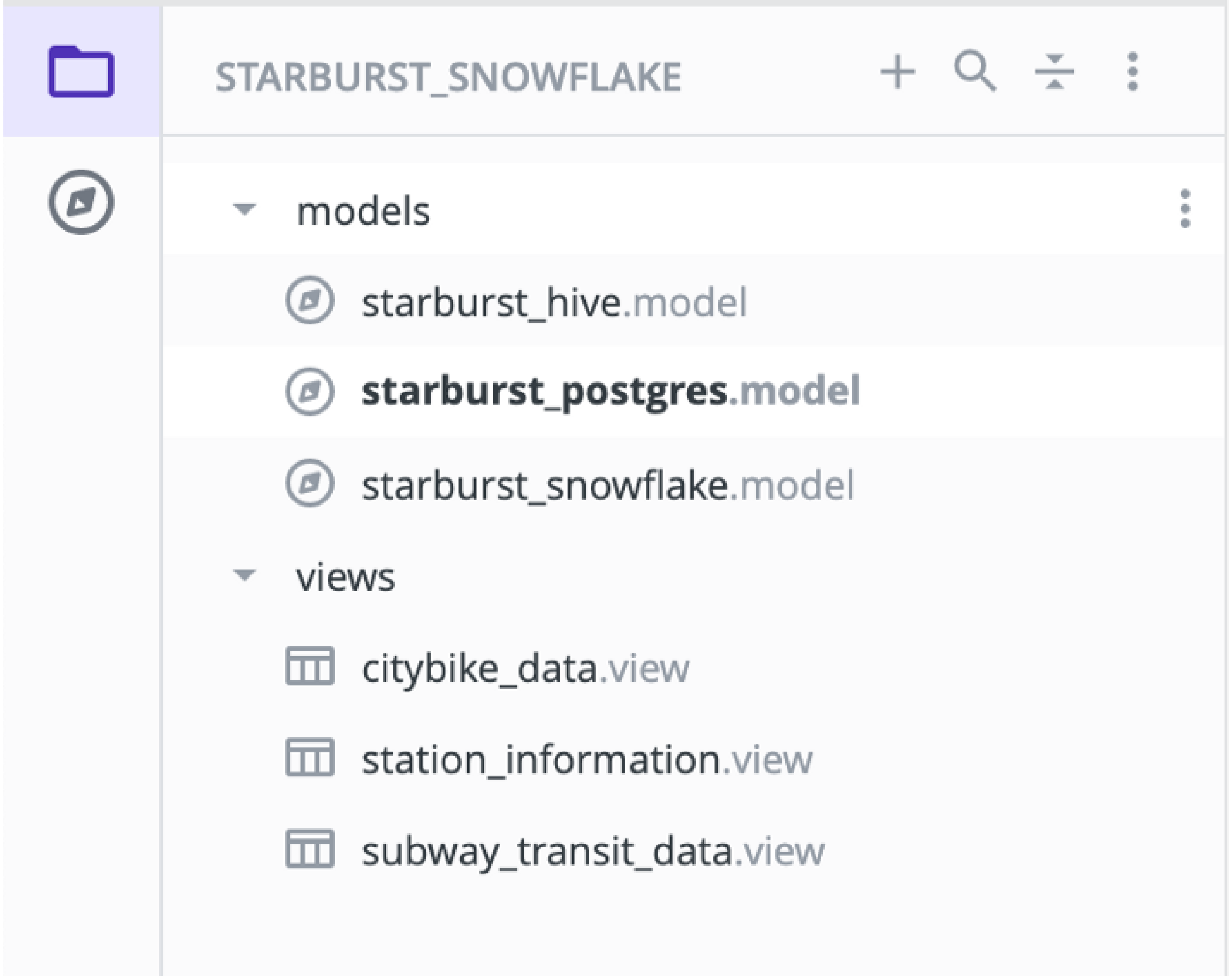
Setup new connections as stated below to connect to multiple data sources. Note, 'hive' is the mandatory data source name for the object sources using Internal MetaStore connections. More detail on this can be found here.



● starburst_multi	hive
● starburst_snowflake	snowflake
● starbust_poc	hive
● starbust_postgres	subwaytransit

Developing LookML

Under "Develop -> (your Project)" you should see the tables appear for writing LookML Models and Views (one model per database):


STARBURST_SNOWFLAKE

models

starburst_hive.model

starburst_postgres.model

starburst_snowflake.model

views

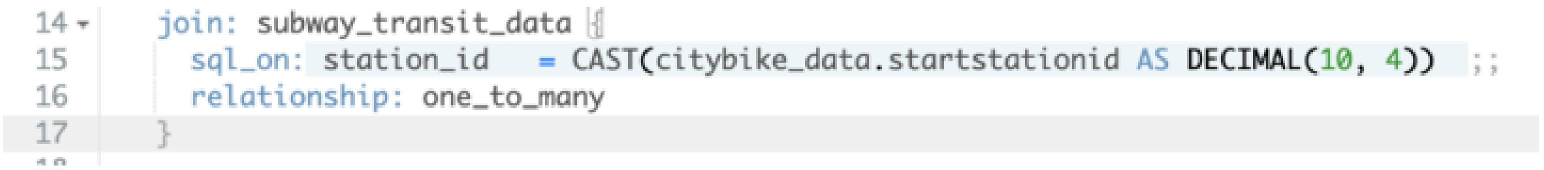
citybike_data.view

station_information.view

subway_transit_data.view

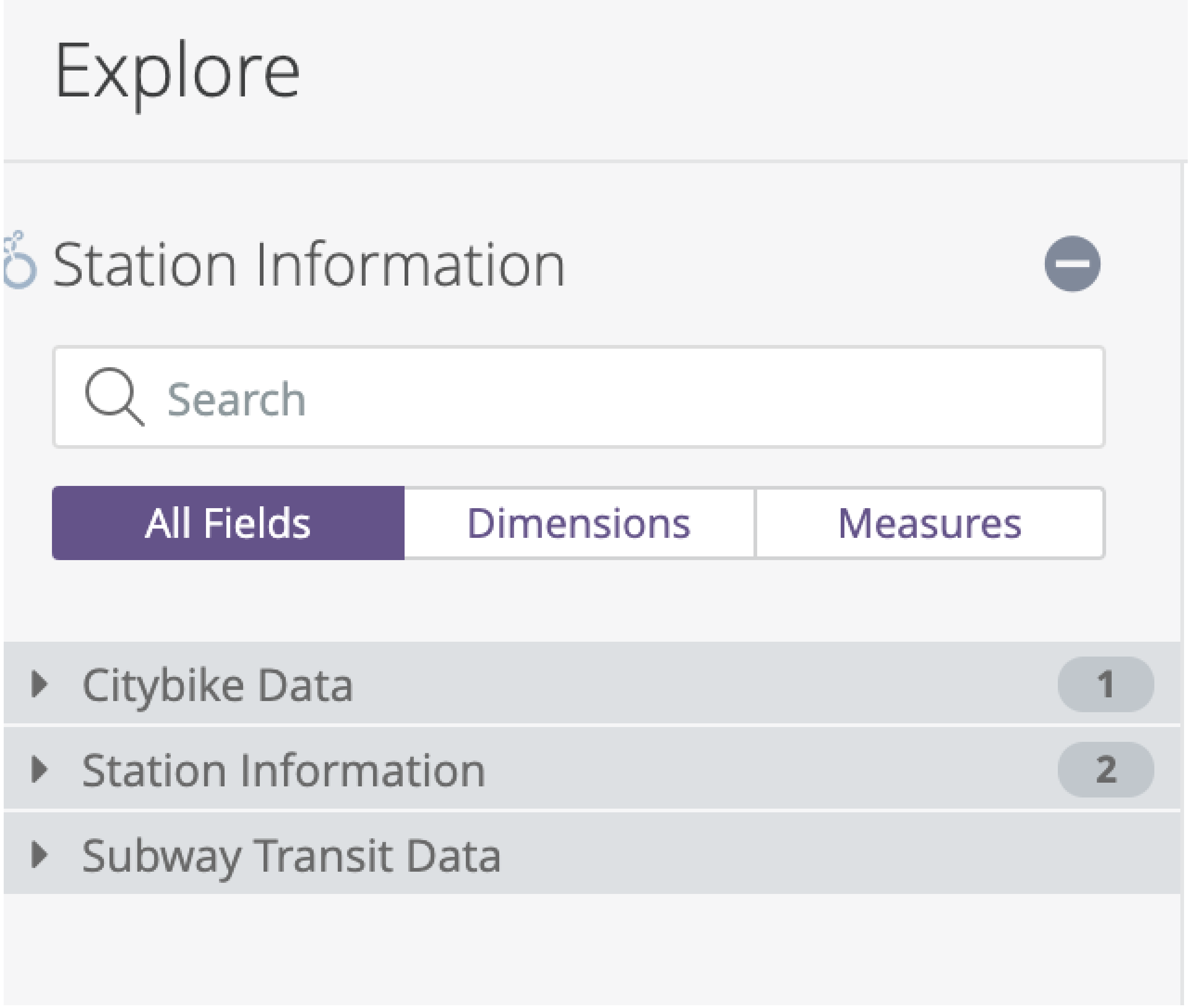
Joining tables across connections

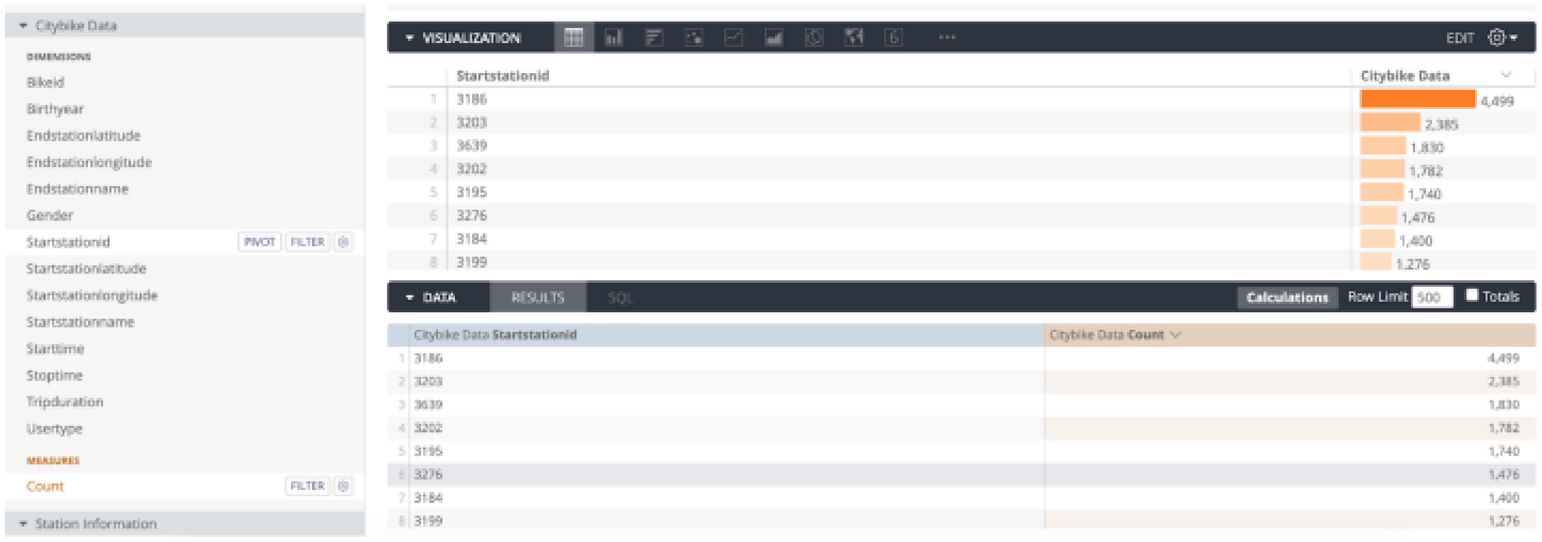
In the model, join the views with primary and foreign keys. If no such relationship exists, join by common field name.


```
  join: subway_transit_data {
    sql_on: station_id   = CAST(citybike_data.startstationid AS DECIMAL(10, 4)) ;;
    relationship: one_to_many
  }
```

Exploring the data in Looker

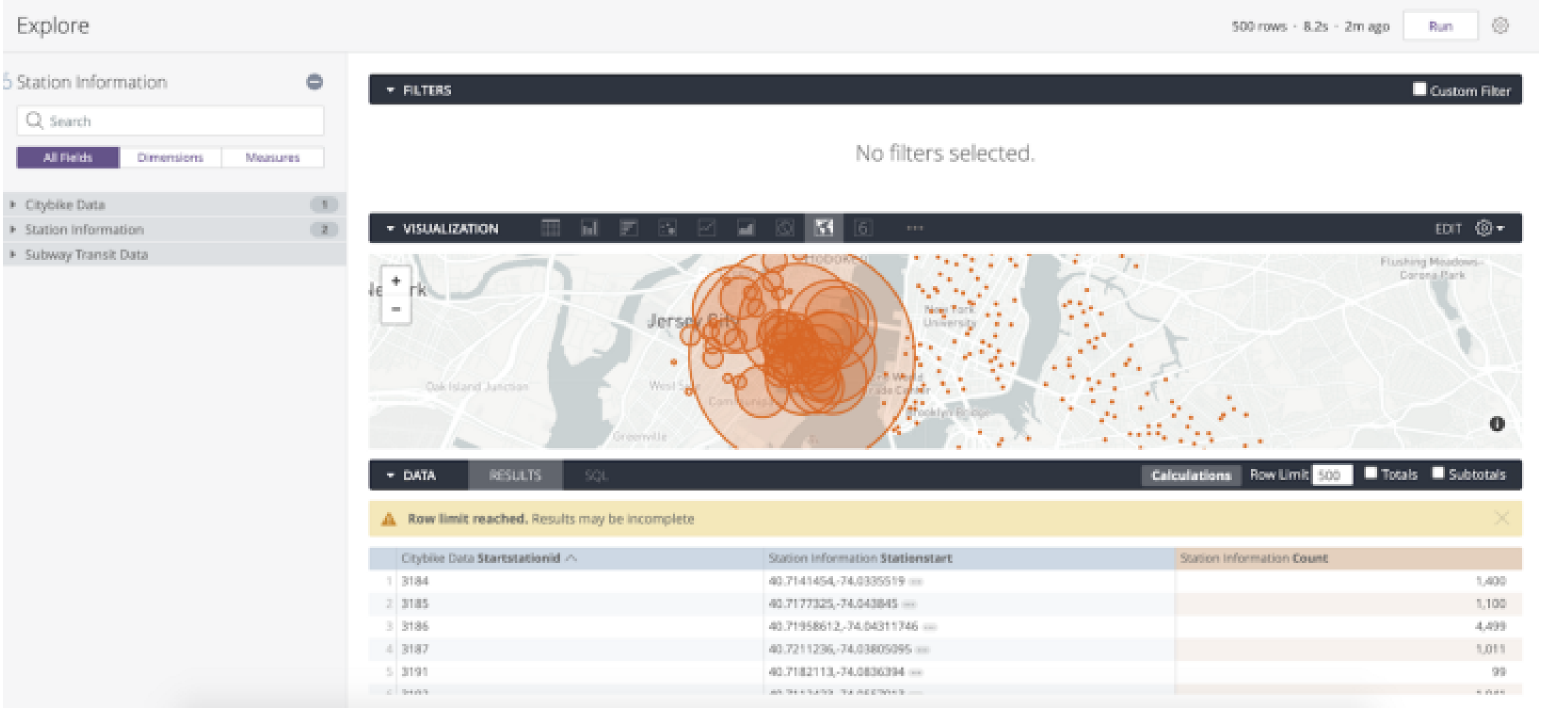
Explore the data before creating a dashboard by selecting multiple dimensions and measures with sorting and filters.





	Startstationid	Citybike Data
1	3186	4,499
2	3203	2,385
3	3639	1,830
4	3202	1,782
5	3195	1,740
6	3276	1,476
7	3184	1,400
8	3199	1,276

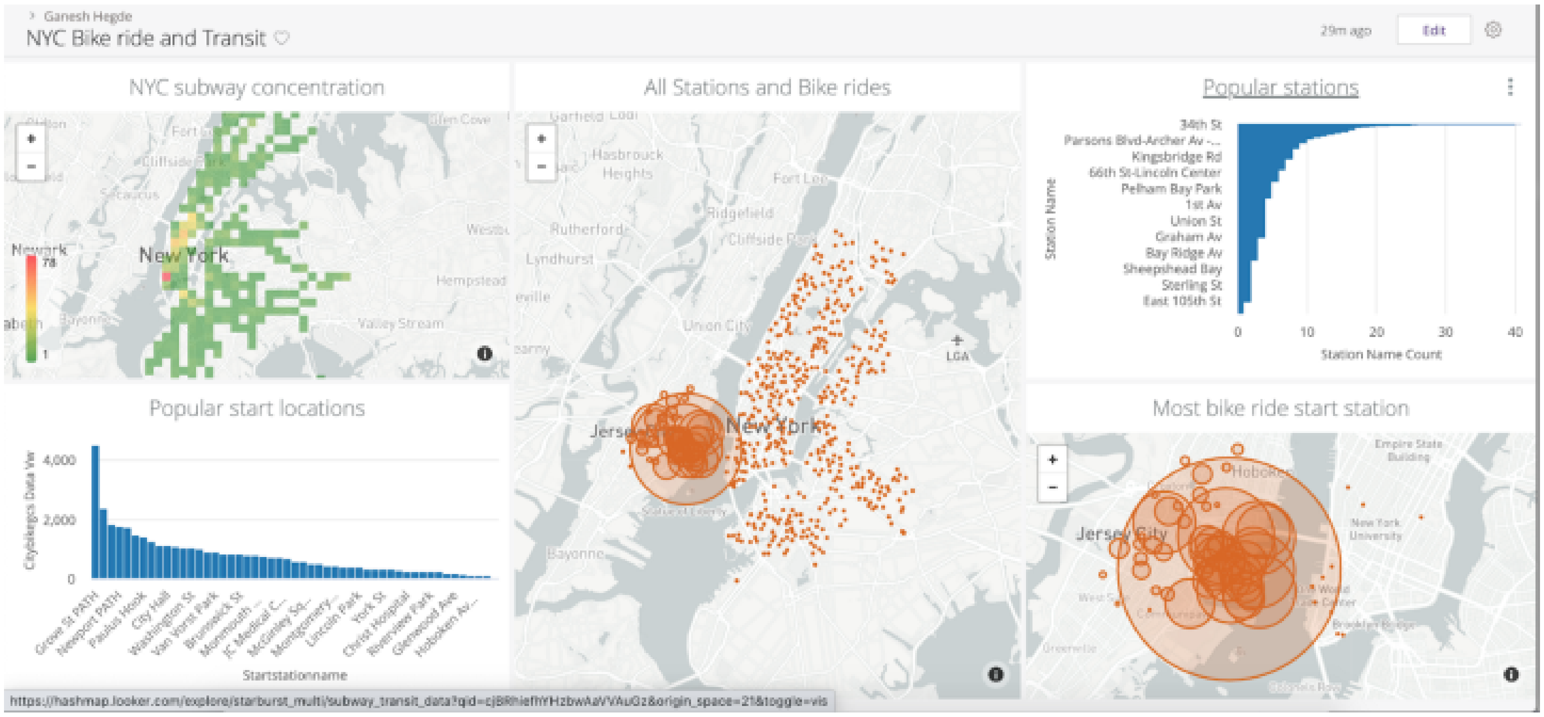
	Citybike Data **Startstationid**	Citybike Data **Count**
1	3186	4,499
2	3203	2,385
3	3639	1,830
4	3202	1,782
5	3195	1,740
6	3276	1,476
7	3184	1,400
8	3199	1,276



Row limit reached. Results may be incomplete

	Citybike Data Startstationid	Station Information Stationstart	Station Information Count
1	3184	40.7141454,-74.0335519	1,400
2	3185	40.7177325,-74.043845	1,100
3	3186	40.71958612,-74.04311746	4,499
4	3187	40.7211236,-74.03805095	1,011
5	3191	40.7182113,-74.0836394	99

This dashboard is built using city bike data from Hive (Google Cloud storage), station data from Snowflake, and transit data from Postgres with Query federation done using Starburst Presto.



Closing Thoughts

With the growing list of data connectors, Presto provides an opportunity to realize data virtualization with federated SQL queries across multiple data sources. Presto's Query Plan viewer is a huge plus for developers to identify SQL execution plans as well as optimize and improve query response times. Share your experiences with us if you have tried connecting other data sources and reporting tools to Starburst Presto!

Need Help with Your Cloud Initiatives?

If you are considering migrating or modernizing data and analytics products and applications in the cloud or if you would like help and guidance and a few best practices in delivering higher value outcomes in your existing cloud program, then please contact us.

Hashmap offers a range of enablement workshops and assessment services, cloud modernization and migration services, and consulting service packages as part of our service offerings.

Feel free to share on other channels and be sure and keep up with all new content from Hashmap ***here****. To listen in on a casual conversation about all things data engineering and the cloud, check out Hashmap's podcast* ***Hashmap on Tap*** *as well on Spotify, Apple, Google, and other popular streaming apps.*

Hashmap on Tap | Hashmap Podcast

A rotating cast of Hashmap hosts and special guests explore technologies from diverse perspectives while enjoying a drink of choice.

Senthil Anandarangan *and Ganesh Hedge* are *Cloud, Data, and UI Engineers with* ***Hashmap*** *providing Data, Cloud, IoT, and AI/ML solutions and consulting expertise across industries with a group of innovative technologists and domain experts accelerating high-value business outcomes for our customers.*

Starburst Presto Looker Data Virtualization Data