databricks

Hands-on Lab

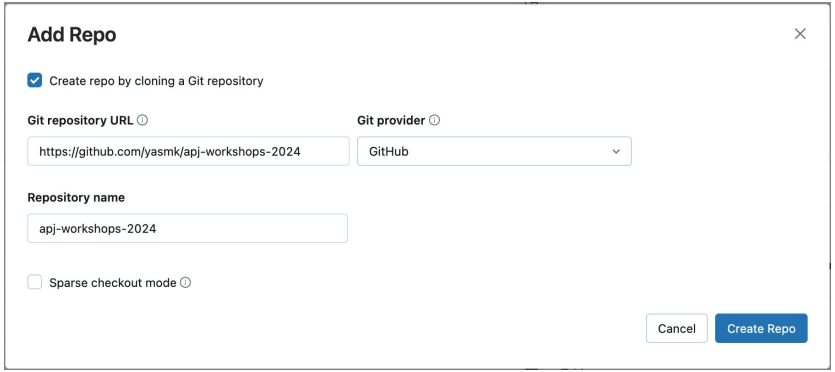
Data Intelligence Day, 2024

Replicate the Code

1. Click on **New** then **Repo**
2. Enter the following **Git URL**

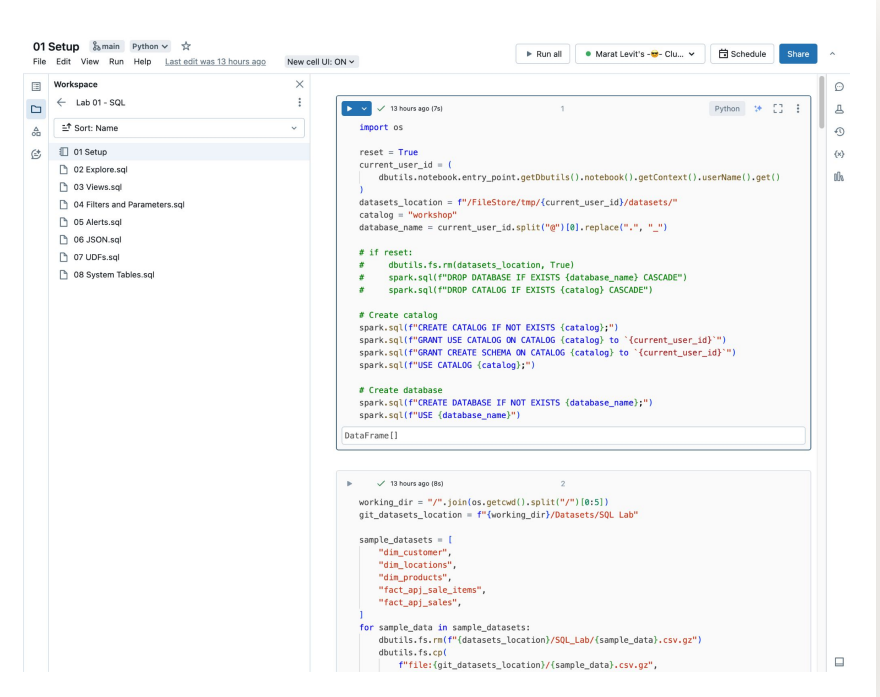
https://github.com/yasmk/apj-workshops-2024

3. Click **Create Repo**



Create Datasets

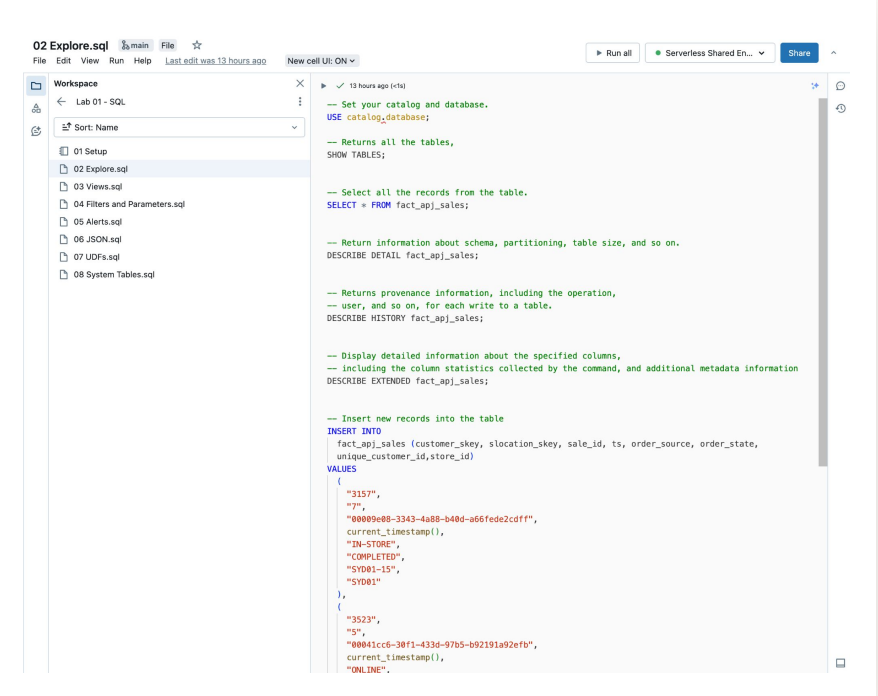
1. Find your cloned repository under **Workspace > Repo**
2. Open the “01 Setup” notebook under the “Lab 01 – SQL” folder
3. Click **Run all**
4. Give Databricks a couple of minutes to create all required datasets



Data Exploration

Data Exploration

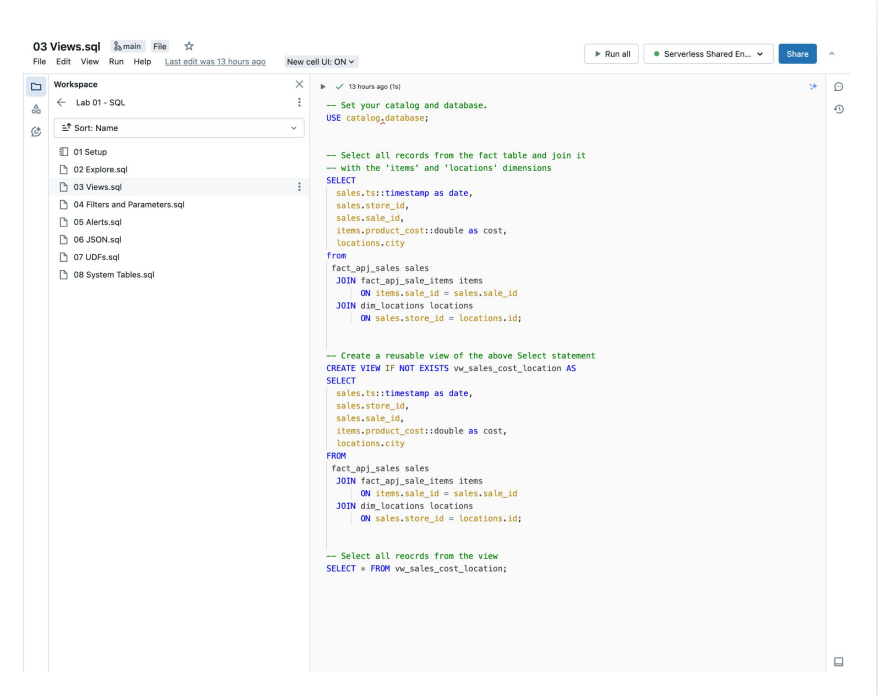
1. Open “02 Explore.sql”
2. Replace “catalog.database” with “workshop.<username>” and run the command
3. Run each command one at a time by highlighting it and hitting **Ctrl + Shift + Enter**



Views

Views

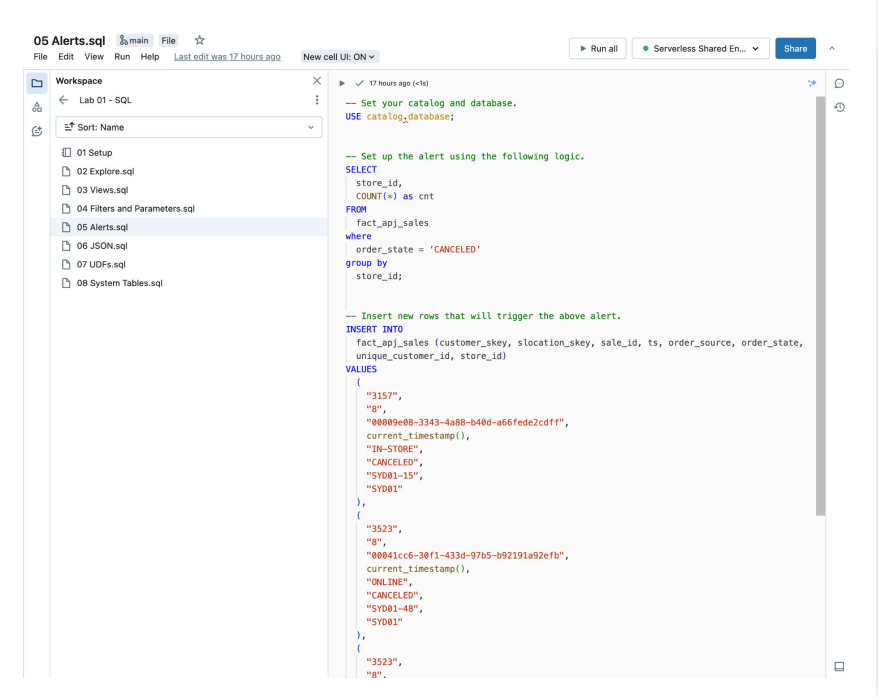
1. Open “03 Views.sql”
2. Replace “catalog.database” with “workshop.<username>” and run the command
3. Run each command one at a time by highlighting it and hitting **Ctrl + Shift + Enter**



Alerts

Alerts

1. Open “05 Alerts.sql”
2. Replace “catalog.database” with “workshop.<username>” and run the command
3. Copy the “USE workshop…” and the following “SELECT” query to clipboard



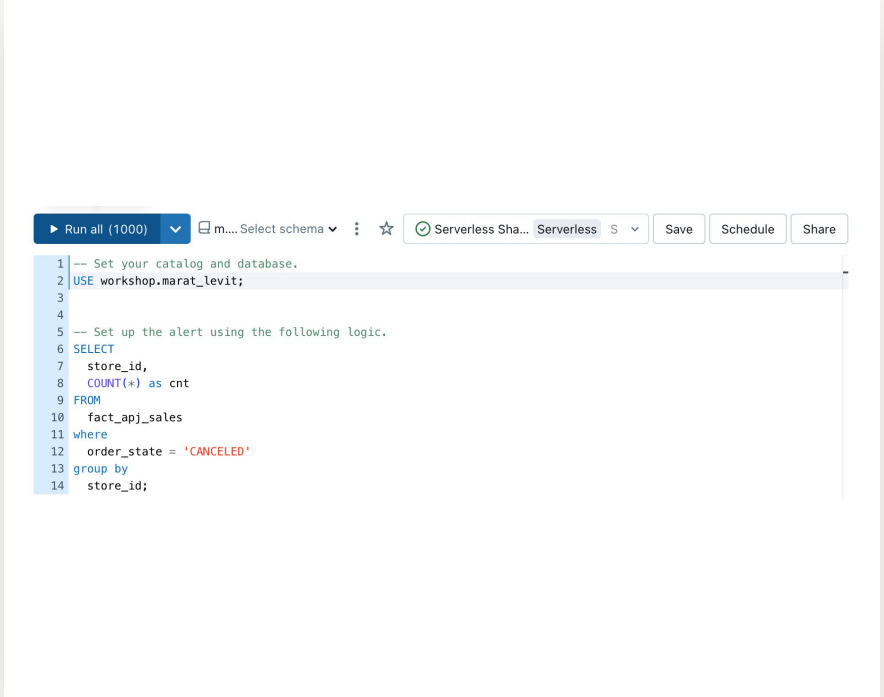
Alerts

Create Query

1. Open **Queries** from the side panel

2. Click **Create query**. In the **SQL Editor** paste the SELECT query

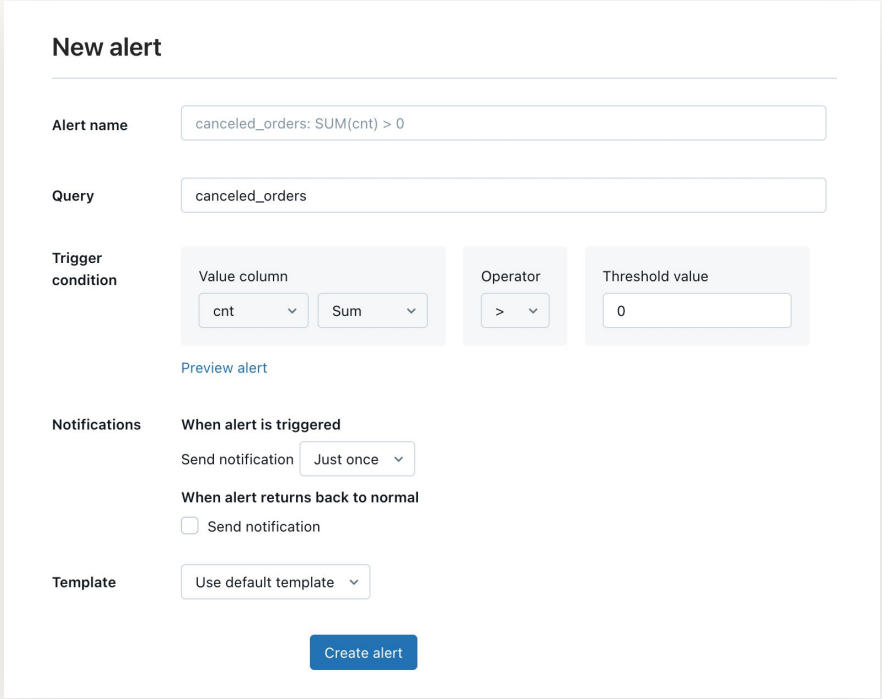
3. Click **Save** and name the query “canceled_orders”



Alerts

Create Alert

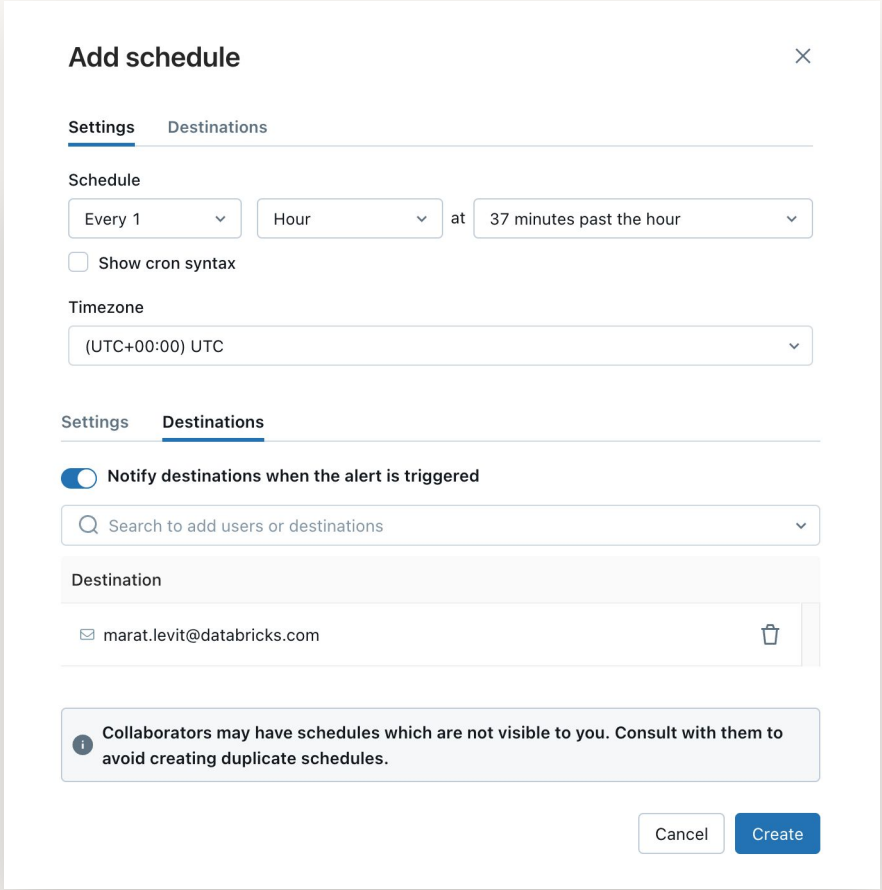
1. Open **Alerts** from the side panel
2. Click **Create alert**
3. Select the “canceled_orders” query from the dropdown
4. Change the trigger condition to **cnt SUM > 0**
5. Click **Create alert**



Alerts

Schedule & Subscribe

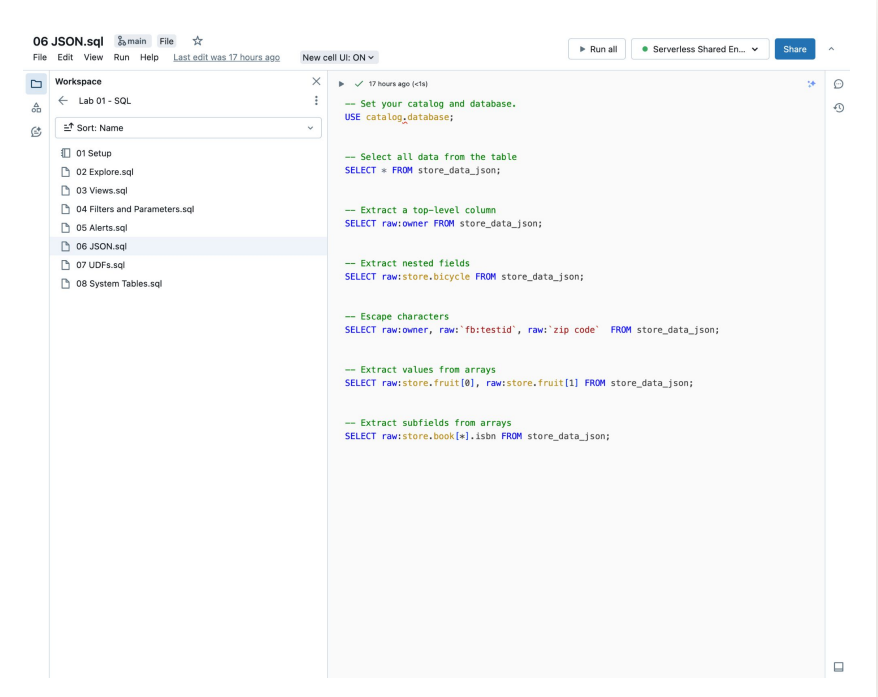
1. Click **Add schedule** after creating the Alert
2. Here you can adjust how often the alert query runs
3. Switch to the **Destinations** tab
4. Search for your username
5. Click **Create**



JSON

JSON

1. Open “06 JSON.sql”
2. Replace “catalog.database” with “workshop.<username>” and run the command
3. Run each command one at a time by highlighting it and hitting **Ctrl + Shift + Enter**



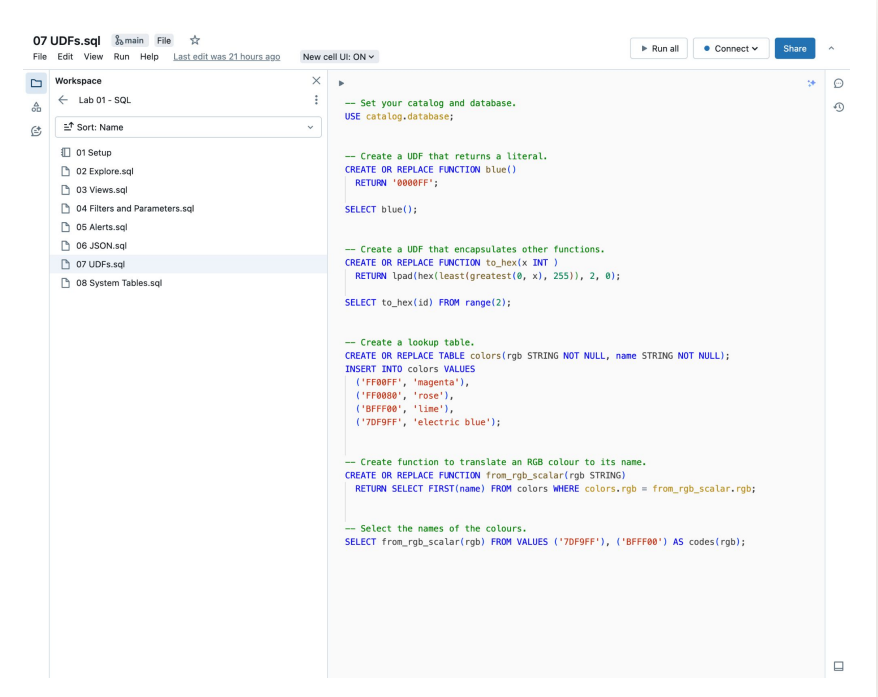
User-defined Function (UDF)

UDF

1. Open “07 UDFs.sql”

2. Replace “catalog.database” with “workshop.<username>” and run the command

3. Run each command one at a time by highlighting it and hitting **Ctrl + Shift + Enter**

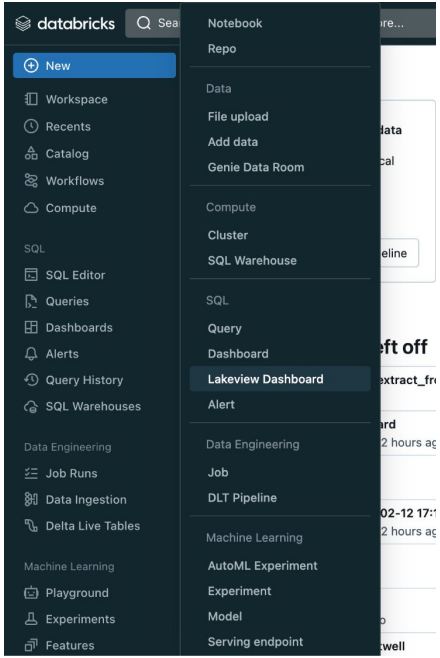


Dashboards

Presented by Junchi

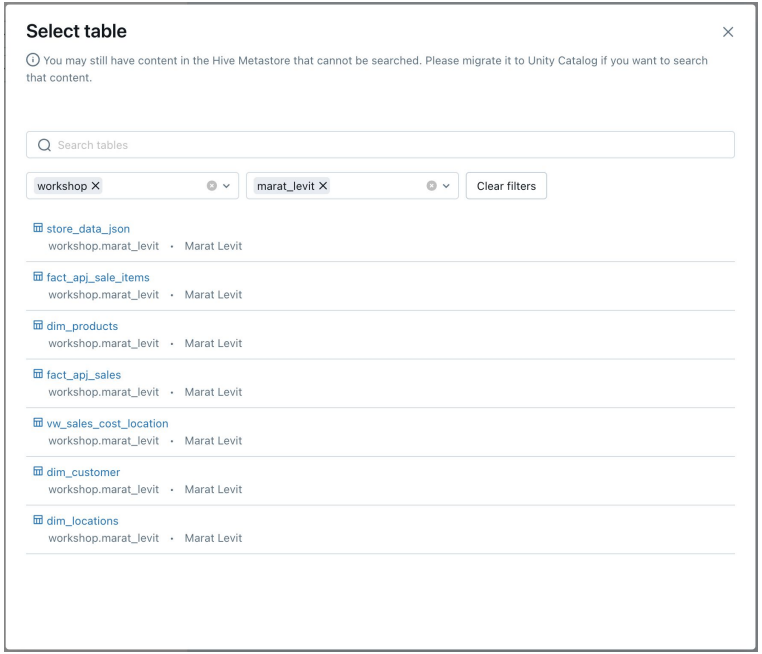
Create a Lakeview Dashboard

1. Click on **New**
2. Select **Lakeview Dashboard**



Select the DIM Customer Dataset

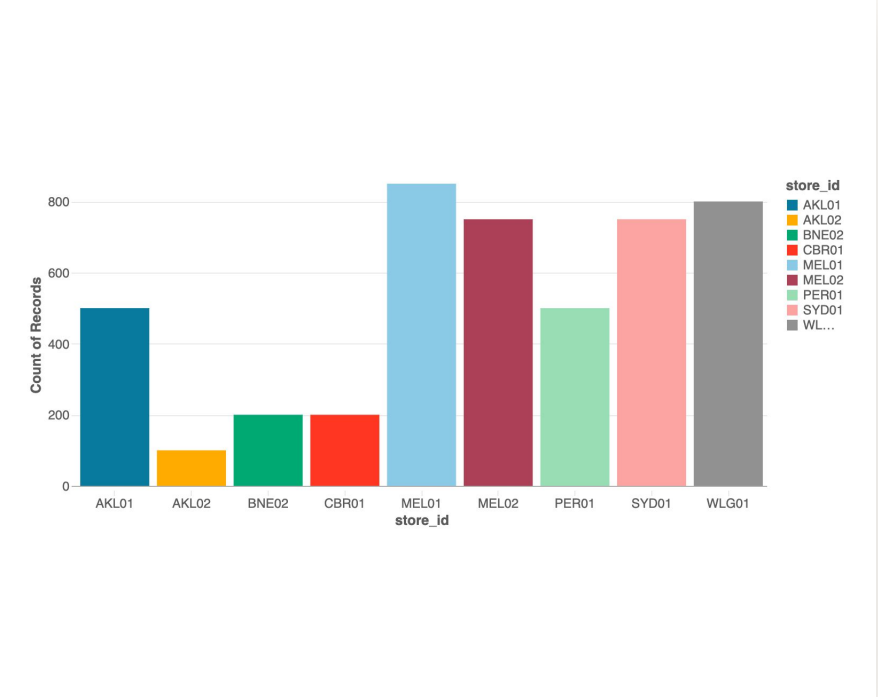
1. Switch over to the **Data** tab and click **Select a table**
2. In **Catalog** enter “workshop”
3. In **Database** enter your username
4. Select the “dim_customer” table



Create Customer by Store

How many customers does each store have?

1. Switch over to the **Canvas** tab and click the **Create a visualization** button
2. Place the visualization onto the canvas
3. For **X Axis** select “store_id”
4. For **Y Axis** select “COUNT(*)”
5. For **Color/Group by** select “store_id”



Create a Sales by Week Dataset

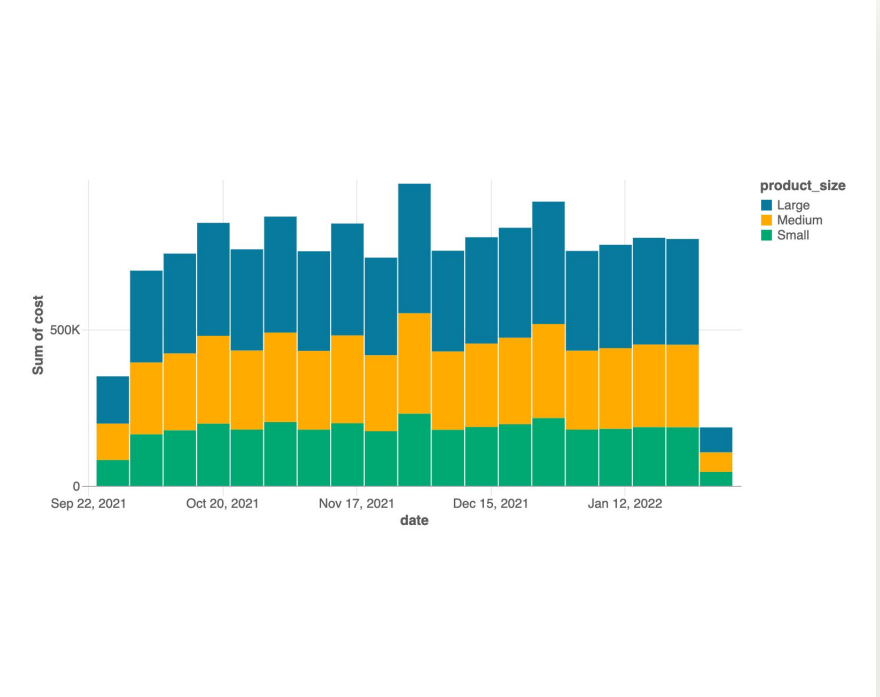
1. Switch over to the **Data** tab and click the **Create from SQL** button
2. Enter the query on the right and run it
3. Rename the dataset to “Sales by Week”

```
select
  sales.ts::timestamp as date,
  items.product_cost::double as cost,
  items.product_id as product,
  items.product_size,
  sales.sale_id,
  sales.store_id
from
  workshop.database.fact_apj_sales sales
  join workshop.database.fact_apj_sale_items items
       on items.sale_id = sales.sale_id
```

Create Sales by Week

Sales by week by product size purchased

1. Switch over to the **Canvas** tab and click the **Create a visualization** button
2. For **Dataset** select “Sales by Week”
3. For **X Axis** select “date”
4. For **Y Axis** select “cost”
5. For **Color/Group by** select “product_size”



Create a Sales by Region Dataset

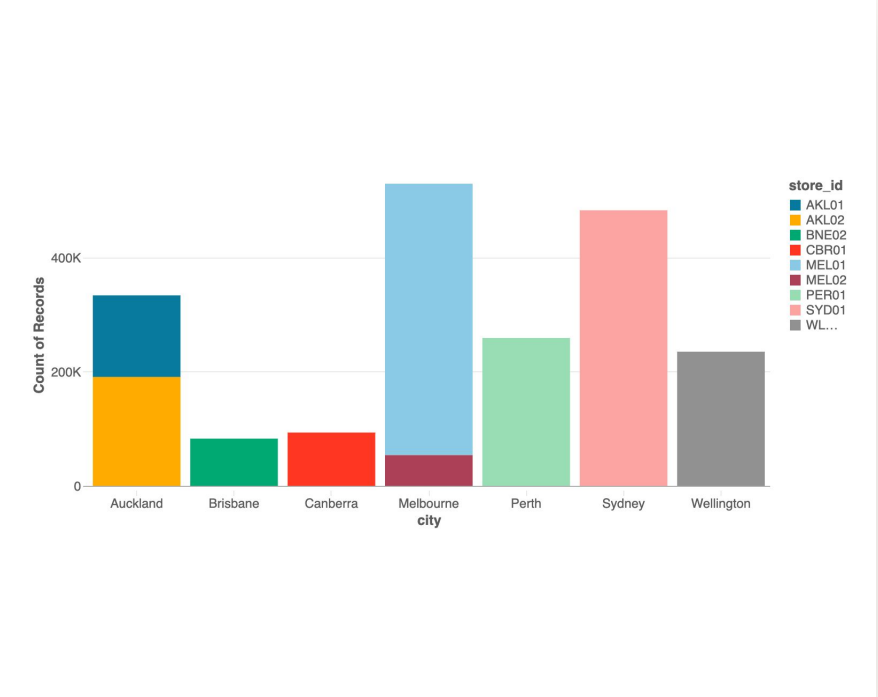
1. Switch over to the **Data** tab and click the **Create from SQL** button
2. Enter the query on the right and run it
3. Rename the dataset to “Sales by Region”

```
select
  sales.ts::timestamp as date,
  sales.store_id,
  sales.sale_id,
  items.product_cost::double as cost,
  locations.city
from
  workshop.database.fact_apj_sales sales
  join workshop.database.fact_apj_sale_items items
       on items.sale_id = sales.sale_id
  join workshop.database.dim_locations locations
       on sales.store_id = locations.id
```

Create Sales by Region

Which stores are performing well or poorly?

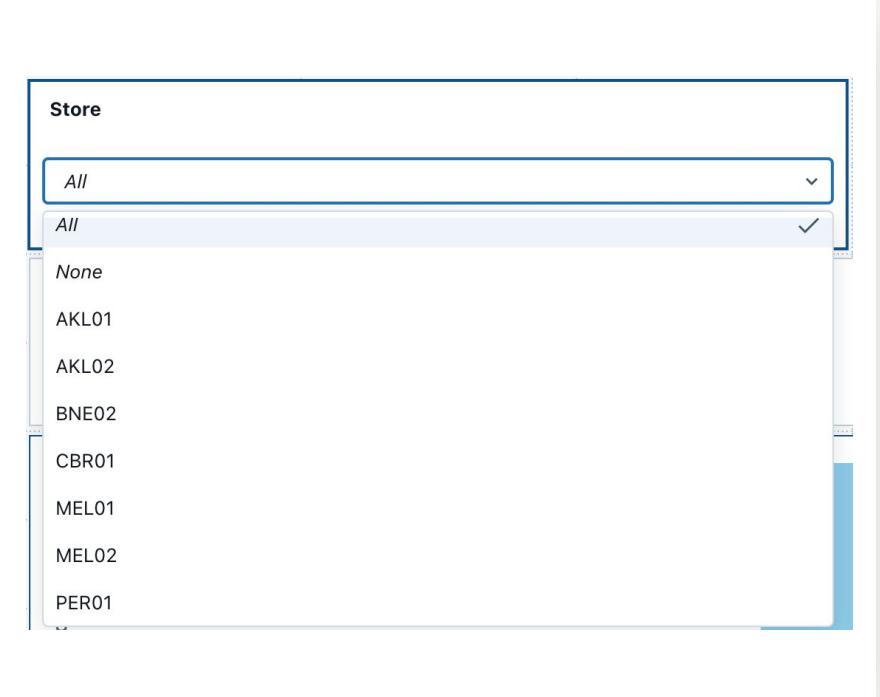
1. Switch over to the **Canvas** tab and click the **Create a visualization** button
2. For **Dataset** select “Sales by Region”
3. For **X Axis** select “city”
4. For **Y Axis** select “COUNT(*)”
5. For **Color/Group by** select “store_id”



Create Store Filter

1. On the **Canvas** tab click the **Add a filter** button

2. Add all three datasets to the **Filter on** section and select “store_id” as the filter column for each dataset



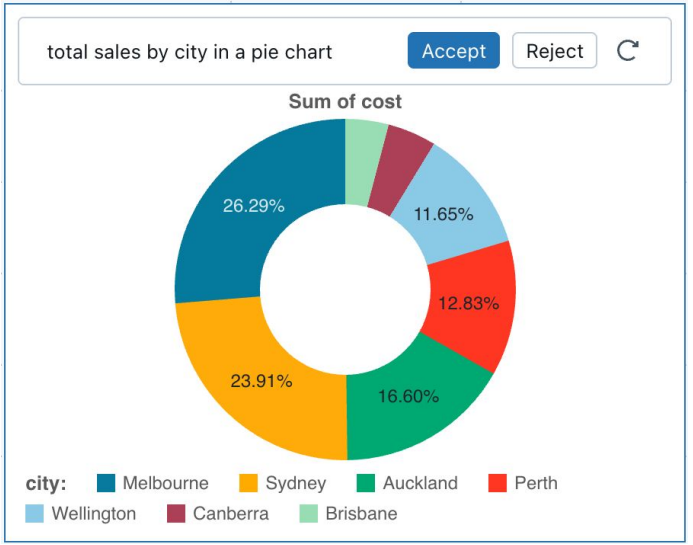
Natural Language Visualisations

Create Sales by City

1. On the **Canvas** tab and click the **Create a visualization** button

2. In the **Describe a chart…** section enter “total sales by city in a pie chart”

3. Hit **Enter**

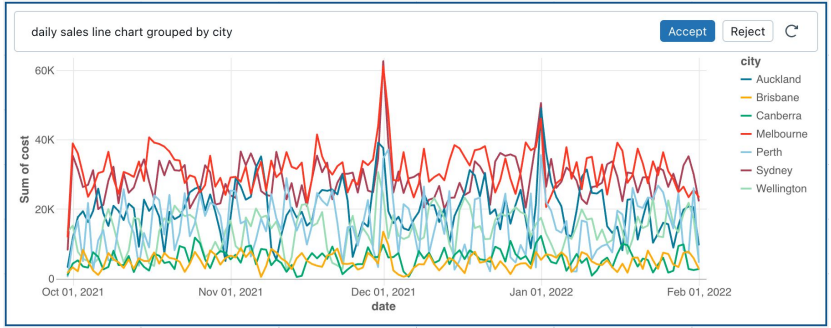


Create Daily Sales by City

1. On the **Canvas** tab and click the **Create a visualization** button

2. In the **Describe a chart…** section enter “daily sales line chart grouped by city”

3. Hit **Enter**

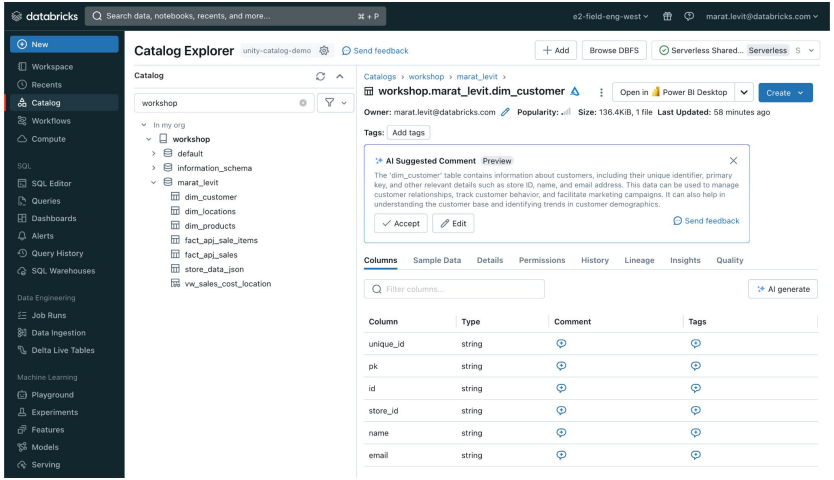


Governance

View Tables in Unity Catalog

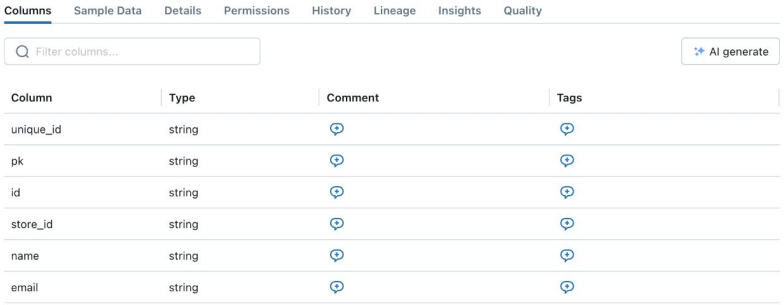
1. From the side panel, click on **Catalog**

2. Navigate to your table by drilling down into the **workshop** catalog, followed by your **username** database, and select “dim_customer”



View Column and Sample Data

1. Click through the **Columns** and **Sample Data** tabs



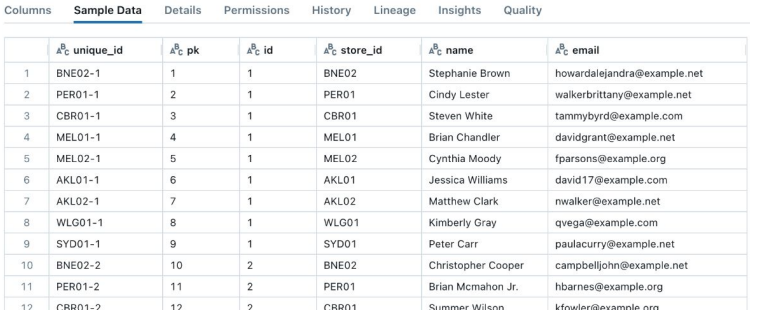
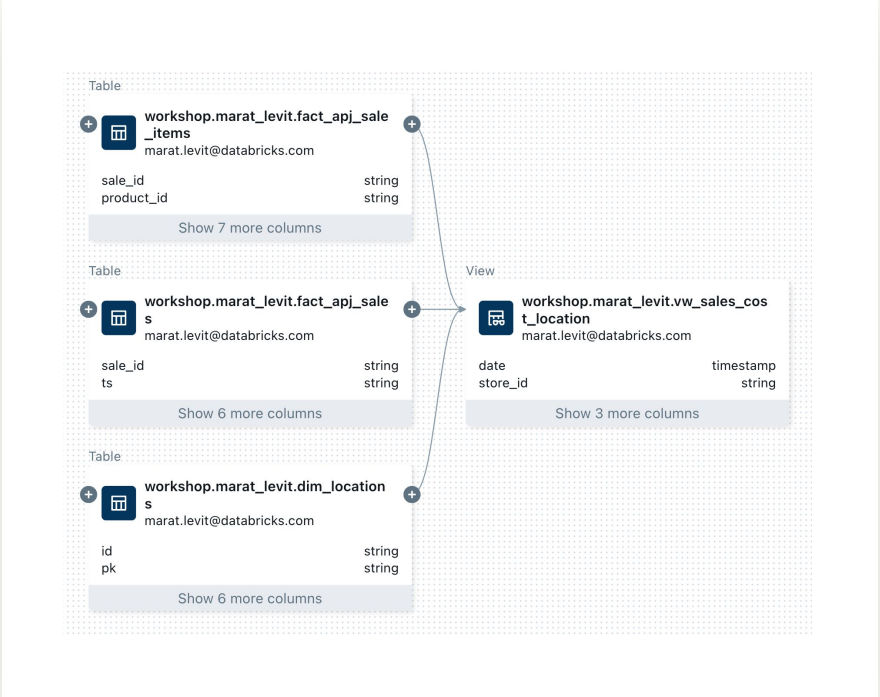


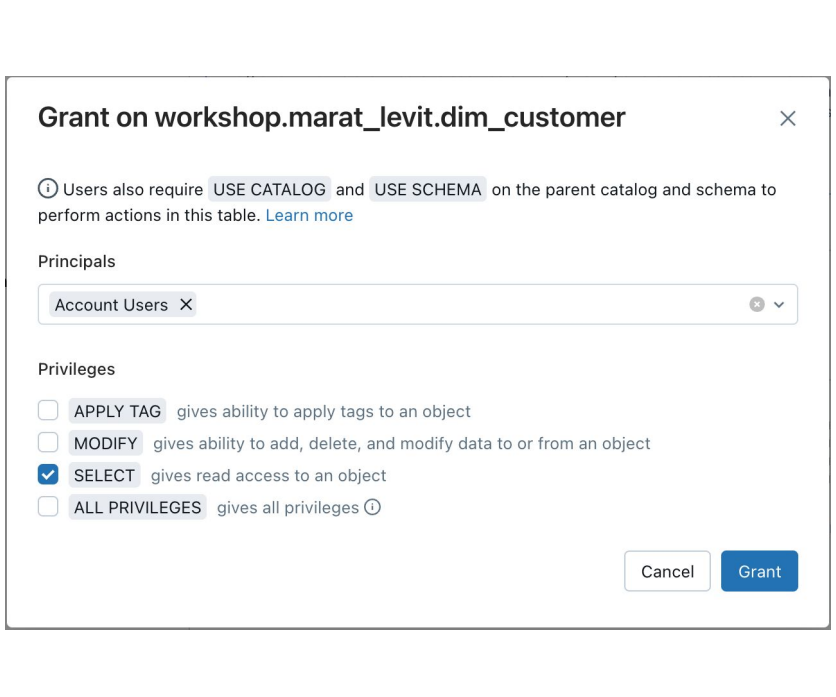
Table and Column Lineage

1. Select the “vw_sales_cost_location” view from the **Catalog Explorer**
2. Click through to the **Lineage** tab
3. Click the **See lineage graph** button



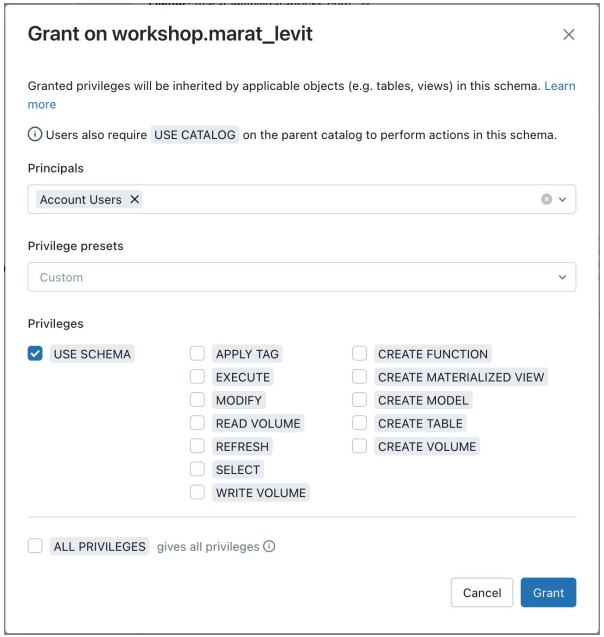
Manage Table Permissions

1. Navigate to the **Permissions** tab
2. Click the **Grant** button
3. Enter “Account Users” into the **Principals** field
4. Tick “SELECT” under **Privileges**
5. Click the **Grant** button to grant those privileges



Manage Catalog Permissions

1. Navigate to the “workshop” catalog in the **Catalog Explorer** panel
2. Navigate to the **Permissions** tab
3. Click the **Grant** button
4. Enter “Account Users” into the **Principals** field
5. Tick “USE SCHEMA” under **Privileges**
6. Click the **Grant** button to grant those privileges



Management

Billing Usage

Analyze and optimize DBU consumption

1. Navigate to the “system” catalog in the **Catalog Explorer** panel
2. Open the “billing” database and click on the “usage” table
3. With system tables, your account’s billable usage data is centralized and routed to all regions, so you can view your account’s global usage from whichever region your workspace is in.
4. Usage is aggregated hourly, so each record in **system.billing.usage** represents an hour of billable usage.

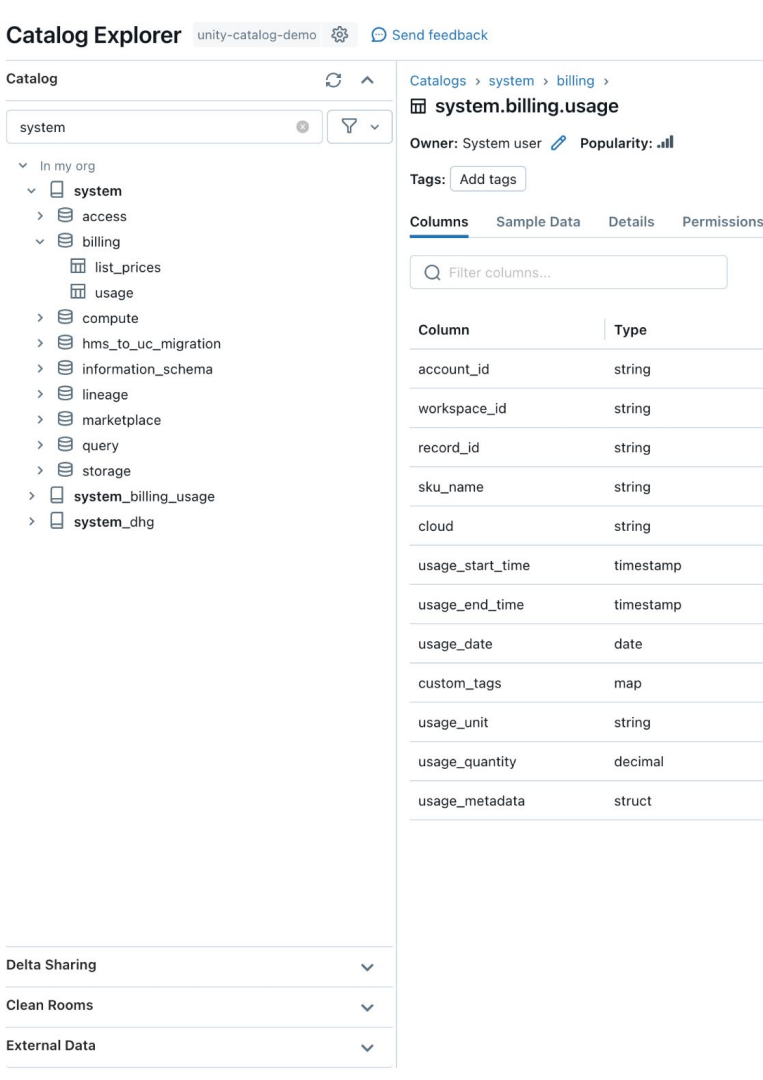
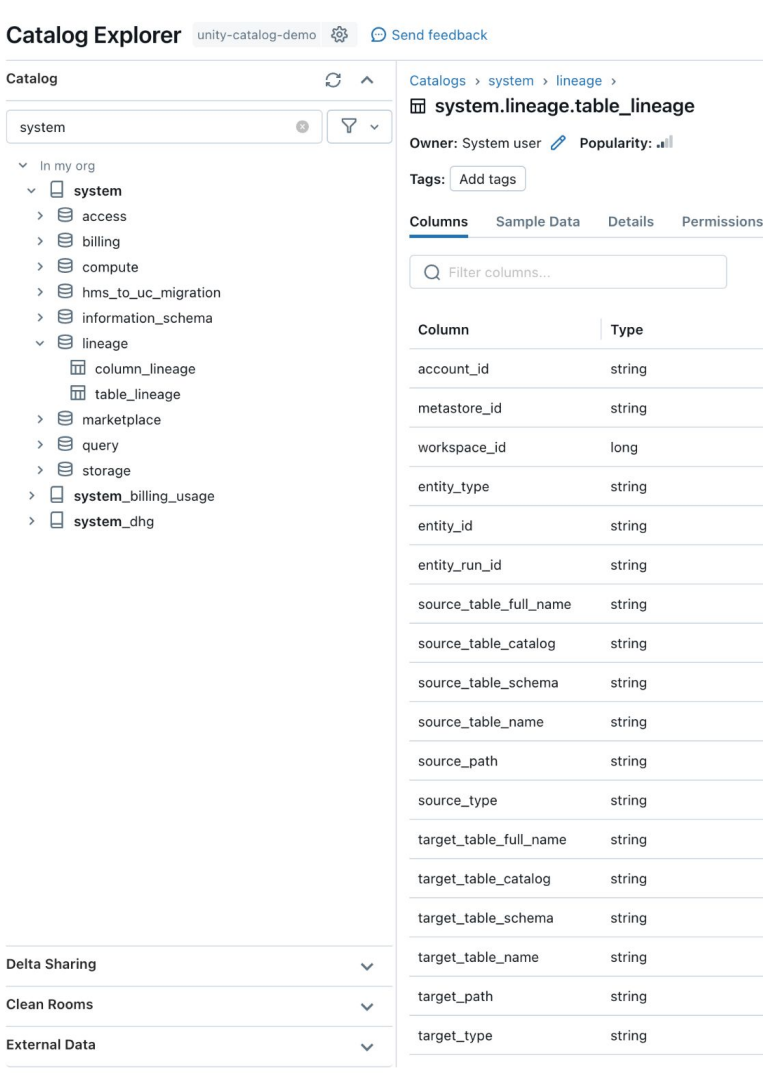


Table and Column Lineage

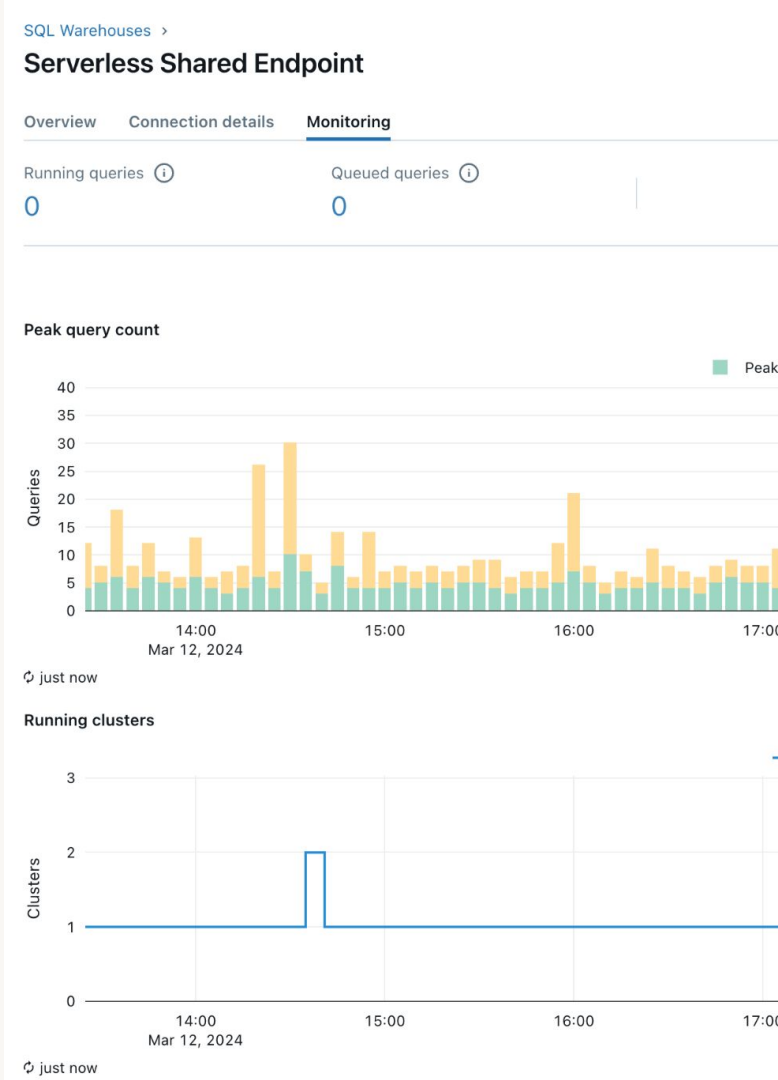
Programmatically query lineage data

1. Navigate to the “system” catalog in the **Catalog Explorer** panel
2. Open the “lineage” database and click on the “column_lineage” or “table_lineage” tables
3. The **table lineage table** includes a record for each read or write event on a Unity Catalog table or path.
4. The **column lineage table** does not include events that do not have a source. For example, if you insert into a column using explicit values, it is not captured.



Warehouse Management

1. Navigate to the **SQL Warehouses** the side panel
2. Click on your SQL Warehouse name and select the **Monitoring** tab. Here you'll find:
    a. **Live statistics** show the currently running and queued queries, active SQL sessions, the warehouse status, and the current cluster count.
    b. The **peak query count chart** shows the maximum number of concurrent queries, either running or queued, on the warehouse during the selected time frame.
    c. The **running clusters chart** shows the number of clusters allocated to the warehouse during the selected time frame.
    d. The **query history table** shows all of the queries active during the selected time frame



Query Inspection

1. Click on any of the successful queries in the **query history table**