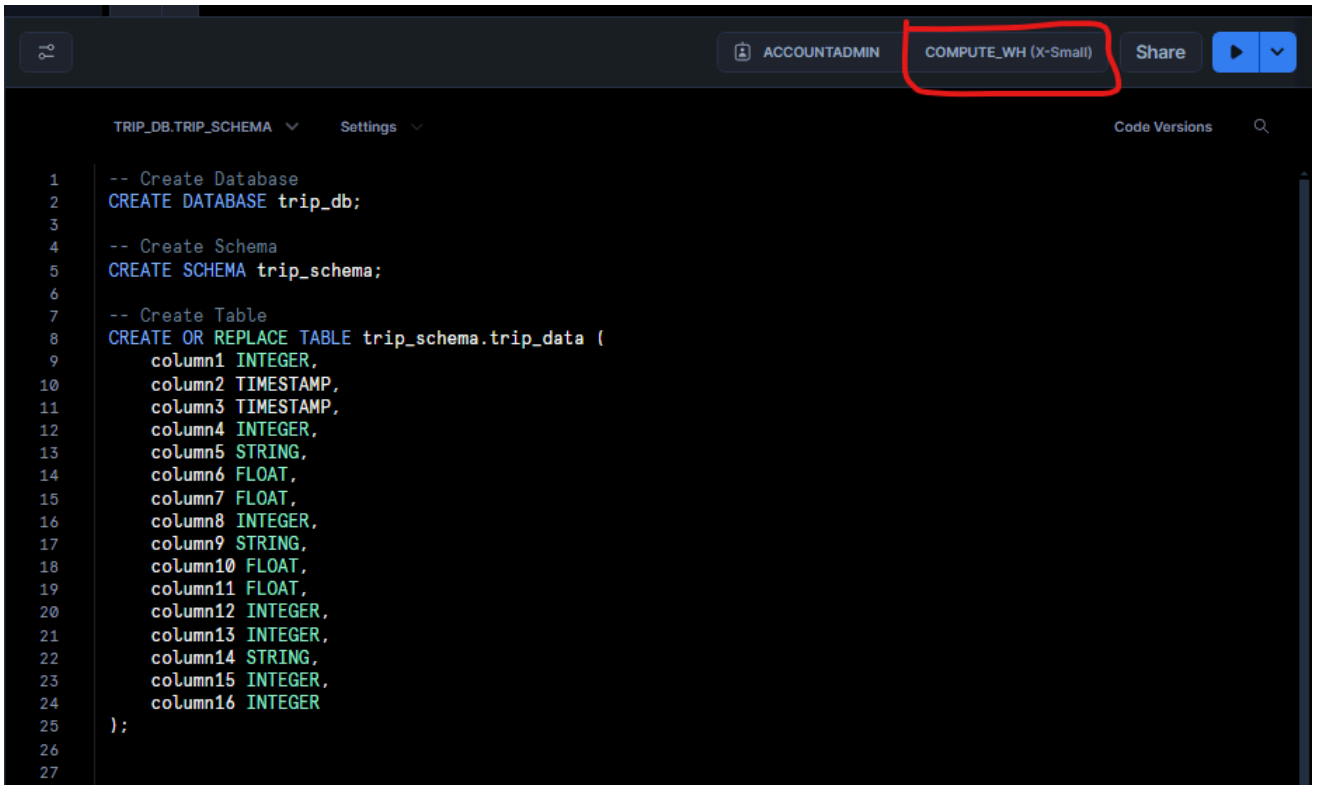
Homework 2

1. (+2) Create Database 'trip_db', Schema 'trip_schema' and Table 'trip_data' in Snowflake. Make sure the schema of trip_data should matches the structure of the file. Use the warehouse of size **XSMALL**




```
-- Create Database
CREATE DATABASE trip_db;

-- Create Schema
CREATE SCHEMA trip_schema;

-- Create Table
CREATE OR REPLACE TABLE trip_schema.trip_data (
    column1 INTEGER,
    column2 TIMESTAMP,
    column3 TIMESTAMP,
    column4 INTEGER,
    column5 STRING,
    column6 FLOAT,
    column7 FLOAT,
    column8 INTEGER,
    column9 STRING,
    column10 FLOAT,
    column11 FLOAT,
    column12 INTEGER,
    column13 INTEGER,
    column14 STRING,
    column15 INTEGER,
    column16 INTEGER
);
```

- Creating database trip_db and schema trip_schema is pretty straight forward, I took a slightly different approach inorder to match the schema to the given S3 database. First using the link I printed out the first 10 rows of the database in snowflake, and using the first row, I concurred that there are 16 columns with different data types which I matched with the columns while creating the table as seen in the screenshot.

2. (+1) Create Stage 'trip_stage' in Snowflake and it can be any stage type of your choice (internal or external)

```
CREATE OR REPLACE FILE FORMAT my_csv_format
TYPE = 'CSV'
FIELD_OPTIONALLY_ENCLOSED_BY = '"'
SKIP_HEADER = 1
NULL_IF = ('', 'null');
```

3. (+1) Create File Format 'trip_file_format' in Snowflake.

```
CREATE OR REPLACE STAGE citibike_stage
URL = 's3://snowflake-workshop-lab/citibike-trips-csv/'
FILE_FORMAT = my_csv_format;
```

4. (+1) Load Data into the 'trip_data' table (COPY INTO)

```
COPY INTO trip_schema.trip_data
FROM @citibike_stage/trips_2013_0_5_0.csv.gz
FILE_FORMAT = (TYPE = 'CSV', FIELD_OPTIONALLY_ENCLOSED_BY = '"', SKIP_HEADER = 1, NULL_IF = ('', 'null'))
ON_ERROR = 'CONTINUE';
```

Results | Chart

	COLUMN1	COLUMN2	COLUMN3	COLUMN4	COLUMN5
1	1717	2013-06-28 18:08:27.000	2013-06-28 18:37:04.000	349	Rivington St & Ridge St
2	838	2013-06-28 18:08:27.000	2013-06-28 18:22:25.000	2012	E 27 St & 1 Ave
3	972	2013-06-28 18:08:30.000	2013-06-28 18:24:42.000	358	Christopher St & Greenwicl
4	602	2013-06-28 18:08:30.000	2013-06-28 18:18:32.000	540	Lexington Ave & E 29 St
5	542	2013-06-28 18:08:31.000	2013-06-28 18:17:33.000	521	8 Ave & W 31 St N
6	1344	2013-06-28 18:08:32.000	2013-06-28 18:30:56.000	477	W 41 St & 8 Ave
7	443	2013-06-28 18:08:34.000	2013-06-28 18:15:57.000	515	W 43 St & 10 Ave
8	903	2013-06-28 18:08:35.000	2013-06-28 18:23:38.000	400	Pitt St & Stanton St
9	525	2013-06-28 18:08:37.000	2013-06-28 18:17:22.000	484	W 44 St & 5 Ave
10	1176	2013-06-28 18:08:39.000	2013-06-28 18:28:15.000	454	E 51 St & 1 Ave

- As you can see the data has been loaded using COPY INTO

5. (+4) Write an SQL query to produce a report that shows, for each hour of the day, the following:
    1. The total number of trips that started during that hour.
    2. The average duration of these trips in mins.
    3. The average distance traveled during these trips in kms.

Results | Chart

	HOUR_OF_DAY	TOTAL_TRIPS	AVG_DURATION_MINS	AVG_DISTANCE_KMS
1	0	1900	422.284211	40.731788075
2	1	1050	423.681905	40.731909883
3	2	699	405.340486	40.732597666
4	3	453	419.456954	40.732713679
5	4	297	414.909091	40.735083739
6	5	470	426.714894	40.734007014
7	6	1671	441.868941	40.73604748
8	7	3507	435.335044	40.73630456
9	8	5332	424.010690	40.733586983
10	9	4066	419.421545	40.732889609
11	10	3797	432.680801	40.733275058
12	11	4502	431.010440	40.732353958
13	12	5673	431.122158	40.732453917
14	13	6014	431.758730	40.73171905
15	14	6424	437.827833	40.731967256
16	15	5655	430.698674	40.732154404
17	16	6270	434.710686	40.733400151
18	17	8861	434.075612	40.734096079
19	18	11363	423.358004	40.733277689
20	19	9291	425.566139	40.732600832
21	20	6036	422.352386	40.731374105
22	21	4573	421.676580	40.732084337
23	22	3714	421.341949	40.732306824
24	23	2738	416.083272	40.732084247

Query Details

Query duration 428ms

Rows 24

Query ID 01b70192-0002-e7a5-0...

Show more

HOUR_OF_DAY
0 23

TOTAL_TRIPS
297 11363

AVG_DURATION_MINS
405.340486 441.868941

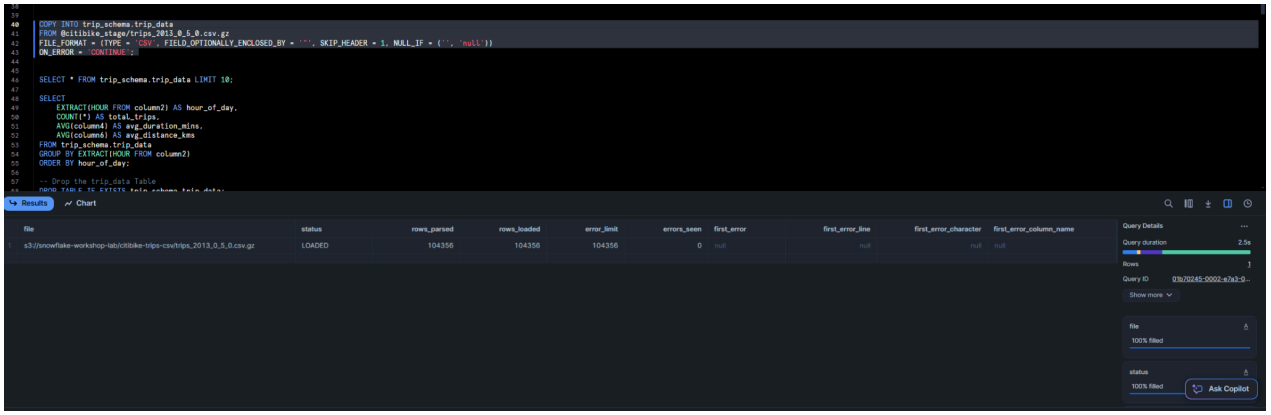
AVG_DISTANCE_KMS
40.731374105 40.73630456

6. (+1) Create a virtual warehouse of XLARGE.

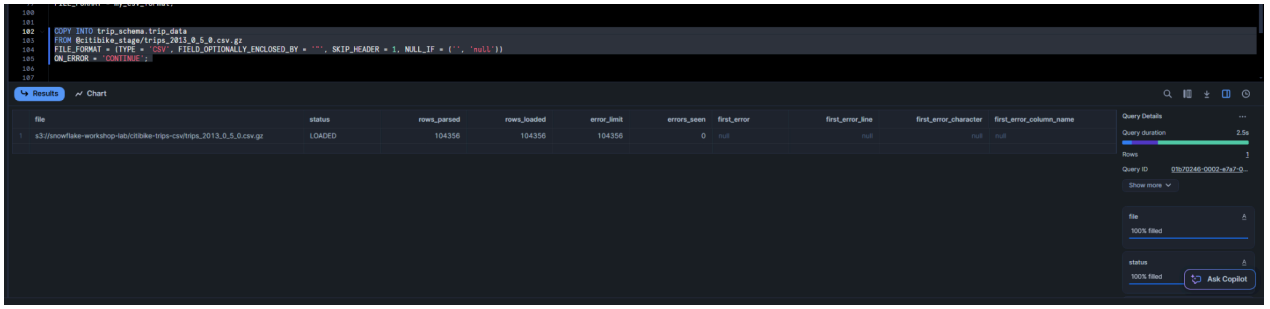
```sql
CREATE OR REPLACE WAREHOUSE trip_warehouse
WITH
    WAREHOUSE_SIZE = 'XLARGE'
    AUTO_SUSPEND = 600
    AUTO_RESUME = TRUE;


USE WAREHOUSE trip_warehouse;
```

7. (+3) Rerun the same query from step 4 & 5 using the XLARGE warehouse **after dropping trip_data table**, Analyze the performance of the node upon changing configurations from x-small to x-large in Snowflake.

x-small:



x-large:



ACCOUNTADMIN | TRIP_WAREHOUSE (X-Large) | Share

TRIP_DB.TRIP_SCHEMA | Settings | Code Versions

```sql
CREATE OR REPLACE STAGE citibike_stage
URL = 's3://snowflake-workshop-lab/citibike-trips-csv/'
FILE_FORMAT = my_csv_format;


COPY INTO trip_schema.trip_data
FROM @citibike_stage/trips_2013_0_5_0.csv.gz
FILE_FORMAT = (TYPE = 'CSV', FIELD_OPTIONALLY_ENCLOSED_BY = '"', SKIP_HEADER = 1, NULL_IF = ('', 'null'))
ON_ERROR = 'CONTINUE';


SELECT * FROM trip_schema.trip_data LIMIT 10;

SELECT
    EXTRACT(HOUR FROM column2) AS hour_of_day,
    COUNT(*) AS total_trips,
    AVG(column4) AS avg_duration_mins,
    AVG(column6) AS avg_distance_kms
FROM trip_schema.trip_data
GROUP BY EXTRACT(HOUR FROM column2)
ORDER BY hour_of_day;


CREATE OR REPLACE WAREHOUSE trip_warehouse
WITH
    WAREHOUSE_SIZE = 'XLARGE'
    AUTO_SUSPEND = 600
    AUTO_RESUME = TRUE;


USE WAREHOUSE trip_warehouse;


SELECT
    EXTRACT(HOUR FROM column2) AS hour_of_day,
    COUNT(*) AS total_trips,
    AVG(column4) AS avg_duration_mins,
    AVG(column6) AS avg_distance_kms
FROM trip_schema.trip_data
GROUP BY EXTRACT(HOUR FROM column2)
ORDER BY hour_of_day;
```

Results | Chart

	HOUR_OF_DAY	TOTAL_TRIPS	AVG_DURATION_MINS	AVG_DISTANCE_KMS
1	0	1900	422.284211	40.731788075
2	1	1050	423.681905	40.731909883
3	2	699	405.340486	40.732597666
4	3	453	419.456954	40.732713679
5	4	297	414.909091	40.735083739
6	5	470	426.714894	40.734007014
7	6	1671	441.868941	40.73604748
8	7	3507	435.335044	40.73630456
9	8	5332	424.010690	40.733586983
10	9	4066	419.421545	40.732889609
11	10	3797	432.680801	40.733275058

Query Details

Query duration 81ms

Rows 24

Query ID 01b701c2-0002-e7a9-0...

Show more

HOUR_OF_DAY

0 23

TOTAL_TRIPS

Ask Copilot

We can see that x-small took 214ms to load the data into the schema whereas after dropping the table and loading the data in x-large took half as much time.
We can see that on the x-small warehouse the query took 428ms to run whereas in the x-large warehouse the same query took just 81ms which is 5x faster.