

Snowflake Introduction

Summary of the tutorial

Load structured .csv data from rider transactions into Snowflake. Work on open-source, semi-structured JSON weather data to determine if there is any correlation between the number of bike rides and the weather.

Preparing to Load Data

Create a Database and Table

Created a database called Citibike

The screenshot shows the Snowflake web interface. At the top, there is a navigation bar with icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets, and History. On the right, it shows the user's name 'SIDDHIP' and 'SYSADMIN'. Below the navigation bar, the main area is titled 'Databases' and shows a list of existing databases: 'SNOWFLAKE_SAMPLE_DATA', 'SFC_SAMPLES.SA...', 'DEMO_DB', and 'UTIL_DB'. A 'Create Database' dialog box is open in the center. It has fields for 'Name*' containing 'Citibike', 'Comment' (empty), and a 'Show SQL' button. There are 'Cancel' and 'Finish' buttons at the bottom right of the dialog. The status bar at the bottom right indicates 'Last refreshed 11:02:21 PM'.

Context settings will help to determine what elements the user can see and run from each worksheet.

The screenshot shows the Snowflake interface with the 'Worksheets' tab selected. A context menu is open over a block of SQL code, displaying options for Role (SYSADMIN), Warehouse (COMPUTE_WH (XS) Suspended), Database (Select Database), and Schema (Schema). The SQL code is a comment block for a lab guide, mentioning 'COMPUTE_WH (XS)' and 'X-Small Resize'.

Go to Worksheet Tab -> Run Create table SQL -> Creates Trip table to store comma delimited data.

The screenshot shows the Snowflake interface with the 'Worksheets' tab selected. A confirmation dialog box is displayed, asking 'Do you want to run the following queries?' It contains the SQL code for creating the 'trips' table. Below the dialog are two buttons: 'Cancel' and 'Run'. The background shows the same context menu as the previous screenshot, indicating the user has permissions to run the query.

Table Trips was successfully created.

The screenshot shows the Snowflake Worksheet interface. At the top, there are navigation icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (which is selected), and History. On the right, there are links for Preview App, Partner Connect, Help, and a user account labeled SIDDHIP SYSADMIN. The main area has tabs for New Worksheet and New Worksheet (with a checkmark). A search bar says "Find database objects Starting with...". Below it, a list of databases includes DEMO_DB, SNOWFLAKE_SAMPLE_DATA, and UTIL_DB. The central workspace contains the following SQL code:

```
13
14 create table trips
15   (tripduration integer,
16    starttime timestamp,
17    stoptime timestamp,
18    start_station_id integer,
19    start_station_name string,
20    start_station_latitude float,
21    start_station_longitude float,
22    end_station_id integer,
23    end_station_name string.
```

The status bar indicates "Run" and "All Queries | Saved 2 minutes ago". The results section shows a single row in a table:

Row	status
1	Table TRIPS successfully created.

There are buttons for Filter result..., Download, Copy, and Columns.

TRIPS table in CITIBIKE Database

Databases > CITIBIKE

Last refreshed 11:24:24 PM

Table Name	Schema	Creation Time	Owner	Rows	Size	Comment
TRIPS	PUBLIC	11:10:49 PM	SYSADMIN			

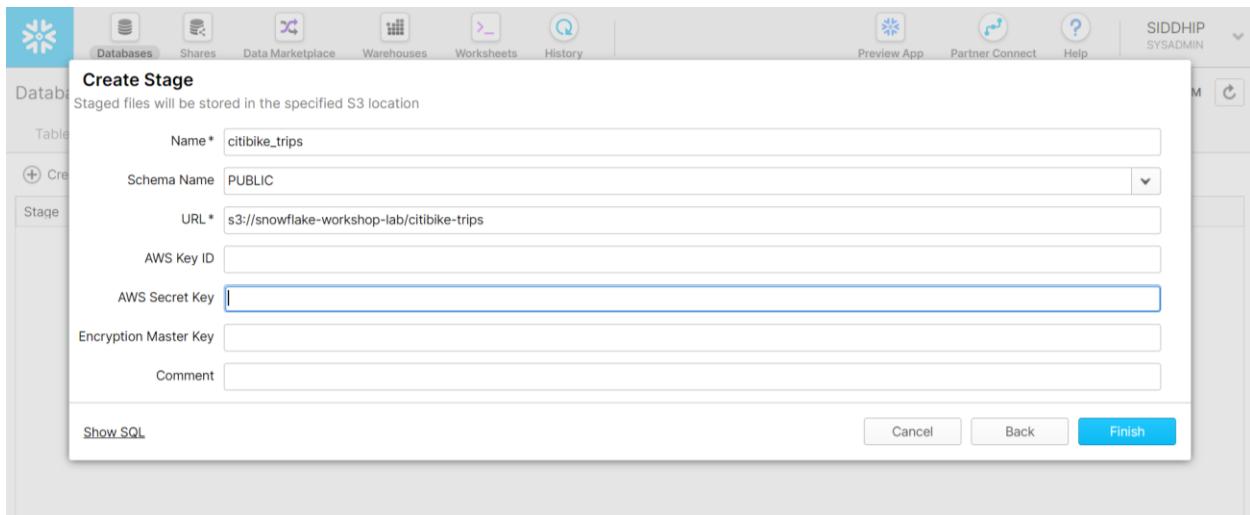
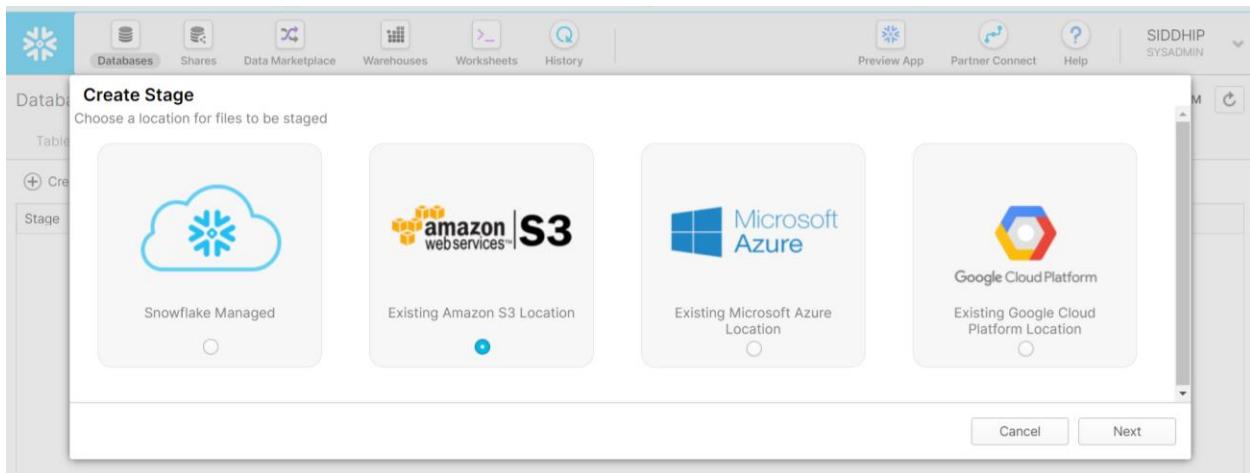
Databases > CITIBIKE > TRIPS (PUBLIC)

Column Name	Ordinal▲	Type	Nullable	Default	Comment
TRIPDURATION	1	NUMBER(38,0)	true	NULL	
STARTTIME	2	TIMESTAMP_NTZ(9)	true	NULL	
STOPTIME	3	TIMESTAMP_NTZ(9)	true	NULL	
START_STATION_ID	4	NUMBER(38,0)	true	NULL	
START_STATION_NAME	5	VARCHAR(16777216)	true	NULL	
START_STATION_LATITUDE	6	FLOAT	true	NULL	
START_STATION_LONGITUDE	7	FLOAT	true	NULL	
END_STATION_ID	8	NUMBER(38,0)	true	NULL	
END_STATION_NAME	9	VARCHAR(16777216)	true	NULL	
END_STATION_LATITUDE	10	FLOAT	true	NULL	

Create an external stage

Comma delimited file already staged in a public external S3 bucket. Create a stage to specify the location of our external bucket.

Go to Database -> Stage -> Create -> Select stage location as AWS S3 ->Click Next



Stage created

Databases > CITIBIKE

Tables Views Schemas **Stages** File Formats Sequences Pipes

+ Create... Clone... Edit... Drop... Transfer Ownership

Stage	Schema	Location	Creation Time	Owner	Comment
CITIBIKE_TRIPS	PUBLIC	s3://snowflake-workshop-lab/citibike-trips	11:40:56 PM	SYSADMIN	

View contents of the stage

New Worksheet New Worksheet +

Find database objects Starting with...

34 create or replace stage citibike_trips url = 's3://snowflake-workshop-lab/citibike-trips';
35 -- 3.2.4
36
37
38 list @citibike_trips;
39
40 -- 3.3
41
42 create or replace file format csv type='csv'

New Worksheet New Worksheet +

Find database objects Starting with...

Do you want to run the following queries?

```
1 list @citibike_trips;
```

Cancel Run

The screenshot shows the Snowflake Worksheet interface. At the top, there are navigation icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (which is selected), and History. On the right, there are links for Preview App, Partner Connect, Help, and a user account for SIDDHIP SYSADMIN.

In the main area, there are two tabs: "New Worksheet" and "New Worksheet". The "New Worksheet" tab is active, showing a query editor with the following SQL command:

```
list @citibike_trips;
```

The results of the query are displayed in a table titled "Results - Data Preview". The table has the following columns:

Row	name	size	md5	last_modified
1	s3://snowflake-workshop-lab/citib...	3072073	cfc69e04228a94d1337ab383a3af...	Wed, 12 Jun 2019 16:30:58 GMT
2	s3://snowflake-workshop-lab/citib...	2877852	92a1c064a3c632f338b57d5c6531...	Wed, 12 Jun 2019 16:30:58 GMT
3	s3://snowflake-workshop-lab/citib...	3174598	39faac098802cb29f2d4d99f313...	Wed, 12 Jun 2019 16:30:58 GMT
4	s3://snowflake-workshop-lab/citib...	3031012	cd0dcadcfca309c0bb4bd40d126...	Wed, 12 Jun 2019 16:30:58 GMT
5	s3://snowflake-workshop-lab/citib...	3005838	fb24c0cc5fb6ee54d2aa4d502657...	Wed, 12 Jun 2019 16:30:58 GMT
6	s3://snowflake-workshop-lab/citib...	3099881	441efeb06352c57a50c4f31afcccb...	Wed, 12 Jun 2019 16:30:58 GMT

Create a File Format

In order to load data in Snowflake database a file format is created that matches the data structure.

Go to database tab -> click on ‘CITIBIKE’ database -> Select file format and Create -> Enter the required details to create the file format -> Click on Finish.

The screenshot shows the Snowflake web interface. At the top, there are navigation icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets, and History. Below the header, the path 'Databases > CITIBIKE' is displayed. A navigation bar below the path includes links for Tables, Views, Schemas, Stages, File Formats (which is underlined), Sequences, and Pipes. Below this is a toolbar with buttons for Create..., Clone..., Edit..., Drop..., and Transfer Ownership. The main area is a table titled 'File Format' with columns for Name, Schema, Type, Creation Time, Owner, and Comment. There are no rows currently listed in the table.

The screenshot shows the 'Create File Format' dialog box. The 'Name*' field is set to 'CSV'. The 'Schema Name' is set to 'PUBLIC'. The 'Format Type' is set to 'CSV'. The 'Compression Method' is set to 'Auto'. The 'Column separator' is set to 'Comma'. The 'Row separator' is set to 'New Line'. The 'Header lines to skip' is set to '0'. The 'Field optionally enclosed by' is set to 'Double Quote'. The 'Null String' field is empty. A checkbox for 'Trim space before and after' is unchecked. At the bottom of the dialog are 'Show SQL', 'Cancel', and 'Finish' buttons. The background shows the same Snowflake interface as the previous screenshot, with the 'File Formats' tab selected in the CITIBIKE database.

Databases > CITIBIKE

Tables Views Schemas Stages File Formats Sequences Pipes

(+) Create... Clone... Edit... Drop... Transfer Ownership

File Format	Schema	Type	Creation Time	Owner	Comment
CSV	PUBLIC	CSV	2:46:32 PM	SYSADMIN	

Loading Data

Using datawarehouse and COPY command to load structured data into Snowflake table.

Resize and Use a Warehouse for Data Loading

Go to warehouse -> Configure->COMPUTE_WH

COMPUTE_WH will be used to load data from AWS S3 into Snowflake table.

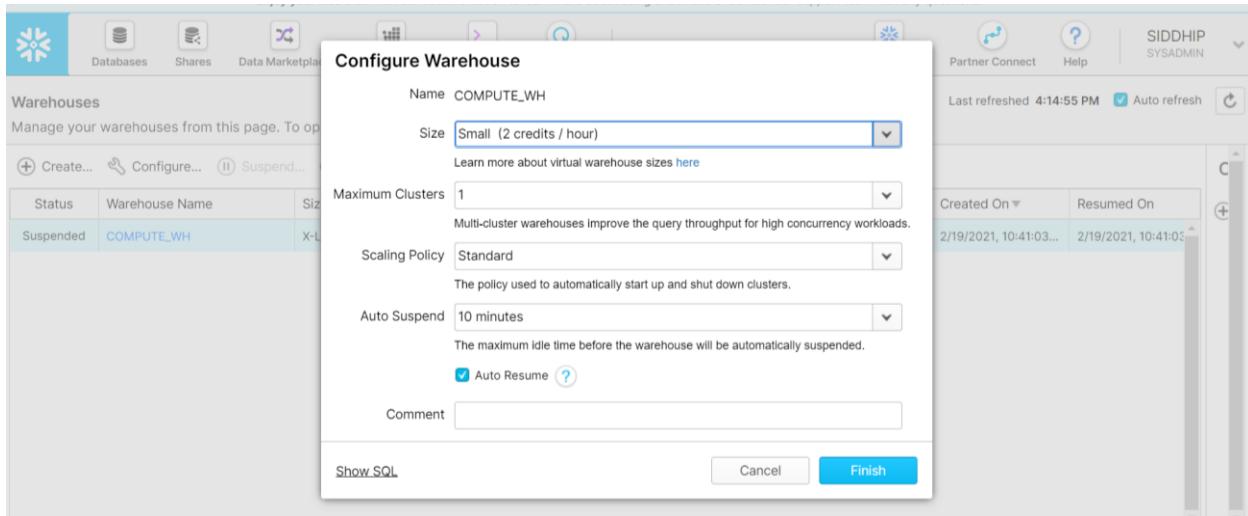
Warehouses

Manage your warehouses from this page. To operate on your data, you need to create one or more warehouses.

Last refreshed 3:35:38 PM Auto refresh

(+) Create... Configure... Suspend... Resume... Drop... Transfer Ownership

Status	Warehouse Name	Size	Clusters	Scaling Poli...	Runn...	Que...	Auto Suspe...	Auto Resume	Created On	Resumed On
Suspended	COMPUTE_WH	X-Large	min: 1, max: 1	Standard	0	0	10 minutes	Yes	2/19/2021, 10:41:03...	2/19/2021, 10:41:03



Load the Data

Copy command to load data into TRIPS table.

Worksheets -> check the context

```

1
2
3 -- This SQL file is for the Hands On Lab Guide for the 30-day fr
4 -- The numbers below correspond to the sections of the Lab Guide
5 -- Modules 1 and 2 of the Lab Guide have no SQL to execute
6
7
8 /* *****
9 /* *** MODULE 3 *****
10 /* *****
11
12 -- 3.1.4
13
14

```

Load staged data into table in worksheet tab

The screenshot shows the Snowflake interface with the 'Worksheets' tab selected. A modal dialog box is open, prompting the user with the message: "Do you want to run the following queries?". Inside the dialog, there is a single query listed:

```
1 copy into trips from @citibike_trips file_format=csv;
```

Below the dialog, there are two buttons: "Cancel" and "Run". On the left side of the main interface, there is a sidebar titled "Find database objects" with a search bar and a list of databases: CITIBIKE, DEMO_DB, SNOWFLAKE_SAMPLE_DATA, and UTIL_DB.

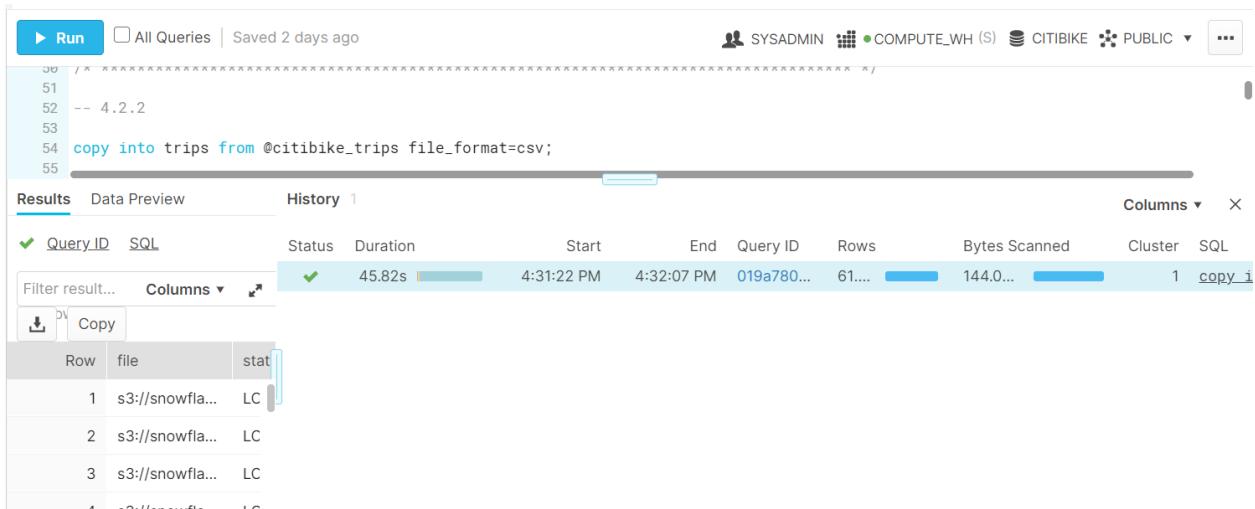
The screenshot shows the Snowflake interface with the 'Worksheets' tab selected. The main area displays the results of the executed query:

```
copy into trips from @citibike_trips file_format=csv;
```

The results are shown in a table titled "Data Preview". The table has the following columns:

Row	file	status	rows_parsed	rows_loaded	error_limit	errors_seen	first_error	first_error_line	first_error_char
1	s3://snowfla...	LOADED	89539	89539	1	0	NULL	NULL	NULL
2	s3://snowfla...	LOADED	114075	114075	1	0	NULL	NULL	NULL
3	s3://snowfla...	LOADED	115155	115155	1	0	NULL	NULL	NULL
4	s3://snowfla...	LOADED	117362	117362	1	0	NULL	NULL	NULL
5	s3://snowfla...	LOADED	144481	144481	1	0	NULL	NULL	NULL

Checking history of Snowflake operations performed in that worksheet



The screenshot shows a Snowflake Worksheet interface. At the top, there is a toolbar with a 'Run' button, a 'All Queries' dropdown, and a status message 'Saved 2 days ago'. On the right, there are user and role icons, cluster names (COMPUTE_WH, CITIBIKE), and a 'PUBLIC' role. Below the toolbar, the query history shows a single operation:

```

51
52 -- 4.2.2
53
54 copy into trips from @citibike_trips file_format=csv;
55

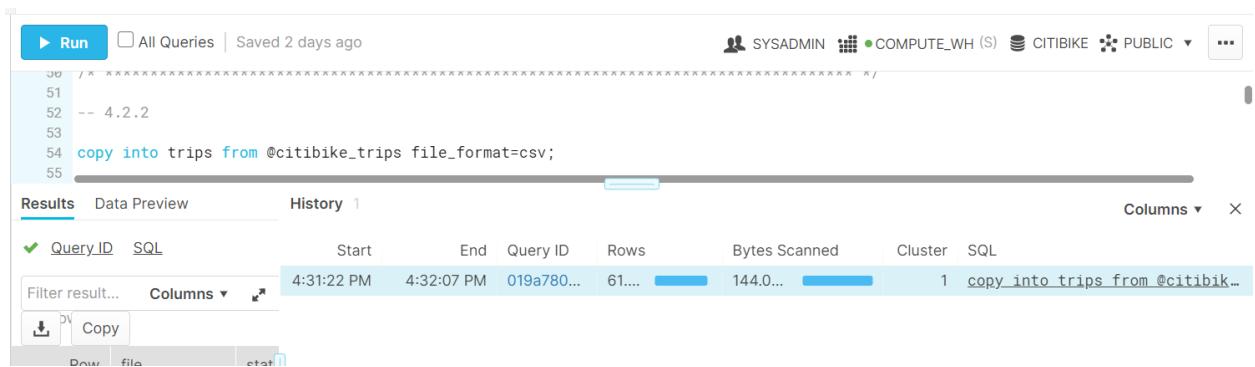
```

The 'History' tab is selected, displaying the following details:

Query ID	SQL	Status	Duration	Start	End	Query ID	Rows	Bytes Scanned	Cluster	SQL
019a780...	copy into trips from @citibike_trips file_format=csv;	✓	45.82s	4:31:22 PM	4:32:07 PM	019a780...	61...	144.0...	1	copy_...

Below the table, a data preview shows the copied data:

Row	file	stat
1	s3://snowfla...	LC
2	s3://snowfla...	LC
3	s3://snowfla...	LC
4	@citibike_trips	LC



This screenshot shows another view of the same or a very similar operation in the history:

```

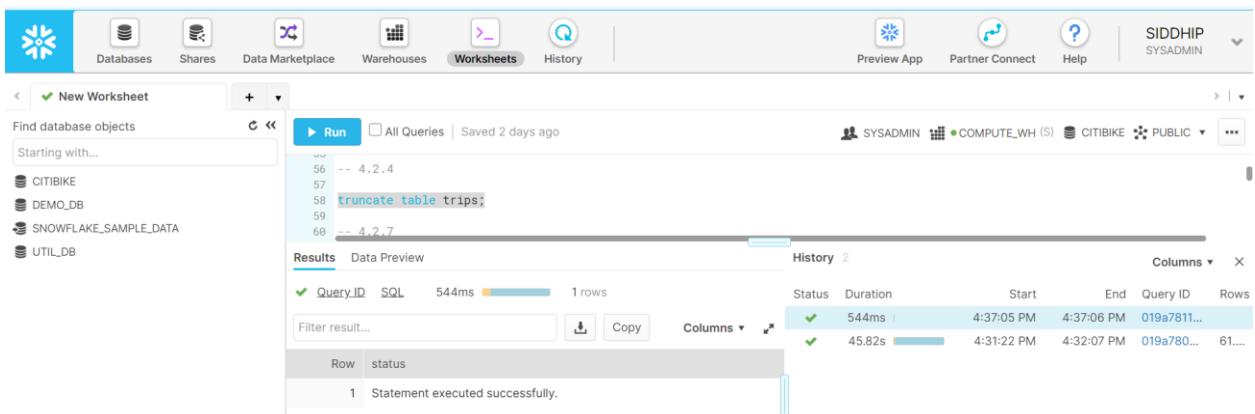
51
52 -- 4.2.2
53
54 copy into trips from @citibike_trips file_format=csv;
55

```

The 'History' tab is selected, showing the following details:

Start	End	Query ID	Rows	Bytes Scanned	Cluster	SQL
4:31:22 PM	4:32:07 PM	019a780...	61...	144.0...	1	copy into trips from @citibike_trips file_format=csv;

Clear the table of all data and metadata using TRUNCATE table command.



The screenshot shows the Snowflake UI with the 'Worksheets' tab selected. The left sidebar shows databases (CITIBIKE, DEMO_DB, SNOWFLAKE_SAMPLE_DATA, UTIL_DB) and a 'New Worksheet' button. The main area shows a query history and a results table.

The query history shows:

```

56 -- 4.2.4
57
58 truncate table trips;
59
60 -- 4.2.7

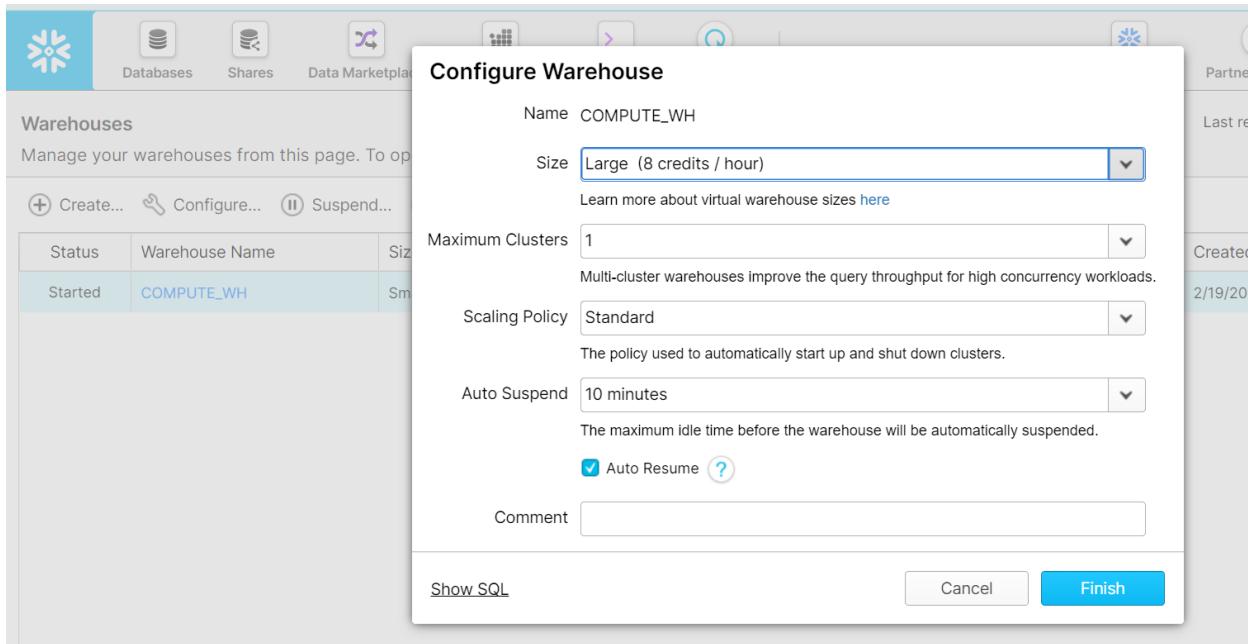
```

The results table shows the following data:

Query ID	SQL	Status	Duration	Start	End	Query ID	Rows
019a781...	truncate table trips;	✓	544ms	4:37:05 PM	4:37:06 PM	019a781...	1 rows
019a780...		✓	45.82s	4:31:22 PM	4:32:07 PM	019a780...	61...

The results table also displays a message: 'Row status Statement executed successfully.'

Increase the size of the warehouse to size Large.



Load completed successfully in larger warehouse.

The screenshot shows the 'Results' tab of a query page. The query was run 2 days ago and completed successfully. The results table has the following columns: Row, file, status, rows_parsed, rows_loaded, error_limit, errors_seen, and first_error. The data is as follows:

Row	file	status	rows_parsed	rows_loaded	error_limit	errors_seen	first_error
1	s3://snowfla...	LOADED	93833	93833	1	0	NULL
2	s3://snowfla...	LOADED	112502	112502	1	0	NULL
3	s3://snowfla...	LOADED	103524	103524	1	0	NULL
4	s3://snowfla...	LOADED	168725	168725	1	0	NULL
5	s3://snowfla...	LOADED	142407	142407	1	0	NULL

Comparing the loads using the two warehouses, load using larger warehouse was quite faster as compared to load using smaller warehouse.

Run All Queries | Saved 2 days ago

SYSADMIN COMPUTE_WH (L) CITIBIKE PUBLIC ...

```
copy into trips from @citibike_trips file_format=csv;
```

Results Data Preview History 3 Columns ▾ X

Query_ID	SQL	Status	Duration	Start	End	Query ID	Rows	Bytes Scanned	Cluster	SQL
Filter result...	Columns ▾	✓	14.12s	4:38:42 PM	4:38:56 PM	019a781...	61...	525.9...	1	copy_i
Copy		✓	544ms	4:37:05 PM	4:37:06 PM	019a7811...				trunca
Row	file	✓	45.82s	4:31:22 PM	4:32:07 PM	019a780...	61....	144.0...	1	copy_i
1	s3://snowfla...	LC								

Run All Queries | Saved 2 days ago

SYSADMIN COMPUTE_WH (L) CITIBIKE PUBLIC ...

```
copy into trips from @citibike_trips file_format=csv;
```

Results Data Preview History 3 Columns ▾ X

Query_ID	SQL	Start	End	Query ID	Rows	Bytes Scanned	Cluster	SQL
Filter result...	Columns ▾	4:38:42 PM	4:38:56 PM	019a781...	61...	525.9...	1	copy into trips from @citibik...
Copy		4:37:05 PM	4:37:06 PM	019a7811...				truncate table trips;
Row	file	4:31:22 PM	4:32:07 PM	019a780...	61....	144.0...	1	copy into trips from @citibik...
1	s3://snowfla...	LC						
2	s3://snowfla...	LC						
3	s3://snowfla...	LC						
4	s3://snowfla...	LC						

Create a New Warehouse for Data Analytics

Go to warehouse tab -> Create -> ANALYTICS_WH with large size -> Finish

The screenshot shows the 'Create Warehouse' dialog box over a background of the Snowflake interface. The dialog box has a blue header 'Create Warehouse'. It contains fields for Name (ANALYTICS_WH), Size (Large), Maximum Clusters (1), Scaling Policy (Standard), Auto Suspend (10 minutes), and Auto Resume (checked). There is also a 'Comment' field and a 'Show SQL' button at the bottom. Below the dialog box, the background shows the 'Warehouses' page with two existing warehouses: 'COMPUTE_WH' (Suspended) and 'ANALYTICS_WH' (Started).

Status	Warehouse Name	Size	Clusters	Scaling Poli...	Runn...	Que...	Auto Suspe...	Auto Resume	Created On	Resumed On
Started	ANALYTICS_WH	Large	1 active (min: 1, ...)	Standard	0	0	10 minutes	Yes	4:53:33 PM	4:53:33 PM
Suspended	COMPUTE_WH	Large	min: 1, max: 1	Standard	0	0	10 minutes	Yes	2/19/2021, 10:41:03...	4:38:20 PM

Warehouse created successfully

The screenshot shows the 'Warehouses' page. The top navigation bar includes icons for Databases, Shares, Data Marketplace, Warehouses (selected), Worksheets, History, Preview App, Partner Connect, Help, and user SIDHHIP SYSADMIN. The main content area displays a table of warehouses. The newly created 'ANALYTICS_WH' is listed as 'Started' with a 'Large' size and 1 active cluster. The table also lists 'COMPUTE_WH' as 'Suspended'.

Status	Warehouse Name	Size	Clusters	Scaling Poli...	Runn...	Que...	Auto Suspe...	Auto Resume	Created On	Resumed On
Started	ANALYTICS_WH	Large	1 active (min: 1, ...)	Standard	0	0	10 minutes	Yes	4:53:33 PM	4:53:33 PM
Suspended	COMPUTE_WH	Large	min: 1, max: 1	Standard	0	0	10 minutes	Yes	2/19/2021, 10:41:03...	4:38:20 PM

Analytical Queries, Results Cache, Cloning

Execute SELECT Statements and Result Cache

Go to Worksheet tab -> check for the context

The screenshot shows the Snowflake Worksheet interface. The top navigation bar includes icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (selected), and History. On the right, there are links for Preview App, Partner Connect, Help, and a user dropdown for SIDDHIP SYSADMIN.

In the main area, a query is being run:

```
copy into trips from @citibike_trips file_format=csv;
```

The results pane shows the query ID, SQL, Start, End, Query ID, and Rows. A tooltip is displayed over the Database and Schema dropdowns, indicating the current context:

- Role: SYSADMIN Change
- Warehouse: ANALYTICS_WH (L) On
- Large Resize
- Database: CITIBIKE
- Schema: PUBLIC

The tooltip also shows the query ID and the command being run.

View sample of select data

The screenshot shows the Snowflake Worksheet interface with a different query:

```
select * from trips limit 20;
```

The results pane displays 20 rows of data from the trips table. The columns shown are Row, TRIPDURATION, STARTTIME, STOPTIME, START_STATION, START_STATION, START_STATION, START_STATION, END_STATION_ID, and END_STATION_NAME.

Row	TRIPDURATION	STARTTIME	STOPTIME	START_STATION	START_STATION	START_STATION	START_STATION	END_STATION_ID	END_STATION_NAME
1	386	2018-01-13 ...	2018-01-13 ...	347	Greenwich S...	40.728846	-74.008591	3224	W 13 St & Hu...
2	1059	2018-01-13 ...	2018-01-13 ...	3643	E 41 St & 5 A...	40.752722014	-73.9812362...	422	W 59 St & 10...
3	1196	2018-01-13 ...	2018-01-13 ...	3472	W 15 St & 10...	40.7427538...	-74.0074735...	376	John St & Wi...
4	886	2018-01-13 ...	2018-01-13 ...	504	1 Ave & E 16 ...	40.73221853	-73.98165557	498	Broadway & ...
5	343	2018-01-13 ...	2018-01-13 ...	262	Washington ...	40.6917823	-73.9737299	323	Lawrence St ...

View the hourly statistics on city bike usage.

The screenshot shows the Snowflake Worksheets interface. At the top, there are navigation icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (which is selected), and History. On the right, there are links for Preview App, Partner Connect, Help, and user information (SIDDHIP SYSADMIN). The main area is a code editor titled "New Worksheet" with a "Run" button. The SQL query is as follows:

```
-- 5.1.3
select date_trunc('hour', starttime) as "date",
       count(*) as "num trips",
       avg(tripduration)/60 as "avg duration (mins)",
       avg(haversine(start_station_latitude, start_station_longitude, end_station_latitude, end_station_longitude)) as "avg distance (km)"
  from trips
 group by 1 order by 1;
```

Output:-

The screenshot shows the results of the executed query. The interface is identical to the previous one, but the results tab is selected. The results table has the following columns: Row, date, num trips, avg duration (mins), and avg distance (km). The data is as follows:

Row	date	num trips	avg duration (mins)	avg distance (km)
1	2013-06-01 00:00:00.000	152	56.058442983333	2.127971476
2	2013-06-01 01:00:00.000	102	26.525163400000	2.067906273
3	2013-06-01 02:00:00.000	67	36.119900500000	2.31784827
4	2013-06-01 03:00:00.000	41	44.485365850000	2.349126632
5	2013-06-01 04:00:00.000	16	23.278125000000	1.840026007
6	2013-06-01 05:00:00.000	13	34.584615383333	3.337844489
7	2013-06-01 06:00:00.000	40	22.397500000000	2.832927896
8	2013-06-01 07:00:00.000	93	66.397849466667	2.691774983

Snowflake has a result cache that holds the results of every query executed in the past 24 hours.

The screenshot shows a database query editor interface. At the top, there is a toolbar with a 'Run' button, a 'All Queries' dropdown, and a note indicating the query was saved 29 minutes ago. To the right are user icons for 'SYSADMIN', 'ANALYTICS_WH (L)', 'CITIBIKE', and 'PUBLIC'. Below the toolbar, the SQL code is displayed:

```

93 -- 5.1.4
94
95 select date_trunc('hour', starttime) as "date",
96 count(*) as "num trips",
97 avg(tripduration)/60 as "avg duration (mins)",
98 avg(haversine(start_station_latitude, start_station_longitude, end_station_latitude, end_station_longitude)) as "avg distance (km)"
99 from trips
100 group by 1 order by 1;

```

The results section shows the output of the query:

Row	date	num trips	avg duration (mins)	avg distance (km)
1	2013-06-01 00:00:00.000	152	56.058442983333	2.127971476
2	2013-06-01 01:00:00.000	102	26.525163400000	2.067906273
3	2013-06-01 02:00:00.000	67	36.119900500000	2.31784827
4	2013-06-01 03:00:00.000	41	44.485365850000	2.349126632

In the History window note that the second query runs significantly faster because the results have been cached.

The screenshot shows the history window of the database query editor. It displays the execution details for the previously run query, highlighting the cached results for subsequent runs. The history table includes columns for Status, Duration, Start, End, Query ID, Rows, Bytes Scanned, Cluster, and SQL.

Status	Duration	Start	End	Query ID	Rows	Bytes Scanned	Cluster	SQL
✓	164ms	5:26:53 PM	5:26:53 PM	019a784...		799.5...	1	select
✓	3.33s	5:14:39 PM	5:14:42 PM	019a783...	44...		1	select
✓	1.51s	4:59:51 PM	4:59:53 PM	019a782...	20	1.3GB	1	select
✓	14.12s	4:38:42 PM	4:38:56 PM	019a781...	61...	525.9...	1	copy_i
✓	544ms	4:37:05 PM	4:37:06 PM	019a7811...				trunca

Verify which months are the busiest :-

Screenshot of the Snowflake Worksheet interface. The top navigation bar includes icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (selected), and History. On the right, it shows user information for SIDDHIP SYSADMIN and links for Preview App, Partner Connect, and Help.

The main area shows a query editor with the following SQL code:

```
184 select monthname(starttime) as "month",
185 count(*) as "num trips"
186 from trips
187 group by 1 order by 2 desc;
```

The Results tab is selected, displaying the output of the query:

Row	month	num trips
1	Jun	7600765
2	Sep	6804899
3	Oct	6550164
4	Aug	6518652
5	May	6388309
6	Jul	6013644

Screenshot of the Snowflake Worksheet interface, identical to the first one but showing a different set of results. The top navigation bar and user information are the same.

The main area shows the same query as before:

```
184 select monthname(starttime) as "month",
185 count(*) as "num trips"
186 from trips
187 group by 1 order by 2 desc;
```

The Results tab is selected, displaying the output of the query:

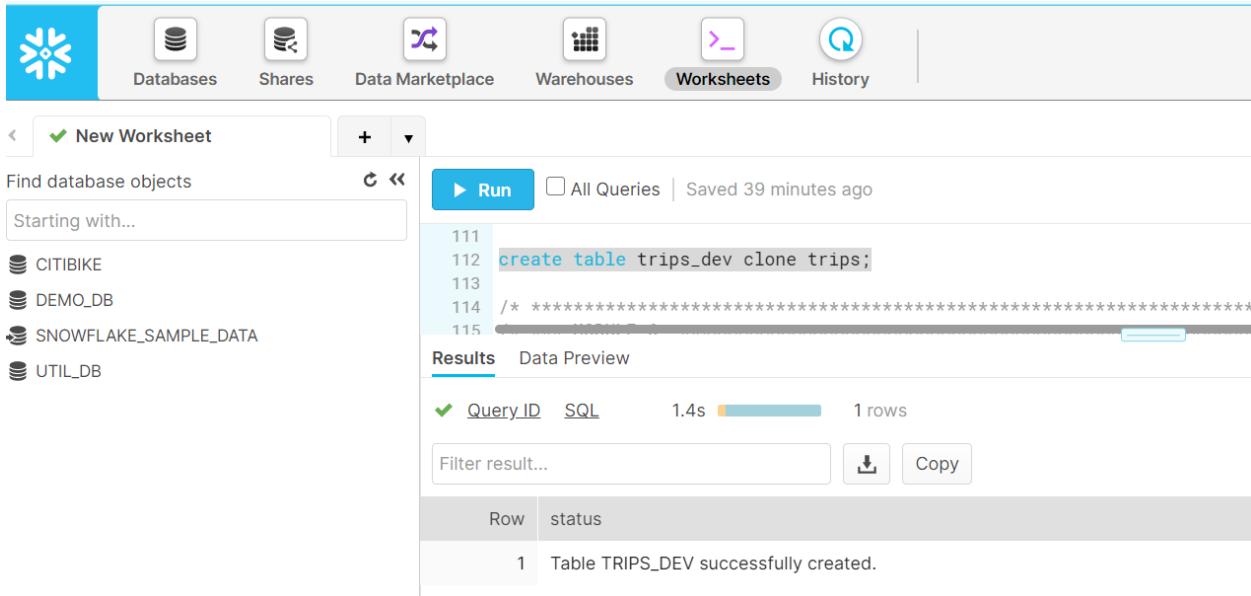
Row	month	num trips
7	Apr	4959244
8	Nov	4719798
9	Mar	3405178
10	Dec	3349319
11	Feb	2617291
12	Jan	2541096

Clone a Table

A snapshot of data present in the source object is taken when the clone is created, and is made available to the cloned object.

The cloned object is writable and independent of the clone source.

Creating a development environment so that we can test the data without affecting changes to prod data and prod data can remain secure .



The screenshot shows the Snowflake Worksheets interface. At the top, there are navigation icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (which is selected), and History. Below the header, a search bar says "New Worksheet" and "Find database objects". A sidebar on the left lists databases: CITIBIKE, DEMO_DB, SNOWFLAKE_SAMPLE_DATA, and UTIL_DB. The main area shows a query editor with the following SQL code:

```
111
112 create table trips_dev clone trips;
113
114 /* ****
115 */

The results section shows a single row of output:
```

Row	status
1	Table TRIPS_DEV successfully created.

New cloned table is present under CITIBIKE database

The screenshot shows the Snowflake interface with a 'New Worksheet' button at the top left. Below it is a search bar with placeholder text 'Starting with...'. The main area displays the database objects for the 'CITIBIKE' schema. It includes the PUBLIC schema and two tables: 'TRIPS' and 'TRIPS_DEV'. A message indicates 'No Views in this Schema'. Other databases listed are 'DEMO_DB', 'SNOWFLAKE_SAMPLE_DATA', and 'UTIL_DB'.

Object Type	Name
Schema	CITIBIKE
Schema	PUBLIC
Table	TRIPS
Table	TRIPS_DEV
View	No Views in this Schema
Database	DEMO_DB
Database	SNOWFLAKE_SAMPLE_DATA
Database	UTIL_DB

Working with Semi-Structured Data, Views, JOIN

The semi structured weather data is staged in AWS S3. Citi bike wants to analyze the impact of weather on the bike rides.

The following 4 steps need to be followed :-

- Load weather data staged in AWS S3 bucket in json format
- Query the data and create view using SQL Dot notation
- Create join on the weather data and trips data
- Analyze the weather and ride count data to determine their relationship

Create a Database and Table

Create weather database to store the semi structured data

The screenshot shows the Snowflake Worksheets interface. At the top, there are navigation icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (which is selected), and History. Below the header, there are two tabs: "New Worksheet" and another "New Worksheet". The main area is titled "Find database objects" with a search bar "Starting with...". On the left, a list of existing databases is shown: CITIBIKE, DEMO_DB, SNOWFLAKE_SAMPLE_DATA, UTIL_DB, and WEATHER. In the center, a query editor window displays the SQL command "create database weather". The results tab shows the output: "Query_ID" (green checkmark), "SQL" (link), "98ms" (execution time), and "1 rows". The result table has columns "Row" and "status", with one row containing the message "Database WEATHER successfully created.".

Row	status
1	Database WEATHER successfully created.

Setting the context

The screenshot shows the Snowflake UI interface. At the top, there are navigation icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (selected), and History. Below the header, two worksheets are visible, both titled "New Worksheet". The left worksheet shows a list of databases: CITIBIKE, DEMO_DB, SNOWFLAKE_SAMPLE_DATA, UTIL_DB, and WEATHER. The right worksheet contains the following SQL code:

```
1 create database weather
2
3 use role sysadmin;
4 use warehouse compute_wh;
5 use database weather;
6 use schema public;
```

The "Results" tab is selected, showing a single row of results:

Row	status
1	Statement executed successfully.

Below the results, there is a modal window with the following settings:

- Role:** SYSADMIN [Change](#)
- Warehouse:** COMPUTE_WH (L) Suspended [Resume](#)
- Database:** WEATHER
- Schema:** PUBLIC

Creating table to store the JSON data which will be stored in Snowflake's variant column type.

The screenshot shows the Snowflake Worksheets interface. At the top, there are navigation icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (which is selected), and History. Below the header, there are two tabs: "New Worksheet" and another "New Worksheet". A search bar labeled "Find database objects" with "Starting with..." dropdown is present. On the left, a sidebar lists databases: CITIBIKE, DEMO_DB, SNOWFLAKE_SAMPLE_DATA, UTIL_DB, and WEATHER. The main area contains a query editor with the following SQL code:

```
1 create database weather
2
3 use role sysadmin;
4 use warehouse compute_wh;
5 use database weather;
6 use schema public;
7
8 create table json_weather_data (v variant);
```

Below the query editor, there are tabs for "Results" and "Data Preview". The "Results" tab is selected, showing a successful execution message: "Table JSON_WEATHER_DATA successfully created." The "Data Preview" tab is also visible. At the bottom, there are buttons for "Filter result...", "Download", and "Copy".

Go to Databases -> Click on WEATHER link

The screenshot shows the Snowflake Databases interface. At the top, there are navigation icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets, and History. The 'Databases' icon is highlighted. Below the header, the title 'Databases' is displayed. A toolbar with actions like 'Create...', 'Clone...', 'Drop...', and 'Transfer Ownership' is present. A search bar shows 'Search Databases' and '1/5 databases'. A table lists the databases:

Database	Origin	↓ Creation Time	Owner	Comment
WEATHER		8:38 PM	SYSADMIN	
CITIBIKE		2/19/2021, 11:03 PM	SYSADMIN	
SNOWFLAKE_SAMPLE_DATA	SFC_SAMPLES.SA...	2/19/2021, 10:40 PM	ACCOUNTADMIN	TPC-H, OpenWeatherMap, etc
DEMO_DB		2/19/2021, 10:40 PM	SYSADMIN	demo database
UTIL_DB		2/19/2021, 10:40 PM	SYSADMIN	utility database

View the newly created table

The screenshot shows the Snowflake Tables interface for the 'WEATHER' database. The top navigation bar includes icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets, History, and a 'Preview Ap...' button. The 'Databases' icon is highlighted. The title 'Databases > WEATHER' is shown. A navigation bar below the title has tabs for Tables, Views, Schemas, Stages, File Formats, Sequences, and Pipes. The 'Tables' tab is selected. A toolbar with actions like 'Create...', 'Create Like...', 'Clone...', 'Load Data...', 'Drop...', and 'Transfer Ownership' is present. A table lists the tables:

Table Name	Schema	Creation Time	Owner	Rows	Size	Comment
JSON_WEATHER_DATA	PUBLIC	8:41:30 PM	SYSADMIN			

Create an External Stage

Go to worksheet tab -> create a stage from where the unstructured data is stored on AWS S3

The screenshot shows the Snowflake Worksheet interface. At the top, there are several navigation icons: Databases, Shares, Data Marketplace, Warehouses, Worksheets (which is selected), and History. Below the header, there are two tabs: 'New Worksheet' and another 'New Worksheet'. A search bar labeled 'Find database objects' with the placeholder 'Starting with...' is present. On the left, a sidebar lists databases: CITIBIKE, DEMO_DB, SNOWFLAKE_SAMPLE_DATA, UTIL_DB, and WEATHER. The main workspace contains a SQL query:

```

4 use warehouse compute_wh;
5 use database weather;
6 use schema public;
7
8 create table json_weather_data (v variant);
9
10 create stage nyc_weather
11 url = 's3://snowflake-workshop-lab/weather-nyc';

```

The 'Run' button is highlighted. Below the code, the results section shows a single row of data:

Row	status
1	Stage area NYC_WEATHER successfully created.

Viewing the contents of the nyc_weather stage

The screenshot shows the Snowflake Worksheet interface again. The top navigation and sidebar are identical to the previous screenshot. The main workspace contains a SQL query:

```

6 use schema public;
7
8 create table json_weather_data (v variant);
9
10 create stage nyc_weather
11 url = 's3://snowflake-workshop-lab/weather-nyc';
12
13 list @nyc_weather;

```

The 'Run' button is highlighted. Below the code, the results section shows the contents of the stage:

Row	name	size	mid5	last_modified
1	s3://snowflake-workshop-lab/wea...	40905	79638b8890d72e7d9bae14e3db6...	Tue, 25 Jun 2019 21:40:55 GMT
2	s3://snowflake-workshop-lab/wea...	42150	76fc16208b4ca385fbafefedff...	Tue, 25 Jun 2019 21:40:55 GMT
3	s3://snowflake-workshop-lab/wea...	43130	57ee5eae2ff354f9ffffe3e3740f9b2...	Tue, 25 Jun 2019 21:40:55 GMT
4	s3://snowflake-workshop-lab/wea...	42132	cfc162be5002f95f71999b287608f...	Tue, 25 Jun 2019 21:40:55 GMT

Results Data Preview

✓ Query_ID SQL 399ms 61 rows

Filter result... Columns ▾

Row	name	size	md5	last_modified
1	s3://snowflake-workshop-lab/wea...	40905	79638b8890d72e7d9bae14e3db6...	Tue, 25 Jun 2019 21:40:55 GMT
2	s3://snowflake-workshop-lab/wea...	42150	76fcfd16208b4ca385fbbafe6fedff...	Tue, 25 Jun 2019 21:40:55 GMT
3	s3://snowflake-workshop-lab/wea...	43130	57ee5eae2ff354f9ffe3e3740f9b2...	Tue, 25 Jun 2019 21:40:55 GMT
4	s3://snowflake-workshop-lab/wea...	42132	cfc162be5002f95f71999b287608f...	Tue, 25 Jun 2019 21:40:55 GMT
5	s3://snowflake-workshop-lab/wea...	80842	ea4b3e62c759f610c9f465052019...	Tue, 25 Jun 2019 21:40:56 GMT
6	s3://snowflake-workshop-lab/wea...	41813	0053b51c2b8569bfbe4381b8f736...	Tue, 25 Jun 2019 21:40:56 GMT
7	s3://snowflake-workshop-lab/wea...	42060	84af8404c59f205ebdf86731c9a3...	Tue, 25 Jun 2019 21:40:56 GMT
8	s3://snowflake-workshop-lab/wea...	41773	08f76d45e3ea474e0b84181b087...	Tue, 25 Jun 2019 21:40:56 GMT
9	s3://snowflake-workshop-lab/wea...	43868	a63bd4cd3afb852a5bc840086ee...	Tue, 25 Jun 2019 21:40:56 GMT

Loading and Verifying the Unstructured Data

Using warehouse loading data into snowflake table from s3 bucket

The screenshot shows the Snowflake interface with the following details:

- Top Navigation:** Databases, Shares, Data Marketplace, Warehouses, **Worksheets** (selected), History.
- User Information:** SIDHHP SYSADMIN.
- Worksheet Content:**
 - New Worksheet button.
 - Run button.
 - SQL code:

```

5 use database weather;
6 use schema public;
7
8 create table json_weather_data (v variant);
9
10 create stage nyc_weather
11 url = 's3://snowflake-workshop-lab/weather-nyc';
12
13 list @nyc_weather;
14
15 copy into json_weather_data
16 from @nyc_weather
17 file_format = (type=json);

```
- Results Tab:**
 - Query_ID: 3.88s, 61 rows.
 - Table output:

Row	file	status	rows_parsed	rows_loaded	error_limit	errors_seen	first_error	first_error_line	first_error_char
1	s3://snowfla...	LOADED	913	913	1	0	NULL	NULL	NULL

Viewing the data that has been loaded

```

6 use schema public;
7
8 create table json_weather_data (v variant);
9
10 create stage nyc_weather
11 url = 's3://snowflake-workshop-lab/weather-nyc';
12
13 list @nyc_weather;
14
15 copy into json_weather_data
16 from @nyc_weather
17 file_format = (type=json);
18
19 select * from json_weather_data limit 10;

```

Results Data Preview

✓ Query ID SQL 567ms 10 rows

Row	V
1	{"city": {"coord": {"lat": 43.000351, "lon": -75.499901}, "country": "US", "findname": "NEW YORK", "id": 5128638, "langs": [{"abbr": "NY"}, {"am": "\u00d6\u00c7h"}, ...], "name": "New York", "place_id": 123456789, "type": "City"}, "id": 5128638, "name": "New York", "place_id": 123456789, "type": "City"} {"city": {"coord": {"lat": 43.000351, "lon": -75.499901}, "country": "US", "findname": "NEW YORK", "id": 5128638, "langs": [{"abbr": "NY"}, {"am": "\u00d6\u00c7h"}, ...], "name": "New York", "place_id": 123456789, "type": "City"}, "id": 5128638, "name": "New York", "place_id": 123456789, "type": "City"} {"city": {"coord": {"lat": 40.714272, "lon": -74.005966}, "country": "US", "findname": "NEW YORK", "id": 5128581, "langs": [{"af": "New York Stad"}, {"an": "Nueba York" ...}, {"ar": "نيويورك"}, {"ca": "Nova York"}, {"cs": "Nov\u00e1 York"}, {"de": "New York City"}, {"el": "Νέα Υόρκη"}, {"es": "Nueva York"}, {"fr": "New York"}, {"he": "ניו יורק"}, {"it": "New York"}, {"ja": "ニューヨーク"}, {"ko": "뉴욕"}, {"nl": "New York"}, {"pt": "Nova York"}, {"ru": "Нью-Йорк"}, {"th": "นิวยอร์ก"}, {"tr": "New York"}, {"zh": "纽约"}], "name": "New York", "place_id": 123456789, "type": "City"}, "id": 5128581, "name": "New York", "place_id": 123456789, "type": "City"} {"city": {"coord": {"lat": 43.000351, "lon": -75.499901}, "country": "US", "findname": "NEW YORK", "id": 5128638, "langs": [{"abbr": "NY"}, {"am": "\u00d6\u00c7h"}, ...], "name": "New York", "place_id": 123456789, "type": "City"}, "id": 5128638, "name": "New York", "place_id": 123456789, "type": "City"} {"city": {"coord": {"lat": 40.714272, "lon": -74.005966}, "country": "US", "findname": "NEW YORK", "id": 5128581, "langs": [{"af": "New York Stad"}, {"an": "Nueba York" ...}, {"ar": "نيويورك"}, {"ca": "Nova York"}, {"cs": "Nov\u00e1 York"}, {"de": "New York City"}, {"el": "Νέα Υόρκη"}, {"es": "Nueva York"}, {"fr": "New York"}, {"he": "ניו יורק"}, {"it": "New York"}, {"ja": "ニューヨーク"}, {"ko": "뉴욕"}, {"nl": "New York"}, {"pt": "Nova York"}, {"ru": "Нью-Йорк"}, {"th": "นิวยอร์ก"}, {"tr": "New York"}, {"zh": "纽约"}], "name": "New York", "place_id": 123456789, "type": "City"}, "id": 5128581, "name": "New York", "place_id": 123456789, "type": "City"} {"city": {"coord": {"lat": 43.000351, "lon": -75.499901}, "country": "US", "findname": "NEW YORK", "id": 5128638, "langs": [{"abbr": "NY"}, {"am": "\u00d6\u00c7h"}, ...], "name": "New York", "place_id": 123456789, "type": "City"}, "id": 5128638, "name": "New York", "place_id": 123456789, "type": "City"} {"city": {"coord": {"lat": 40.714272, "lon": -74.005966}, "country": "US", "findname": "NEW YORK", "id": 5128581, "langs": [{"af": "New York Stad"}, {"an": "Nueba York" ...}, {"ar": "نيويورك"}, {"ca": "Nova York"}, {"cs": "Nov\u00e1 York"}, {"de": "New York City"}, {"el": "Νέα Υόρκη"}, {"es": "Nueva York"}, {"fr": "New York"}, {"he": "ניו יורק"}, {"it": "New York"}, {"ja": "ニューヨーク"}, {"ko": "뉴욕"}, {"nl": "New York"}, {"pt": "Nova York"}, {"ru": "Нью-Йорк"}, {"th": "นิวยอร์ก"}, {"tr": "New York"}, {"zh": "纽约"}], "name": "New York", "place_id": 123456789, "type": "City"}, "id": 5128581, "name": "New York", "place_id": 123456789, "type": "City"} {"city": {"coord": {"lat": 43.000351, "lon": -75.499901}, "country": "US", "findname": "NEW YORK", "id": 5128638, "langs": [{"abbr": "NY"}, {"am": "\u00d6\u00c7h"}, ...], "name": "New York", "place_id": 123456789, "type": "City"}, "id": 5128638, "name": "New York", "place_id": 123456789, "type": "City"}}

✓ Query ID SQL 567ms 10 rows

Row	V
1	{"city": {"coord": {"lat": 43.000351, "lon": -75.499901}, "country": "US", "findname": "NEW YORK", "id": 5128638, "langs": [{"abbr": "NY"}, {"am": "\u00d6\u00c7h"}, ...], "name": "New York", "place_id": 123456789, "type": "City"}, "id": 5128638, "name": "New York", "place_id": 123456789, "type": "City"} {"city": {"coord": {"lat": 43.000351, "lon": -75.499901}, "country": "US", "findname": "NEW YORK", "id": 5128638, "langs": [{"abbr": "NY"}, {"am": "\u00d6\u00c7h"}, ...], "name": "New York", "place_id": 123456789, "type": "City"}, "id": 5128638, "name": "New York", "place_id": 123456789, "type": "City"} {"city": {"coord": {"lat": 40.714272, "lon": -74.005966}, "country": "US", "findname": "NEW YORK", "id": 5128581, "langs": [{"af": "New York Stad"}, {"an": "Nueba York" ...}, {"ar": "نيويورك"}, {"ca": "Nova York"}, {"cs": "Nov\u00e1 York"}, {"de": "New York City"}, {"el": "Νέα Υόρκη"}, {"es": "Nueva York"}, {"fr": "New York"}, {"he": "ניו יורק"}, {"it": "New York"}, {"ja": "ニューヨーク"}, {"ko": "뉴욕"}, {"nl": "New York"}, {"pt": "Nova York"}, {"ru": "Нью-Йорк"}, {"th": "นิวยอร์ก"}, {"tr": "New York"}, {"zh": "纽约"}], "name": "New York", "place_id": 123456789, "type": "City"}, "id": 5128581, "name": "New York", "place_id": 123456789, "type": "City"} {"city": {"coord": {"lat": 43.000351, "lon": -75.499901}, "country": "US", "findname": "NEW YORK", "id": 5128638, "langs": [{"abbr": "NY"}, {"am": "\u00d6\u00c7h"}, ...], "name": "New York", "place_id": 123456789, "type": "City"}, "id": 5128638, "name": "New York", "place_id": 123456789, "type": "City"} {"city": {"coord": {"lat": 40.714272, "lon": -74.005966}, "country": "US", "findname": "NEW YORK", "id": 5128581, "langs": [{"af": "New York Stad"}, {"an": "Nueba York" ...}, {"ar": "نيويورك"}, {"ca": "Nova York"}, {"cs": "Nov\u00e1 York"}, {"de": "New York City"}, {"el": "Νέα Υόρκη"}, {"es": "Nueva York"}, {"fr": "New York"}, {"he": "ניו יורק"}, {"it": "New York"}, {"ja": "ニューヨーク"}, {"ko": "뉴욕"}, {"nl": "New York"}, {"pt": "Nova York"}, {"ru": "Нью-Йорк"}, {"th": "นิวยอร์ก"}, {"tr": "New York"}, {"zh": "纽约"}], "name": "New York", "place_id": 123456789, "type": "City"}, "id": 5128581, "name": "New York", "place_id": 123456789, "type": "City"} {"city": {"coord": {"lat": 43.000351, "lon": -75.499901}, "country": "US", "findname": "NEW YORK", "id": 5128638, "langs": [{"abbr": "NY"}, {"am": "\u00d6\u00c7h"}, ...], "name": "New York", "place_id": 123456789, "type": "City"}, "id": 5128638, "name": "New York", "place_id": 123456789, "type": "City"}}

Clicking on one of the data -> view data stored in raw json format

The screenshot shows a data analysis interface with a top navigation bar featuring icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (selected), and History. On the right, there are links for Preview App, Partner Connect, and Help.

The main area displays two tabs: 'Results' (selected) and 'Data Preview'. The 'Results' tab shows a table with 10 rows of data, each containing a 'Row' number and a 'Value' (V) column. The 'Data Preview' tab shows a single row of data.

A modal window titled 'Details' is open, displaying a JSON object with 13 numbered lines. The JSON structure includes fields like 'city', 'coord', 'country', 'findname', 'id', and 'langs'.

Row	V
2	{ "city": { "coord": { "lat": 43.000351, "lon": -75.499901 }, "country": "US", "findname": "NEW YORK", "id": 5128638, "langs": [{ "abbr": "NY" }, { "af": "Nueba Yorkstad", "an": "Nueba Yorkstad", "am": "Նյու Պարքի" }, { "an": "Nueba Yorkstad", "af": "Nueba Yorkstad", "am": "Նյու Պարքի" }, { "an": "Nueba Yorkstad", "af": "Nueba Yorkstad", "am": "Նյու Պարքի" }] }
3	{ "city": { "coord": { "lat": 40.714272, "lon": -74.005966 }, "country": "US", "findname": "NEW YORK", "id": 5128581, "langs": [{ "abbr": "NY" }, { "af": "New York Stad", "an": "Nueba Yorkstad", "am": "Նյու Պարքի" }, { "an": "Nueba Yorkstad", "af": "Nueba Yorkstad", "am": "Նյու Պարքի" }, { "an": "Nueba Yorkstad", "af": "Nueba Yorkstad", "am": "Նյու Պարքի" }] }
4	{ "city": { "coord": { "lat": 43.000351, "lon": -75.499901 }, "country": "US", "findname": "NEW YORK", "id": 5128638, "langs": [{ "abbr": "NY" }, { "af": "Nueba Yorkstad", "an": "Nueba Yorkstad", "am": "Նյու Պարքի" }, { "an": "Nueba Yorkstad", "af": "Nueba Yorkstad", "am": "Նյու Պարքի" }, { "an": "Nueba Yorkstad", "af": "Nueba Yorkstad", "am": "Նյու Պարքի" }] }
5	{ "city": { "coord": { "lat": 40.714272, "lon": -74.005966 }, "country": "US", "findname": "NEW YORK", "id": 5128581, "langs": [{ "abbr": "NY" }, { "af": "New York Stad", "an": "Nueba Yorkstad", "am": "Նյու Պարքի" }, { "an": "Nueba Yorkstad", "af": "Nueba Yorkstad", "am": "Նյու Պարքի" }, { "an": "Nueba Yorkstad", "af": "Nueba Yorkstad", "am": "Նյու Պարքի" }] }
6	{ "city": { "coord": { "lat": 40.714272, "lon": -74.005966 }, "country": "US", "findname": "NEW YORK", "id": 5128638, "langs": [{ "abbr": "NY" }, { "af": "Nueba Yorkstad", "an": "Nueba Yorkstad", "am": "Նյու Պարքի" }, { "an": "Nueba Yorkstad", "af": "Nueba Yorkstad", "am": "Նյու Պարքի" }, { "an": "Nueba Yorkstad", "af": "Nueba Yorkstad", "am": "Նյու Պարքի" }] }
7	{ "city": { "coord": { "lat": 43.000351, "lon": -75.499901 }, "country": "US", "findname": "NEW YORK", "id": 5128638, "langs": [{ "abbr": "NY" }, { "af": "Nueba Yorkstad", "an": "Nueba Yorkstad", "am": "Նյու Պարքի" }, { "an": "Nueba Yorkstad", "af": "Nueba Yorkstad", "am": "Նյու Պարքի" }, { "an": "Nueba Yorkstad", "af": "Nueba Yorkstad", "am": "Նյու Պարքի" }] }
8	{ "city": { "coord": { "lat": 40.714272, "lon": -74.005966 }, "country": "US", "findname": "NEW YORK", "id": 5128581, "langs": [{ "abbr": "NY" }, { "af": "New York Stad", "an": "Nueba Yorkstad", "am": "Նյու Պարքի" }, { "an": "Nueba Yorkstad", "af": "Nueba Yorkstad", "am": "Նյու Պարքի" }, { "an": "Nueba Yorkstad", "af": "Nueba Yorkstad", "am": "Նյու Պարքի" }] }
9	{ "city": { "coord": { "lat": 43.000351, "lon": -75.499901 }, "country": "US", "findname": "NEW YORK", "id": 5128638, "langs": [{ "abbr": "NY" }, { "af": "Nueba Yorkstad", "an": "Nueba Yorkstad", "am": "Նյու Պարքի" }, { "an": "Nueba Yorkstad", "af": "Nueba Yorkstad", "am": "Նյու Պարքի" }, { "an": "Nueba Yorkstad", "af": "Nueba Yorkstad", "am": "Նյու Պարքի" }] }
10	{ "city": { "coord": { "lat": 40.714272, "lon": -74.005966 }, "country": "US", "findname": "NEW YORK", "id": 5128581, "langs": [{ "abbr": "NY" }, { "af": "New York Stad", "an": "Nueba Yorkstad", "am": "Նյու Պարքի" }, { "an": "Nueba Yorkstad", "af": "Nueba Yorkstad", "am": "Նյու Պարքի" }, { "an": "Nueba Yorkstad", "af": "Nueba Yorkstad", "am": "Նյու Պարքի" }] }

Details

```
1 | {
2 |   "city": {
3 |     "coord": {
4 |       "lat": 43.000351,
5 |       "lon": -75.499901
6 |     },
7 |     "country": "US",
8 |     "findname": "NEW YORK",
9 |     "id": 5128638,
10 |    "langs": [
11 |      {
12 |        "abbr": "NY"
13 |      }
14 |    ]
15 |  }
16 |}
```

Done

Create a View and Query Semi-Structured Data

SQL dot notation `v.city.coord.lat` is used in this command to pull out values at lower levels within the JSON hierarchy. This allows us to treat each field as if it were a column in a relational table.

The screenshot shows the Snowflake Worksheets interface. At the top, there are navigation icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (which is selected), and History. Below the header, there are two tabs: "New Worksheet" and "New Worksheet". The "Run" button is highlighted. To the right of the run button are checkboxes for "All Queries" and "Saved 25 seconds ago". The main area contains the following SQL code:

```
20
21 create view json_weather_data_view as
22 select
23 v:time::timestamp as observation_time,
24 v:city.id::int as city_id,
25 v:city.name::string as city_name,
26 v:city.country::string as country,
27 v:city.coord.lat::float as city_lat,
28 v:city.coord.lon::float as city_lon,
29 v:clouds.all::int as clouds,
30 (v:main.temp::float)-273.15 as temp_avg,
31 (v:main.temp_min::float)-273.15 as temp_min,
32 (v:main.temp_max::float)-273.15 as temp_max,
33 weather.main::string as weather
```

Below the code, there are two tabs: "Results" (selected) and "Data Preview". The results table has columns: "Query_ID", "SQL", "152ms", and "1 rows". There is a "Filter result..." input field and download/copy buttons. The results table shows one row:

Row	status
1	View JSON_WEATHER_DATA_VIEW successfully created.

The screenshot shows the Snowflake UI interface. At the top, there is a navigation bar with a blue logo on the left and three main tabs: "Databases", "Shares", and "Data Ma". Below the navigation bar, there is a search bar with a green checkmark icon and the text "New Worksheet". To the right of the search bar, there is another green checkmark icon and the text "New".

Below the search bar, there is a section titled "Find database objects" with a sub-section "Starting with...".

The main content area displays a list of database objects:

- CITIBIKE
- DEMO_DB
- SNOWFLAKE_SAMPLE_DATA
- UTIL_DB
- WEATHER
- INFORMATION_SCHEMA
- PUBLIC
 - Tables
 - JSON_WEATHER_DATA
 - Views
 - JSON_WEATHER_DATA_VIEW

The screenshot shows the Snowflake Worksheet interface. The top navigation bar includes icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (selected), and History. On the right, it shows user information: SYSADMIN, COMPUTE_WH (L), WEATHER, PUBLIC, and Siddhiprakash (SYSADMIN). The main area has tabs for 'New Worksheet' and 'New Worksheet'. A sidebar on the left lists database objects: CITIBIKE, DEMO_DB, SNOWFLAKE_SAMPLE_DATA, UTIL_DB, and WEATHER (with sub-schemas INFORMATION_SCHEMA and PUBLIC, and tables JSON_WEATHER_DATA and Views JSON_WEATHER_DATA_VIEW). The central workspace contains a SQL query:

```

31 (v:main.temp_min::float)-273.15 as temp_min,
32 (v:main.temp_max::float)-273.15 as temp_max,
33 v:weather[0].main::string as weather,
34 v:weather[0].description::string as weather_desc,
35 v:weather[0].icon::string as weather_icon,
36 v:wind.deg::float as wind_dir,
37 v:wind.speed::float as wind_speed
38 from json_weather_data
39 where city_id = 5128638;
40
41 select * from json_weather_data_view
42 where date_trunc( 'month' ,observation_time) = '2018-01-01'
43 limit 20;

```

The results section shows a single row of data:

Row	OBSERVATION_DATE	CITY_ID	CITY_NAME	COUNTRY	CITY_LAT	CITY_LON	CLOUDS	TEMP_AVG	TEMP_MIN
1	2018-01-11 00:00:00	5128638	New York	US	43.000351	-75.499901	90	9.05	8

The screenshot shows the Snowflake Worksheet interface, identical to the first one but with a larger result set. The results section shows 20 rows of data:

Row	OBSERVATION_DATE	CITY_ID	CITY_NAME	COUNTRY	CITY_LAT	CITY_LON	CLOUDS	TEMP_AVG	TEMP_MIN
1	2018-01-11 00:00:00	5128638	New York	US	43.000351	-75.499901	90	9.05	8
2	2018-01-11 08:00:00	5128638	New York	US	43.000351	-75.499901	20	10.05	9
3	2018-01-12 00:00:00	5128638	New York	US	43.000351	-75.499901	90	10.58	9
4	2018-01-12 08:00:00	5128638	New York	US	43.000351	-75.499901	90	11.05	10
5	2018-01-29 00:00:00	5128638	New York	US	43.000351	-75.499901	90	-2.53	-3
6	2018-01-29 08:00:00	5128638	New York	US	43.000351	-75.499901	90	-3	-3
7	2018-01-10 00:00:00	5128638	New York	US	43.000351	-75.499901	90	-6.8	-11
8	2018-01-10 08:00:00	5128638	New York	US	43.000351	-75.499901	1	-7.8	-12

Use a Join Operation to Correlate Against Data Sets

Joining weather and trips data to analyze the impact of weather on the bike rides by counting number of trips associated with certain weather conditions.

The screenshot shows the Snowflake Worksheets interface. On the left, there's a sidebar with database navigation. The main area contains a query editor with the following SQL code:

```

34 v:weather[0].description::string as weather_desc,
35 v:weather[0].icon::string as weather_icon,
36 v:wind.deg::float as wind_dir,
37 v:wind.speed::float as wind_speed
38 from json_weather_data
39 where city_id = 5128638;
40
41 select * from json_weather_data_view
42 where date_trunc('month', observation_time) = '2018-01-01'
43 limit 20;
44
45 select weather as conditions
46 ,count(*) as num_trips
47 from citibike.public.trips
48 left outer join json_weather_data_view
49 on date_trunc('hour', observation_time) = date_trunc('hour', starttime)
50 where conditions is not null
51 group by 1 order by 2 desc;

```

So, in the output below we can observe that the number of bike rides is quite higher when the climate is good.

The screenshot shows the results of the query in a table format. The columns are 'Row' and 'CONDITIONS' (with a secondary header) and 'NUM_TRIPS'. The data is as follows:

Row	CONDITIONS	NUM_TRIPS
1	Clear	9249531
2	Clouds	8934964
3	Rain	3206720
4	Snow	1380418
5	Mist	798487
6	Fog	362496
7	Thunderstorm	230539
8	Drizzle	98779
9	Haze	40724
10	Smoke	2871

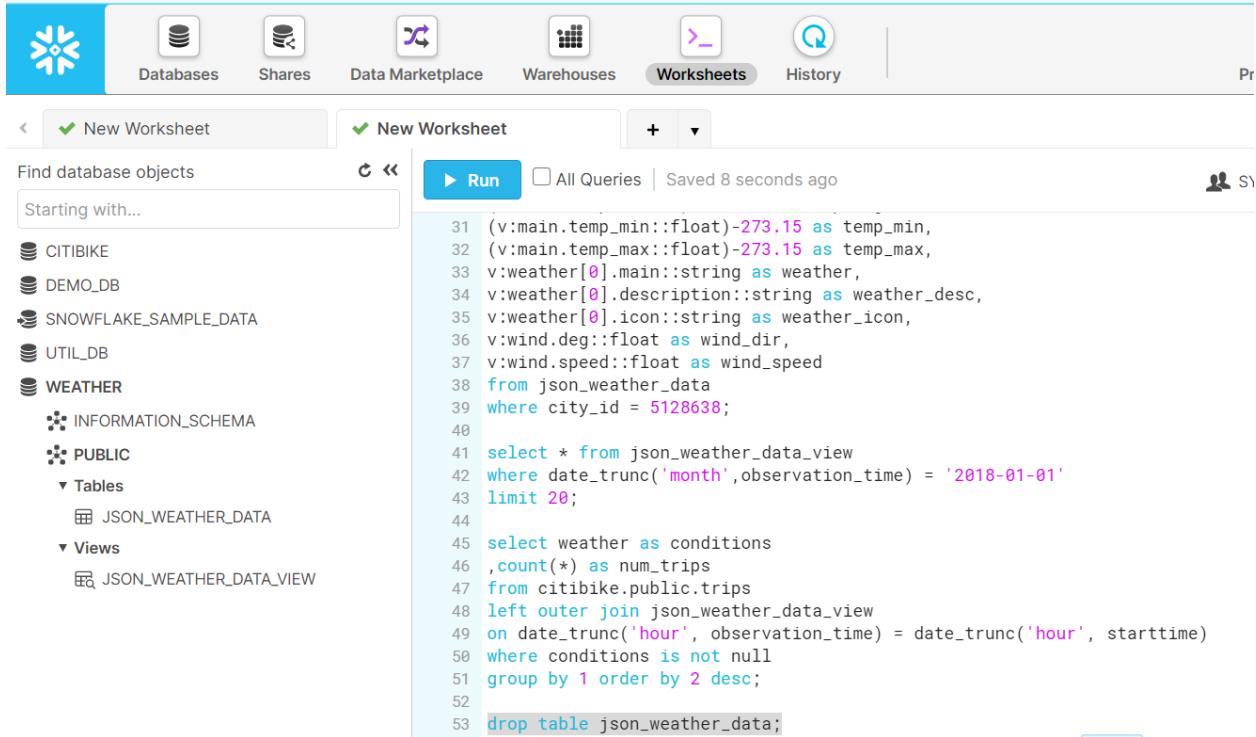
Using Time Travel

This feature allows us to access the historical data. Below are the applications of time travel in snowflake.

- Restoring data-related objects such as tables, schemas, and databases that may have been deleted
- Duplicating and backing up data from key points in the past
- Analyzing data usage and manipulation over specified periods of time

Drop and Undrop a Table

Recover data that has been accidentally dropped.



The screenshot shows the Snowflake Worksheets interface. At the top, there are navigation icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (which is selected), and History. Below the header, there are two tabs: "New Worksheet" and another "New Worksheet". A sidebar on the left lists database objects: CITIBIKE, DEMO_DB, SNOWFLAKE_SAMPLE_DATA, UTIL_DB, WEATHER, INFORMATION_SCHEMA, and PUBLIC. Under PUBLIC, there are Tables (JSON_WEATHER_DATA) and Views (JSON_WEATHER_DATA_VIEW). The main workspace contains a SQL query:

```
31 (v:main.temp_min::float)-273.15 as temp_min,
32 (v:main.temp_max::float)-273.15 as temp_max,
33 v:weather[0].main::string as weather,
34 v:weather[0].description::string as weather_desc,
35 v:weather[0].icon::string as weather_icon,
36 v:wind.deg::float as wind_dir,
37 v:wind.speed::float as wind_speed
38 from json_weather_data
39 where city_id = 5128638;
40
41 select * from json_weather_data_view
42 where date_trunc('month',observation_time) = '2018-01-01'
43 limit 20;
44
45 select weather as conditions
46 ,count(*) as num_trips
47 from citibike.public.trips
48 left outer join json_weather_data_view
49 on date_trunc('hour', observation_time) = date_trunc('hour', starttime)
50 where conditions is not null
51 group by 1 order by 2 desc;
52
53 drop table json_weather_data;
```

Results Data Preview

✓ [Query ID](#) [SQL](#) 77ms 1 rows

Filter result...



Copy

Row	status
1	JSON_WEATHER_DATA successfully dropped.

New Worksheet N

Find database objects

Starting with...

- CITIBIKE
- DEMO_DB
- SNOWFLAKE_SAMPLE_DATA
- UTIL_DB
- WEATHER

INFORMATION_SCHEMA

PUBLIC

No Tables in this Schema

▼ Views

JSON_WEATHER_DATA_VIEW

Verify the table has been dropped successfully by running the select command

Databases Shares Data Marketplace Warehouses Worksheets History

New Worksheet Run All Queries | Saved 7 seconds ago

Find database objects

Starting with...

- CITIBIKE
- DEMO_DB
- SNOWFLAKE_SAMPLE_DATA
- UTIL_DB
- WEATHER

INFORMATION_SCHEMA

PUBLIC

```
52
53 drop table json_weather_data;
54
55 select * from json_weather_data limit 10;
```

Results Data Preview

✖ Query_ID SQL 28ms

SQL compilation error: Object 'JSON_WEATHER_DATA' does not exist or not authorized.

Restoring the dropped table.

The screenshot shows the Snowflake Worksheets interface. At the top, there are navigation icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (which is selected), and History. Below the header, there are two tabs: "New Worksheet" and "New Worksheet". A search bar labeled "Find database objects" with the placeholder "Starting with..." is present. On the left, a sidebar lists databases: CITIBIKE, DEMO_DB, SNOWFLAKE_SAMPLE_DATA, UTIL_DB, WEATHER, INFORMATION_SCHEMA, PUBLIC, Tables (with JSON_WEATHER_DATA listed), and Views (with JSON_WEATHER_DATA_VIEW listed). The main area contains a query editor with the following SQL code:

```
54 select * from json_weather_data limit 10;
55
56 undrop table json_weather_data;
```

The "Results" tab is selected, showing the output of the query:

Row	status
1	Table JSON_WEATHER_DATA successfully restored.

Below the results table are buttons for "Filter result...", "Download", and "Copy".

Find database objects



Starting with...

CITIBIKE

DEMO_DB

SNOWFLAKE_SAMPLE_DATA

UTIL_DB

WEATHER

INFORMATION_SCHEMA

PUBLIC

▼ Tables

JSON_WEATHER_DATA

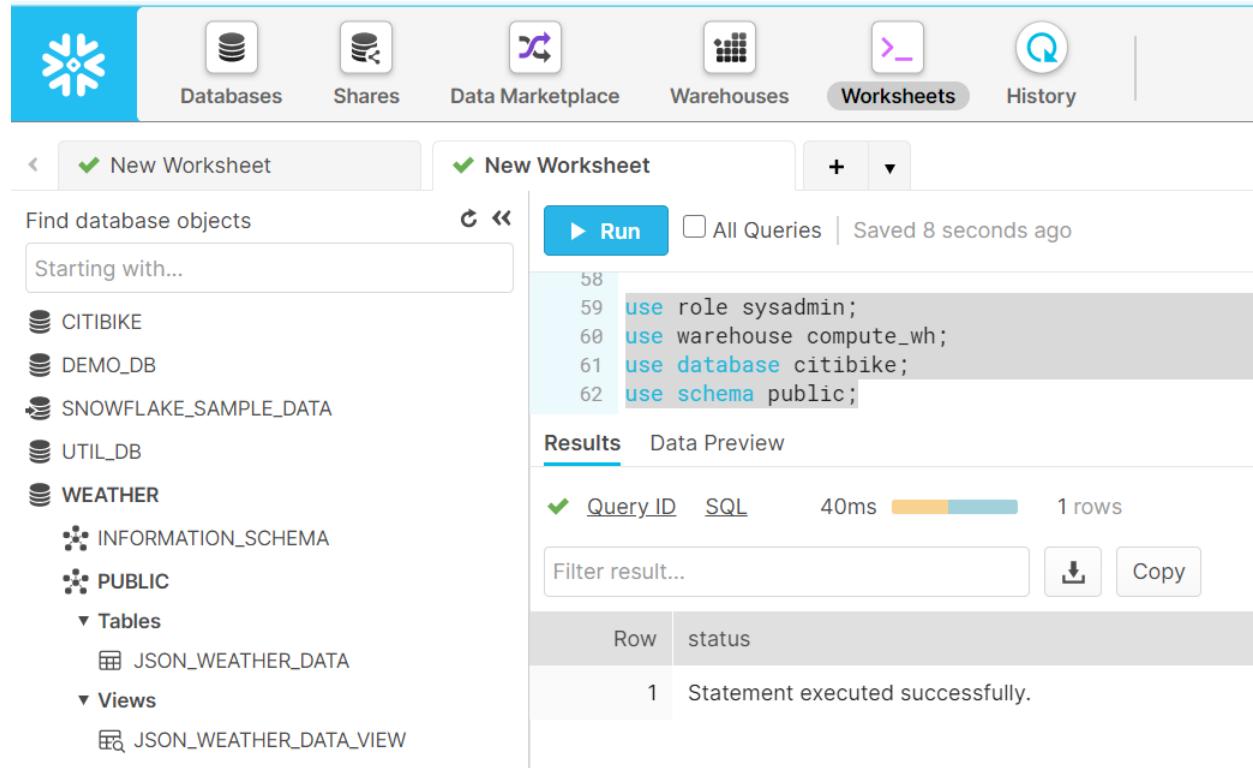
▼ Views

JSON_WEATHER_DATA_VIEW

Roll Back a Table

Roll back a table to a previous state to fix an unintentional DML error that replaces all the station names in the CITIBIKE database's TRIPS table with the word "oops."

Verify the context of the worksheet.



The screenshot shows the Snowflake web interface. At the top, there are navigation icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (which is selected), and History. Below the header, there are two tabs labeled "New Worksheet". On the left, the "Find database objects" sidebar lists databases: CITIBIKE, DEMO_DB, SNOWFLAKE_SAMPLE_DATA, UTIL_DB, WEATHER, INFORMATION_SCHEMA, PUBLIC, Tables (JSON_WEATHER_DATA), and Views (JSON_WEATHER_DATA_VIEW). On the right, a query editor window displays the following SQL code:

```
58
59 use role sysadmin;
60 use warehouse compute_wh;
61 use database citibike;
62 use schema public;
```

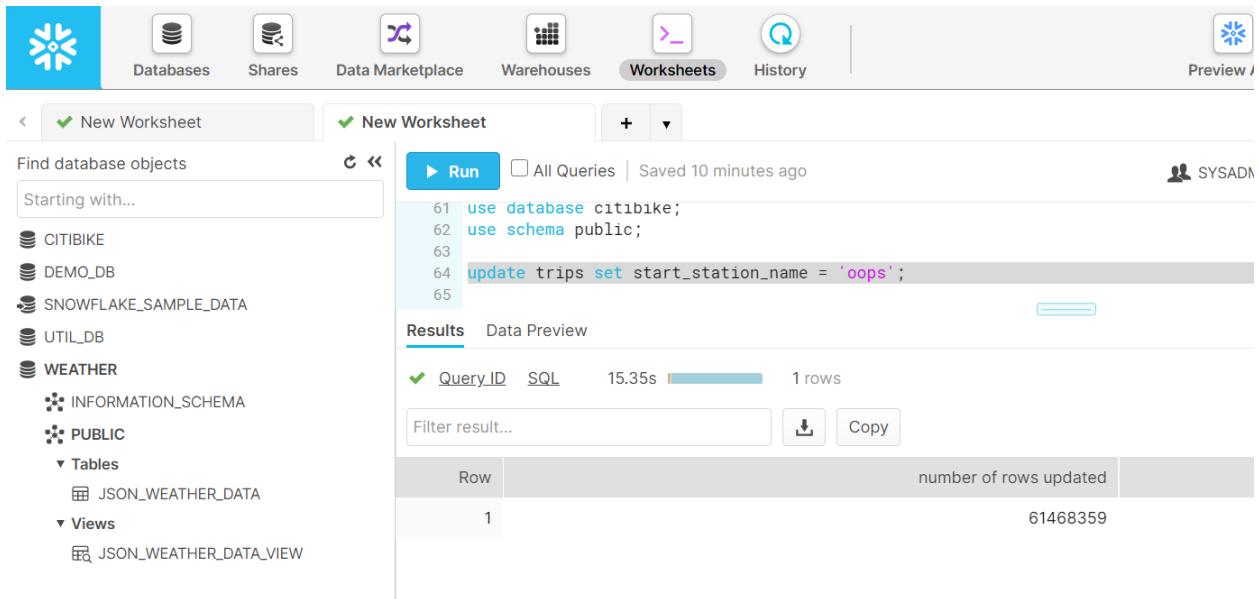
The "Results" tab is selected, showing a single row of output:

Row	status
1	Statement executed successfully.

At the bottom of the interface, there is a session context panel:

Role	SYSADMIN	Change
Warehouse	COMPUTE_WH (L)	Suspended
Large	Resize	Resume
Database	CITIBIKE	
Schema	PUBLIC	

Run the following command to replace all of the station names in the table with the word "oops".



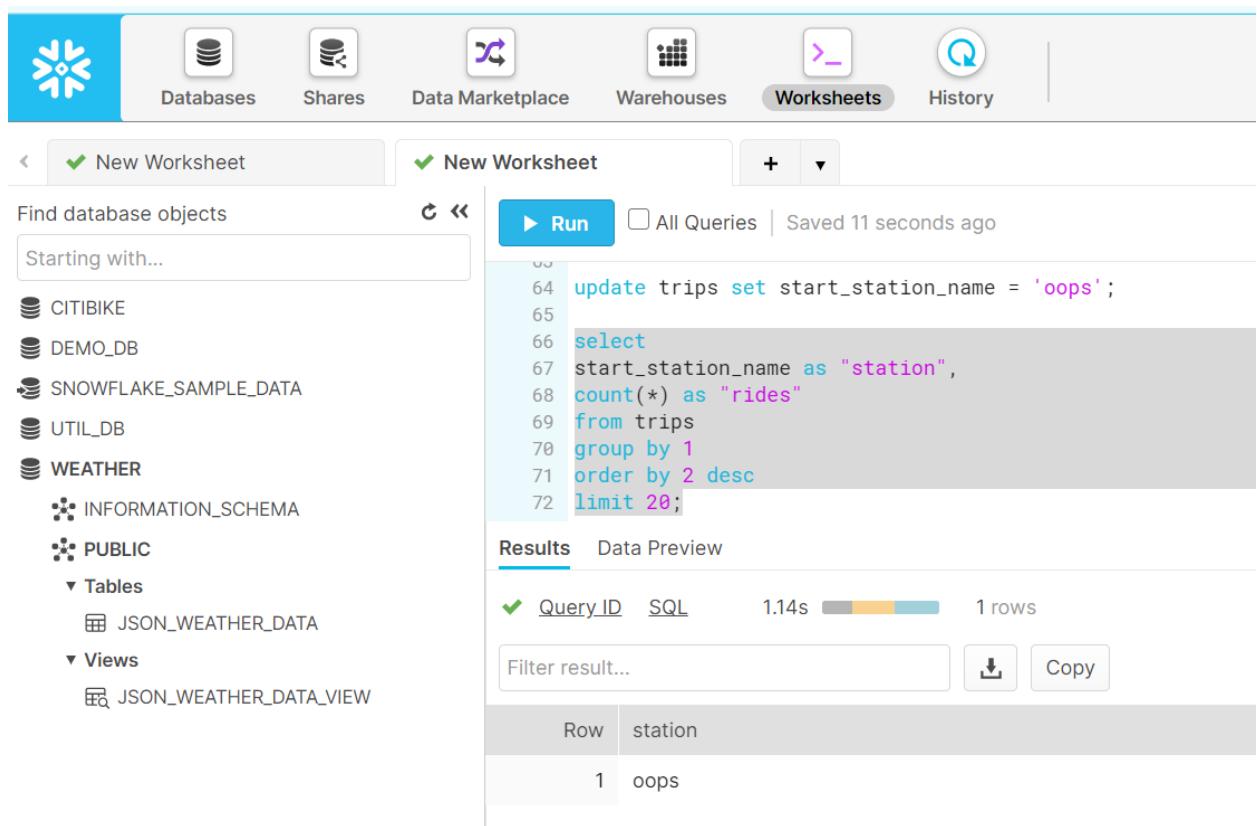
The screenshot shows the Snowflake UI interface. At the top, there are navigation icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (which is selected), and History. On the right, there's a 'Preview' button. Below the header, there are two tabs: 'New Worksheet' and another 'New Worksheet'. The main area shows a query editor with the following SQL code:

```
use database citibike;
use schema public;
update trips set start_station_name = 'oops';
```

The results section shows the output of the query:

Query_ID	SQL	Time	Rows
1	number of rows updated	15.35s	1 rows
			61468359

Run a query that returns the top 20 stations by number of rides. Notice that the station names result is only one row.



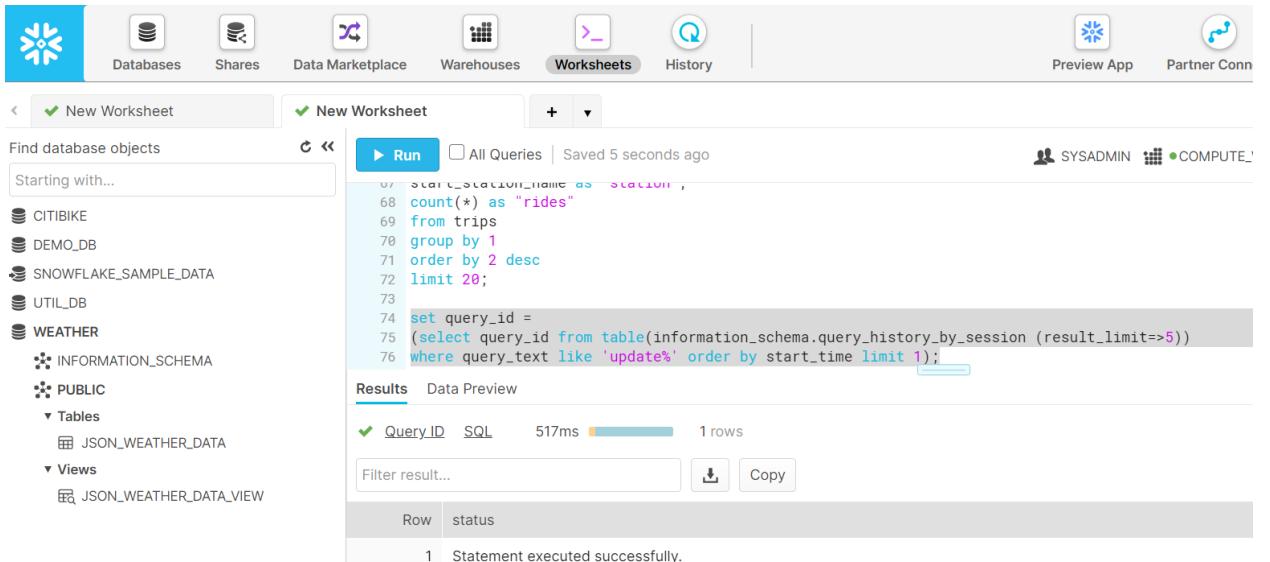
The screenshot shows the Snowflake UI interface. At the top, there are navigation icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (selected), and History. Below the header, there are two tabs: 'New Worksheet' and another 'New Worksheet'. The main area shows a query editor with the following SQL code:

```
update trips set start_station_name = 'oops';
select
    start_station_name as "station",
    count(*) as "rides"
from trips
group by 1
order by 2 desc
limit 20;
```

The results section shows the output of the query:

Query_ID	SQL	Time	Rows
1	station	1.14s	1 rows
	oops		

In Snowflake, we can simply run commands to find the query ID of the last UPDATE command and store it in a variable called \$QUERY_ID.

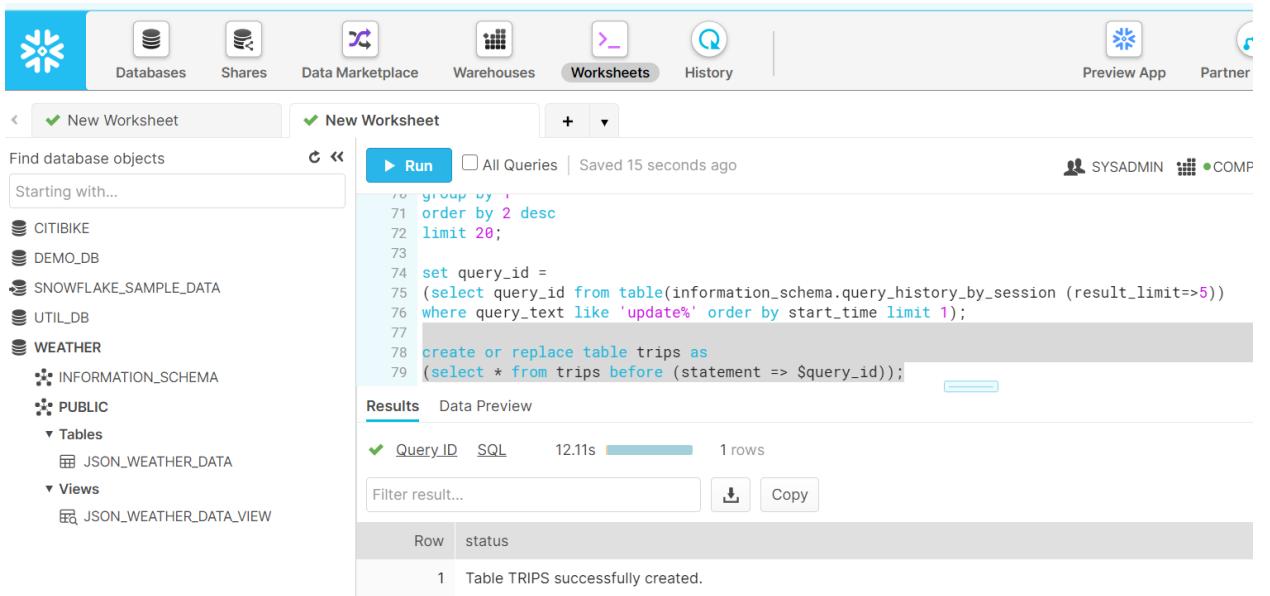


The screenshot shows the Snowflake interface with the 'Worksheets' tab selected. On the left, a sidebar lists databases like CITIBIKE, DEMO_DB, SNOWFLAKE_SAMPLE_DATA, UTIL_DB, WEATHER, INFORMATION_SCHEMA, and PUBLIC. Under PUBLIC, there are tables (JSON_WEATHER_DATA) and views (JSON_WEATHER_DATA_VIEW). The main area contains a code editor with the following SQL:

```
67 start_station_name as "station",
68 count(*) as "rides"
69 from trips
70 group by 1
71 order by 2 desc
72 limit 20;
73
74 set query_id =
75 (select query_id from table(information_schema.query_history_by_session (result_limit=>5))
76 where query_text like 'update%' order by start_time limit 1);
```

The results show a single row with the message: "Statement executed successfully."

Re-create the table with the correct station names:



The screenshot shows the Snowflake interface with the 'Worksheets' tab selected. The sidebar and code editor are identical to the previous screenshot. The results show a single row with the message: "Table TRIPS successfully created."

Run the SELECT statement again to verify that the station names have been restored:

The screenshot shows the Snowflake Worksheet interface. The top navigation bar includes icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (selected), and History. On the right, there are links for Preview App, Partner Connect, Help, and the user SIDHIP SYSADMIN.

The main area has two tabs: "New Worksheet" and "New Worksheet". A search bar at the top left says "Find database objects Starting with...". The sidebar on the left lists databases: CITIBIKE, DEMO_DB, SNOWFLAKE_SAMPLE_DATA, UTIL_DB, and WEATHER, along with their respective schemas and tables like JSON_WEATHER_DATA and JSON_WEATHER_DATA_VIEW.

The central workspace contains a SQL query:

```

70 create or replace table trips as
71 (select * from trips before (statement => $query_id));
72
73 select
74 start_station_name as "station",
75 count(*) as "rides"
76 from trips
77 group by 1
78 order by 2 desc
79 limit 20;

```

The results pane shows the output of the query:

Row	station	rides
1	Pershing Square North	491951
2	E 17 St & Broadway	481065
3	W 21 St & 6 Ave	458626
4	8 Ave & W 31 St	438001
5	West St & Chambers St	432518
6	Broadway & E 22 St	421812
7	Lafayette St & E 8 St	397724
8	Broadway & E 14 St	394995
9	8 Ave & W 33 St	379843
10	W 41 St & 8 Ave	359838
11	Cleveland Pl & Spring St	358485
12	W 20 St & 11 Ave	352099

This screenshot shows the same Snowflake Worksheet interface as the first one, but with a different query result. The results pane displays a table with columns "Row" and "station" on the left, and "rides" on the right.

Row	station	rides
1	Pershing Square North	491951
2	E 17 St & Broadway	481065
3	W 21 St & 6 Ave	458626
4	8 Ave & W 31 St	438001
5	West St & Chambers St	432518
6	Broadway & E 22 St	421812
7	Lafayette St & E 8 St	397724
8	Broadway & E 14 St	394995
9	8 Ave & W 33 St	379843
10	W 41 St & 8 Ave	359838
11	Cleveland Pl & Spring St	358485
12	W 20 St & 11 Ave	352099

Role-Based Access Controls, Account Usage, and Account Admin

Create a New Role and Add a User

Go to the worksheet -> change to ACCOUNTADMIN role to create a new role.

ACCOUNTADMIN encapsulates the SYSADMIN and SECURITYADMIN system-defined roles. It is the top-level role in the system and should be granted only to a limited number of users in your account.

SYSADMIN role cannot create any new roles and hence if we try to create new role while being in SYSADMIN role we will get an error so we need to switch to ACCOUNTADMIN role.

The screenshot shows the Snowflake web interface. At the top, there are navigation icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (which is selected), and History. Below the header, there are two tabs labeled "New Worksheet". A search bar says "Find database objects Starting with...". On the left, a sidebar lists databases: CITIBIKE, DEMO_DB, SNOWFLAKE, SNOWFLAKE_SAMPLE_DATA, UTIL_DB, and WEATHER. The main area shows a query result for "use role accountadmin;". The results table has columns for Query_ID, SQL, and status, showing 1 row with the message "Statement executed successfully.". A modal dialog is open in the foreground, titled "Role ACCOUNTADMIN Change". It shows the current warehouse set to ANALYTICS_WH (L) Suspended. There are buttons for "Large Resize" and "Resume". Below that, it shows "Database Select Database" and "Schema Schema".

Assigning my username to the newly created user role - junior_dba

Create the role and add your username to it:

The screenshot shows the Snowflake Worksheets interface. At the top, there are several navigation icons: Databases, Shares, Data Marketplace, Warehouses, Worksheets (which is selected), and History. Below the header, there are two tabs: 'New Worksheet' and another 'New Worksheet'. A search bar labeled 'Find database objects' with 'Starting with...' dropdown is present. On the left, a list of databases is shown: CITIBIKE, DEMO_DB, SNOWFLAKE, SNOWFLAKE_SAMPLE_DATA, UTIL_DB, and WEATHER. In the main workspace, a query has been run:

```
2 create role junior_dba;
3 grant role junior_dba to user SIDDHIP;
```

The results show one row with the status 'Statement executed successfully.'

Changing worksheet context to new Junior DBA role.

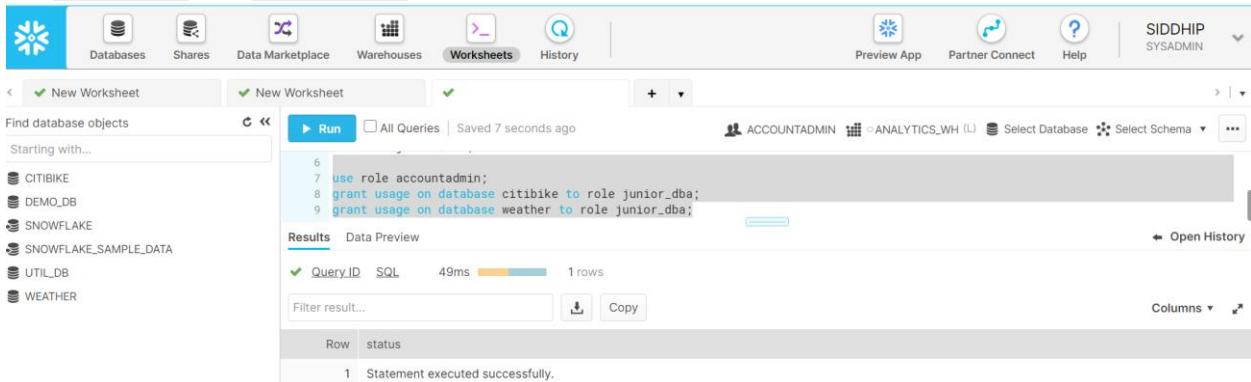
Also, in the left pane CITIBIKE and WEATHER databases no longer appear. This is because the junior_dba role does not have access to view them.

The screenshot shows the Snowflake Worksheets interface with a different user context. The top bar shows the user 'SIDDHIP' with roles 'SYSADMIN'. The left sidebar shows the same database list: DEMO_DB, SNOWFLAKE_SAMPLE_DATA, and UTIL_DB. The main workspace shows a query running:

```
4
5 use role junior_dba;
```

The results show one row with the status 'Statement executed successfully.'

Switch back to the ACCOUNTADMIN role and grant the junior_db to the ability to view and use the CITIBIKE and WEATHER databases:

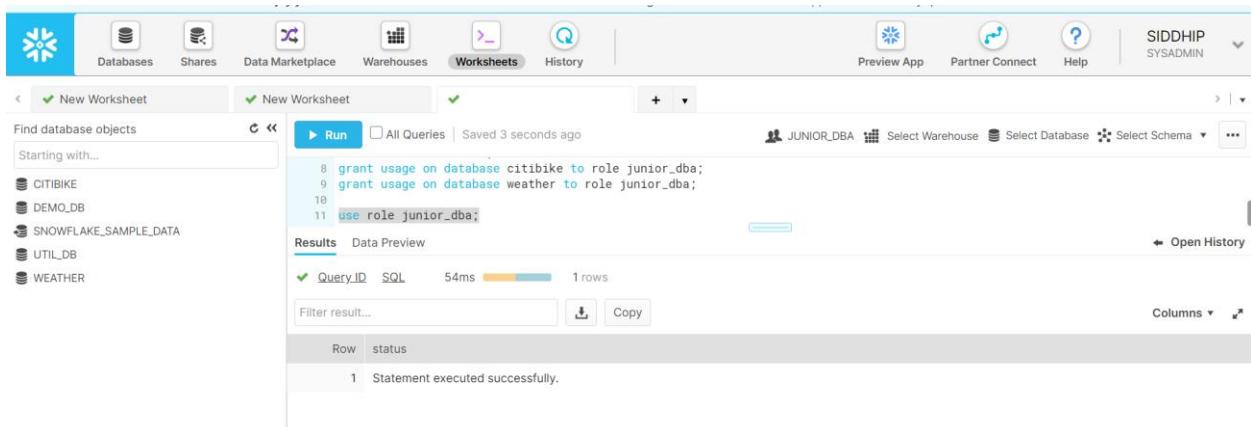


The screenshot shows the Snowflake Worksheet interface. The top navigation bar includes 'Databases', 'Shares', 'Data Marketplace', 'Warehouses', 'Worksheets' (selected), and 'History'. On the right, it shows 'SIDDHIP SYSADMIN'. The main area has two tabs: 'New Worksheet' and 'New Worksheet'. A 'Run' button is highlighted. The query editor contains the following SQL:

```
6
7 use role accountadmin;
8 grant usage on database citibike to role junior_db;
9 grant usage on database weather to role junior_db;
```

The results pane shows a single row with status 'Statement executed successfully.'

Switching to junior admin role, now we can view CITIBIKE and WEATHER databases on the left side pane.



The screenshot shows the Snowflake Worksheet interface. The top navigation bar includes 'Databases', 'Shares', 'Data Marketplace', 'Warehouses', 'Worksheets' (selected), and 'History'. On the right, it shows 'SIDDHIP SYSADMIN'. The main area has two tabs: 'New Worksheet' and 'New Worksheet'. A 'Run' button is highlighted. The query editor contains the following SQL:

```
8 grant usage on database citibike to role junior_db;
9 grant usage on database weather to role junior_db;
10
11 use role junior_db;
```

The results pane shows a single row with status 'Statement executed successfully.'

Account Administrator View

In the top right corner of the UI, click on your username to show the User Preferences menu. Go to Switch Role, then select the ACCOUNTADMIN role.

The screenshot shows the Snowflake web interface. At the top, there's a navigation bar with icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (which is selected), and History. On the far right of the top bar, the user's name 'SIDHHP' and role 'SYSADMIN' are displayed, along with links for Preview App, Partner Connect, Help, Change Password, Switch Role, Preferences, and Log Out. Below this, a sidebar shows 'Find database objects' with options like CITIBIKE, DEMO_DB, SNOWFLAKE_SAMPLE_DATA, UTIL_DB, and WEATHER. The main workspace shows a query editor with the following SQL code:

```
8 grant usage on database citibike to role junior_dba;
9 grant usage on database weather to role junior_dba;
10
11 use role junior_dba;
```

The results pane indicates 1 row was returned in 54ms. At the bottom of the interface, there are buttons for Filter result..., Download, and Copy.

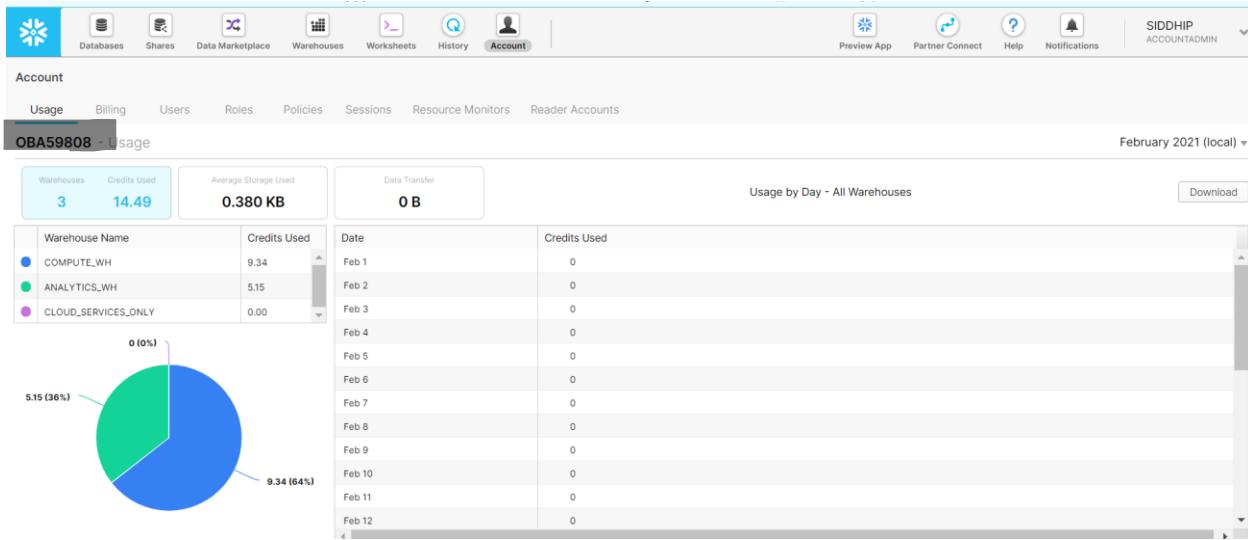
Roles in user preference can be simultaneously different from the role used in the worksheets. Also, we can see that a new tab – ‘ACCOUNT’ has been added at the top which is only added for ACCOUNTADMIN user.

This screenshot shows the expanded User Preferences menu for the user 'SIDHHP' who is now listed as 'ACCOUNTADMIN'. The menu includes options for Change Password, Switch Role, Preferences, and Log Out. To the left, a sidebar lists roles: ACCOUNTADMIN (selected), JUNIOR_DBA, and SYSADMIN (Default). The top navigation bar remains the same as the previous screenshot.

This screenshot shows the Snowflake UI with the 'Account' tab selected in the top navigation bar. The user 'SIDHHP' is now listed as 'ACCOUNTADMIN'. The main workspace shows a query editor with the same SQL code as before. The top navigation bar also includes a 'Notifications' icon.

Click on Account tab -> Click on Usage -> Information on credits, storage, and daily or hourly usage for each warehouse, including cloud services. Select a day to review its usage.

Information on Users, Roles, and Resource Monitors is also present. The latter set limits on your account's credit consumption so you can appropriately monitor and manage your credits.



Secure Data Sharing & Data Marketplace

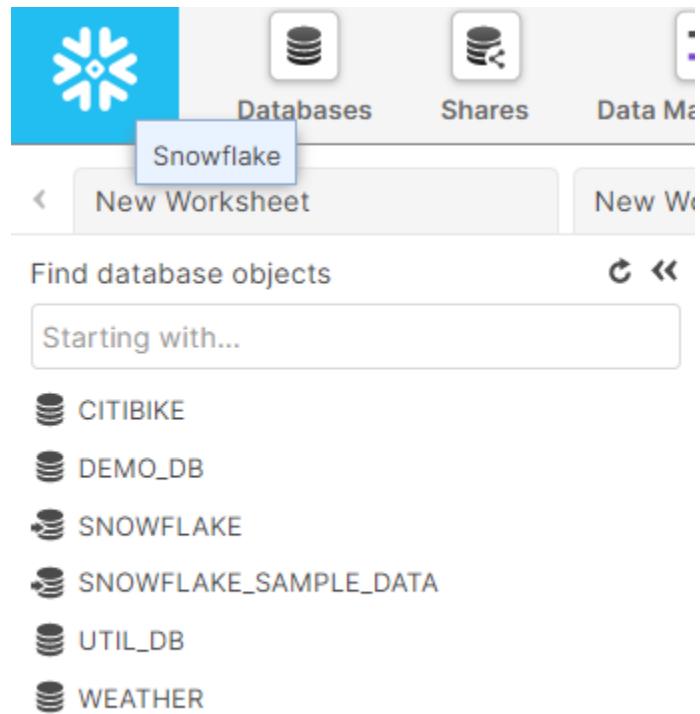
We can access data as shares in snowflake. Provider provides data as shares which consumers(external entity or different internal business unit) import. There can be snowflake reader account or consumer's own account to import data.

With Secure Data Sharing:

- There is only one copy of the data that lives in the data provider's account
- Shared data is always live, real-time, and immediately available to consumers
- Providers can establish revocable, fine-grained access to shares
- Data sharing is simple and safe, especially compared to older data sharing methods which were often manual and insecure, involving the transfer of large .csv files across the internet

See Existing Shares

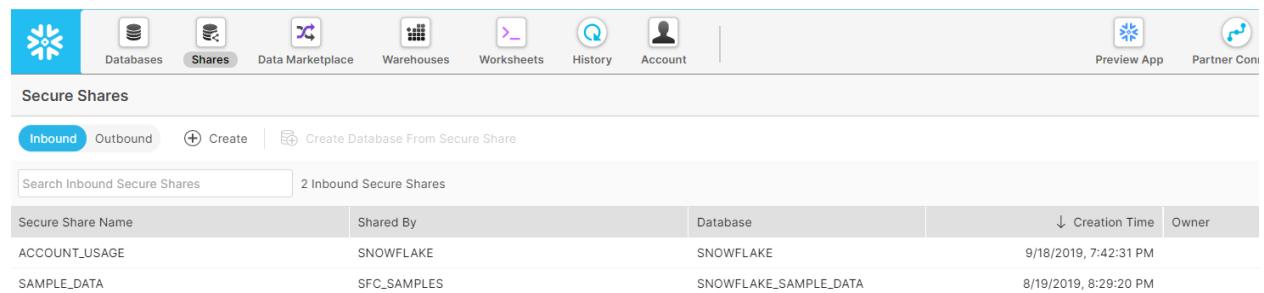
Click on the blue Snowflake logo -> On the left side of the page in the database object browser, notice the database SNOWFLAKE_SAMPLE_DATA. The small arrow on the database icon indicates this is a share.



The screenshot shows the Snowflake Database Object Browser interface. At the top, there are three main tabs: 'Databases' (selected), 'Shares' (disabled), and 'Data Marketplace'. Below the tabs, there are two buttons: 'New Worksheet' and 'New Work'. A search bar labeled 'Find database objects' with placeholder text 'Starting with...' is present. The main list displays several databases: CITIBIKE, DEMO_DB, SNOWFLAKE, SNOWFLAKE_SAMPLE_DATA (which has a small arrow icon indicating it is shared), UTIL_DB, and WEATHER.

Database	Shared By	Database	Creation Time	Owner
ACCOUNT_USAGE	SNOWFLAKE	SNOWFLAKE	9/18/2019, 7:42:31 PM	
SAMPLE_DATA	SFC_SAMPLES	SNOWFLAKE_SAMPLE_DATA	8/19/2019, 8:29:20 PM	

We can go to shares tab to view the existing shares. Our account acts as consumer of these shares.



The screenshot shows the 'Secure Shares' page in the Snowflake interface. The top navigation bar includes icons for Snowflake, Databases, Shares (selected), Data Marketplace, Warehouses, Worksheets, History, and Account, along with links for Preview App and Partner Connect. Below the navigation, a header bar shows 'Inbound' selected, with options for 'Outbound', 'Create', and 'Create Database From Secure Share'. A search bar 'Search Inbound Secure Shares' shows '2 Inbound Secure Shares'. The main table lists the inbound secure shares:

Secure Share Name	Shared By	Database	Creation Time	Owner
ACCOUNT_USAGE	SNOWFLAKE	SNOWFLAKE	9/18/2019, 7:42:31 PM	
SAMPLE_DATA	SFC_SAMPLES	SNOWFLAKE_SAMPLE_DATA	8/19/2019, 8:29:20 PM	

Create an Outbound Share

Create an outbound share to give access to your company's trusted partner in order to analyze the data. Also, not extra copies of the data was created. One secure data can be accessed by multiple people.

We can also share UDF's, views and secure joins. We can share data by preventing access to sensitive information as well.

Create a Secure Share and add Database objects to it

X

A Secure Share is a package or container comprising Database objects that contain the data you want to share. Once you setup Tables or Secure Views in the Database, ready with the data you intend to share, you can add them to the Secure Share.

Have you prepared the data? Identify and prepare data you want to include in the Secure Share. [Learn more](#) about preparing data.

Create

Select a Database and Schemas. Select Tables or Views within each Schema to add to the Secure Share.

Secure Share Name

Database ▼

Tables & Views 1 Table, 0 Secure Views

CITIBIKE.PUBLIC.TRIPS ✖ [Remove All](#)

Comment

[Show SQL](#)[Cancel](#)Create

Secure share was created successfully. Account Administrator would click on the Next: Add Consumers button to input their partner's Snowflake account name and type.

[Edit Share](#)

Review the Secure Share, Preview Tables & Validate Secure Views

[X](#)

Help

_SHARI

Before you add consumer accounts to access the Secure Share, it's always a good practice to review the contents of your share and validate the data they will see.

✓ "TRIPS_SHARE" Secure share was created successfully!

[Data Preview](#) [Secure Share Overview](#)

Warehouse ANALYTICS_WH (L) Suspended (auto resume)

Large [Resize](#)[Resume](#)

Table or View Select a Table or View

[Preview Data](#)

With a warehouse running, we run

```
SELECT * FROM <NAME> LIMIT 10;
```

to generate a preview of the data in your Table

[Show SQL](#)[Done](#)[Next: Add Consumers](#)

The screenshot shows the Snowflake interface with the 'Shares' tab selected in the top navigation bar. The main area displays a table titled 'Secure Shares' under the 'Outbound' tab. The table has columns: Secure Share Name, Shared With, Database, Creation Time, Owner, and Comment. One row is selected, showing 'TRIPS_SHARE' as the name, 'CITIBIKE' as the database, and 'ACCOUNTADMIN' as the owner. To the right of the table, a modal window provides detailed information about the selected share, including its type (Outbound), owner (ACCOUNTADMIN), creation time (2/22/2021), and database (CITIBIKE). The top right corner of the modal shows the user's name (SIDDHIP ACCOUNTADMIN) and a refresh icon.

Secure Share Name	Shared With	Database	Creation Time	Owner	Comment
TRIPS_SHARE	Add Consumers	CITIBIKE	11:06:40 PM	ACCOUNTADMIN	

Last refreshed 11:06:40 PM↻

TRIPS_SHARE

Type: Outbound
Owner: ACCOUNTADMIN
Creation Time: 2/22/2021
Database: CITIBIKE

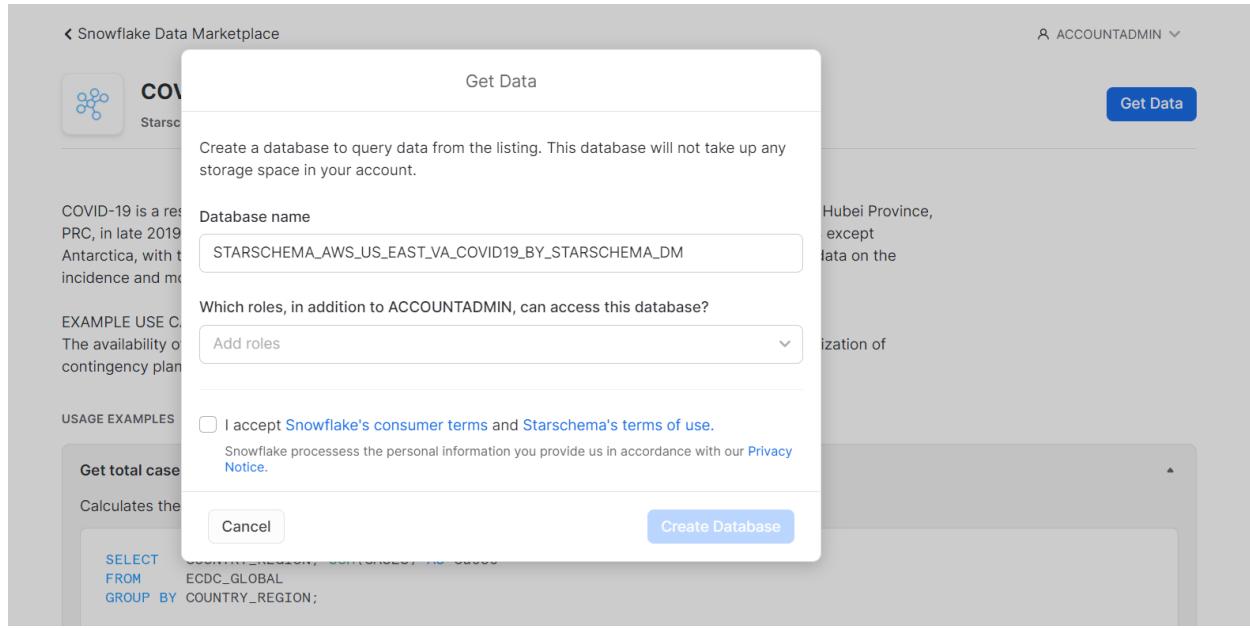
Snowflake Data Marketplace

Go to Data Marketplace tab -> Enter your credentials to login -> view categories of data available on top -> click on 'Health'

The screenshot shows the Snowflake Data Marketplace interface. At the top, there is a navigation bar with icons for Databases, Shares, Data Marketplace (which is highlighted in blue), Warehouses, Worksheets, History, Account, Preview App, Partner Connect, Help, Notifications, and a user account labeled 'SIDDHIP ACCOUNTADMIN'. Below the navigation bar is a large central callout box with a blue gradient background. The text inside reads: 'Discover new external data. Drive business insights, instantly, with data from the world's best data and software providers on the Snowflake Data Marketplace.' It includes a button labeled 'Explore the Snowflake Data Marketplace →' and a note 'Now, on the new Snowflake Experience'. Below this callout is a grid of eight placeholder cards, each showing a blue gradient icon and a small preview area. The main content area below the callout has a search bar with the placeholder 'Search Data Marketplace', a 'My Requests' button, and filters for 'Providers All' and 'Filters'. The main content area is titled 'Health' and displays four data provider cards: 'S&P Global Market Intelligence' (MedMine Medical Device Transactions), 'IQVIA Data-as-a-Service' (Fit-for-purpose global data at scale), 'Starschema' (COVID-19 Epidemiological Data), and 'Knoema' (COVID-19 Data Atlas). Each card provides a brief description and a 'Personalized' button.

We will use Starschema's COVID-19 Epidemiological Data. Make sure you are in the **ACCOUNTADMIN** role.

select the data and click on the Get Data button to access this information within your Snowflake Account.



Select all roles and click on terms -> click on create database

Get Data

Create a database to query data from the listing. This database will not take up any storage space in your account.

Database name

STARSCHEMA_AWS_US_EAST_VA_COVID19_BY_STARSCHEMA_DM

Which roles, in addition to ACCOUNTADMIN, can access this database?

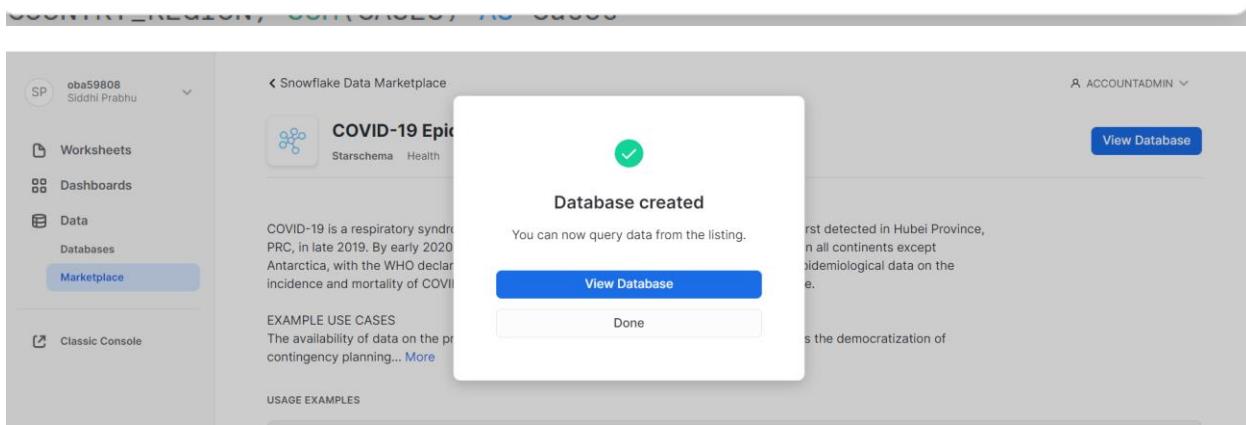
JUNIOR_DB, PUBLIC, SECURITYADMIN, SYSADMIN, USERADMIN

I accept [Snowflake's consumer terms](#) and [Starschema's terms of use](#).

Snowflake processes the personal information you provide us in accordance with our [Privacy Notice](#).

[Cancel](#)

[Create Database](#)



We can view Database Details including the owner of the share, number of tables and views in this database, source, share name, and data provider.

View the schemas and details. Select PUBLIC Schema to view available tables.

ACCOUNTADMIN ▾

- ▶ CITIBIKE
- ▶ DEMO_DB
- ▶ SNOWFLAKE
- ▶ SNOWFLAKE_SAMPLE_DATA
- ▶ UTIL_DB
- ▶ WEATHER
- ▶ STARSHEMA_AWS_US_EAST_VA_COVID19_BY_STARSHEMA_DM
- ▶ INFORMATION_SCHEMA
- ▶ PUBLIC

Databases

STARSHEMA_AWS_US_EAST_VA_COVID19_BY_STARSHEMA_DM

[Database Details](#) [Schemas](#)

About

Type	Database
Owner	ACCOUNTADMIN
Created	just now

Source details

Source	Share
Share	COVID19_BY_STARSHEMA_DM
Shared by	STARSHEMA_AWS_US_EAST_VA

Databases / STARSHEMA_AWS_US_EAST_VA_COVID19_BY_STARSHEMA_DM

PUBLIC

[Schema Details](#) [Tables](#) [Views](#) [Stages](#) [Data Pipelines](#) [Functions](#) [Procedures](#)

32 tables

NAME ↑	TYPE	OWNER	ROWS	BYTES	CREATED
CT_US_COVID_TESTS	Table	—	20.0K	640.0KB	11 months ago
DATABANK_DEMOGRAPHICS	Table	—	216	9.0KB	10 months ago
DEMOGRAPHICS	Table	—	3.1K	124.0KB	11 months ago
ECDC_GLOBAL	Table	—	61.9K	639.5KB	9 months ago
GOOG_GLOBAL_MOBILITY_RE...	Table	—	4.3M	39.4MB	10 months ago
HDX_ACAPS	Table	—	23.9K	2.8MB	10 months ago
HS_BULK_DATA	Table	—	727.8K	18.8MB	11 months ago
HUM_RESTRICTIONS_AIRLINE	Table	—	651	64.0KB	10 months ago
HUM_RESTRICTIONS_COUNT...	Table	—	235	601.5KB	10 months ago
IHME_COVID_19	Table	—	189.2K	31.9MB	6 months ago

Type JHU in the search bar to view all tables with data sourced from [John Hopkins University](#). We have now successfully subscribed to the COVID-19 dataset from StarSchema which is updated daily with global COVID data.

Databases / STARSCHHEMA_AWS_US_EAST_VA_COVID19_BY_STARSCHHEMA_DM

PUBLIC

Schema Details **Tables** Views Stages Data Pipelines Functions Procedures

4 tables

X All Tables C

NAME ↑	TYPE	OWNER	ROWS	BYTES	CREATED
JHU_COVID_19	Table	—	3.6M	37.7MB	11 months ago
JHU_COVID_19_TIMESERIES	Table	—	4.3M	51.9MB	9 months ago
JHU_DASHBOARD_COVID_19_G...	Table	—	4.6K	214.5KB	10 months ago
JHU_VACCINES	Table	—	4.0K	66.0KB	1 month ago

Resetting Your Snowflake Environment

First set the worksheet context:

```
1 use role accountadmin;
2 use warehouse compute_wh;
3 use database weather;
4 use schema public;
```

Results Data Preview

✓ [Query ID](#) [SQL](#) 38ms  1 rows

Filter result...



Copy

Row	status
1	Statement executed successfully.

Then run this SQL to drop all the objects we created in the lab:

The screenshot shows the Snowflake Worksheet interface. The top navigation bar includes icons for Databases, Shares, Data Marketplace, Warehouses, Worksheets (selected), History, and Account. Below the navigation is a search bar labeled "Find database objects" with "Starting with..." and a dropdown menu listing databases: CITIBIKE, DEMO_DB, SNOWFLAKE, SNOWFLAKE_SAMPLE_DATA, STARSCHHEMA_AWS_US_EAST_VA_COVID19_B..., UTIL_DB, and WEATHER. The main workspace contains a query editor with the following SQL code:

```
1 use role accountadmin;
2 use warehouse compute_wh;
3 use database weather;
4 use schema public;
5
6 drop share if exists trips_share;
7 drop database if exists citibike;
8 drop database if exists weather;
9 drop warehouse if exists analytics_wh;
10 drop role if exists junior_dba;
```

The results section shows a table with one row, indicating the success of the query:

Row	status
1	JUNIOR_DBA successfully dropped.