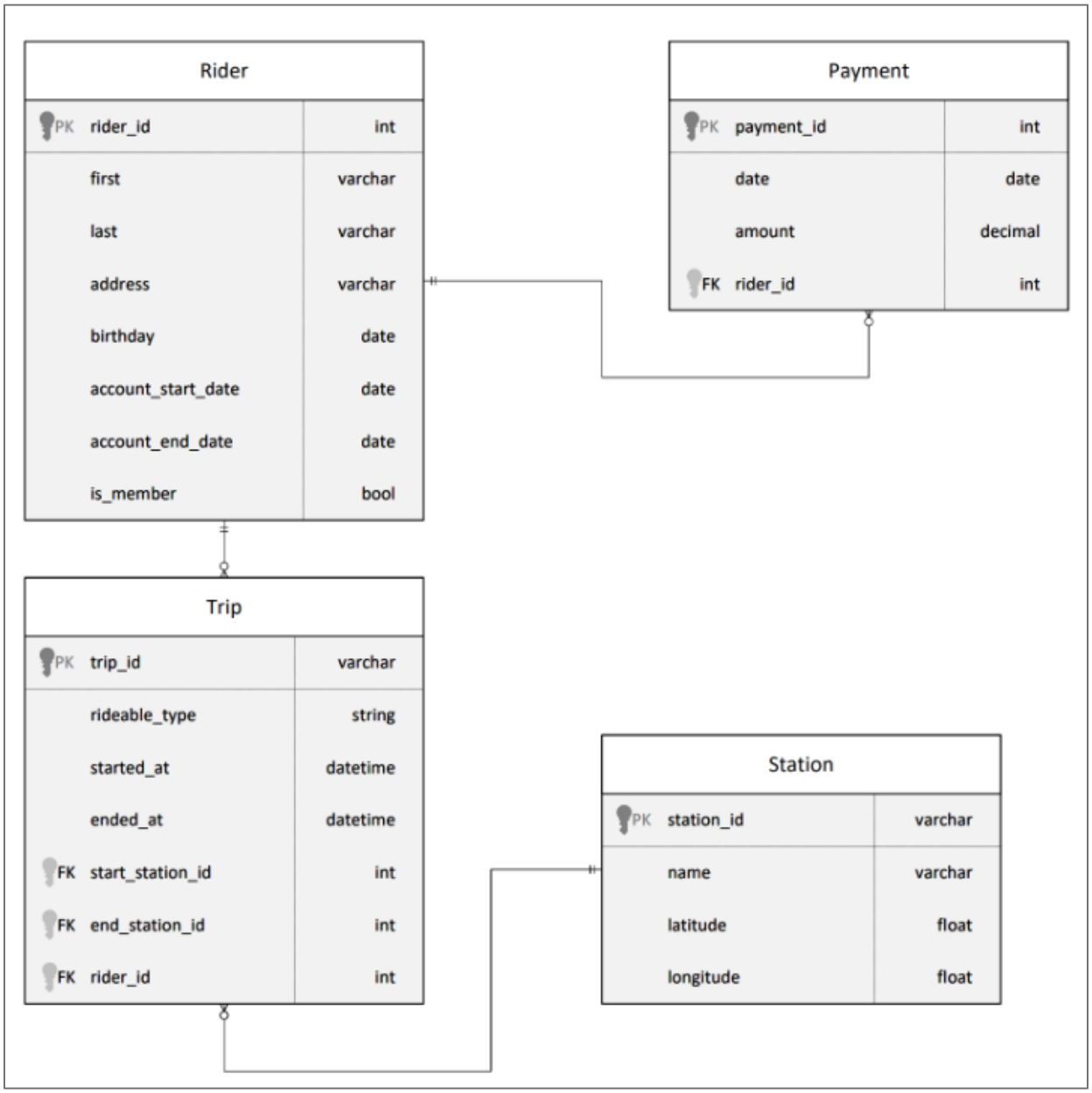
Building an Azure Data Lake for Bike Share Data Analytics

Divvy is a bike sharing program in Chicago, Illinois USA that allows riders to purchase a pass at a kiosk or use a mobile application to unlock a bike at stations around the city and use the bike for a specified amount of time. The bikes can be returned to the same station or to another station. The City of Chicago makes the anonymized bike trip data publicly available for projects like this where we can analyze the data. The dataset looks like this:



The goal of this project is to develop a data lake solution using Azure Databricks using a lake house architecture.

- Design a star schema based on the business outcomes listed below
- Import the data into Azure Databricks using Delta Lake to create a Bronze data store
- Create a gold data store in Delta Lake tables
- Transform the data into the star schema for a Gold data store

The business outcomes we are designing for are as follows:

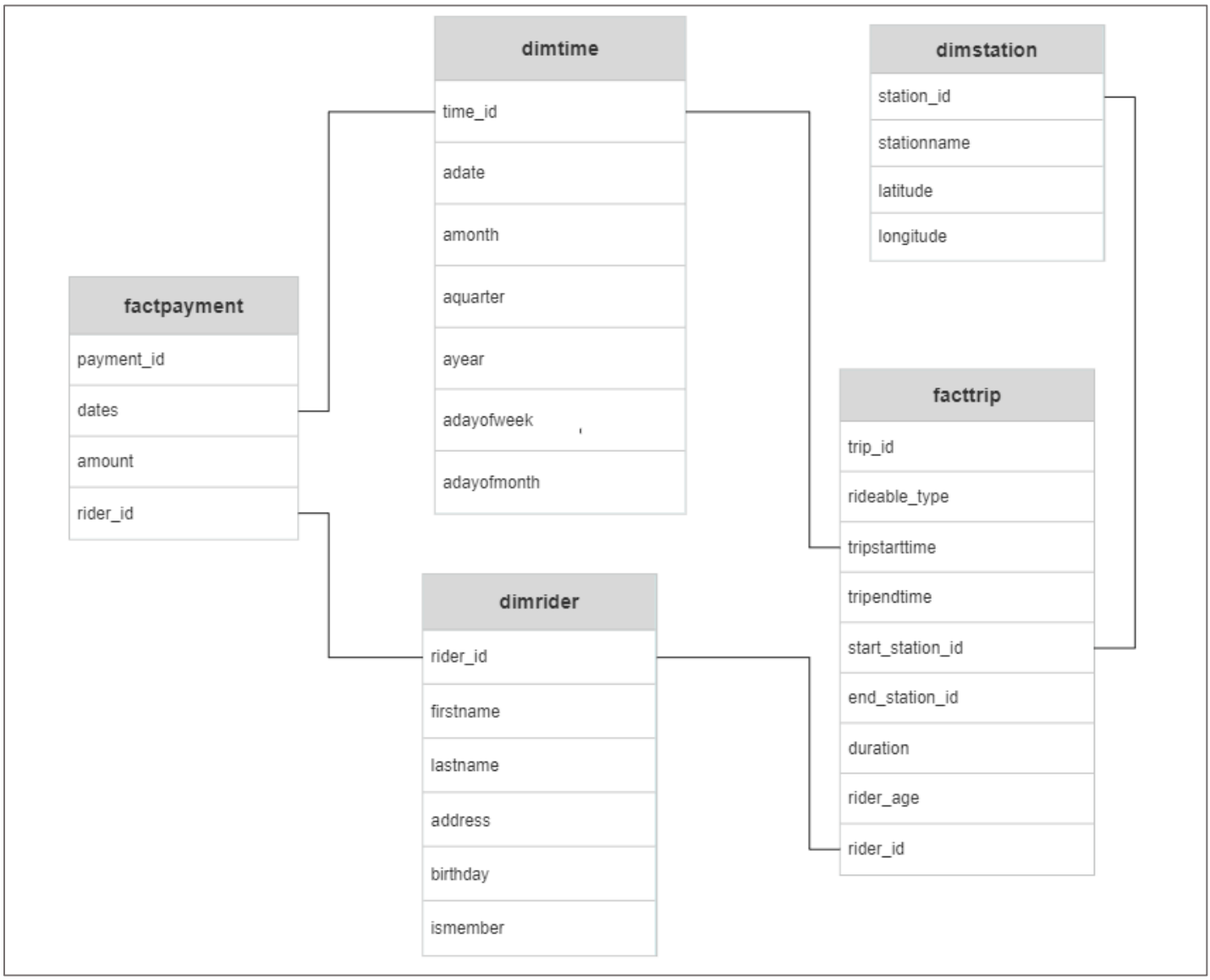
1. Analyze how much time is spent per ride
   - Based on date and time factors such as day of week and time of day
   - Based on which station is the starting and / or ending station
   - Based on age of the rider at time of the ride
   - Based on whether the rider is a member or a casual rider

2. Analyze how much money is spent
   - Per month, quarter, year
   - Per member, based on the age of the rider at account start

3. Analyze how much money is spent per member
   - Based on how many rides the rider averages per month
   - Based on how many minutes the rider spends on a bike per month

Task 1: Design a star schema

Designing a Star Schema based on the relation diagram and the business problems outlined above:



Star Schema design comprising 2 fact tables, and 3 dimension tables:

FACT TABLES:

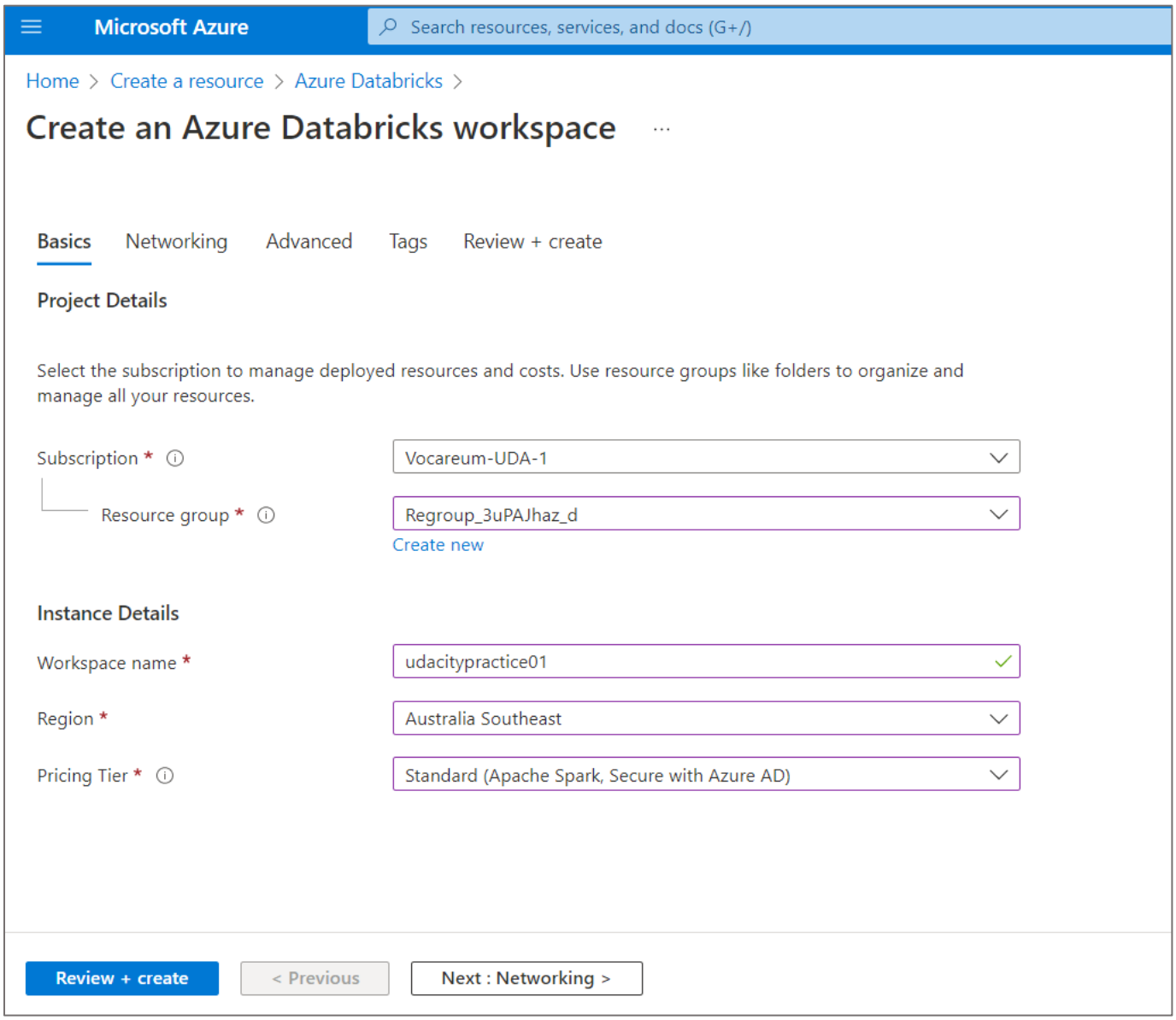
- facttrip
- factpayment

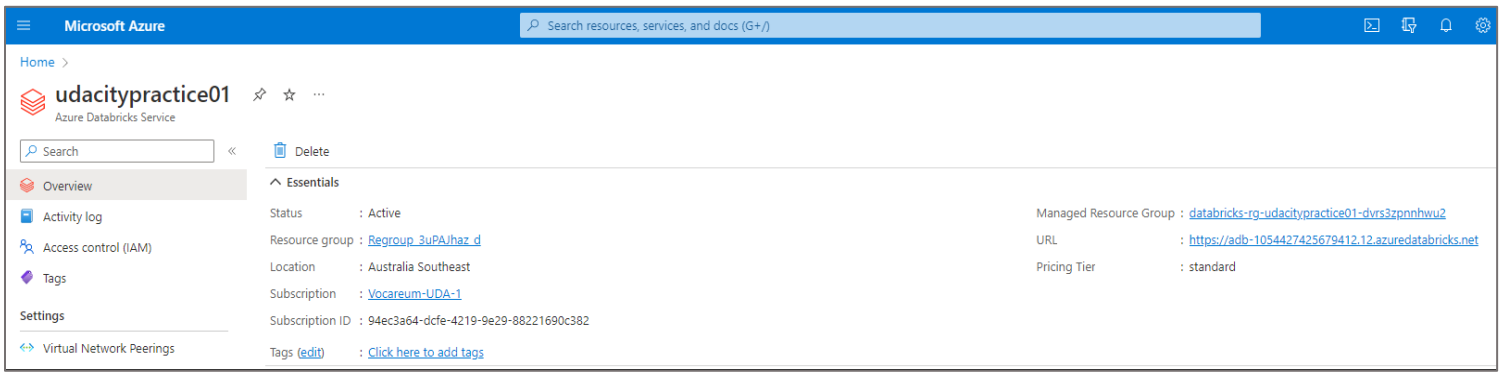
DIMENSION TABLES:

- dimrider
- dimtime
- dimstation

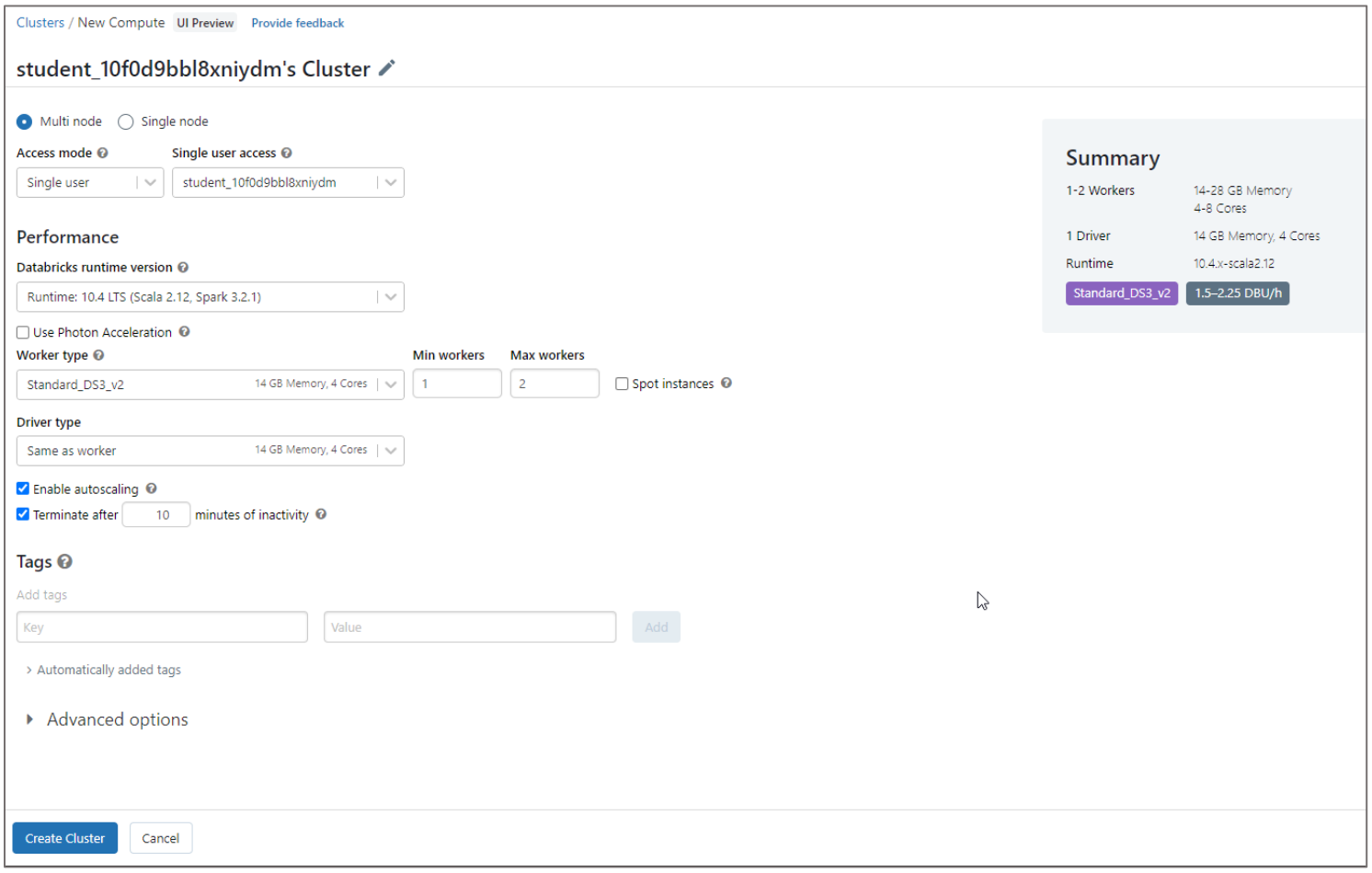
Task 2: Create Resources

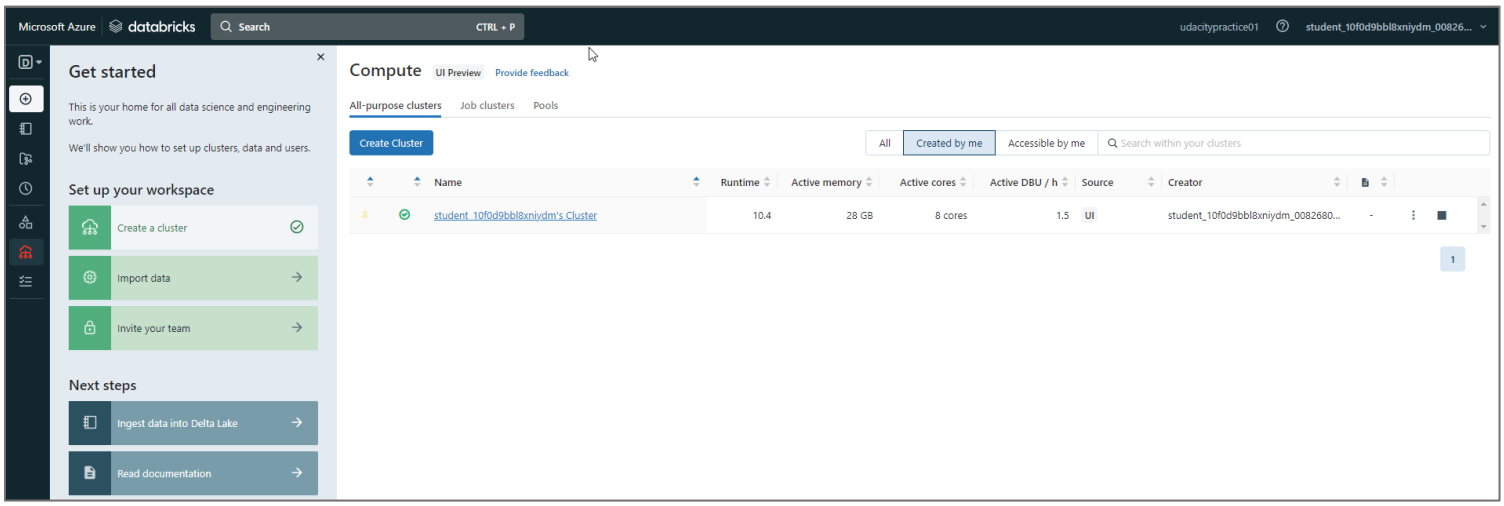
1. Azure Databricks Workspace





2. Create a Spark Cluster in the Databricks Workspace

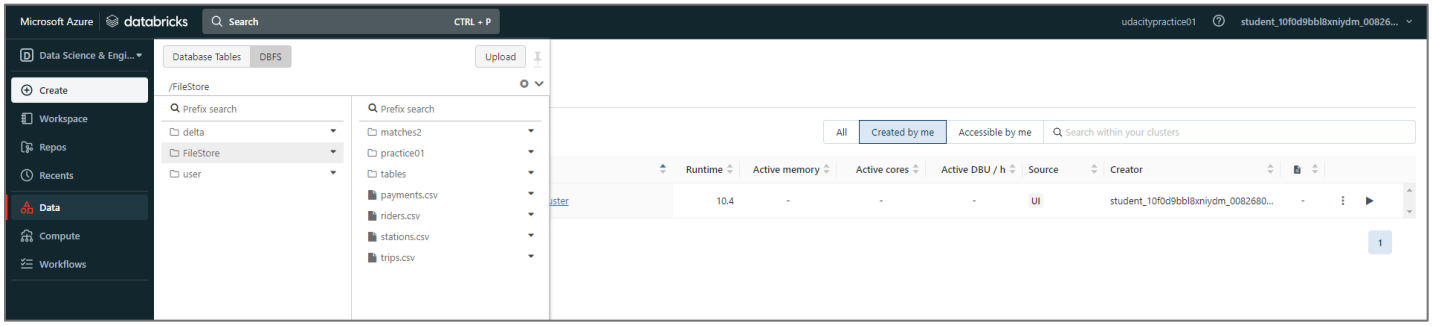




Task 3: Extract step

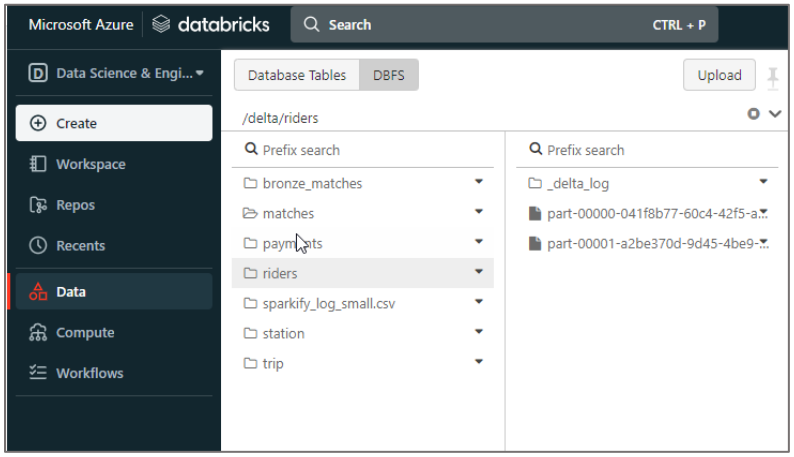
1. Uploaded the 4 csv files to the DBFS Filestore manually

payments
riders
stations
trips

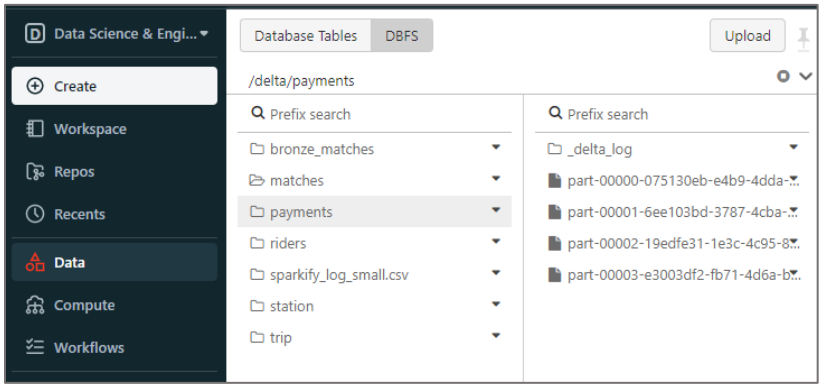


2. Using the Extract Python Notebook, Extracted the data to the DBFS DELTA Storage

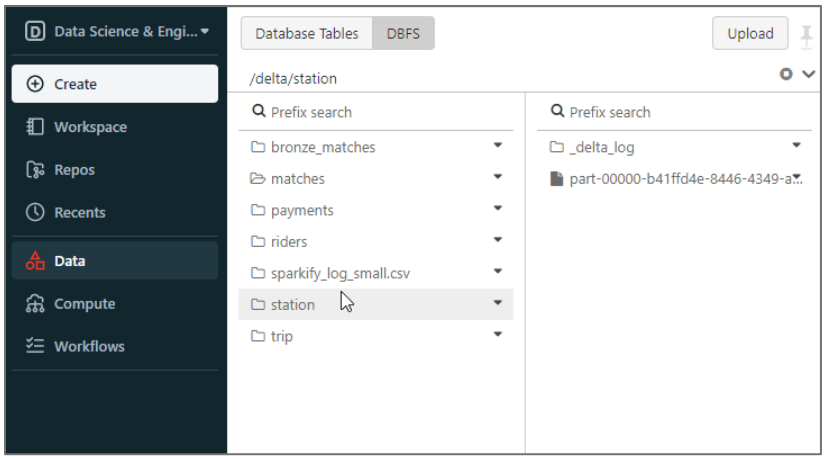
Riders



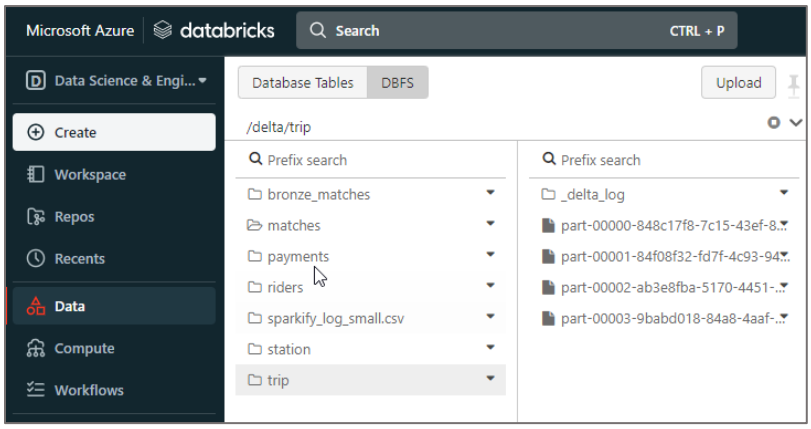
Payments



Station



Trip



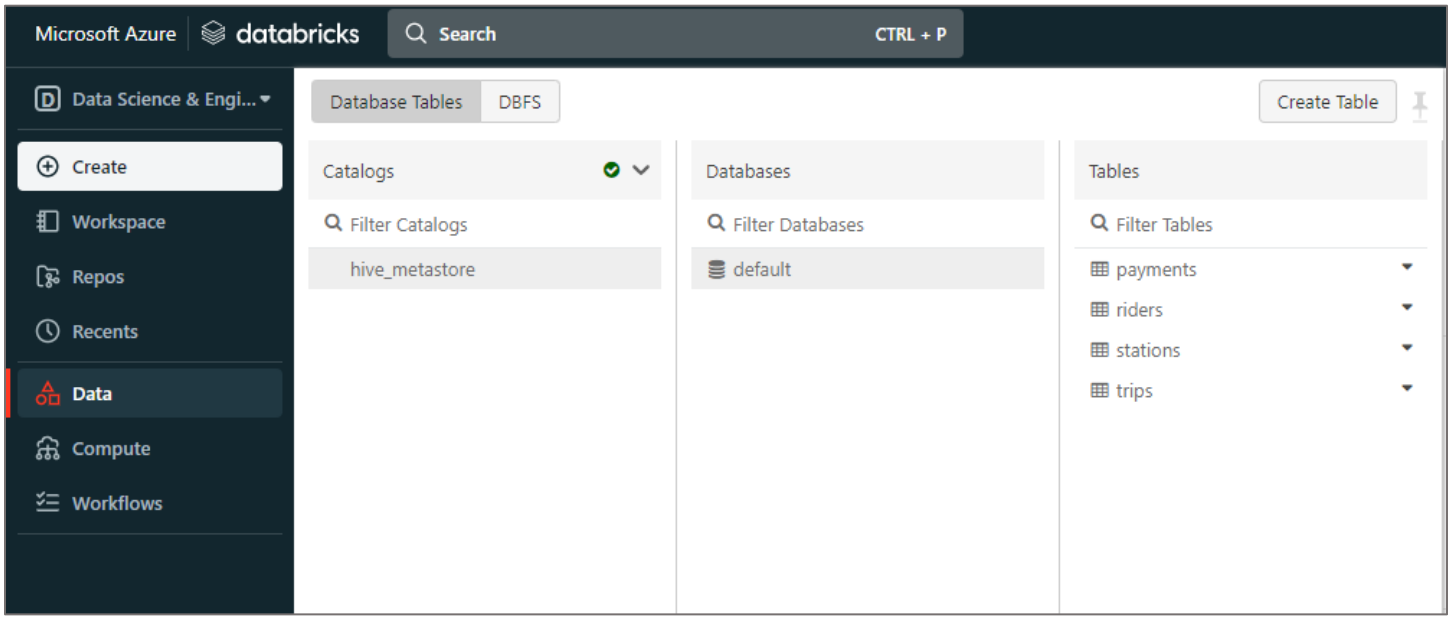
Task 4: Load

-Complete Extraction by loading data from DBFS Delta to DBFS Databasetables

Cmd 6

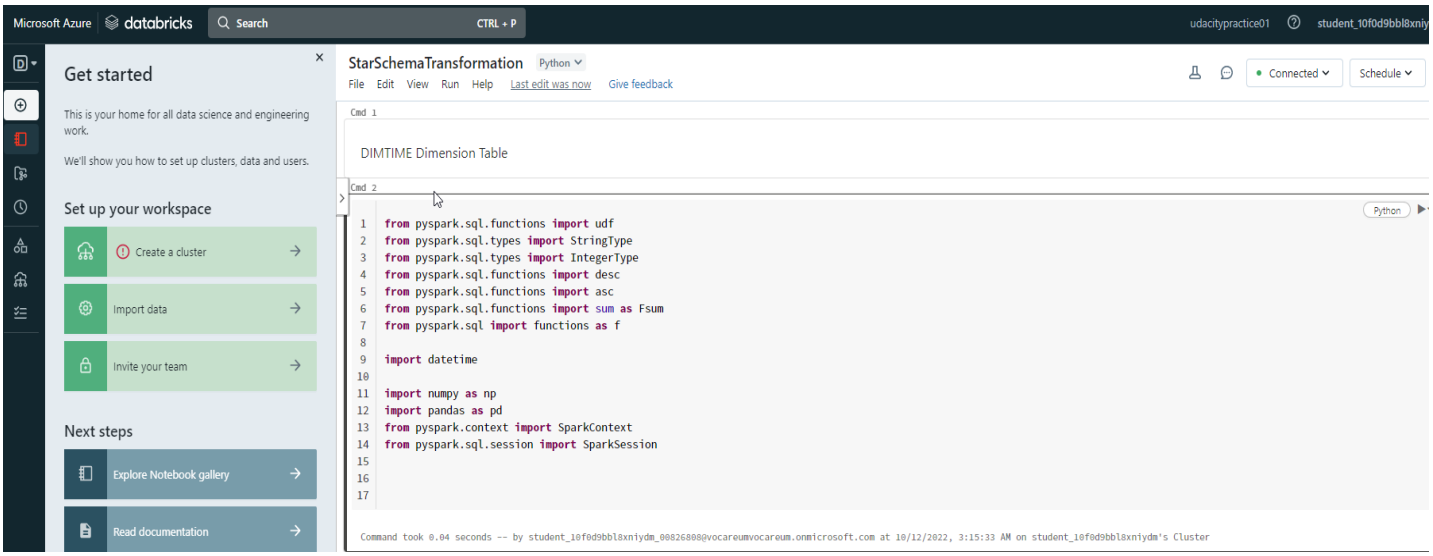
```
1  spark.sql("CREATE TABLE payments USING DELTA LOCATION '/delta/payments'")
2  spark.sql("CREATE TABLE riders USING DELTA LOCATION '/delta/riders'")
3  spark.sql("CREATE TABLE stations USING DELTA LOCATION '/delta/station'")
4  spark.sql("CREATE TABLE trips USING DELTA LOCATION '/delta/trip'")
```

Task 5: Transform



Using the Extract Python Notebook - StarSchemaTransformation, transformed from DBFS DELTA Storage to DBFS Database Tables

Using below dependencies created the fact and dimension tables



DIMTIME table:

Create dimtime table using dimtime dataframe in the databricks delta storage

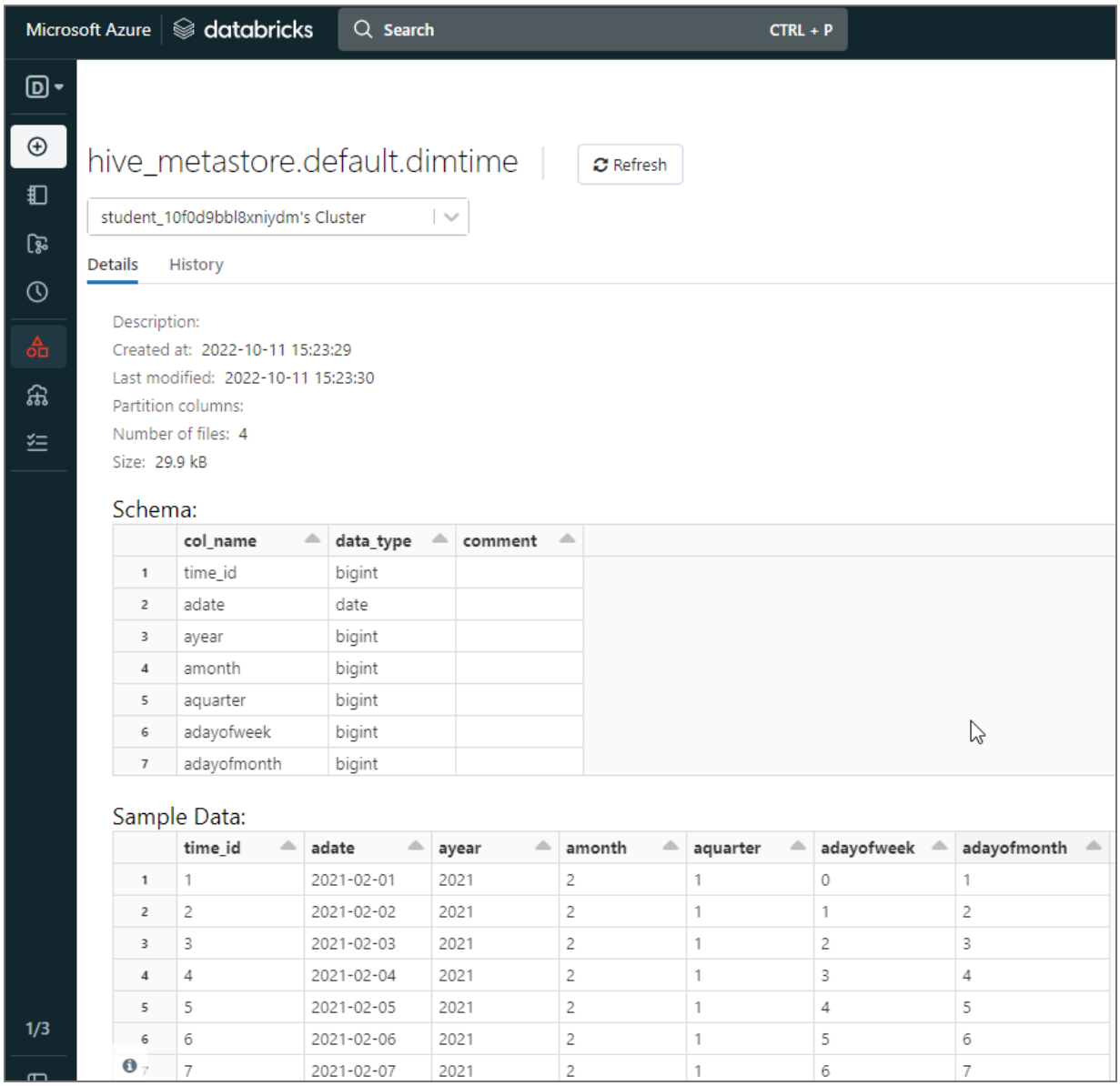
Cmd 7

```
1  dimtime.write.format("delta").mode("overwrite").saveAsTable("dimtime")
```

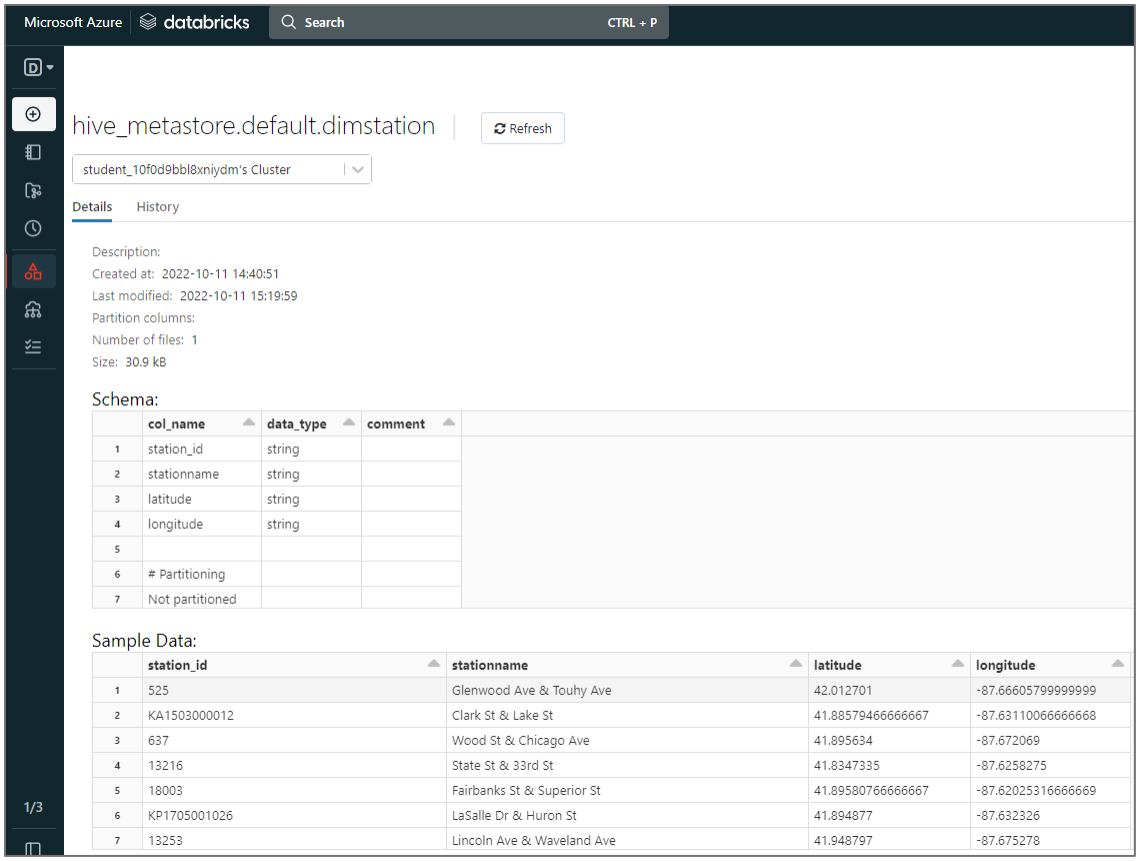
(4) Spark Jobs

- Job 11 View (Stages: 1/1)
- Job 13 View (Stages: 1/1)
- Job 14 View (Stages: 1/1, 1 skipped)
- Job 15 View (Stages: 1/1, 2 skipped)

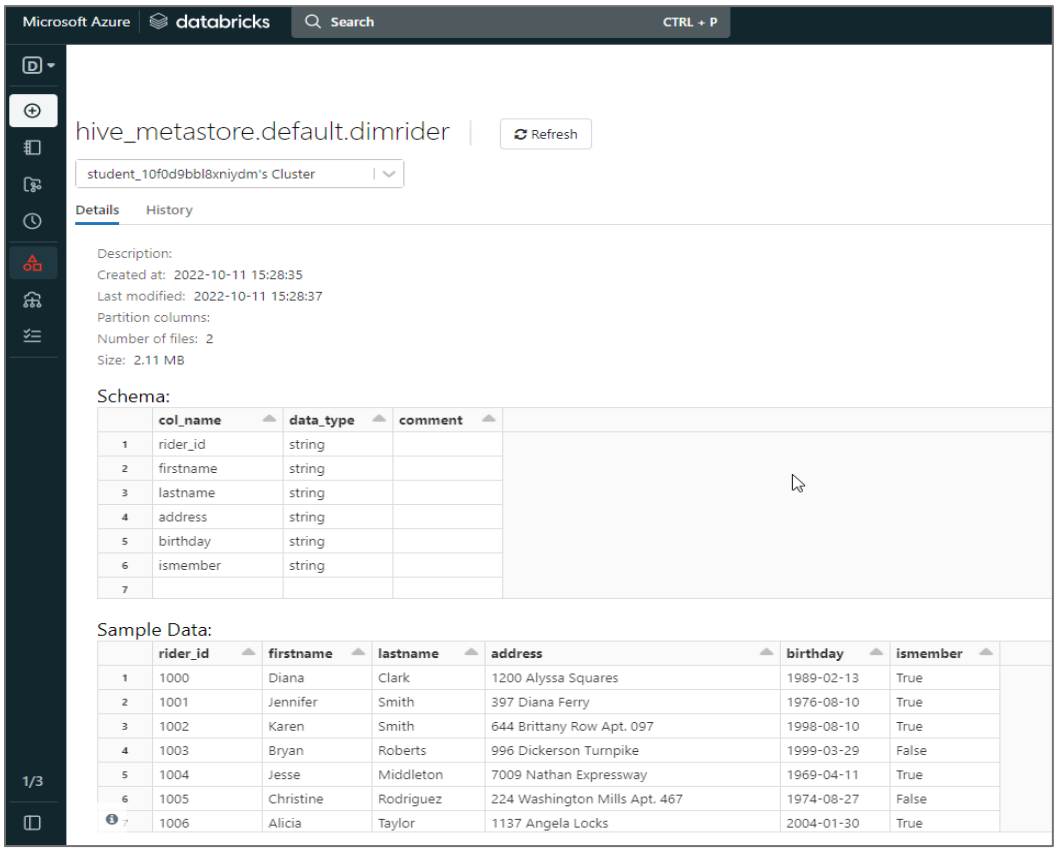
Command took 3.74 seconds -- by student_10f0d9bbl8xniydm_00826808@vocareumvocareum.onmicrosoft.com at 10/12/2022, 4:23:28 AM on student_10f0d9bbl8xniydm's Cluster



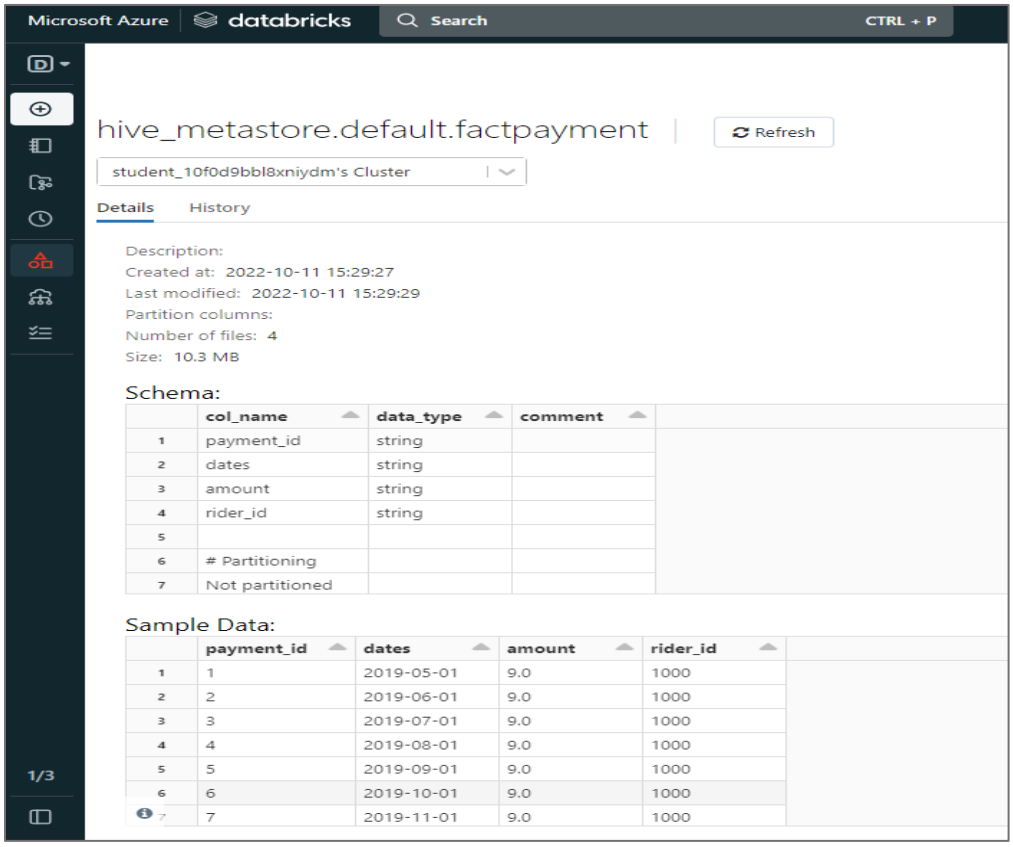
DIMSTATION table:



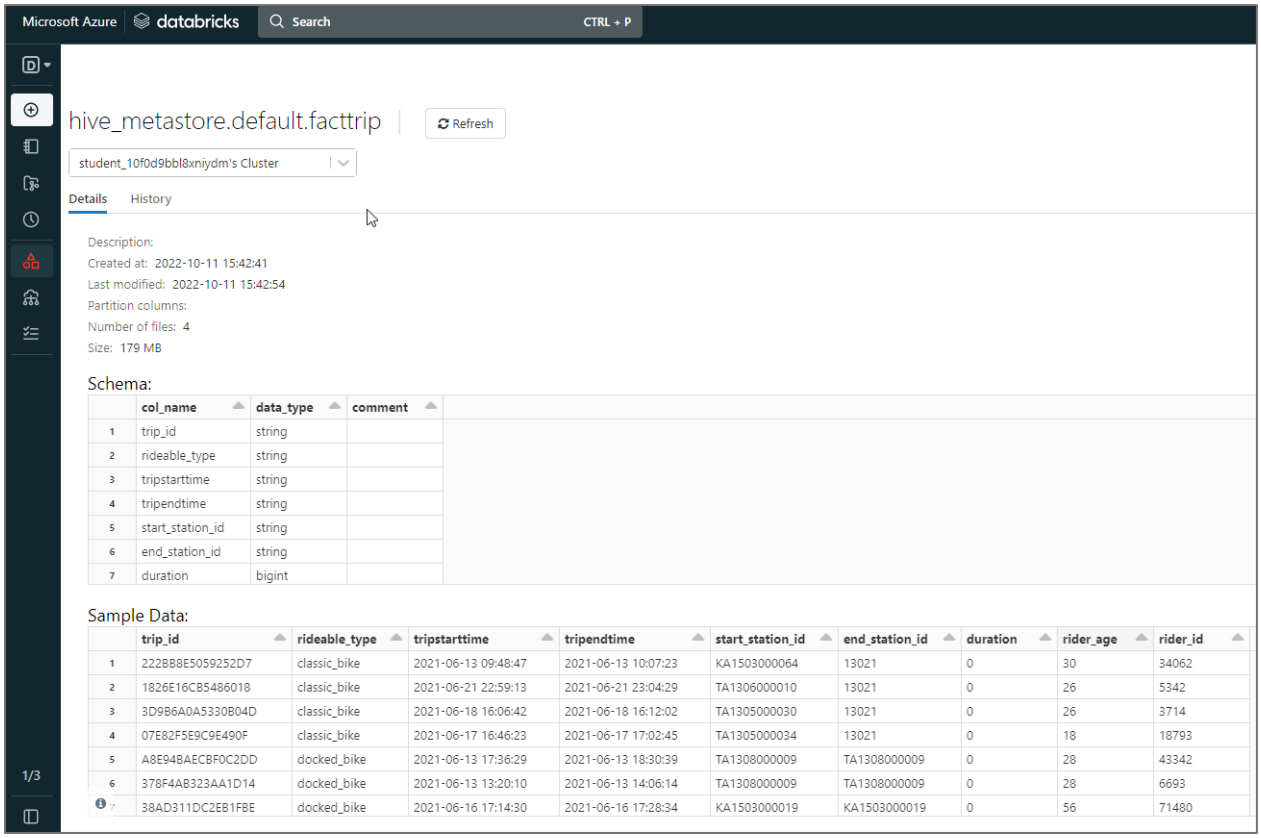
DIMRIDER table:



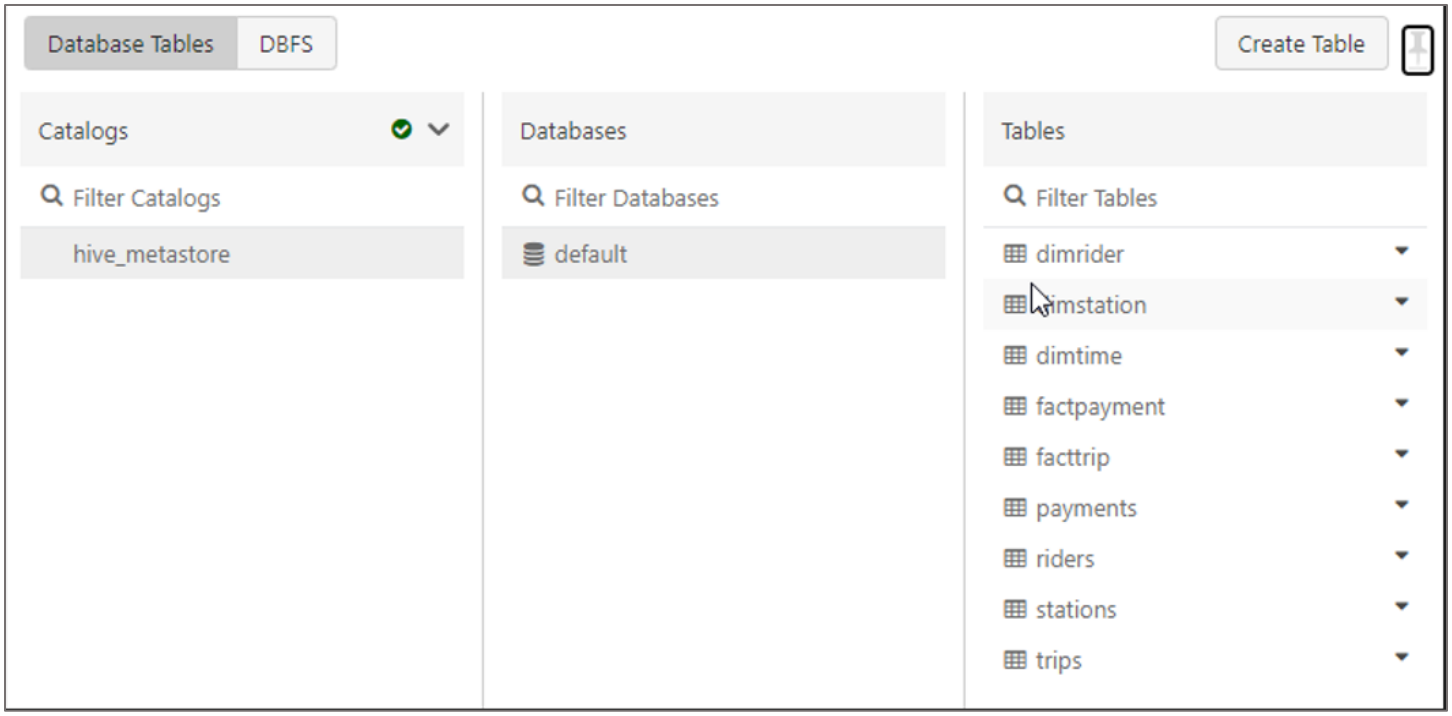
FACTPAYMENT table:



FACTTRIP table:



All Fact and Dimension tables have been Created



NOTEBOOKS:

1. ExtractNotebook
2. StarSchemaTransformation