

# Data pipelines, views & data products [SQL-based]

Data Universe 2024

## Table of Contents

<b>Lab 1: Introduction and setup</b>	<b>1</b>
Part 1: Overview of labs	1
Part 2: Create a Starburst Galaxy account	2
Part 3: Housekeeping	3
<b>Lab 2: Connect to data sources</b>	<b>5</b>
Part 1: Create Amazon S3 catalog	5
Part 2: Create Snowflake catalog	9
<b>Lab 3: Build within your data lake</b>	<b>14</b>
Part 1: Use schema discovery	14
Part 2: Discover the Snowflake data source	20
Part 3: Build your reporting structure in S3	24
Part 4: Build the consume layer	29
Part 5: Secure access to your consume layer	35
<b>Lab 4: Create data products</b>	<b>42</b>
Part 1: Execute global search	42
Part 2: Create a data product	44
Part 3: Create tags (Bonus)	50

# Lab 1: Introduction and setup

## Learning objectives

- Describe the lab scenarios and goals.
- Setup a free Starburst Galaxy account.
- Understand how to continue using Starburst Galaxy after the end of the lab.

## Activities

1. Lab overview
2. Create a Starburst Galaxy account
3. Housekeeping items

## Part 1: Overview of labs

You are a data engineer at Nintendo. You were asked to gather some data about Pokemon Go and help the marketing team figure out which Pokemon spawns are most common in the San Francisco Bay Area. You need to help both teams by discovering, transforming, and cleaning the data from multiple sources.

### Step 1 - Purpose of labs

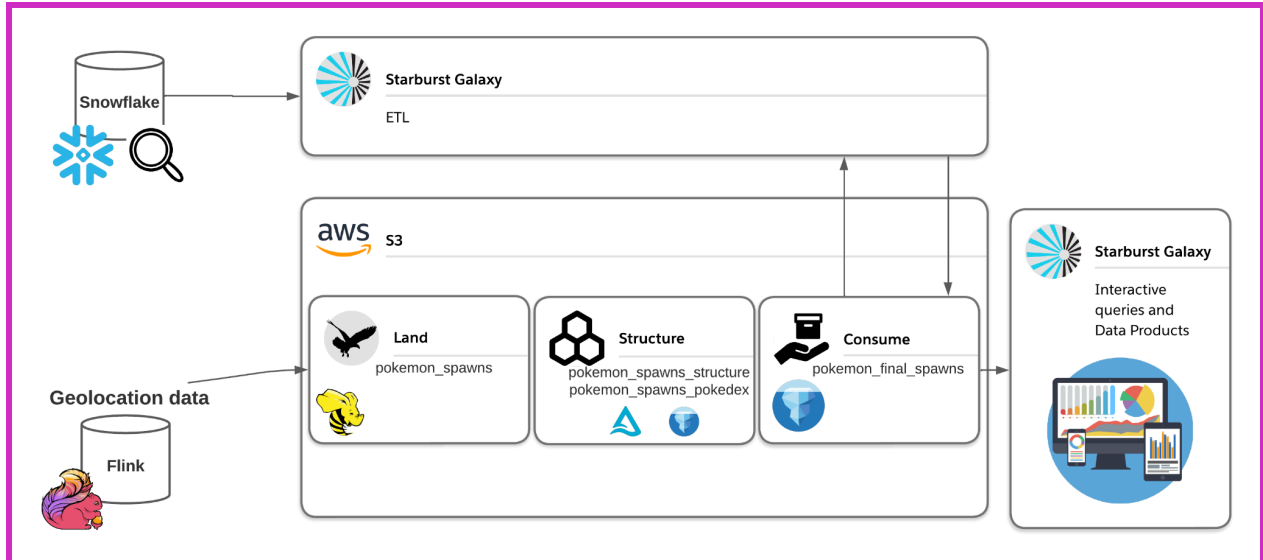
These labs use Pokemon Go data being ingested into S3, which contains the encounter information of the Pokemon including geolocation data of where the Pokemon spawned, and how long the Pokemon was at that location.

Importantly, you do not have any information about the Pokemon's abilities, that's all contained in the Pokedex in Snowflake. This has all the stats on your desired Pokemon including type\_1, type\_2, catch rate, and more.

### Step 2 - Description of activities

To make sense of data from multiple sources, you will create a reporting structure in your data lake. First, you will use schema discovery to understand the data in your data lake. You will then use Starburst Galaxy to read the data in the land layer, then clean and optimize that data into more optimal ORC files in the structure layer. In the last step, you will join the geolocation information from AWS S3 with the Pokedex lookup table in Snowflake into a single table that is cleaned and ready to be utilized by our teams. After completing the discovery, location, governance, and query stages, you will end the lab by creating data products, which package the dataset in a curated way for easy consumption.

You will also be introduced to [Gravity](#) and [Great Lakes connectivity](#) in Starburst Galaxy, which are two awesome features that make it easy to run data lake analytics. Both these features will be demonstrated throughout the lab guide.



### Step 3 - Data challenge

This data challenge involves two key missions:

- Create a final table output combining data from both structure tables.
- Create a data product answering two specific business questions from the marketing department.
  - a. What are the easiest and most popular Pokemon to catch in San Francisco by Type\_1?
  - b. Find the total number of Pokemon caught for each Type\_1 and Type\_2 pairing. Also, find the average catch rate.

*Note: Easiest is defined by having a high catch rate. A high catch rate is greater than or equal to 100. Also consider that in the structure layer, you filtered out data that did not exist in the San Francisco Bay Area.*

## Part 2: Create a Starburst Galaxy account

**If you have already registered for Starburst Galaxy, you may skip this FIRST step.**

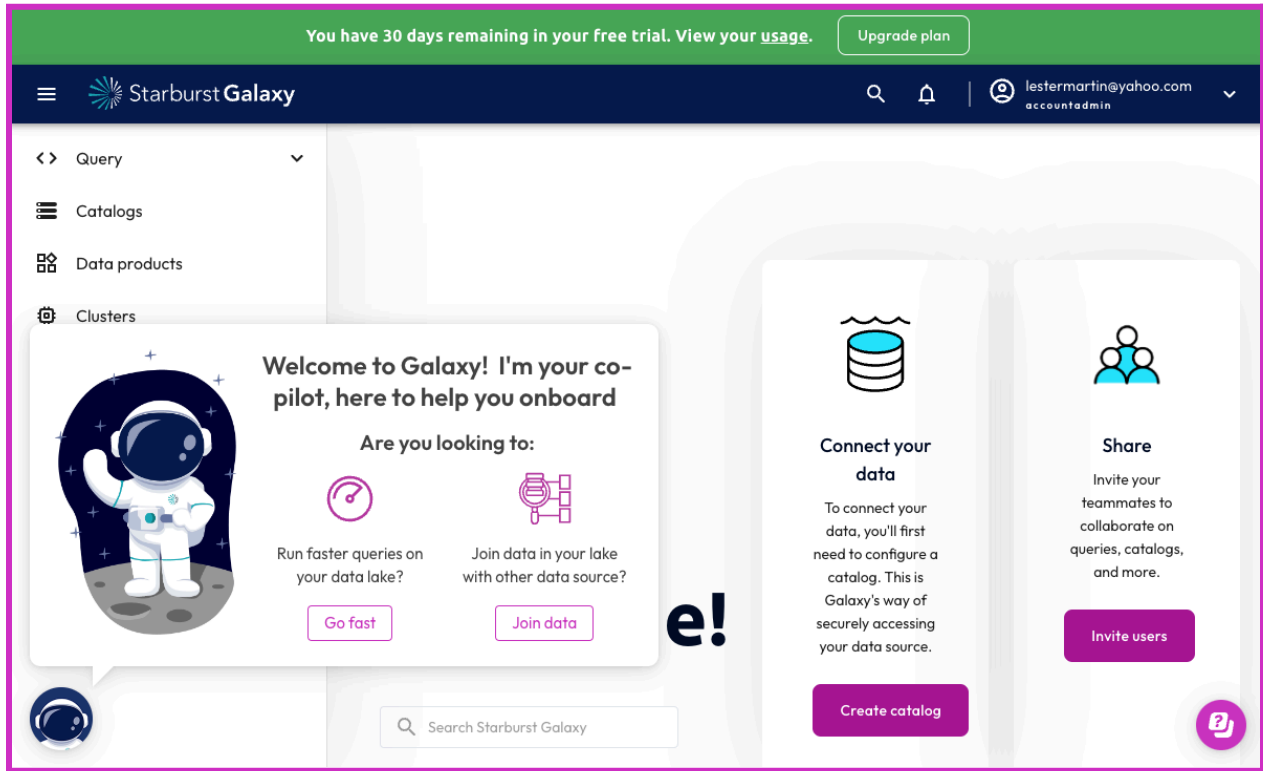
To sign up for Starburst Galaxy, follow the instructions on the free registration page at <https://www.starburst.io/platform/starburst-galaxy/start/>.

**Note:** You will receive an “invitation” email. Please check your spam or junk folder if it does not immediately arrive in your inbox.

After you have entered your confirmation code, set a password, and selected your new domain name, you will be presented with a series of questions about your desired usage for Starburst Galaxy. Complete these with whatever you choose to share.

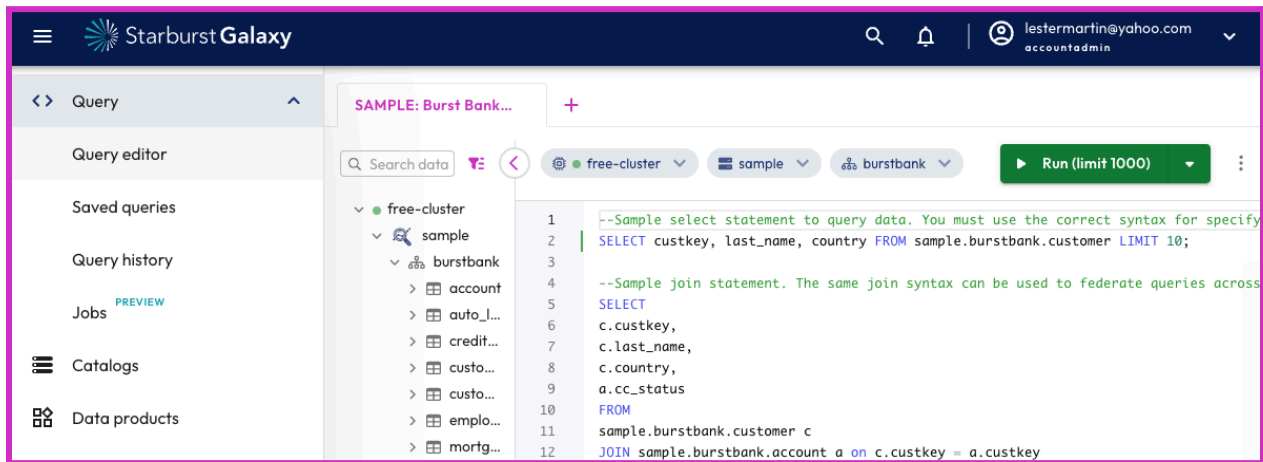
Eventually, you will likely be presented with a page similar to the following screenshot.

## Workshop: Data pipelines, views & data products [SQL-based] (Data Universe 2024)



Click on the astronaut helmet icon in the lower-left to silence the pop-up coming from it.

At this point, you should see something similar to this to indicate you are fully configured.



## Part 3: Housekeeping

As part of the Galaxy Lunch and Lab, the credentials for Amazon S3 and Snowflake will be available for up to 1 week. It is critical to understand that time, **YOU WILL BE UNABLE TO RUN ANY QUERIES AGAINST THE TWO CATALOGS CREATED IN THIS LAB.**

If you want to continue exploring Starburst Galaxy, here are some other free projects and helpful links you can utilize with your Starburst Galaxy account:

- [Federate multiple data sources tutorial](#)
- [Starburst Academy](#)
  - [Starburst Galaxy courses](#)
  - [Data foundations courses](#)
  - [Learn SQL courses](#)
  - [Starburst foundations](#)
- [Starburst Galaxy documentation](#)
- [Near Real-Time Ingestion tutorial](#)

## Lab 2: Connect to data sources

### Learning objectives

- Describe the process for creating catalogs that connect AWS S3 and Snowflake.
- Demonstrate how to connect a catalog to a cluster in Starburst Galaxy.

### Activities

1. Create Amazon S3 catalog
2. Create Snowflake catalog

## Part 1: Create Amazon S3 catalog

### Objective

You're going to begin by setting up an AWS S3 catalog in Starburst Galaxy and connect the Pokemon spawns geolocation data.

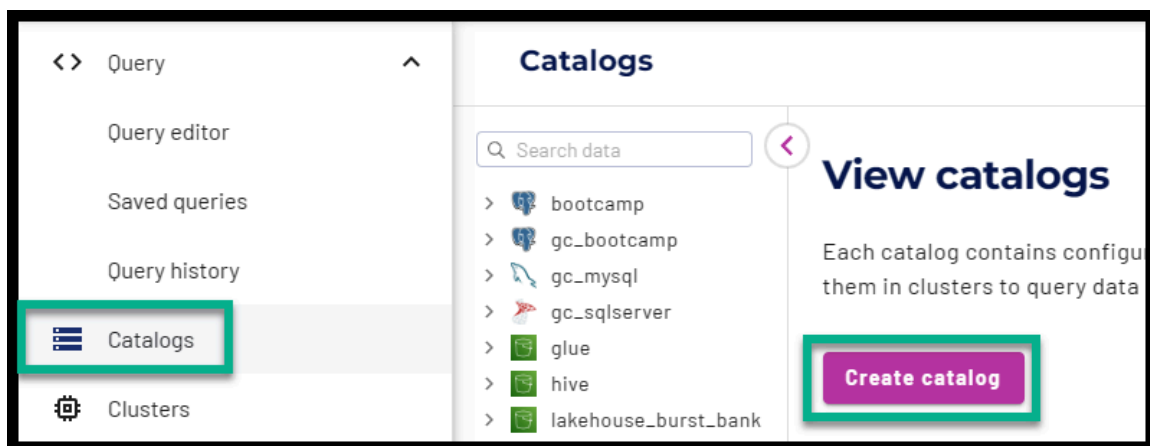
***If you configured a data lake catalog in one of the other workshops, you may skip these steps in this Part and move on to Part #2 of this Lab.***

### Step 1 - Sign in and verify your role

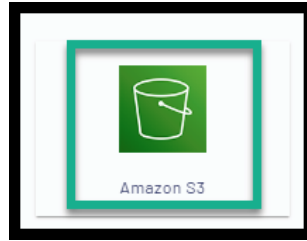
Sign in to Starburst Galaxy. Use the account credentials you previously created. In the upper right corner of the screen, confirm that your role is set as `accountadmin`.

### Step 2 - Create Amazon S3 catalog

Click **Catalogs** in the menu on the left and then click the **Create catalog** button.



Click the **Amazon S3** tile.



Use the information below to configure your catalog.

**Catalog name:** du2024

**Description:** AWS catalog for Data Universe 2024 workshops

**Name and description**

Provide a unique name to identify the catalog in your SQL queries in the query editor and other client tools. The namespace for a table is typically <catalog\_name>.<schema\_name>.<table\_name>

Catalog name \*  
du2024

Must start with a letter and only use lowercase letters (a-z), numbers (0-9), and underscores

Description  
AWS catalog for Data Universe 2024 workshops

**Authentication with:** select the radio button **AWS access key**

**AWS access key for S3:** AKIAYUW62MUV2GXVDL5R

**AWS secret key for S3:** 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

**Note:** These AWS credentials will only be operational through the weekend following the webinar. You will not be able to utilize this catalog beyond that point and should remove it from your Galaxy configuration.

**Authentication to S3**

Choose the authentication mechanism to connect to S3.

Authentication with \*

☐ Cross account IAM role ☒ AWS access key

AWS access key for S3 \*  
AKIAYUW62MUV2GXVDL5R

AWS secret key for S3 \*  
.....

**Metastore type:** select the radio button **Starburst Galaxy**

**Default S3 bucket name:** starburst101-handsonlab

**Default directory name:** du-fname-lname-postalcode (ex: du-lester-martin-90120)

**Allow creating external tables:** enable the slider

**Allow writing to external tables:** enable the slider

### Metastore configuration

Configure access to the metastore to provide metadata and mapping information about the objects stored in Amazon S3.

Metastore type \*

Starburst Galaxy

Default S3 bucket name \*

starburst101-handsonlab

Default directory name \*

du-lester-martin-90120

☒

Allow creating external tables

☒

Allow writing to external tables

**Default table format:** ensure the radio button is selected to **Iceberg**

### Default table format

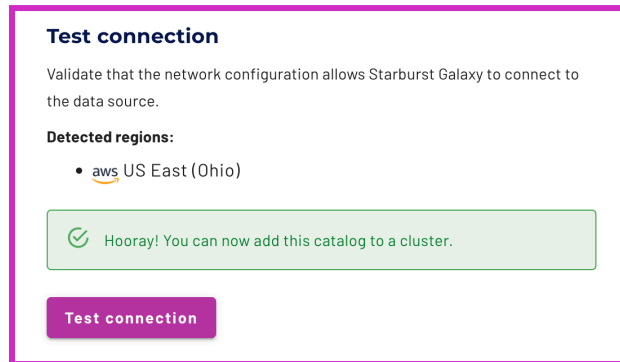
Select the default table format used for creating new tables. The catalog will be able to read from any type. [Check out our docs](#) to learn more.

Default table format \*

☒ Iceberg ☐ Hive ☐ Delta Lake



Validate the connection by hitting **Test connection**. Your catalog should return the same message indicating that you can now add the catalog. Confirm you see the **Hooray! You can now add this catalog to a cluster** message.



**Test connection**

Validate that the network configuration allows Starburst Galaxy to connect to the data source.

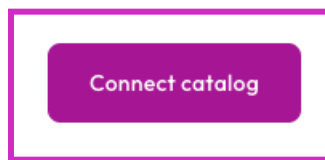
**Detected regions:**

- aws US East (Ohio)

✓ Hooray! You can now add this catalog to a cluster.

Test connection

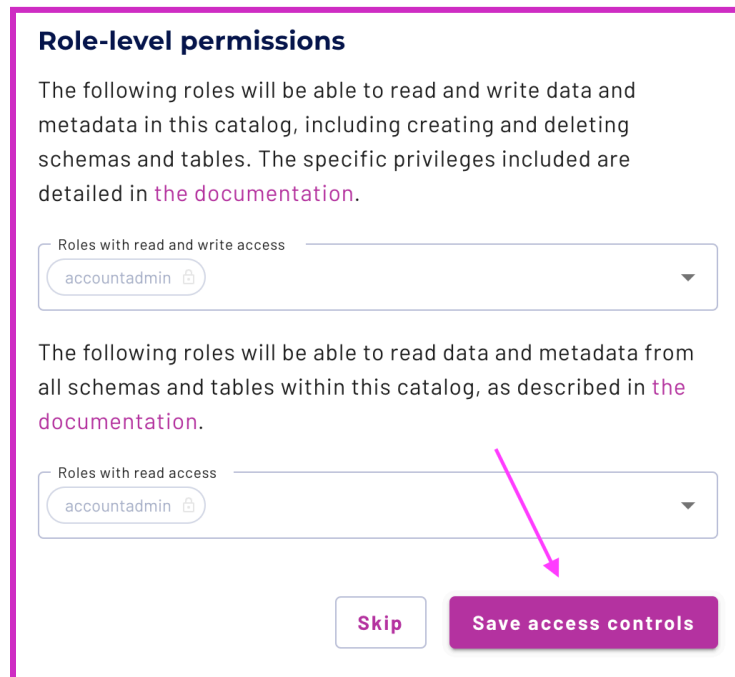
Select **Connect catalog**. This will save the credentials for your Amazon S3 catalog.



Connect catalog

### Step 3 - Set permissions

Next, accept the default permissions for your catalog by selecting the button **Save access controls**.



**Role-level permissions**

The following roles will be able to read and write data and metadata in this catalog, including creating and deleting schemas and tables. The specific privileges included are detailed in [the documentation](#).

Roles with read and write access

accountadmin

The following roles will be able to read data and metadata from all schemas and tables within this catalog, as described in [the documentation](#).

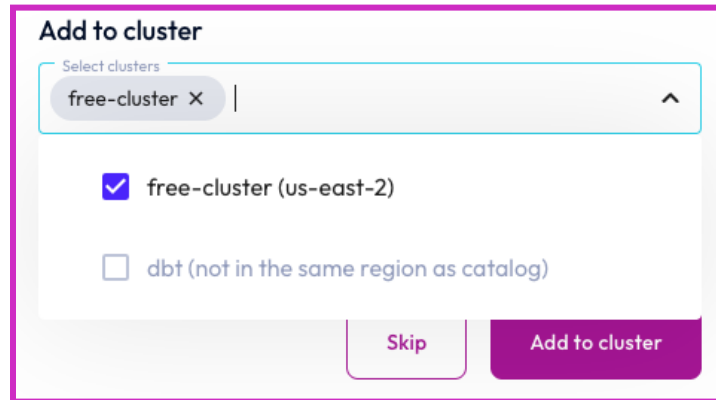
Roles with read access

accountadmin

Skip Save access controls

## Step 4 - Add to cluster

Select `free-cluster` in the **Select clusters** pulldown and then click on **Add to cluster**.



**Add to cluster**

Select clusters

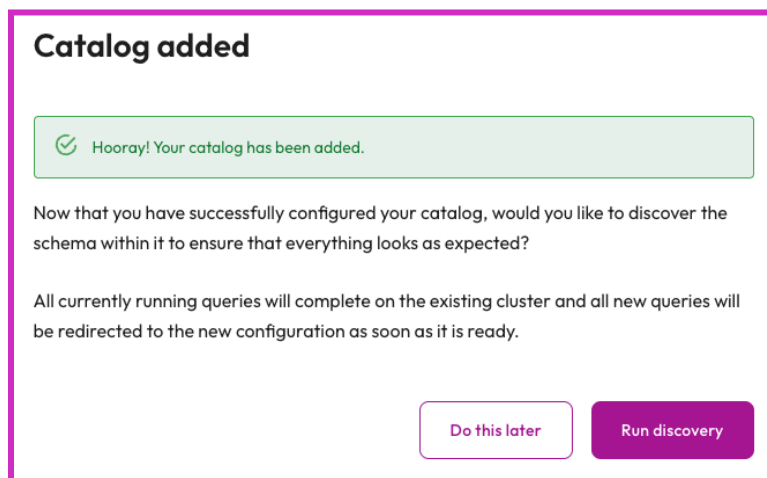
free-cluster x

☒ free-cluster (us-east-2)

☐ dbt (not in the same region as catalog)

Skip Add to cluster

Click **Do this later** in the **Catalog added** pop-up.



**Catalog added**

✓ Hooray! Your catalog has been added.

Now that you have successfully configured your catalog, would you like to discover the schema within it to ensure that everything looks as expected?

All currently running queries will complete on the existing cluster and all new queries will be redirected to the new configuration as soon as it is ready.

Do this later Run discovery

## Part 2: Create Snowflake catalog

### Objective

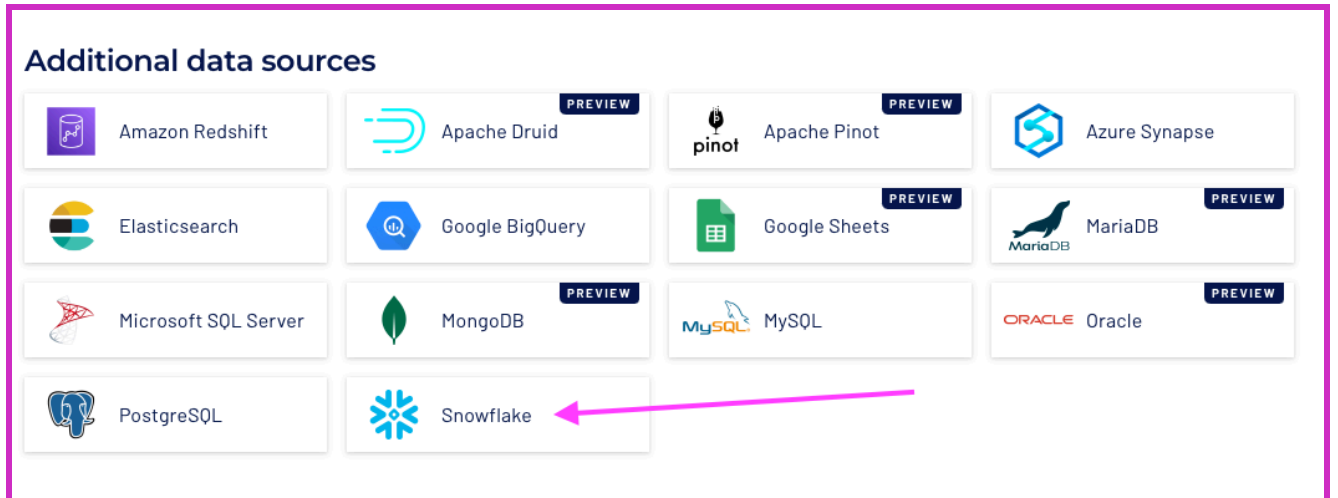
Now it's time to create a Snowflake catalog alongside your AWS S3 catalog. Later, this will allow us to federate across the two data sources.

***If you configured a Snowflake catalog in one of the other workshops, you may skip the REMAINDER of the steps in this Part and move on to Lab #3.***

### Step 1 - Create Snowflake catalog

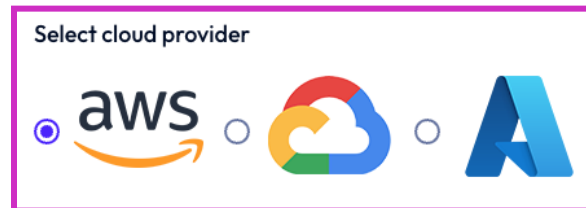
From the catalog page, select the **Create catalog** button to create the second catalog.

Choose **Snowflake**.



Using the list below as a guide, configure your catalog to query objects in Snowflake, specifically the Pokedex information. Provide the necessary credentials to authenticate the connection.

**Cloud Provider:** AWS



**Catalog Name:** pokemon\_lkp

**Description:** Lookup table containing pokemon stats

**Name and description**

Provide a unique name to identify the catalog in your SQL queries in the query editor and other client tools. The namespace for a table is typically <catalog\_name>.<schema\_name>.<table\_name>

Catalog name \*  
pokemon\_lkp

Must start with a letter and only use lowercase letters (a-z), numbers (0-9), and underscores

Description  
Lookup table containing pokemon stats

**Snowflake account identifier:** TB03263.us-east-2.aws

**Username:** DU\_USER

**Password:** Bl@nkEt&P!gg9

**Database name:** POKEMON

**Warehouse name:** SB\_101

**Snowflake role:** STARBURST\_101

### Snowflake connection

Connection type \*

☒ Connect directly

Snowflake account identifier \*  
TB03263.us-east-2.aws

Commonly formatted as <orgname>-<account\_name>. If unsuccessful, try other formats recommended by [Snowflake](#).

Username \*  
DU\_USER

Password \*  
Bl@nkEt&P!gg9

Database name \*  
POKEMON

Warehouse name  
SB\_101

Snowflake role  
STARBURST\_101

☐ Enable parallel mode

**Test** the connection to ensure that the setup is correct.

✓ Hooray! You can now add this catalog to a cluster.

Test connection

Select **Connect catalog** to save the credentials for your Snowflake catalog.

A rectangular button with rounded corners, colored purple, with the text "Connect catalog" in white. It is enclosed in a thin purple rectangular border.

## Step 2 - Save access controls

Next, set the default permissions for your catalog by selecting the **Save access controls** button.

### Role-level permissions

The following roles will be able to read and write data and metadata in this catalog, including creating and deleting schemas and tables. The specific privileges included are detailed in [the documentation](#).

Roles with read and write access

accountadmin

The following roles will be able to read data and metadata from all schemas and tables within this catalog, as described in [the documentation](#).

Roles with read access

accountadmin

Skip

Save access controls

## Step 3 - Add to cluster

Select `free-cluster` in the **Select clusters** pulldown and then click on **Add to cluster**.

### Add to cluster

Select clusters

free-cluster X

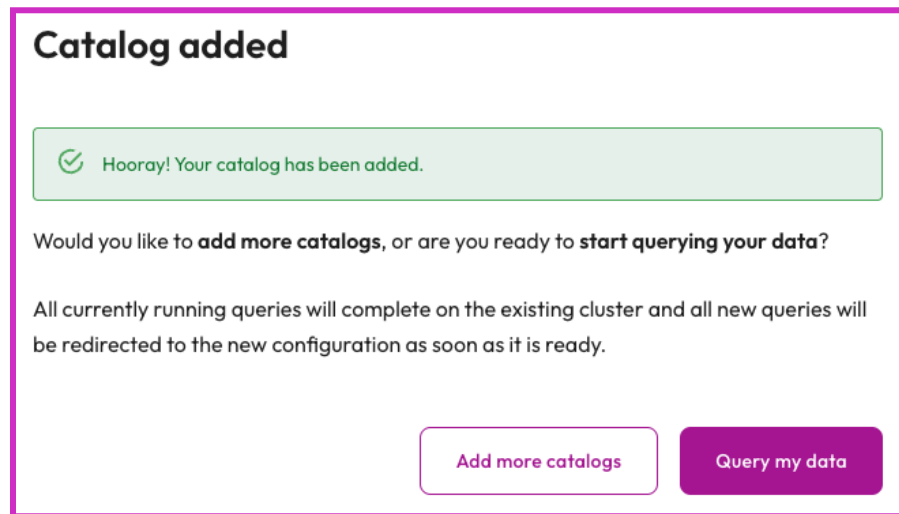
☒ free-cluster (us-east-2)

☐ dbt (not in the same region as catalog)

Skip

Add to cluster

Click **Query my data** in the **Catalog added** pop-up.



## Lab 3: Build within your data lake

### Learning objectives

- Demonstrate the process needed to run schema discovery to analyze a root object in an object storage location.
- Show how to use open table formats.
- Demonstrate the steps needed to build a reporting structure in your data lake, and secure your team's access.

### Prerequisites

- [Lab 1: Introduction and setup](#)
- [Lab 2: Connect to data sources](#)

### Activities

1. Use schema discovery
2. Discover the lookup data
3. Build the structure layer
4. Build the consume layer
5. Secure access to your consume layer

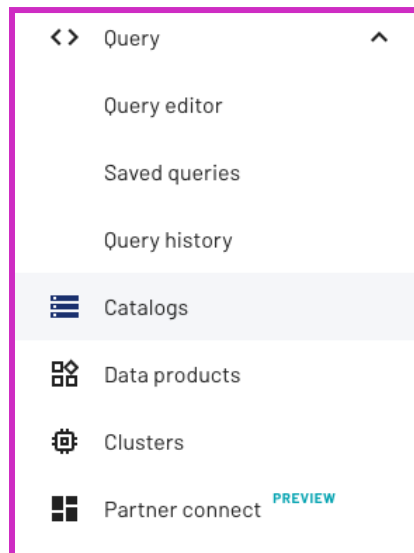
## Part 1: Use schema discovery

### Objective

You're going to begin by utilizing schema discovery to create your schema and table. Schemas control the structure of the data inside them. Luckily for us, Starburst Gravity will take care of the discovery work.

### Step 1 - Navigate to the catalogs page

In the left hand navigation pane, select **Catalogs**.





Select the **du2024** catalog to navigate within it.

### Catalogs

## View catalogs


Each catalog contains configuration for Starburst Galaxy to access a data source. Configure catalogs and use them in clusters to query data sources in Starburst Galaxy.

Create catalog 8 catalogs

Name ↑	Kind	Description	Cloud	Region	Tags
du2024		AWS catalog for Data Universe 202...		US East (Ohio)	No tags assigned.

## Step 2 - Run Schema discovery

As part of Gravity, you can see all the metrics, schemas, query history, audit log, privileges, and more! Click on the **Schema discovery** tab.

 **du2024**

DESCRIPTION

AWS catalog for Data Universe 2024 workshops

Schemas 1


Schema discovery


Metrics

Query history

Audit log

Privileges

 Refresh

 Auto tag

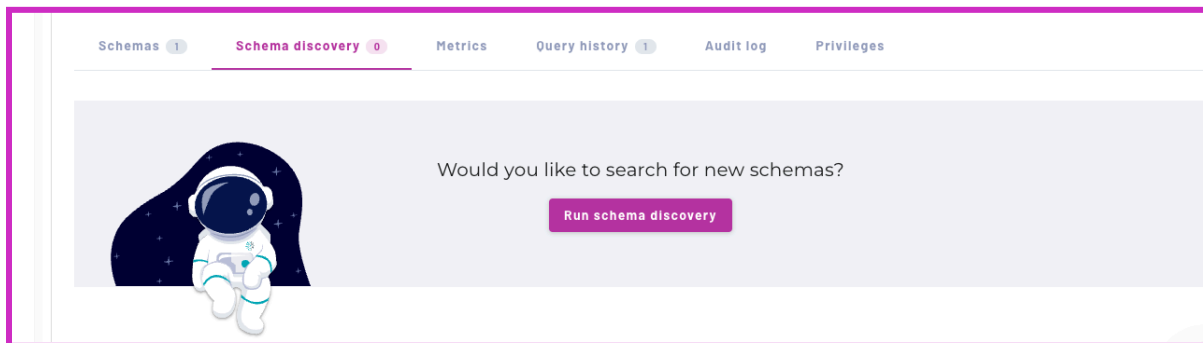
Schema ↑	Tags
information_schema	No tags assigned.



The Schema discovery pane lets you examine the metadata of the specified object storage location. Schema discovery is for catalogs in object storage data sources only.

Use schema discovery to identify and register tables or views that are newly added to a known schema location. For example, a logging process might drop a new log file every hour, rolling over from the previous hour's log file. The purpose of schema discovery is to find the newly added files to make sure Starburst Galaxy knows how to query them.

Select **Run schema discovery**.



Add the following information:

**Catalog location URL:**

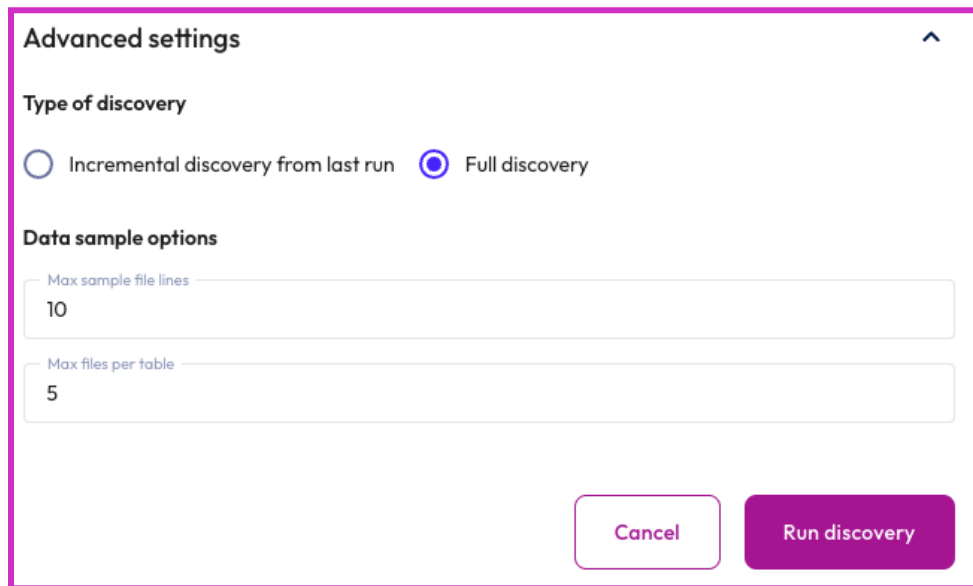
`s3://starburst101-handsonlab-nyc-uber-rides/pokemon/`

**Add location privilege:** Leave it checked

**Default schema:** `discovered_schema`

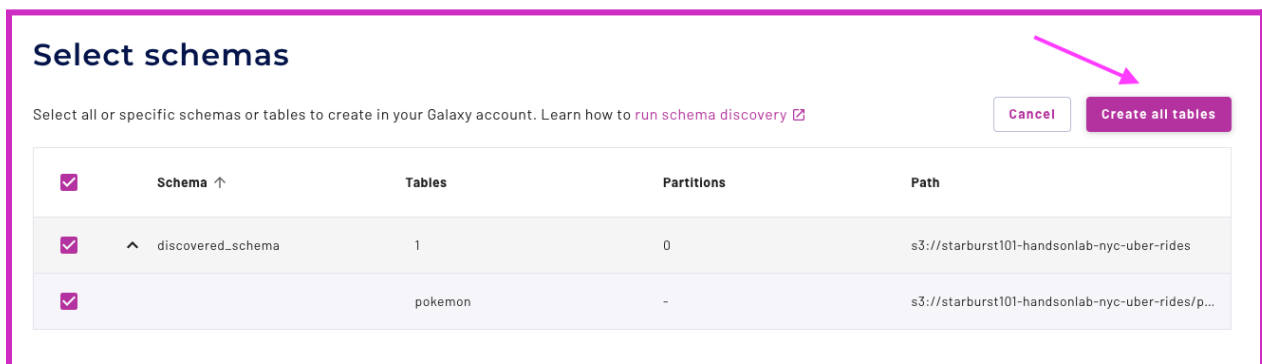
A screenshot of the Starburst Galaxy configuration form for schema discovery. The form is enclosed in a light blue border. At the top, there is a text input field labeled 'Catalog location URI \*' containing the value 's3://starburst101-handsonlab-nyc-uber-rides/pokemon/'. Below this, a message states: 'This role does not have privileges to access this location. Would you like us to add the missing location privilege for this catalog?'. Underneath the message is a checkbox labeled 'Add location privilege', which is checked. At the bottom, there is another text input field labeled 'Default schema \*' containing the value 'discovered\_schema'.

Toggle the **Advanced settings** arrow down and select **Full discovery** for the **Type of discovery**.



The 'Advanced settings' dialog box is shown with a blue border. It has a title bar with an upward arrow. The 'Type of discovery' section has two radio buttons: 'Incremental discovery from last run' (unselected) and 'Full discovery' (selected). The 'Data sample options' section has two input fields: 'Max sample file lines' with the value '10' and 'Max files per table' with the value '5'. At the bottom right are two buttons: 'Cancel' and 'Run discovery'.

Click **Run discovery**. Starburst will start scanning for you. Then, it will return code to create your desired schema and table. Toggle open `discovered_schema` and then check the top checkbox to select the two below it. Select **Create all tables**.



The 'Select schemas' dialog box is shown with a blue border. It has a title bar. Below the title bar is a text prompt: 'Select all or specific schemas or tables to create in your Galaxy account. Learn how to [run schema discovery](#)'. There are two buttons: 'Cancel' and 'Create all tables'. Below this is a table with four columns: 'Schema', 'Tables', 'Partitions', and 'Path'. The table has three rows. The first row is for 'discovered\_schema' and has a checkbox checked. The second row is for 'pokemon' and has a checkbox checked. The third row is for 'pokemon' and has a checkbox checked. A red arrow points to the 'Create all tables' button.

Schema	Tables	Partitions	Path
discovered_schema	1	0	s3://starburst101-handsonlab-nyc-uber-rides
pokemon	-	-	s3://starburst101-handsonlab-nyc-uber-rides/p...

Schema discovery has done the heavy lifting so you don't have to spend time trying to investigate what columns exist, or bother your AWS administrator to give you details about the file.

### Log events

Events for: `s3://starburst101-handsonlab-nyc-uber-rides/pokemon/` Close


**Summary**

✓ 2 query executions completed successfully.

Status	Timestamp ↓	Query text	Message
✓	Jul 11, 2023, 10:48:11 AM	<code>CREATE TABLE "aws_pokemon"."discovered_schema"."pokemon" (</code>	Created table: [pokemon], with location: [s3://starburst101...
✓	Jul 11, 2023, 10:48:08 AM	<code>CREATE SCHEMA "aws_pokemon"."discovered_schema" WITH (location = 's3://starburst101-handsonlab-nyc-uber-</code>	Created schema: [discovered_schema], with location: [s3://...

Click on the Query text to view the full queries. Your first query created the desired schema, and the second query created a Hive table. Hit **Close**.

### Step 3 - Set up the Query editor

Navigate to the **Query editor**. If you already have queries, add a new tab  using the fuschia plus sign. Change the location drop-downs in the top left-hand corner to match the cluster and catalog previously created.

**Cluster:** free-cluster

**Catalog:** du2024



Run the following query to validate the table you created using schema discovery.

```
SELECT * from du2024.discovered_schema.pokemon LIMIT 100;
```

Your data sample should look something like the following:

s2_id	s2_token	num	name	lat	lng
-918579452294725...	8085808cc6d	13	Weedle	37.7935915752623	-122.408720633183
-9185794529389707...	8085808b51d	16	Pidgey	37.7947455405929	-122.406419649564
-918579452938970...	8085808b271	41	Zubat	37.794999066064	-122.404384122075
-9185794082713108...	808580f3587	16	Pidgey	37.7956444102582	-122.407127649888
-918579407627065...	808580f4b1d	60	Poliwag	37.7955915257874	-122.406331149188
-9182922218470900...	808fb4e54b3	50	Diglett	37.3011286952679	-122.048453380601

Run a command to view the CREATE TABLE statement:

```
SHOW CREATE TABLE discovered_schema.pokemon;
```

You should return the same code as run with Schema discovery. Notice that the columns are already utilizing proper data types. Also notice that the table format is HIVE. You will update this as you build your reporting structure in your data lake using Great Lakes connectivity.

## Step 4 - Create your schema

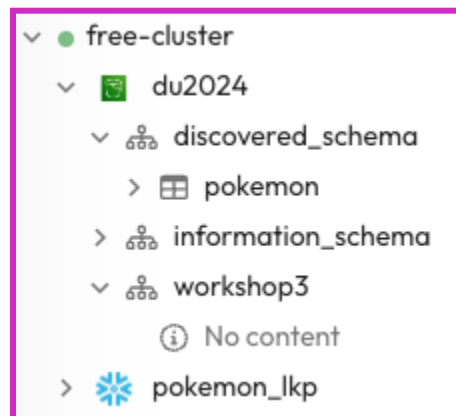
You have already created one schema, named `discovered_schema`. Now, create another schema so that you can separate out the table that was created from an external location and your own work.

Create a new schema by running the following command.

```
CREATE SCHEMA du2024.workshop3;
```

## Step 5 - Verify schema in catalog

In the catalog explorer, close and expand the `aws_pokemon` catalog. In addition to the schema and table created earlier, `discovered_schema.pokemon`, verify your new `workshop3` schema is present. There will be other schemas listed.



## Step 6 - Select schema after creation

Choose `workshop3` in the **Select schema** pulldown just above the editor.



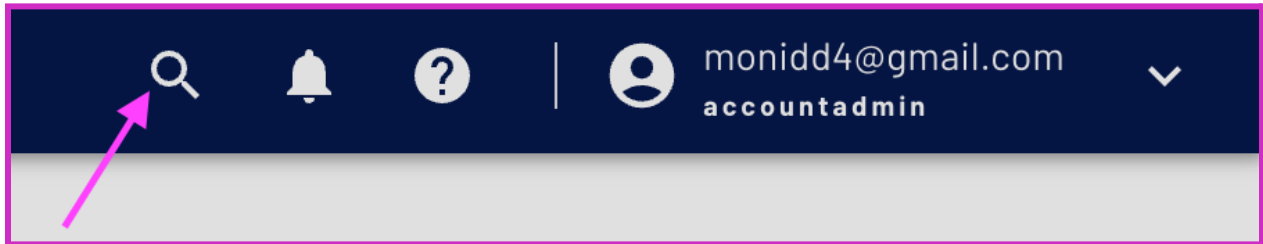
## Part 2: Discover the Snowflake data source

### Objective

Learn about the Pokedex lookup table stored in Snowflake using global search. Global search lets users find datasets quickly and intuitively. It is a powerful tool that helps keep better track of your data. Use global search to discover the Pokedex data in Snowflake and validate that connection.

### Step 1 - Navigate Starburst UI

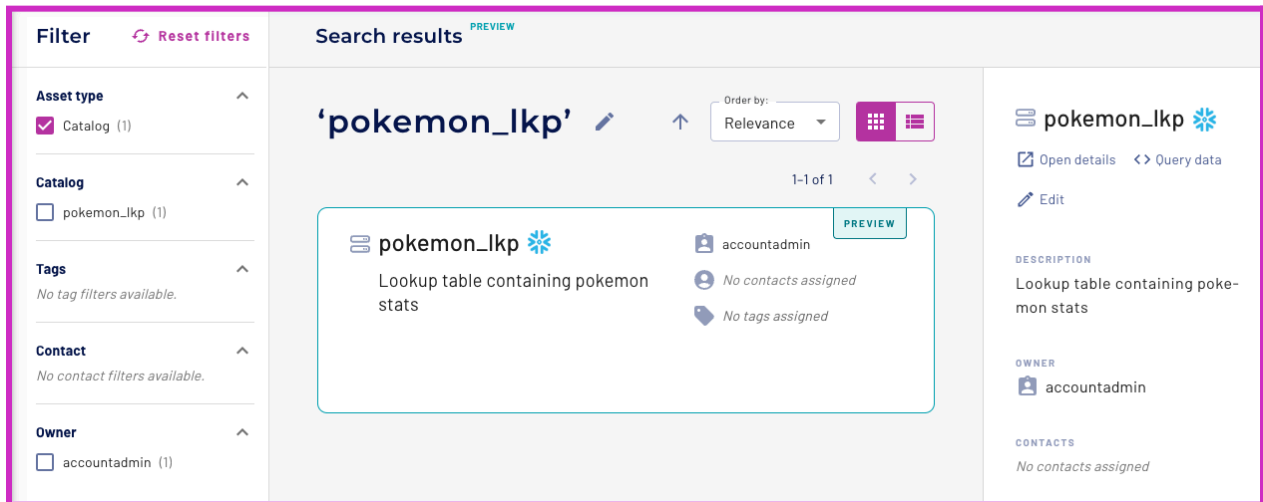
To use global search, select the magnifying glass icon in the upper-right corner.



### Step 2 - Execute global search

Enter `pokemon_lkp` and select **View all results** at the bottom of the pop-up window.

Starburst Galaxy lets you filter and organize your search results. This is handy when you have a bigger environment with more results.



Click on the **Open details** in the top right corner.

The screenshot shows the 'Search results' page in Starburst Galaxy. On the left, there's a 'Filter' sidebar with sections for 'Asset type' (Catalog (1)), 'Catalog' (pokemon\_lkp (1)), 'Tags' (No tag filters available), 'Contact' (No contact filters available), and 'Owner' (accountadmin (1)). The main area displays the search results for 'pokemon\_lkp', showing it's a 'Lookup table containing pokemon stats' owned by 'accountadmin'. A 'PREVIEW' button is visible. On the right, a sidebar for 'pokemon\_lkp' includes an 'Open details' button (highlighted with a red box), 'Query data', 'Edit', 'DESCRIPTION' (Lookup table containing pokemon stats), 'OWNER' (accountadmin), and 'CONTACTS' (No contacts assigned).

### Step 3 - Explore your Snowflake data source

Notice that Starburst Galaxy automatically places you within the catalog page. You can see the catalog has a place to add additional details, as well as shared information regarding **Schemas, Metrics, Query history, Audit log, and Privileges**. Click within the `pokemon_lookup` schema to learn more.

The screenshot shows the 'Catalogs / pokemon\_lkp' page. At the top, there's a header with the 'pokemon\_lkp' logo and a 'Show details' link. Below this, a summary bar shows 'DESCRIPTION: Lookup table containing pokemon stats', 'TAGS: 0', 'CONTACTS: 0', and 'OWNER: accountadmin'. A navigation bar includes 'Schemas (2)', 'Metrics', 'Query history (0)', 'Audit log', and 'Privileges'. The 'Schemas' tab is active, showing a list of schemas. A 'Refresh' button and '2 schemas' count are at the top of the list. The list has columns for 'Schema', 'Tags', and an expand/collapse icon. The schemas listed are 'information\_schema' and 'pokemon\_lookup'. A red arrow points to the 'pokemon\_lookup' schema row.

Schema	Tags	
information_schema	No tags assigned.	+
pokemon_lookup	No tags assigned.	+

## Workshop: Data pipelines, views & data products [SQL-based] (Data Universe 2024)

Within the schema, you can see more information available to you. Stay tuned throughout the lab as you will come back and utilize these features of Starburst Gravity. For now, click within the `pokedex` table.

Catalogs / `pokemon_lkp` / `pokemon_lookup`

### pokemon\_lookup

Promote to data product

Show details

DESCRIPTION: No description provided.

TAGS: 0 | LINKS: 0 | CONTACTS: 0 | OWNER: accountadmin

Tables 1 | Views | Metrics | Definition | Query history 0 | Audit log | Privileges

Refresh

1 table

Search tables

Table ↑	Tags
<code>pokedex</code>	No tags assigned.

You can see a preview of the **Columns** in the table. You also see all the **Metrics**, the **Definition**, the **Data preview**, the **Query history**, the **Audit log**, and the **Privileges**.

Catalogs PREVIEW

Catalogs / `pokedex` / `pokemon_lookup` / `pokedex`

### pokedex

Show details

DESCRIPTION: No description provided.

TAGS: 0 | CONTACTS: 0 | OWNER: accountadmin

Columns 17 | Metrics | Definition | Data preview | Query history 0 | Audit log | Privileges

Refresh

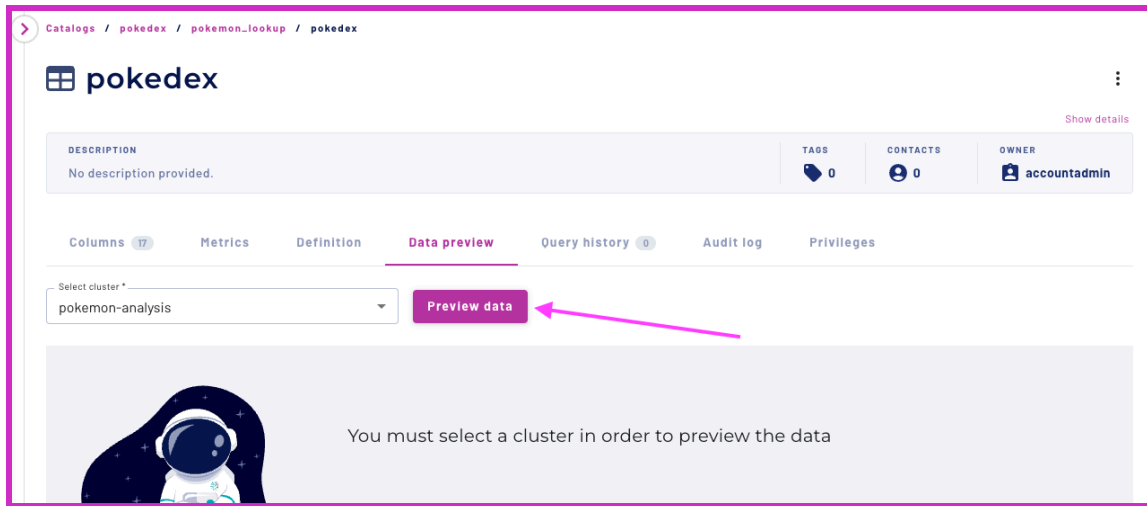
17 columns

Search columns

Column ↑	Type	Nullable	Default	Tags	Description
abilities	varchar(16777216)	yes	null	No tags assigned.	+ No description provided.
att	varchar(16777216)	yes	null	No tags assigned.	+ No description provided.
catch_rate	varchar(16777216)	yes	null	No tags assigned.	+ No description provided.
def	varchar(16777216)	yes	null	No tags assigned.	+ No description provided.

## Workshop: Data pipelines, views & data products [SQL-based] (Data Universe 2024)

Navigate to the **Data preview** tab. Make sure the `free-cluster` cluster is selected, then hit **Preview data**.



The data is available to be previewed without ever having to run a query. This is handy if you have any data consumers who want access to the data but do not want to use the **Query editor**.

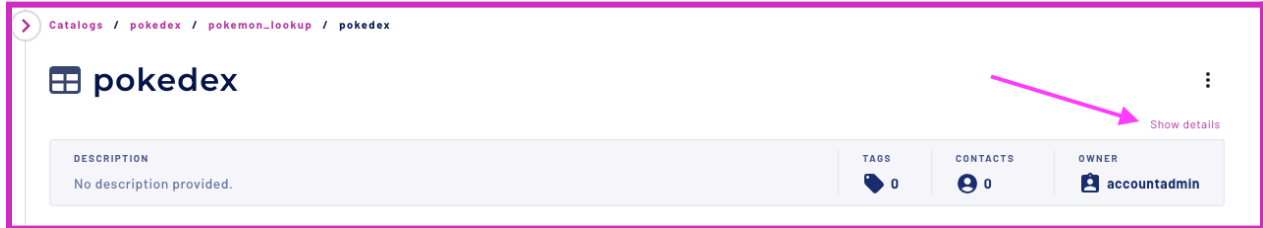
The screenshot shows the 'pokedex' interface with the 'Data preview' tab selected. The 'Preview data' button has been clicked, and a table of Pokemon data is displayed.

name	number	type_1	type_2	hp	att	def
Bulbasaur	1	Grass	Poison	45	49	49
Ivysaur	2	Grass	Poison	60	62	63
Venusaur	3	Grass	Poison	80	82	83
Mega Venusaur	3	Grass	Poison	80	100	123
Charmander	4	Fire	NULL	39	52	43



## Step 4 - Enter in data definitions

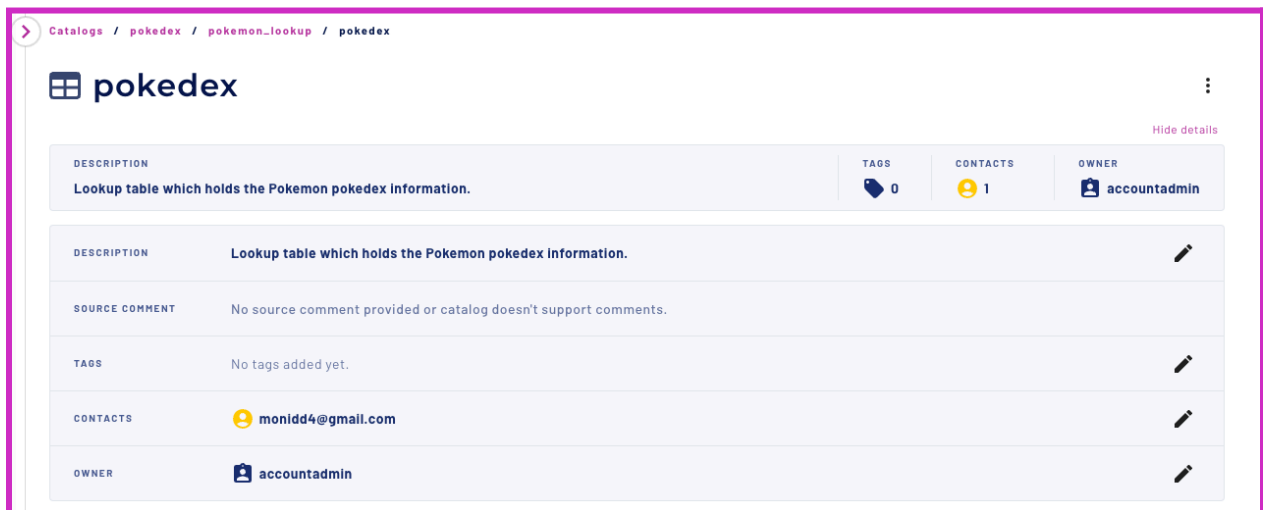
Add the following information to your table so that anyone else who looks at the table has some basic understanding of the data. Hit **Show details**.



Click on the pencil icon on the right of the 2 entries below to enter the information provided below:

**Description:** Lookup table which holds the Pokemon pokedex information.

**Contacts:** yourself



Now those reading the dataset for the first time will have more context.

## Part 3: Build your reporting structure in S3

### Objective

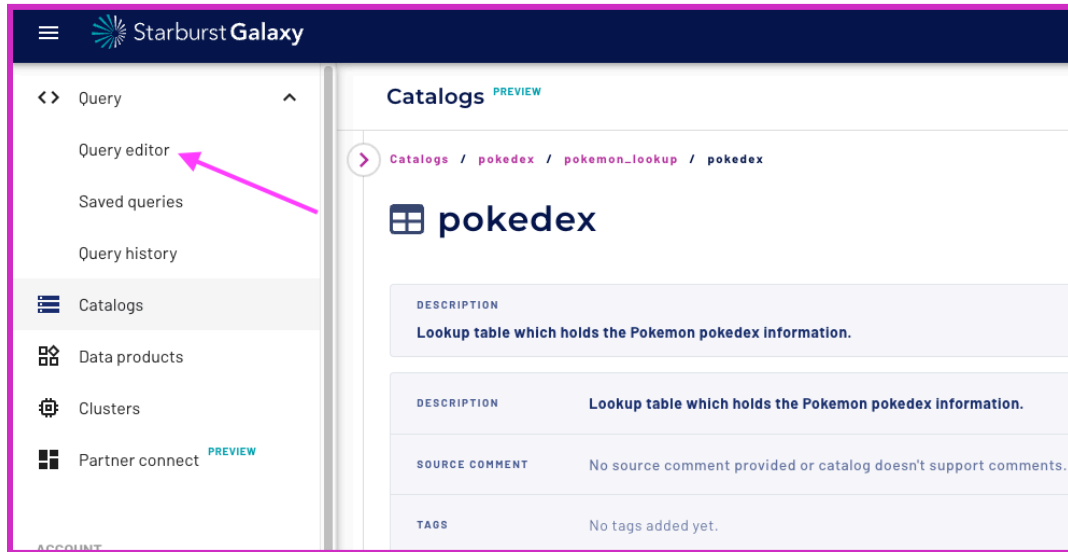
Now it's time to use both your data sources and create a reporting structure in S3.

- **Land layer** - This is the raw data you were ingesting that's landing in S3. Thanks to schema discovery, this layer is already created.
- **Structure layer** - This is the enriched, cleaned, and cleansed data.
- **Consume layer** - This is the data that is ready to be queried and utilized by consumers.

Starburst is special because it allows you to build this reporting structure not just with data that already exists in your data lake, but also with data that exists in other data sources in your orbit - like our Snowflake pokedex data.

## Step 1 - Set up the query editor

Using the left-hand pane, navigate back to the **Query editor**.



Make sure you are in the tab you previously created, or use a new tab that has the cluster, catalog, and schema selected like below.



## Step 2 - Create the pokemon\_spawns structure table

Recall that the default table format you set for this catalog was Iceberg. However, you are still able to create a table with the Hive table format due to [Great Lakes connectivity](#) in Starburst Galaxy. Great Lakes connectivity abstracts the details of using different table formats and file types when using certain write access statements for object storage systems.

Part of your cleansing in the structure layer is to update the table format from Hive to Iceberg and convert the text file to an ORC file. The table you just created with schema discovery is not yet fully optimized and transformed.

To fix this, you must build a structure layer by creating a new table and casting the necessary values into more accurate data types. To help with this, you will also create the table using the ORC file format. This improves performance when using the Iceberg table format.

Run the following command to create the structure layer table.

```
CREATE TABLE pokemon_spawns_structure
(
    number INTEGER,
    name VARCHAR,
    latitude DOUBLE,
    longitude DOUBLE,
    encounter_seconds BIGINT
)
WITH (
    format = 'ORC',
    type = 'ICEBERG'
);
```

### Step 3 - Insert data into the pokemon\_spawns structure table

Once the table is created, you need to insert the data into it.

**Note:** After doing this manually the first time, you can automate the process in the future using an orchestration tool like Airflow, Prefect, or Dagster.

```
INSERT INTO
    pokemon_spawns_structure
SELECT
    CAST(num AS INTEGER) AS number,
    name,
    round(lat, 2) AS latitude,
    round{lng, 2) AS longitude,
    CASE
        WHEN encounter_ms = -1 THEN encounter_ms
        ELSE (encounter_ms / 1000)
    END AS encounter_seconds
FROM du2024.discovered_schema.pokemon
WHERE lat >= 37.62 and lat <= 37.86
    AND lng >= -122.51 and lng <= -122.12;
```

You should insert 95197 rows.

### Step 4 - Validate the pokemon\_spawns\_structure table

It is considered best practice to validate that the new table has been created. To do this, run a simple select statement.

```
SELECT * FROM pokemon_spawns_structure LIMIT 10;
```

num	name	latitude	longitude	encounter_ms
13	Weedle	37.79	-122.41	1469520187
16	Pidgey	37.79	-122.41	1469520297
41	Zubat	37.79	-122.4	1469520709
16	Pidgey	37.8	-122.41	-1
60	Poliwag	37.8	-122.41	1469520741
46	Paras	37.76	-122.43	1469520422

## Step 5 - Preview the lookup table

The data in the lookup table also needs to be cleaned and optimized. It's best practice to clean up the data in the structure layer before creating the consume layer. First, run a quick join to show the federation capabilities available within Starburst Galaxy and join the newly created S3 `pokemon_spawns_structure` table with the unoptimized Snowflake lookup table `pokemon_lkp`.

Run the following command:

```
SELECT
    S.number,
    S.name,
    S.latitude,
    S.longitude,
    P.type_1,
    P.type_2,
    P.catch_rate,
    p.generation
FROM pokemon_lkp.pokemon_lookup.pokedex p
JOIN pokemon_spawns_structure s
    ON CAST(p.number AS INTEGER) = s.number;
```

number	name	latitude	longitude	type_1	type_2	catch_rate	generation
13	Weedle	37.79	-122.41	Bug	Poison	255	1.0
16	Pidgey	37.79	-122.41	Normal	Flying	255	1.0
41	Zubat	37.79	-122.4	Poison	Flying	255	1.0
16	Pidgey	37.8	-122.41	Normal	Flying	255	1.0
60	Poliwag	37.8	-122.41	Water	NULL	255	1.0
46	Paras	37.76	-122.43	Bug	Grass	190	1.0
41	Zubat	37.76	-122.42	Poison	Flying	255	1.0
25	Pikachu	37.76	-122.42	Electric	NULL	190	1.0

You can see that data from two different data sources (catalogs) are returned with one federated query. This is beneficial when performing interactive analytics; specifically for data consumers who could not otherwise get this information without a data engineer.

## Step 6 - Create a pokedex structure table

Now, create a table which will store an optimized version of this table. Create this table as a delta lake table. Why would you create a Delta Lake table? Because Starburst Galaxy's [Great Lakes connectivity](#) gives you the ability to query multiple table formats at once. You will test this out later in the lab. Run the following command:

```
CREATE TABLE pokemon_pokedex_structure
(
  name VARCHAR,
  number INTEGER,
  type_1 VARCHAR,
  type_2 VARCHAR,
  catch_rate INTEGER,
  final_evolution DOUBLE,
  generation DOUBLE,
  abilities DOUBLE
)
WITH (
  type = 'DELTA'
);
```

## Step 7 - Insert data into the pokemon\_pokedex structure table

Once the table is created, you need to insert the data into it. Run the following command.

```
INSERT INTO pokemon_pokedex_structure
SELECT
    name,
    CAST(number AS INTEGER) as number,
    type_1,
    type_2,
    CAST(catch_rate AS INTEGER) as catch_rate,
    CAST(final_evolution AS DOUBLE) as final_evolution,
    CAST(generation AS DOUBLE) as generation,
    CAST(abilities AS DOUBLE) as abilities
FROM pokemon_lkp.pokemon_lookup.pokedex;
```

You should insert 1032 rows.

## Step 8 - Validate the pokemon\_pokedex\_structure table

It is considered best practice to validate that the new table has been created. To do this, run a simple select statement.

```
SELECT * FROM pokemon_pokedex_structure LIMIT 10;
```

name	number	type_1	type_2	catch_rate	final_evolution	generation	abilities
Bulbasaur	1	Grass	Poison	45	0	1	6.9
Ivysaur	2	Grass	Poison	45	0	1	13
Venusaur	3	Grass	Poison	45	1	1	100
Mega Venusaur	3	Grass	Poison	45	1	6	155.5
Charmander	4	Fire	NULL	45	0	1	8.5
Charmeleon	5	Fire	NULL	45	0	1	19
Charizard	6	Fire	Flying	45	1	1	90.5

## Part 4: Build the consume layer

### Objective

Now it's time to construct the last of the three layers of your data lake reporting structure, the consume layer. This will ready the layer for final consumption by data consumers.

## Step 1 - Revisit the business requirements

Take this opportunity to refamiliarize yourself with the business case. This will inform the kinds of questions that you will ask about your dataset.

In this scenario, the business case required you to:

1. Create a final table output combining data from both structure tables.
2. Create a data product answering two specific business questions from the marketing department.
  - c. What are the easiest and most popular Pokemon to catch in San Francisco by Type\_1?
  - d. Find the total number of Pokemon caught for each Type\_1 and Type\_2 pairing. Also, find the average catch rate.

*Note: Easiest is defined by having a high catch rate. A high catch rate is greater than or equal to 100. Also consider that in the structure layer, you filtered out data that did not exist in the San Francisco Bay Area.*

## Step 2 - Create the consume table

Run the following query to create the consume table. This constructs a new table from two separate tables in S3. Note that the `pokemon_spawns_structure` table is an Iceberg table and the `pokemon_pokedex_structure` table is a Delta Lake table.

```
CREATE TABLE pokemon_final_spawns AS
SELECT
    s.number,
    s.name,
    s.latitude,
    s.longitude,
    p.type_1,
    p.type_2,
    p.catch_rate
FROM pokemon_pokedex_structure p
JOIN pokemon_spawns_structure s ON p.number = s.number
WHERE catch_rate > 100;
```

The results should show 94626 rows were added.

Run a `SELECT` statement to validate that the table's values are similar to the image below.

```
SELECT * FROM pokemon_final_spawns LIMIT 10;
```

number	name	latitude	longitude	type_1	type_2	catch_rate
13	Weedle	37.79	-122.41	Bug	Poison	255
16	Pidgey	37.79	-122.41	Normal	Flying	255
41	Zubat	37.79	-122.4	Poison	Flying	255
16	Pidgey	37.8	-122.41	Normal	Flying	255
60	Poliwag	37.8	-122.41	Water	NULL	255

### Step 3 - Find the easiest and most popular Pokemon in San Francisco

Now, you need to derive two different views for the marketing department. These will be used to answer the business questions. Remember, easiest is defined by a catch rate of greater than or equal to 100. The most popular is defined as the most number of appearances for a certain Pokemon for each Type\_1.

Create a [window function](#) to rank the most popular Pokemon for each Type\_1.

```
SELECT
    type_1,
    name,
    COUNT(*) AS total_appearances,
    RANK() OVER (PARTITION BY type_1 ORDER BY count(name) DESC
    ) AS rank_column
FROM
    pokemon_final_spawns
GROUP BY
    type_1,
    name
ORDER BY
    type_1,
    COUNT(*) DESC;
```

type_1	name	total_appearances	rank_column
Bug	Weedle	5352	1
Bug	Caterpie	3059	2
Bug	Paras	1582	3



Next, utilize a WITH statement to run a subquery and only select the most popular Pokemon types.

```
WITH
popular_types AS (
  SELECT
    type_1,
    name,
    COUNT(*) AS total_appearances,
    RANK() OVER (PARTITION BY type_1 ORDER BY COUNT(name) DESC
    ) AS rank_column
  FROM
    pokemon_final_spawns
  GROUP BY
    type_1,
    name
  ORDER BY
    type_1,
    COUNT(*) DESC
)
SELECT
  type_1,
  name,
  total_appearances
FROM
  popular_types
WHERE
  rank_column = 1
ORDER BY
  total_appearances DESC;
```

type_1	name	total_appearances
Poison	Zubat	17444
Normal	Pidgey	10636
Bug	Weedle	5352
Dark	Rattata	5198
Water	Magikarp	3090
Fairy	Clefairy	1909

## Step 4 - Create the first marketing view

Finally, you need to create a view to make the data visible to the marketing team. Use the SQL statement below. It is the same as the last step, but with the **first line added** to it.

```
CREATE OR REPLACE VIEW popular_types_sf_vw AS
WITH
popular_types AS (
  SELECT
    type_1,
    name,
    COUNT(*) AS total_appearances,
    RANK() OVER (PARTITION BY type_1 ORDER BY COUNT(name) DESC
    ) AS rank_column
  FROM
    pokemon_final_spawns
  GROUP BY
    type_1,
    name
  ORDER BY
    type_1,
    COUNT(*) DESC
)
SELECT
  type_1,
  name,
  total_appearances
FROM
  popular_types
WHERE
  rank_column = 1;
```

Run a select statement to validate the view was created properly.

```
SELECT * FROM popular_types_sf_vw;
```

type_1	name	total_appearances
Bug	Weedle	5352
Water	Magikarp	3090
Poison	Zubat	17444
Grass	Bellsprout	1278

## Step 5 - Create the second marketing view

Use a [Grouping set](#) to find the total number of Pokemon caught for each Type\_1 and Type\_2 pairing.

```
CREATE OR REPLACE VIEW counts_by_types_sf_vw AS
SELECT
    type_1,
    type_2,
    ROUND(AVG(catch_rate), 2) AS avg_catch_rate,
    COUNT(name) AS total_count
FROM
    pokemon_final_spawns
GROUP BY
    GROUPING SETS ((type_1, type_2))
ORDER BY
    type_1,
    total_count DESC;
```

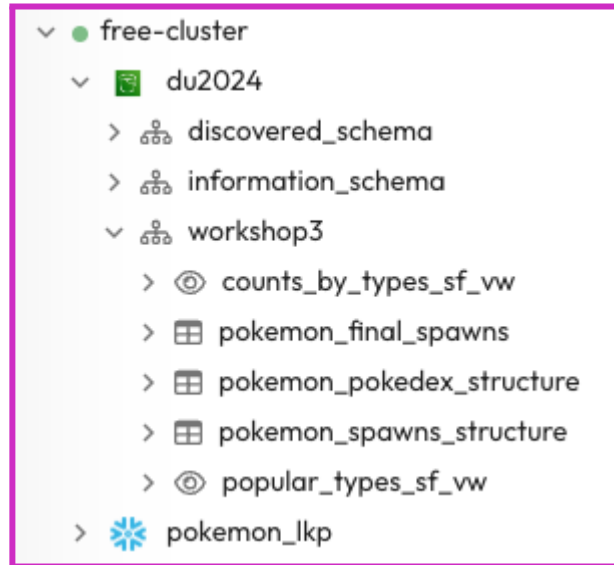
Run a select statement to validate the view was created properly. The results should be similar to the image below.

```
SELECT * FROM counts_by_types_sf_vw;
```

type_1	type_2	avg_catch_rate	total_count
Bug	Poison	235.58	7180
Normal	Flying	236.84	19739
Poison	Flying	255	17444
Bug	Grass	190	1582
Electric	NULL	190	979
Fighting	NULL	187.32	1127
Fairy	NULL	150	1909
Poison	NULL	233.24	5884
Grass	Poison	249.86	2627
Dark	Normal	251.34	5351

## Step 6 - Visualize the structure and consume layers

Collapse and expand the `workshop3` schema to see the 3 structure tables and 2 consume views that you have created.



## Part 5: Secure access to your consume layer

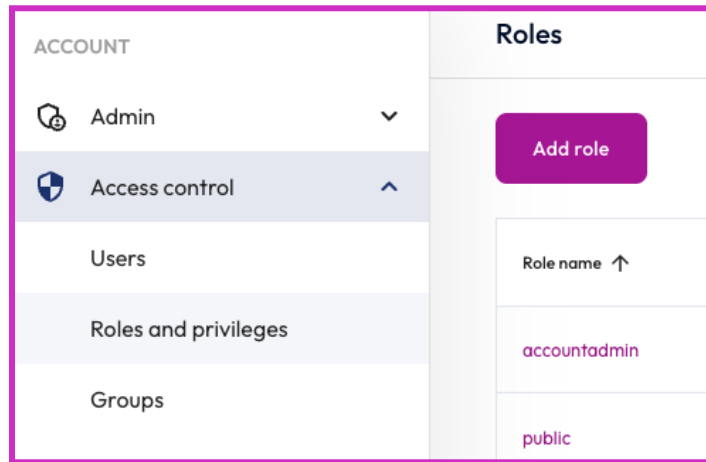
### Objective

The consume layer has been created. Now it's time to ensure that access to this data is restricted to the appropriate users.

### Step 1 - Create a marketing role

To restrict access to the consume layer, you're going to create a specific role for the marketing department. This will restrict access to the data to team members with the appropriate rights, and restrict their access to the two newly created views.

To do this, navigate to Access control near the bottom of the left navigation and select the **Roles and privileges** submenu item.



Select **Add role** and enter the following information:

**Role name:** marketing

**Description:** This role is specifically for the marketing department granting select access to two aggregated views.

**Grant to the creating role?** Yes

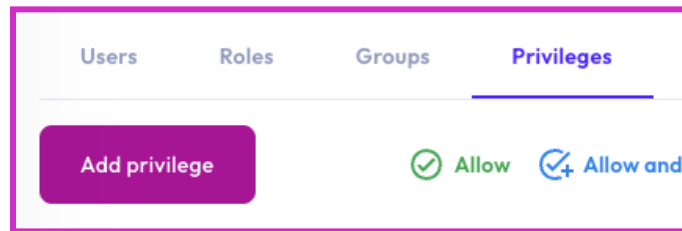
A screenshot of a 'Add a new role' dialog box. The dialog has a title bar with the text 'Add a new role' and a close button (X). Inside, there is a 'Role name' field with a red asterisk, containing the text 'marketing'. Below it is a 'Description' field containing the text 'This role is specifically for the marketing department granting select access on two Pokemon views.' At the bottom, there is a checkbox labeled 'Grant to the creating role?' which is checked. At the bottom right, there are two buttons: 'Cancel' and 'Add role'.

Next, select the newly created marketing role. This allows you to assign proper privileges.

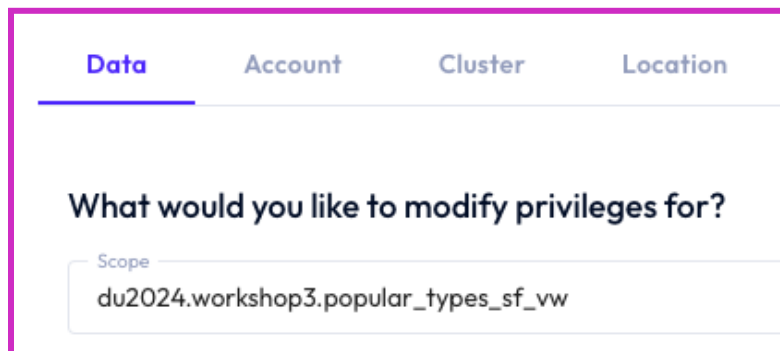


Roles		
Add role		
Role name ↑	Description	Granted to roles
accountadmin	-	_system
marketing	This role is specifically for the marketi...	accountadmin
public	-	_system

Navigate to the **Privileges** tab. Select **Add privilege**.



With the **Data** tab selected, use the explorer in the **What would you like to modify privileges for?** pulldown to be set to `du2024.workshop3.popular_types_sf_vw`.



Leave the default values for **allow/deny access** and **allow role to grant to others**.



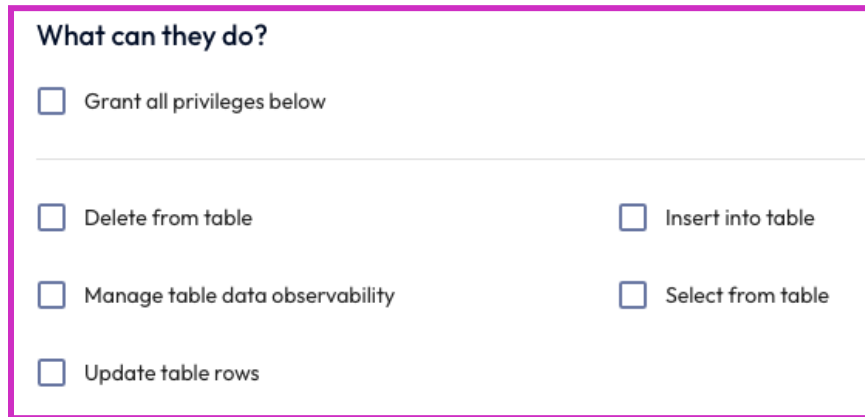
Do you want to allow or deny access?

☒ Allow ☐ Deny

Do you want to allow this role to grant to others?

☒ Allow role receiving privilege to grant to others

Scroll down if needed and then choose **Select from table** options in the bottom right of the **What can they do?** section.



What can they do?

☐ Grant all privileges below

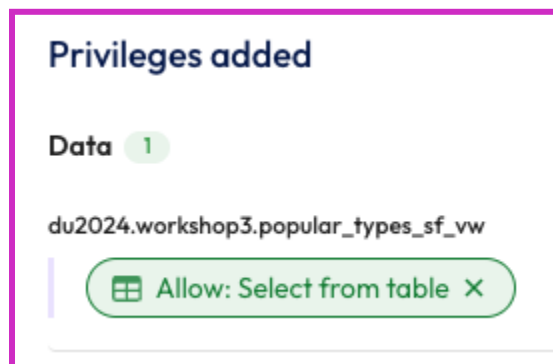
---

☐ Delete from table ☐ Insert into table

☐ Manage table data observability ☐ Select from table

☐ Update table rows

The **Select from table** option will disappear once you select it. It will appear on the right, under **Privileges added**.



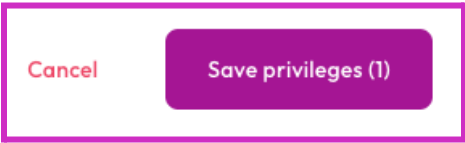
Privileges added

Data 1

du2024.workshop3.popular\_types\_sf\_vw

☒ Allow: Select from table X

Press the **Save privileges** button below the information just presented.



Repeat this process for `counts_by_types_sf_vw`.

Once back to the **Privileges** tab, expand the **Catalogs** line in the list below.

Account	1 privilege	▼
Clusters	1 privilege	▼
Catalogs	5 privileges	▼
Locations	0 privileges	▼
Functions	0 privileges	▼

Toggling `du2024` and then `workshop3` will show you that the **Select from table** privilege has been granted to both views.

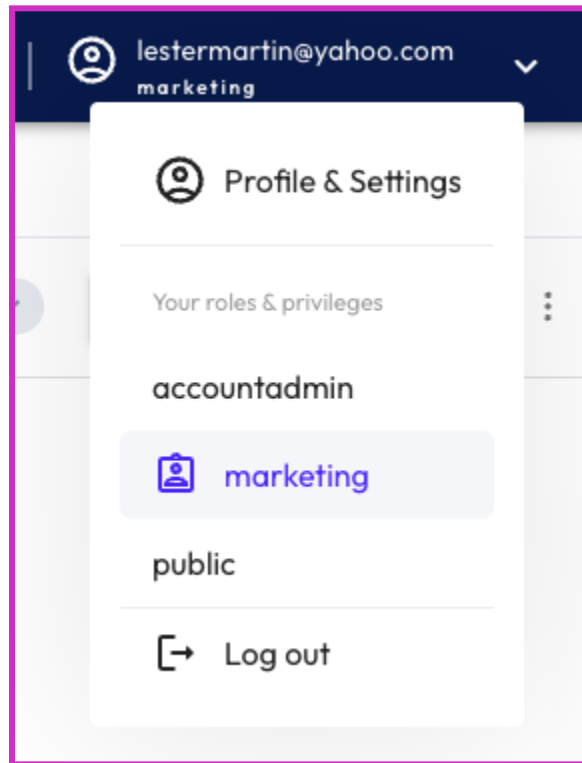
Entity name ↑	Create schema	Create table	Select from table
^ du2024 2 privileges			
^ workshop3 2 privileges			
counts_by_types_sf_vw			✓
popular_types_sf_vw			✓



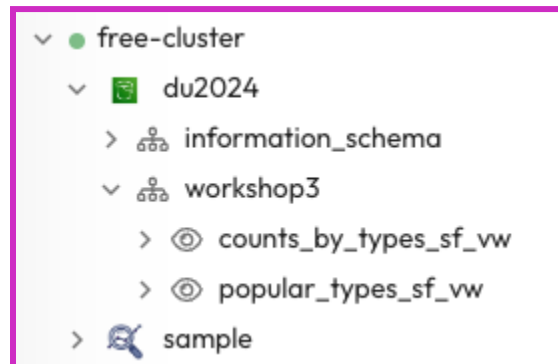
## Step 2 - Test the marketing role

Now it's time to test that the new role is working correctly.

Navigate back to the **Query editor** and switch to the **marketing** role in the top right-hand corner.



Notice that the cluster explorer shows only 2 views in the `workshop3` schema. This is exactly what you would expect. Additionally, the `pokemon_1kp` Snowflake catalog is not present.



Run a select statement to validate the newly created role has access to view the tables.

```
SELECT * FROM counts_by_types_sf_vw;
```

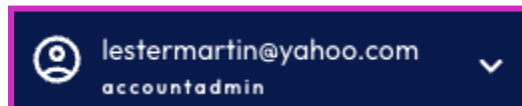
Now, try recreating the `popular_types_sf_vw` view using the marketing role.

```
CREATE OR REPLACE VIEW popular_types_sf_vw AS
WITH
popular_types AS (
  SELECT
    type_1,
    name,
    COUNT(*) AS total_appearances,
    RANK() OVER (PARTITION BY type_1 ORDER BY COUNT(name) DESC
    ) AS rank_column
  FROM
    pokemon_final_spawns
  GROUP BY
    type_1,
    name
  ORDER BY
    type_1,
    COUNT(*) DESC
)
SELECT
  type_1,
  name,
  total_appearances
FROM
  popular_types
WHERE
  rank_column = 1;
```

This will fail because you have only granted the marketing role select permissions.

Access Denied: Cannot create view aws\_pokemon.webinar3.popular\_types\_sf\_vw: Role marketing does not have the privilege CREATE\_TABLE on the schema aws\_pokemon.webinar3

Navigate back to the **accountadmin** role in the upper right-hand corner.



## Lab 4: Create data products

### Learning objectives

- Demonstrate how to execute a global search.
- Demonstrate how to create a data product.
- Demonstrate how to create tags.

### Prerequisites

- [Lab 1: Introduction and setup](#)
- [Lab 2: Connect to data sources](#)
- [Lab 3: Build within your data lake](#)

### Activities

1. Execute global search
2. Create a data product
3. Create tags

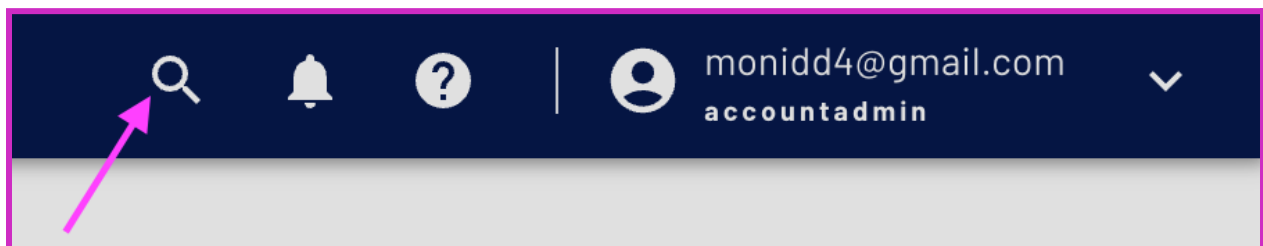
## Part 1: Execute global search

### Objective

Global search lets users find datasets quickly and intuitively. It is a powerful tool that helps keep better track of your data.

### Step 1 - Navigate Starburst UI

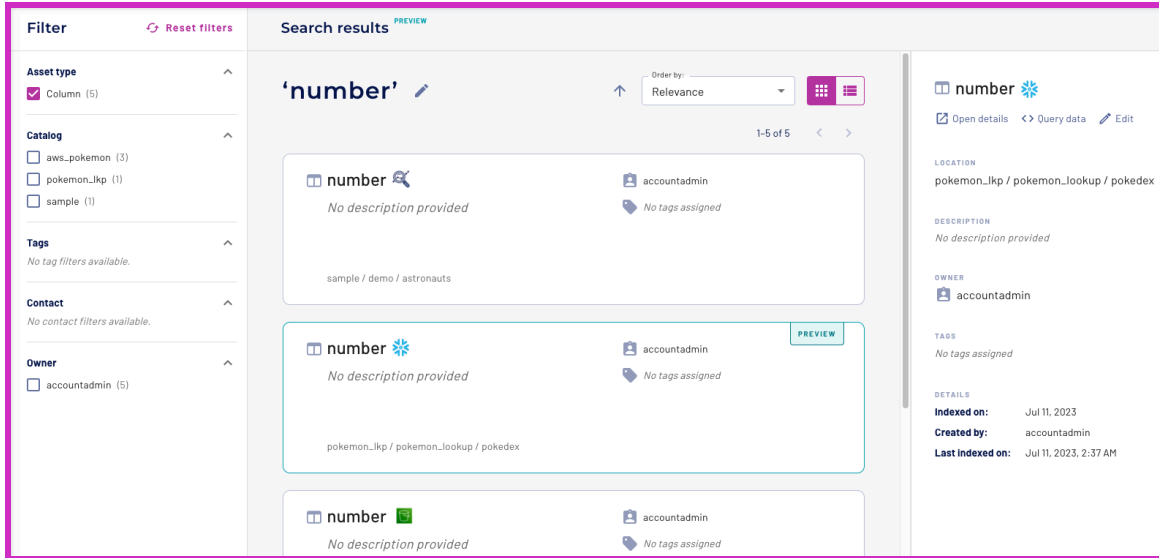
To use global search, select the magnifying glass icon in the upper-right corner.



### Step 2 - Execute global search

Enter the word `number` and select **View all results** at the bottom of the pop-up window.

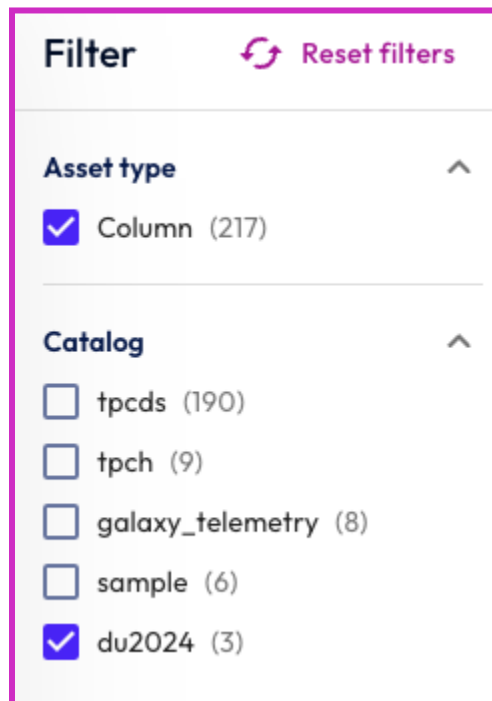
Starburst Galaxy displays multiple instances matching your search criteria drawn from multiple data sources. Some occur in tables and views you created throughout the lab. Others are simply populated through your catalog connection.



### Step 3 - Filter global search

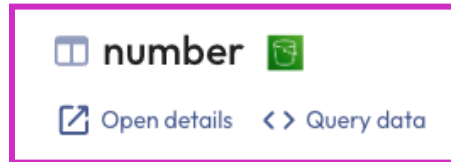
Global search can also be filtered to help refine your search.

Select the Catalog filter for du2024 on the left to see how different search criteria impact the results.



### Step 3 - Navigate to your newly created view

Select **Open details** on the far right for any of the `du2024` results .



Notice that Starburst Galaxy automatically routes you to the **Catalogs** page. Navigate to your `workshop3` schema by clicking on it.



The information for the entire schema is available to you, including **Tables, Views, Metrics, Definition, Query history**, and **Privileges**. Navigate through each tab to see the available features.

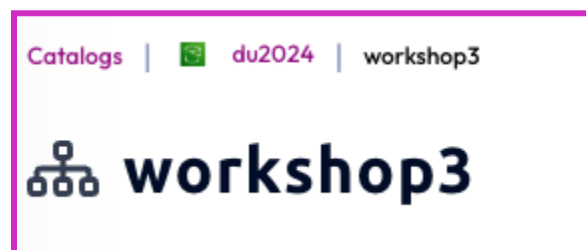
## Part 2: Create a data product

### Objective

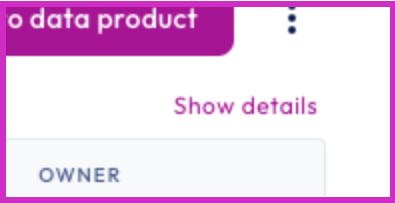
Now it's time to create a data product using your datasets. Data products curate data in a way that makes it more accessible and useful, and can be shared across teams.

### Step 1 - Enter additional information

Click on the schema name so you are looking at its metadata.



Before creating a data product, enter some information into your catalog. Select **Show details** on the right-hand side.



Edit the following information:


**Description:** Data evaluating Pokemon catches in San Francisco.

**Links:** Text to display: National Pokedex

Link URL: <https://www.serebii.net/pokemon/nationalpokedex.shtml>





**Contacts:** yourname

Catalogs | du2024 | workshop3

 workshop3


Promote to data product

Hide details

DESCRIPTION	TAGS	LINKS	CONTACTS	OWNER
Data evaluating Pokemon catches in San Francisco.	 0	 1	 1	 accountadmin


DESCRIPTION

Data evaluating Pokemon catches in San Francisco.





TAGS

No tags added yet.





LINKS

 National Pokedex





CONTACTS

 lestermartin@yahoo.com



OWNER

 accountadmin



If you have extra time, go through the tables and views created and add meaningful descriptions to each table/view and the columns within them.

counts\_by\_types\_sf\_vw

Show details

DESCRIPTION

TOTALS

CONTACTS

OWNER

Total number of Pokemon appearances for each type\_1 and type\_2 pairing.

0

0

accountadmin

Columns 4

Metrics

Definition

Data preview

Query history 3

Audit log

Privileges

Refresh

4 columns

Search columns

Column ↑	Type	Nullable	Default	Tags	Description
avg_catch_rate	double	yes	null	No tags assigned.	+ Average catch rate of each pok...
total_count	bigint	yes	null	No tags assigned.	+ Total appearance count for ea...
type_1	varchar	yes	null	No tags assigned.	+ Primary type
type_2	varchar	yes	null	No tags assigned.	+ Secondary type

## Step 2 - Promote your data product

Navigate back to your schema and select **Promote to data product**.

The screenshot shows the 'workshop3' data product page. At the top, there's a breadcrumb trail: 'Catalogs | du2024 | workshop3'. The main header area includes the 'workshop3' logo and a 'Promote to data product' button. Below this, there's a 'Show details' link. The main content area is divided into five sections: 'DESCRIPTION' (Data evaluating Pokemon catches in San Francisco.), 'TAGS' (0), 'LINKS' (1), 'CONTACTS' (1), and 'OWNER' (accountadmin).

Import all the information you've already added to the schema. Select **Import**.

The modal dialog is titled 'Import monica\_miller metadata?'. It asks: 'Would you like to import the monica\_miller metadata? Select which fields you would like to import.' There are three checkboxes, all of which are checked: 'Description', 'Links', and 'Contacts'. At the bottom right, there are two buttons: 'Skip' and 'Import'.

Add a descriptive name like `Workshop #3: Pokemon Analysis` for the **Data product name** input field.

The form is titled 'Name, summary, and description'. It has a single input field labeled 'Data product name \*'. The text 'Workshop #3: Pokemon Analysis' is entered into the field.



The summary has already been populated based on the information you added to the schema.

**Add the following description:**

Use this data product to plan marketing activity around SF.

- **Question:** What are the easiest and most popular Pokemon to catch? Which are the most prevalent type pairings?
- **Objective:** Plan marketing campaigns for different Pokemon based on popularity.
- **Approach:** Create two specific views.

*Note: Easiest is defined by having a high catch rate. A high catch rate is greater than or equal to 100. Geolocation data is filtered to only be within the San Francisco Bay Area.*

Description

Use this data product to plan marketing activity around SF.

-- Question: What are the easiest and most popular Pokemon to catch? Which are the most prevalent type pairings?

-- Objective: Plan marketing campaigns for different Pokemon based on popularity.

-- Approach: Create two specific views.

Note: Easiest is defined by having a high catch rate. A high catch rate is greater than or equal to 100. Geolocation data is

Select `free-cluster` as the **Default cluster**.

**Default cluster**

Select cluster

`free-cluster`

The **Contacts** and **Supporting information** have been automatically populated from the schema. Select **Promote to data product**.

Cancel

Promote to Data Product

Congratulations! You have created and promoted your first data product. You were routed to the Data products page where you can view your work.



## Part 3: Create tags (Bonus)

### Objective

Last step. It's time to create tags. These can be used to identify the attributes of a dataset so they can be easily searched later. Tagging is flexible and allows you to create the level of granularity that works best for you. Tags can be assigned to the data product, the tables/views within the data product, or the columns within the tables/views.

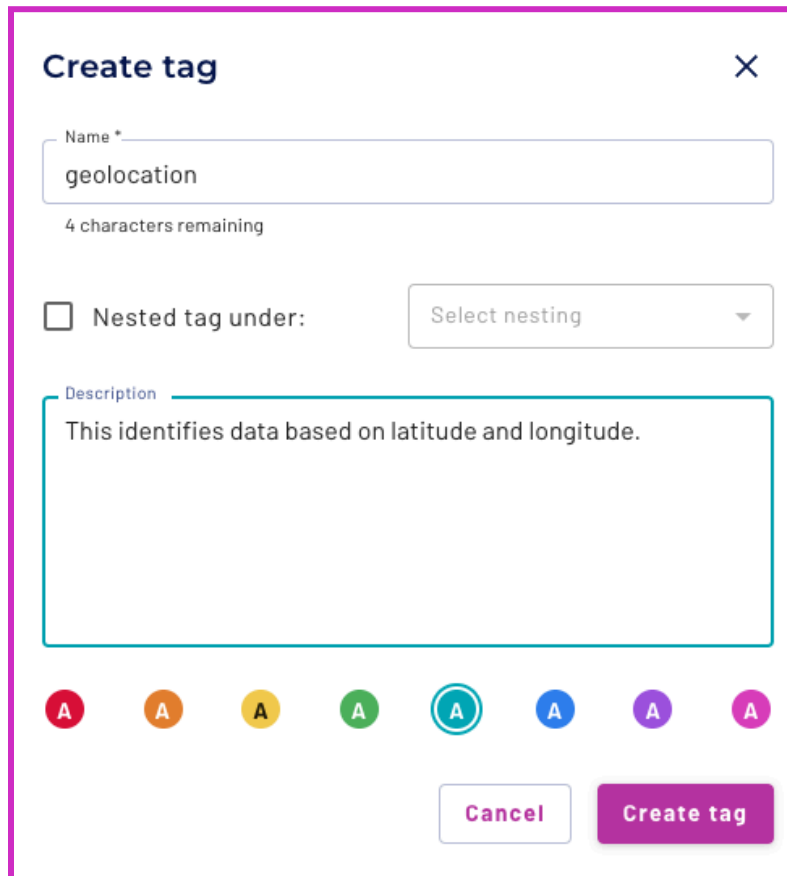
### Step 1 - Create a new tag

Navigate to Admin in the left nav and then select the **Tags** submenu item. Select **Create tag** then fill in the input fields as identified below.

**Name:** geolocation

**Description:** This identifies data based on latitude and longitude.

**Color:** Your choice



**Create tag** ✕

Name \*  
geolocation  
4 characters remaining

☐ Nested tag under: Select nesting ▼

Description  
This identifies data based on latitude and longitude.

A A A A A A A

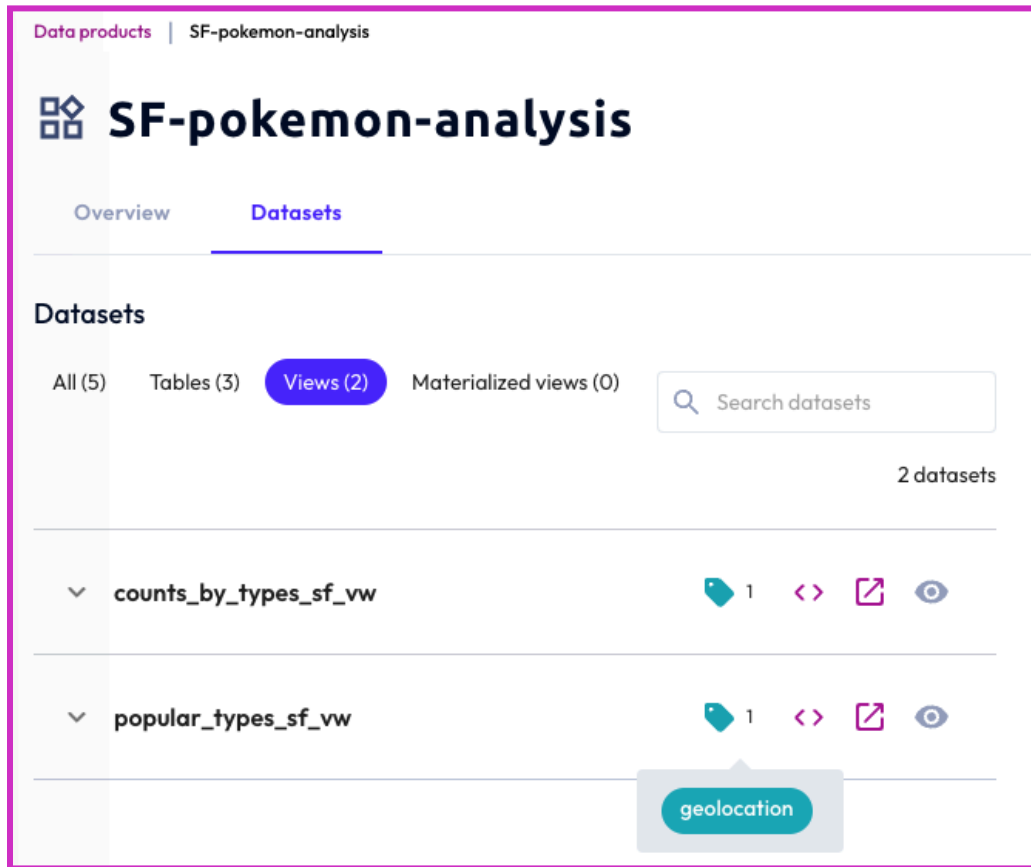
Cancel Create tag

## Step 2 - Assign the new tag appropriately

Your mission is to navigate to both views within the data product you created and correctly assign the tag to the created views. Can you figure it out on your own?

**Hint:** Assign the tags in the **Catalogs** page then verify they are present in the **Data products** page.

Your results should look like following. Ask for help if you need it! :)



If you have more time, add additional information to your data product to make it a more meaningful and curated dataset for your end users.