Content

Introduction

Snowflake is a cloud-based advanced data platform system, provided as Software-as-a-Service (SaaS). Snowflake provides features of data storage from AWS S3, Azure, and Google Cloud, processing complex queries and different analytic solutions. The analytic solutions provided by Snowflake are faster, easy to use, and more flexible than traditional databases and their analytics features. Snowflake stores and provides data near time not in actual real time.

Snowflake is an advanced solution for OLAP (Online Analytical Processing) technology. OLAP is also known as an online data retrieving and data analysis system using historical data. It processes complex and aggregated queries with a low number of transactions. For Ex: Getting the number of orders, sales amount in the last month for a company, number of new users listed in the company in the last quarter, etc. Snowflake is not used as an OLTP (Online Transactional Processing) database. OLTP databases usually contain real-time data with a high volume of small data transactions. For Ex: Inserting customer's order details, registering a new customer, tracking order delivery status, etc.

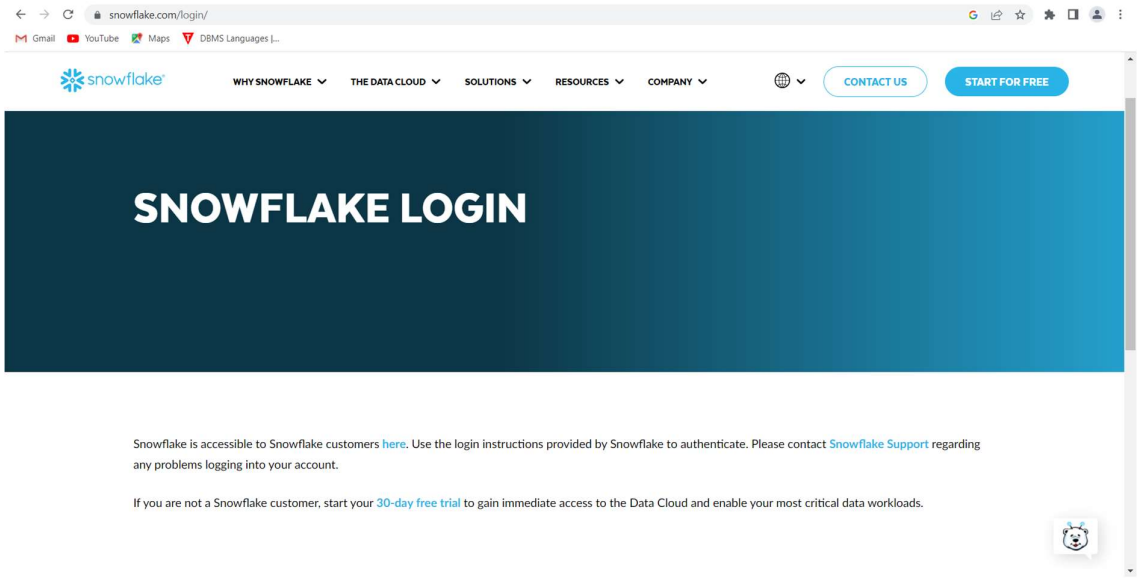
Why Use Snowflake?

Snowflake provides a Data Platform as a Cloud Service.

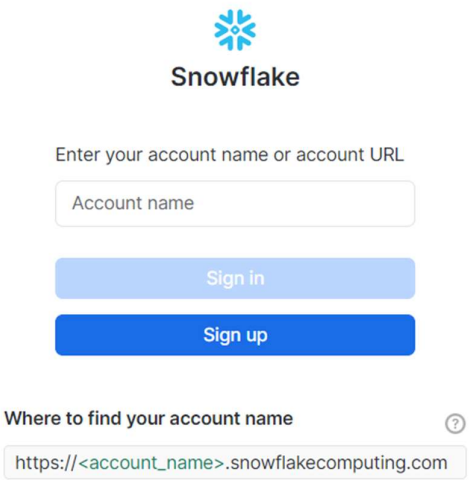
- There is no hardware neither virtual nor physical to select, install, configure, or manage from the client side.
- There is no software to install, configure, or manage to access it.
- All ongoing maintenance, management, upgrades, and patching are owned by Snowflake itself.

Create a Snowflake Account

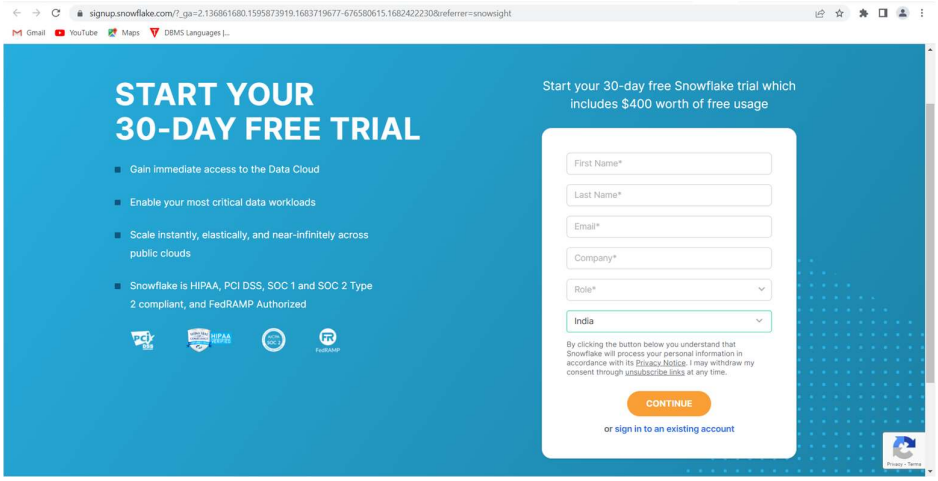
1. To create a Snowflake account please click on this link https://www.snowflake.com/login/
2. Next you will see a window like below, click on the Snowflake customer here option as shown below.



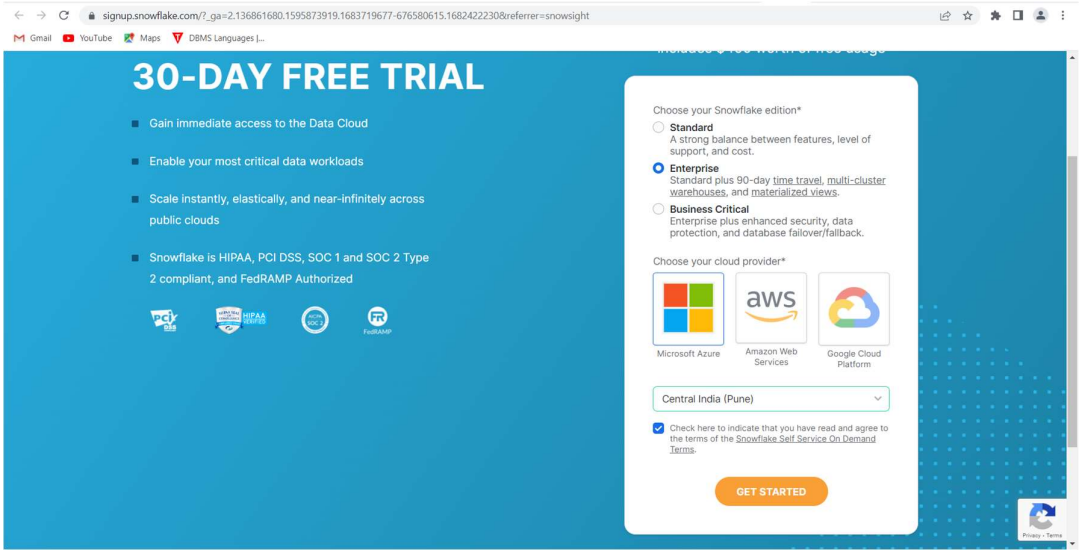
3. Next click on the signup option as shown below.



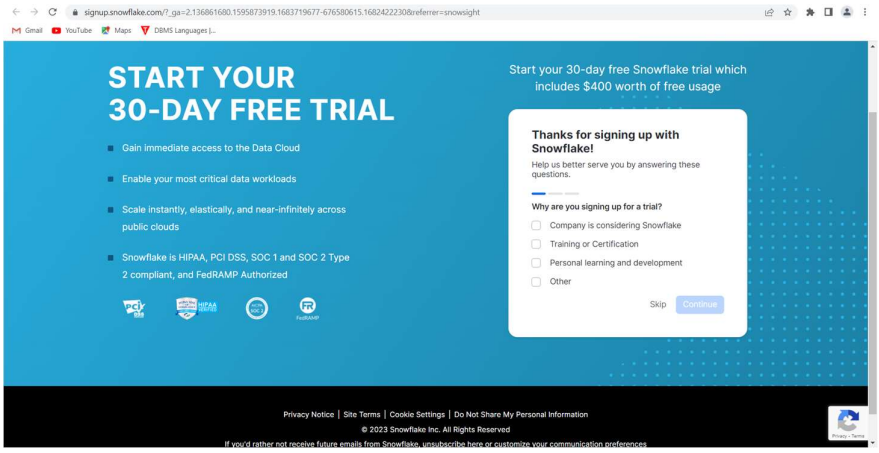
4. Next fill up the following details and click on continue as shown below.



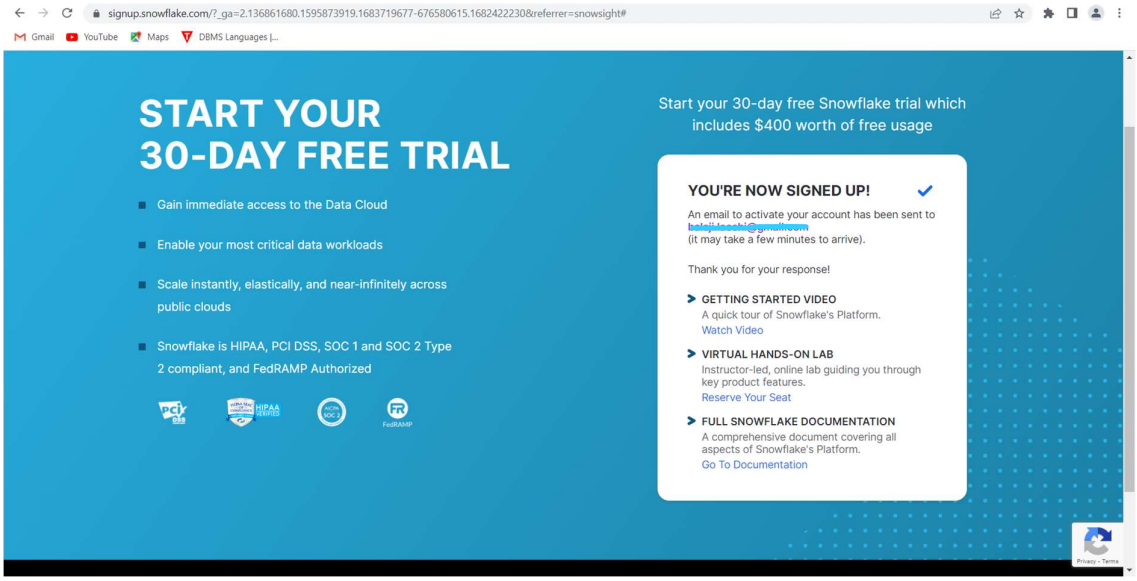
5. Next check the enterprise edition click on Microsoft Azure give central India(Pune) and click on Get Started as shown below.



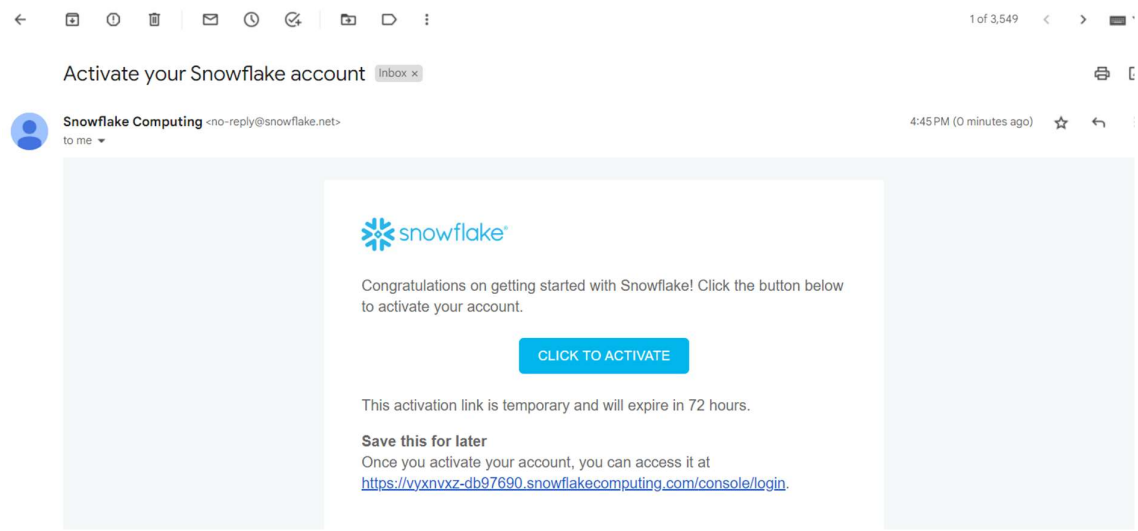
6. Skip the conditions as shown below.



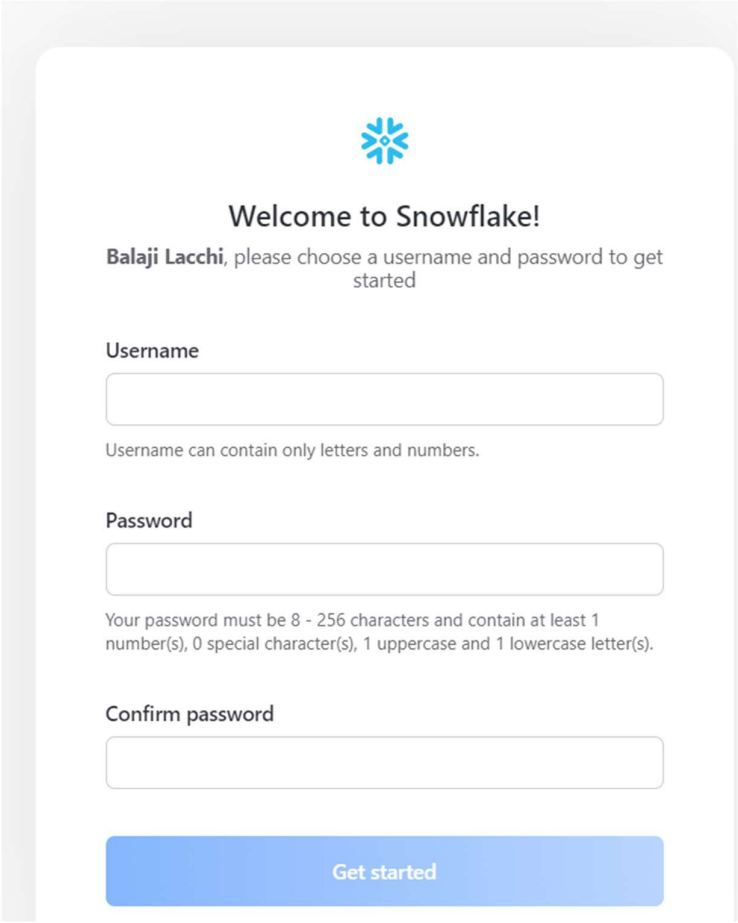
7. Next you will see a message like this with your mail ID as shown below.



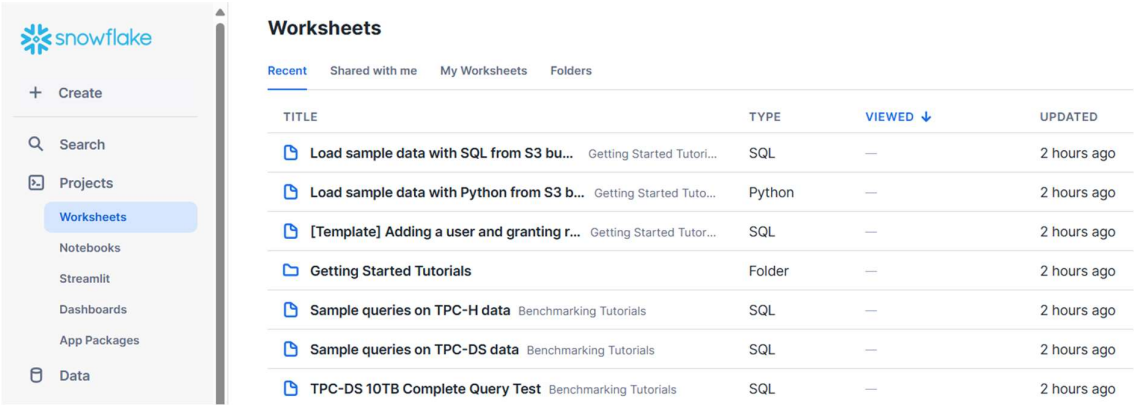
8. Next you will receive one mail from Snowflake below click on the Click to Activate option as shown below.



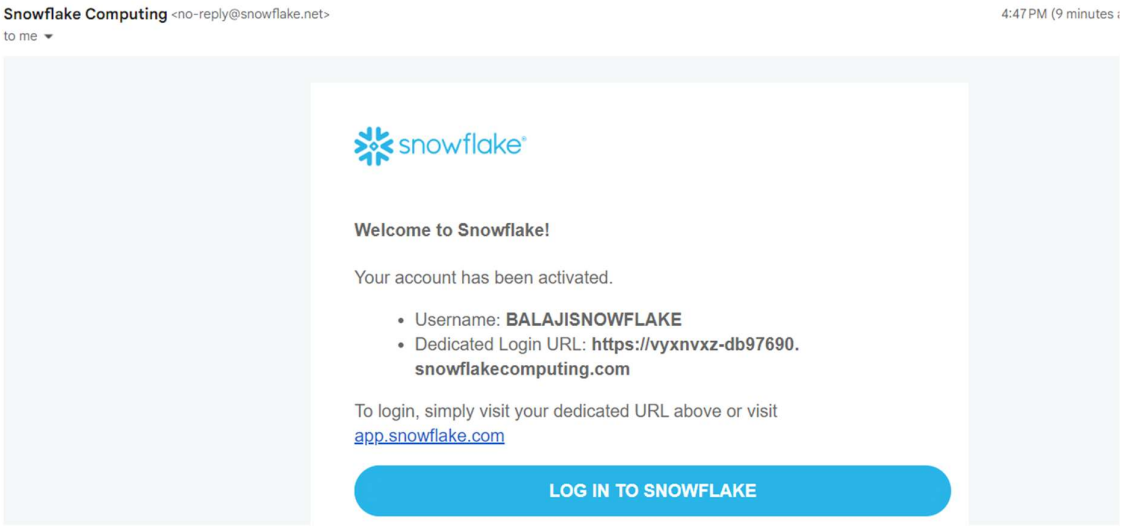
9. Now give the username and password then confirm the password as shown below.
10. Next click on Get Started.



11. It will take you to the snowflake home window as shown below.



12. In case you don't see the above window. You will receive another mail from Snowflake like below.
13. Just click on login then give Username and password.



Create Database

1. Write the following query to create a database "SAMPLE_SNOW"

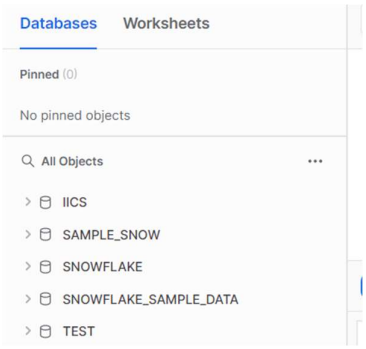
Query: Create database SAMPLE_SNOW

```
1    Create DATABASE "SAMPLE_SNOW"
```

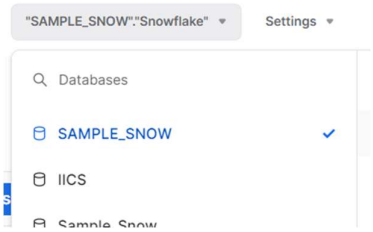
Results | Chart

	status
1	Database SAMPLE_SNOW successfully created.

2. As you can see on the left side Our database has been created.



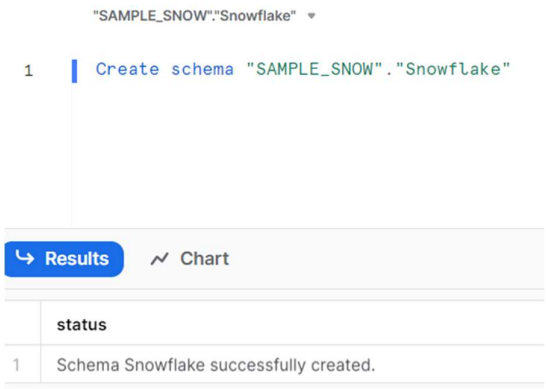
3. Now on the top of the workspace select the Database as shown below.



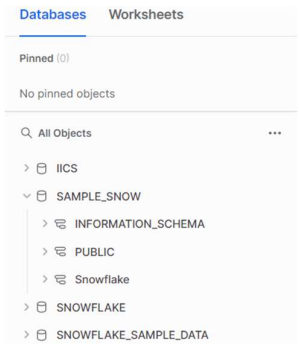
Create Schema

1. Write the following query to create a schema Snowflake under the database SAMPLE_SNOW.

Query: CREATE SCHEMA "SAMPLE_SNOW"."Snowflake"



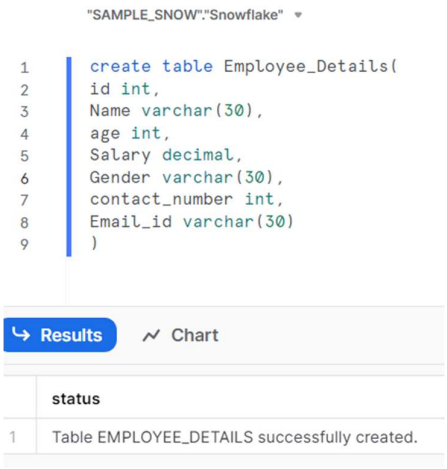
2. As you can see under the SAMPLE_SNOW database Snowflake schema has been created.



Create TABLE and COLUMNS

1. Use the following query to create a table and columns under the database SAMPLE_SNOW and schema Snowflake.

Query: create table Employee_Details(id int,
Name varchar(30),
age int,
Salary decimal,
Gender varchar(30),
contact_number int,
Email_id varchar(30)
)



2. Now Insert some sample data into our table using the Insert command as shown below.

Query: insert into Employee_Details(id, Name, age, Salary, Gender,contact_number, Email_id) values (1,'sagar',24,120000,'male',89942345,'sagar@gmail.com');

insert into Employee_Details(id, Name, age, Salary, Gender,contact_number, Email_id) values (2,'ratan',22,130000,'male',80052345,'ratan@gmail.com') ;

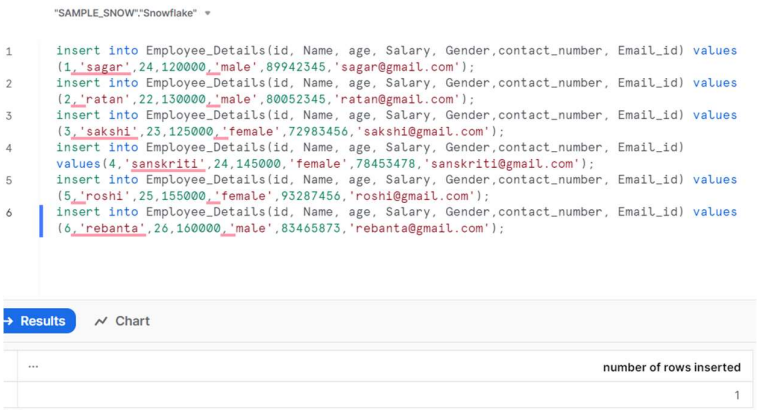
insert into Employee_Details(id, Name, age, Salary, Gender,contact_number, Email_id) values (3,'sakshi',23,125000,'female',72983456,'sakshi@gmail.com');

insert into Employee_Details(id, Name, age, Salary, Gender,contact_number, Email_id) values(4,'sanskriti',24,145000,'female',78453478,'sanskriti@gmail.com');

insert into Employee_Details(id, Name, age, Salary, Gender,contact_number, Email_id) values (5,'roshi',25,155000,'female',93287456,'roshi@gmail.com');

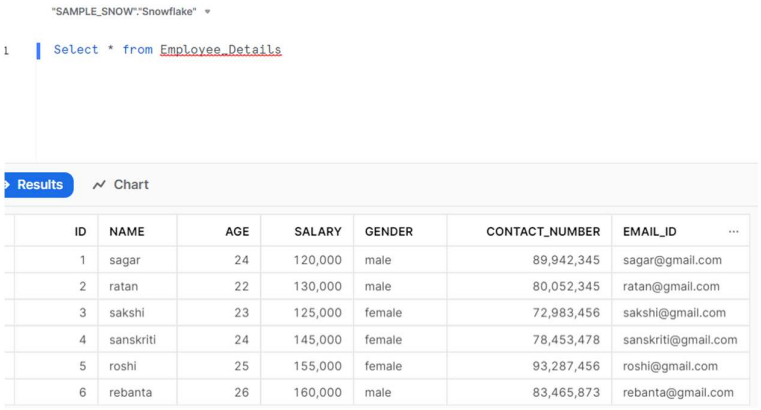
insert into Employee_Details(id, Name, age, Salary, Gender,contact_number, Email_id) values (6,'rebanta',26,160000,'male',83465873,'rebanta@gmail.com');

3. Copy the query and paste it in the workspace then click on Run.



4. Use to check the table use SELECT command.

Query: Select * from Employee_Deatils



ID	NAME	AGE	SALARY	GENDER	CONTACT_NUMBER	EMAIL_ID
1	sagar	24	120,000	male	89,942,345	sagar@gmail.com
2	ratan	22	130,000	male	80,052,345	ratan@gmail.com
3	sakshi	23	125,000	female	72,983,456	sakshi@gmail.com
4	sanskriti	24	145,000	female	78,453,478	sanskriti@gmail.com
5	roshi	25	155,000	female	93,287,456	roshi@gmail.com
6	rebanta	26	160,000	male	83,465,873	rebanta@gmail.com

Types of Snowflake Tables

Tables are database objects logically structured as a collection of rows and columns. All data in Snowflake is stored in database tables. Apart from standard database tables, Snowflake supports other table types that are especially useful for storing data that does not need to be maintained for extended periods of time.

Snowflake supports three types of tables

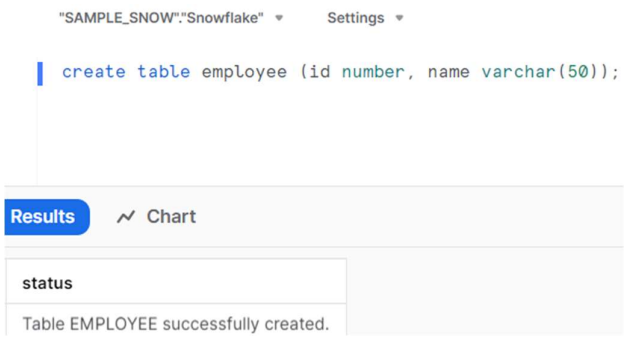
- Permanent table
- Transient table
- Temporary table

Permanent Table

These are the standard, regular database tables. Permanent tables are the default table type in Snowflake and do not need any additional syntax while creating to make them permanent.

1. The data stored in permanent tables consumes space and contributes to the storage charges that Snowflake bills your account.
2. It also comes with additional features like Time-Travel and Fail-Safe which helps in data availability and recovery.
3. To create a Permanent table in Snowflake.

Query: create table employee (id number, name varchar(50));



Transient Table

Transient tables in Snowflake are similar to permanent tables except that that they do not have a Fail-safe period and only have a very limited Time-Travel period. These are best suited in scenarios where the data in your table is not critical and can be recovered from external means if required.

1. To create a Transient table in Snowflake, You need to mention transient in the create table syntax.

Query: create transient table employee_transient (id number, name varchar(50));

Temporary Table

Temporary Tables in Snowflake exist only within the session in which they were created and are available only for the remainder of the session. They are not visible to other users or sessions. Once the session ends, data stored in the table is dropped completely and is not recoverable.

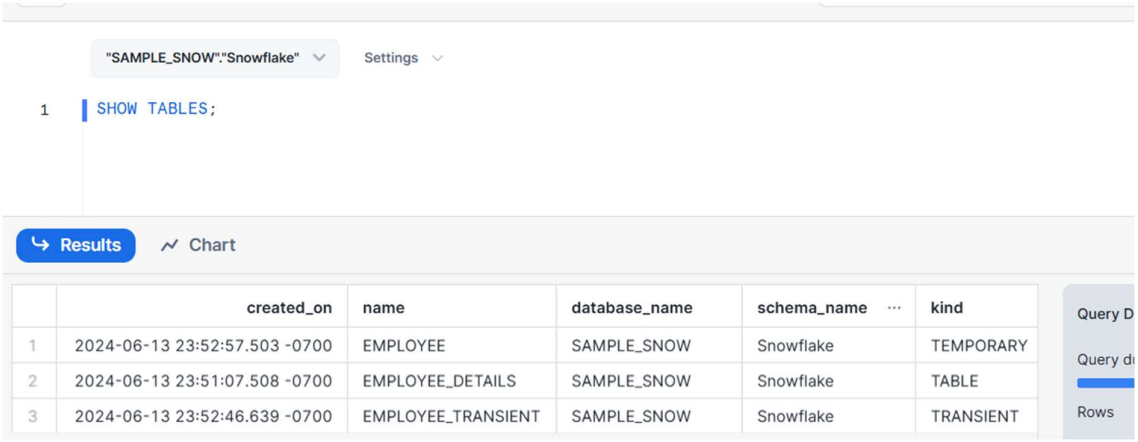
2. To create a Temporary table in Snowflake, you need to mention temporary in the create table syntax.

Query: create temporary table employee (id number, name varchar(50));

SHOW TABLES

1. Lists the tables for which you have access privileges, including dropped tables that are still within the Time Travel retention period and, therefore, can be undropped.
2. Syntax to execute show tables in Snowflake is as below.

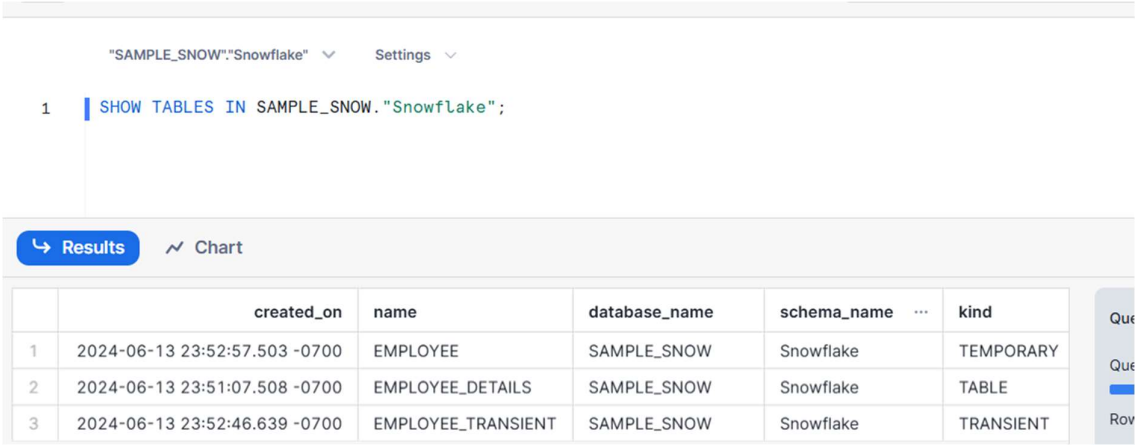
Query: SHOW TABLES;



	created_on	name	database_name	schema_name	kind
1	2024-06-13 23:52:57.503 -0700	EMPLOYEE	SAMPLE_SNOW	Snowflake	TEMPORARY
2	2024-06-13 23:51:07.508 -0700	EMPLOYEE_DETAILS	SAMPLE_SNOW	Snowflake	TABLE
3	2024-06-13 23:52:46.639 -0700	EMPLOYEE_TRANSIENT	SAMPLE_SNOW	Snowflake	TRANSIENT

3. To get table details from a particular database and Schema.

Query: SHOW TABLES IN SAMPLE_SNOW."Snowflake";



	created_on	name	database_name	schema_name	kind
1	2024-06-13 23:52:57.503 -0700	EMPLOYEE	SAMPLE_SNOW	Snowflake	TEMPORARY
2	2024-06-13 23:51:07.508 -0700	EMPLOYEE_DETAILS	SAMPLE_SNOW	Snowflake	TABLE
3	2024-06-13 23:52:46.639 -0700	EMPLOYEE_TRANSIENT	SAMPLE_SNOW	Snowflake	TRANSIENT

GET_DDL

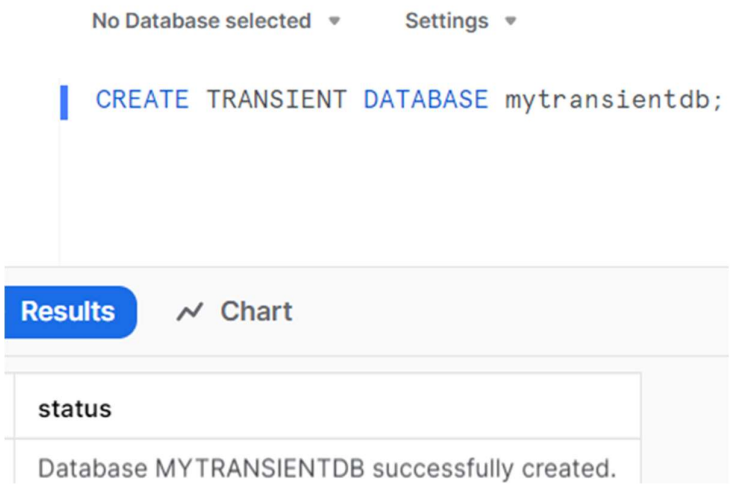
4. Returns a DDL statement that can be used to recreate the specified object. The DDL statement contains the type of the Snowflake table.
5. Syntax to get DDL of a table.

Query: select get_ddl('table','employee');

Create a transient database

1. To create a Transient database use the below query.

Query: CREATE TRANSIENT DATABASE mytransientdb;

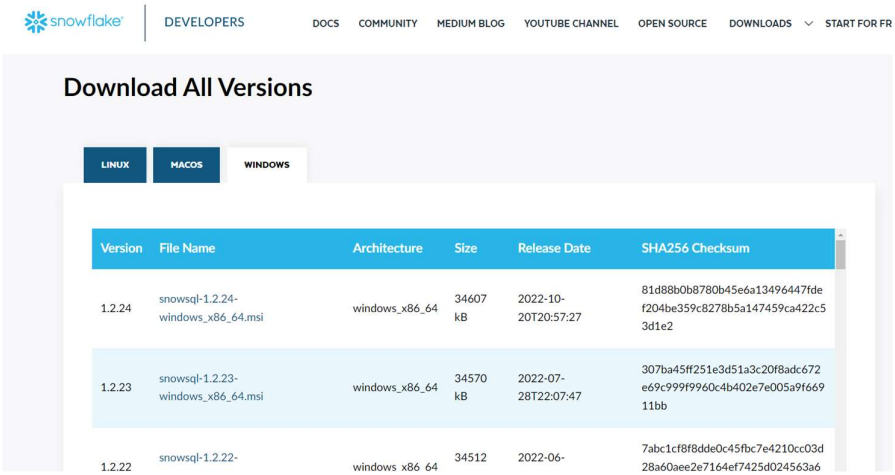


2. Show the database use below query.

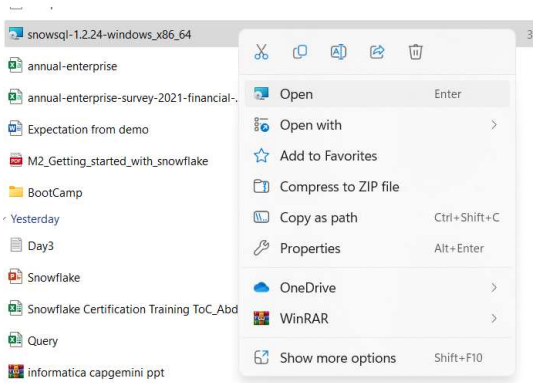
Query: SHOW DATABASES LIKE 'my%';

SnowSQL Installation

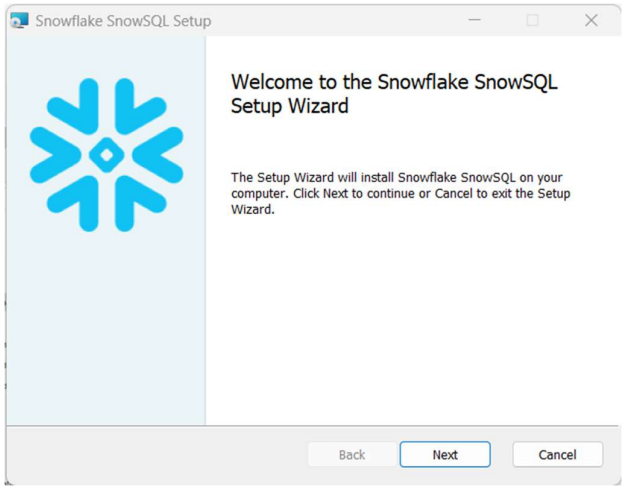
1. Click on the link to download the SnowSQL file https://developers.snowflake.com/snowsql/.
2. Scroll down a bit and choose the windows then click on latest version File Name to download.



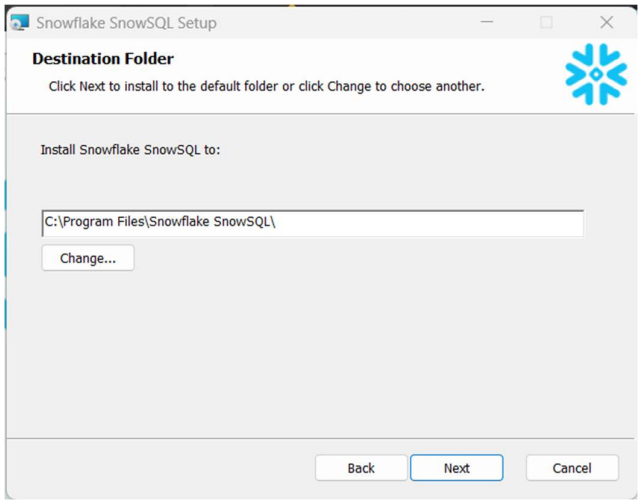
3. After downloading complete open the file as shown below.



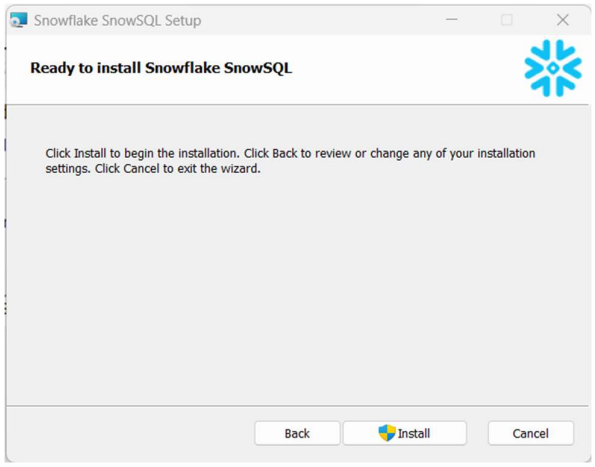
4. Click on Next.



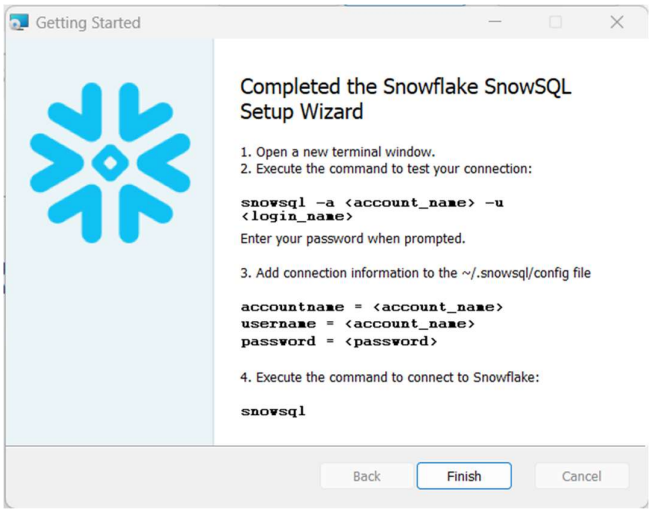
5. Now choose the location then click on next.



6. Click on Install.

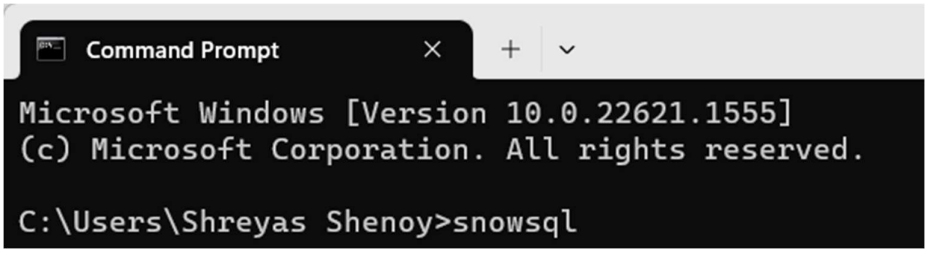


7. After some time you will get this.

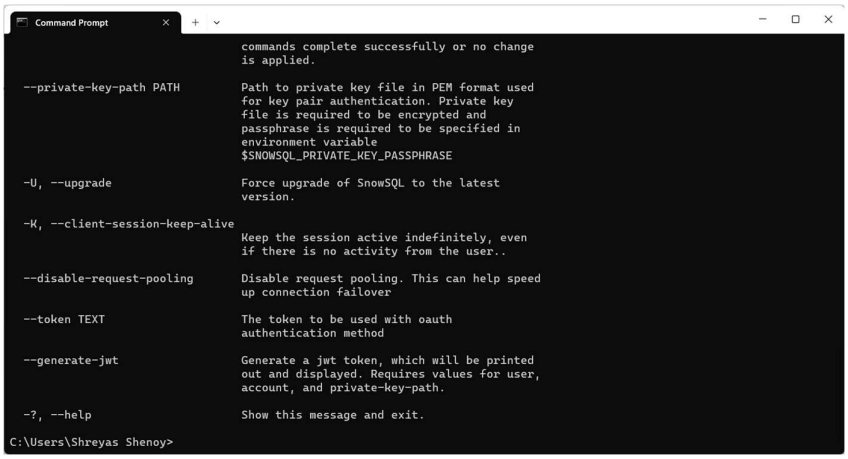


8. Open the command prompt and give the following command as shown below.

Command: snowsql



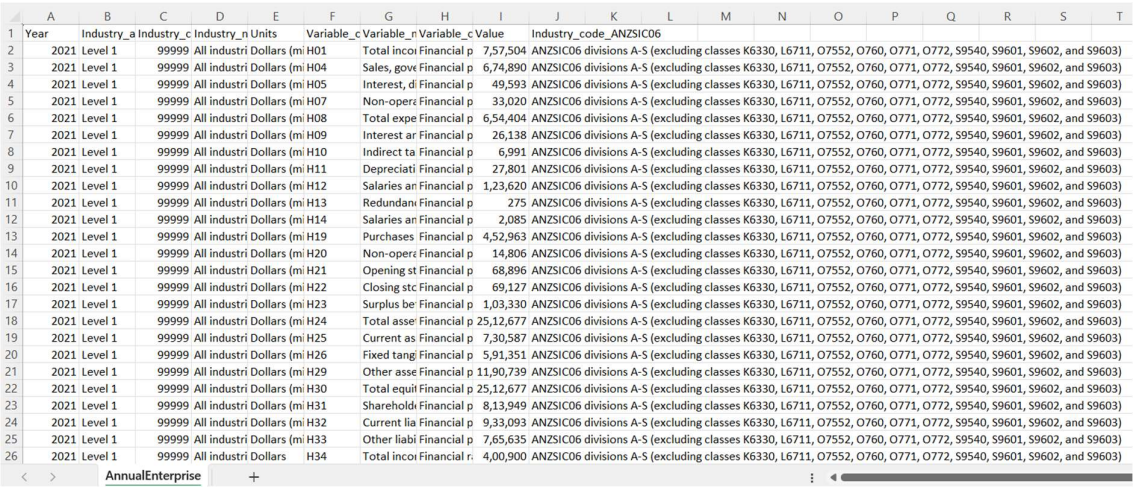
9. Click Enter then you will see a message as shown below.



Load CSV data from the local to Snowflake

For this **SnowSql** should be installed in your system.

In this example we are going load this CSV data into snowflake.



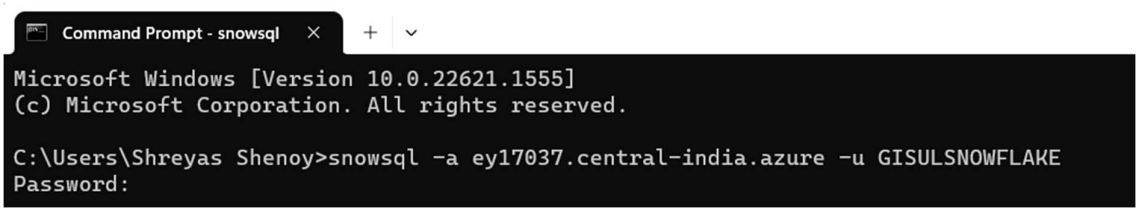
	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	Year	Industry_a	Industry_c	Industry_n	Units	Variable_c	Variable_n	Variable_c	Value	Industry_code_ANZSIC06										
2	2021	Level 1	99999	All industri	Dollars (mi	H01	Total incor	Financial p	7,57,504	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
3	2021	Level 1	99999	All industri	Dollars (mi	H04	Sales, gove	Financial p	6,74,890	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
4	2021	Level 1	99999	All industri	Dollars (mi	H05	Interest, di	Financial p	49,593	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
5	2021	Level 1	99999	All industri	Dollars (mi	H07	Non-opera	Financial p	33,020	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
6	2021	Level 1	99999	All industri	Dollars (mi	H08	Total expe	Financial p	6,54,404	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
7	2021	Level 1	99999	All industri	Dollars (mi	H09	Interest ar	Financial p	26,138	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
8	2021	Level 1	99999	All industri	Dollars (mi	H10	Indirect ta	Financial p	6,991	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
9	2021	Level 1	99999	All industri	Dollars (mi	H11	Depreciati	Financial p	27,801	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
10	2021	Level 1	99999	All industri	Dollars (mi	H12	Salaries an	Financial p	1,23,620	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
11	2021	Level 1	99999	All industri	Dollars (mi	H13	Redundan	Financial p	275	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
12	2021	Level 1	99999	All industri	Dollars (mi	H14	Salaries an	Financial p	2,085	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
13	2021	Level 1	99999	All industri	Dollars (mi	H19	Purchases	Financial p	4,52,963	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
14	2021	Level 1	99999	All industri	Dollars (mi	H20	Non-opera	Financial p	14,806	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
15	2021	Level 1	99999	All industri	Dollars (mi	H21	Opening st	Financial p	68,896	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
16	2021	Level 1	99999	All industri	Dollars (mi	H22	Closing stc	Financial p	69,127	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
17	2021	Level 1	99999	All industri	Dollars (mi	H23	Surplus be	Financial p	1,03,330	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
18	2021	Level 1	99999	All industri	Dollars (mi	H24	Total asse	Financial p	25,12,677	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
19	2021	Level 1	99999	All industri	Dollars (mi	H25	Current as	Financial p	7,30,587	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
20	2021	Level 1	99999	All industri	Dollars (mi	H26	Fixed tang	Financial p	5,91,351	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
21	2021	Level 1	99999	All industri	Dollars (mi	H29	Other asse	Financial p	11,90,739	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
22	2021	Level 1	99999	All industri	Dollars (mi	H30	Total equit	Financial p	25,12,677	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
23	2021	Level 1	99999	All industri	Dollars (mi	H31	Shareholde	Financial p	8,13,949	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
24	2021	Level 1	99999	All industri	Dollars (mi	H32	Current lia	Financial p	9,33,093	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
25	2021	Level 1	99999	All industri	Dollars (mi	H33	Other liabi	Financial p	7,65,635	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										
26	2021	Level 1	99999	All industri	Dollars	H34	Total incor	Financial r	4,00,900	ANZSIC06 divisions A-S (excluding classes K6330, L6711, O7552, O760, O771, O772, S9540, S9601, S9602, and S9603)										

AnnualEnterprise

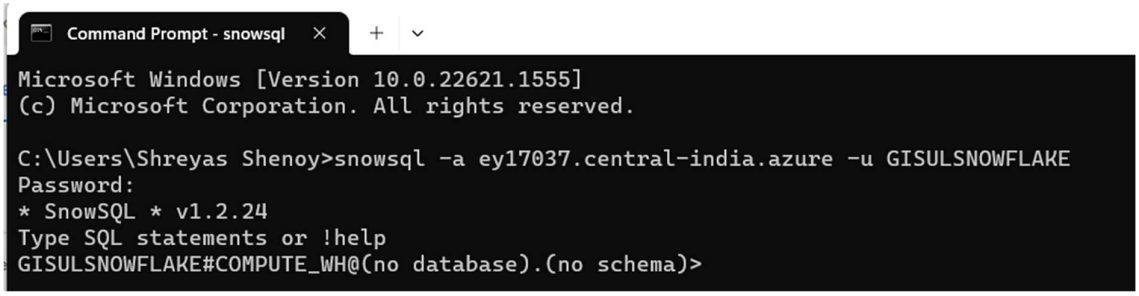
1. Open command prompt and give the following command as shown below.

Command syntax: snowsql -a <Account> -u <Username>

Command: snowsql -a ey17037.central-india.azure -u GISULSNOWFLAKE



2. Give the account Password and click enter.



3. Now to connect with the Warehouse give the below command.

Command: use warehouse COMPUTE_WH;

4. I am using my Warehouse.

```
Microsoft Windows [Version 10.0.22621.1555]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Shreyas Shenoy>snowsql -a ey17037.central-india.azure -u GISULSNOWFLAKE
Password:
* SnowSQL * v1.2.24
Type SQL statements or !help
GISULSNOWFLAKE#COMPUTE_WH@(no database).(no schema)>use warehouse COMPUTE_WH;
+----------------------------------+
| status                           |
|----------------------------------|
| Statement executed successfully. |
+----------------------------------+
1 Row(s) produced. Time Elapsed: 0.204s
GISULSNOWFLAKE#COMPUTE_WH@(no database).(no schema)>
```

5. Now to connect with database give below command.

Command: Use database SAMPLE_SNOW;

```
1 Row(s) produced. Time Elapsed: 0.204s
GISULSNOWFLAKE#COMPUTE_WH@(no database).(no schema)>use database SAMPLE_SNOW;
+----------------------------------+
| status                           |
|----------------------------------|
| Statement executed successfully. |
+----------------------------------+
1 Row(s) produced. Time Elapsed: 0.124s
GISULSNOWFLAKE#COMPUTE_WH@SAMPLE_SNOW.PUBLIC>
```

6. Now to connect with the schema give the below command.

Command: Use schema “Sample”;

```
1 Row(s) produced. Time Elapsed: 0.124s
GISULSNOWFLAKE#COMPUTE_WH@SAMPLE_SNOW.PUBLIC>use schema "Sample";
+----------------------------------+
| status                           |
|----------------------------------|
| Statement executed successfully. |
+----------------------------------+
1 Row(s) produced. Time Elapsed: 0.126s
GISULSNOWFLAKE#COMPUTE_WH@SAMPLE_SNOW.Sample>
```

7. Create a table with the create command as shown below.

Query: create or Replace table ENTERPRICE(
YEAR VARCHAR(250),
INDUSTRY_AGGREGATION VARCHAR(250),
INDUSTRY_CODE VARCHAR(25),
INDUSTRY_NAME VARCHAR(250),
UNITS VARCHAR(150),
VARIABLE_CODE VARCHAR(250),
VARIABLE_NAME VARCHAR(250),
VARIABLE_CATEGORY VARCHAR(25),

VALUE VARCHAR(250),
INDUSTRY_CODE_ANZSIC06 VARCHAR(250)
);

```
GISULSNOWFLAKE#COMPUTE_WH@SAMPLE_SNOW.Sample>create or Replace table ENTERPRICE (
                                             YEAR VARCHAR(250),
                                             INDUSTRY_AGGREGATION VARCHAR(250),
                                             INDUSTRY_CODE VARCHAR(25),
                                             INDUSTRY_NAME VARCHAR(250),
                                             UNITS VARCHAR(150),
                                             VARIABLE_CODE VARCHAR(250),
                                             VARIABLE_NAME VARCHAR(250),
                                             VARIABLE_CATEGORY VARCHAR(25),
                                             VALUE VARCHAR(250),
                                             INDUSTRY_CODE_ANZSIC06 VARCHAR(250)
                                             );
+----------------------------------------+
| status                                 |
|----------------------------------------|
| Table ENTERPRICE successfully created. |
+----------------------------------------+
1 Row(s) produced. Time Elapsed: 0.437s
GISULSNOWFLAKE#COMPUTE_WH@SAMPLE_SNOW.Sample>
```

8. We will load the CSV data file from your local system to the staging of the Snowflake as shown below.

Command: put file://C:\Users\Public\AnnualEnterprise.csv @"SAMPLE_SNOW"."Sample".%Enterprice;

```
1 Row(s) produced. Time Elapsed: 0.437s
GISULSNOWFLAKE#COMPUTE_WH@SAMPLE_SNOW.Sample>put file://C:\Users\Public\AnnualEnterprise.csv @"SAMPLE_SNOW"."Sample".%Enterprice;
+----------------------+-----------------------+-------------+-------------+--------------------+--------------------+----------+---------+
| source               | target                | source_size | target_size | source_compression | target_compression | status   | message |
|----------------------+-----------------------+-------------+-------------+--------------------+--------------------+----------+---------|
| AnnualEnterprise.csv | AnnualEnterprise.csv.gz |     6618545 |      310256 | NONE               | GZIP               | UPLOADED |         |
+----------------------+-----------------------+-------------+-------------+--------------------+--------------------+----------+---------+
1 Row(s) produced. Time Elapsed: 2.072s
GISULSNOWFLAKE#COMPUTE_WH@SAMPLE_SNOW.Sample>
```

9. To check the load details use the below command.

Command: list @"SAMPLE_SNOW"."Sample".%Enterprice;

```
1 Row(s) produced. Time Elapsed: 2.072s
GISULSNOWFLAKE#COMPUTE_WH@SAMPLE_SNOW.Sample>list @"SAMPLE_SNOW"."Sample".%Enterprice;
+-------------------------+--------+----------------------------------+-------------------------------+
| name                    |   size | md5                              | last_modified                 |
|-------------------------+--------+----------------------------------+-------------------------------|
| AnnualEnterprise.csv.gz | 310256 | 17b9709e2f9456d06bd4dfd0d93c15c0 | Wed, 26 Apr 2023 08:23:28 GMT |
+-------------------------+--------+----------------------------------+-------------------------------+
1 Row(s) produced. Time Elapsed: 0.200s
GISULSNOWFLAKE#COMPUTE_WH@SAMPLE_SNOW.Sample>
```

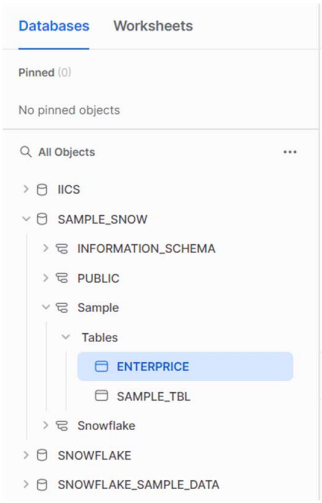
10. Copy the data into the Target Table using the below command.

Command: copy into Enterprice from @%Enterprice file_format = (type = csv skip_header = 1 field_optionally_enclosed_by='"') pattern = '.*AnnualEnterprise.csv.gz' on_error = 'skip_file';

```
1 Row(s) produced. Time Elapsed: 0.159s
GISULSNOWFLAKE#COMPUTE_WH@SAMPLE_SNOW.Sample>copy into Enterprice from @%Enterprice file_format = (type = csv skip_header = 1 field_optionally_enclosed_by=
                                             '"') pattern = '.*AnnualEnterprise.csv.gz' on_error = 'skip_file';
+-------------------------+--------+-------------+-------------+-------------+-------------+------------+------------------+----------------------+-------------
-----------------+
| file                    | status | rows_parsed | rows_loaded | error_limit | errors_seen | first_error | first_error_line | first_error_character | first_
error_column_name |
|-------------------------+--------+-------------+-------------+-------------+-------------+------------+------------------+----------------------+-------------
-----------------|
| AnnualEnterprise.csv.gz | LOADED |       41715 |       41715 |           1 |           0 | NULL       |             NULL |                 NULL | NULL
                  |
+-------------------------+--------+-------------+-------------+-------------+-------------+------------+------------------+----------------------+-------------
-----------------+
1 Row(s) produced. Time Elapsed: 1.500s
GISULSNOWFLAKE#COMPUTE_WH@SAMPLE_SNOW.Sample>
```

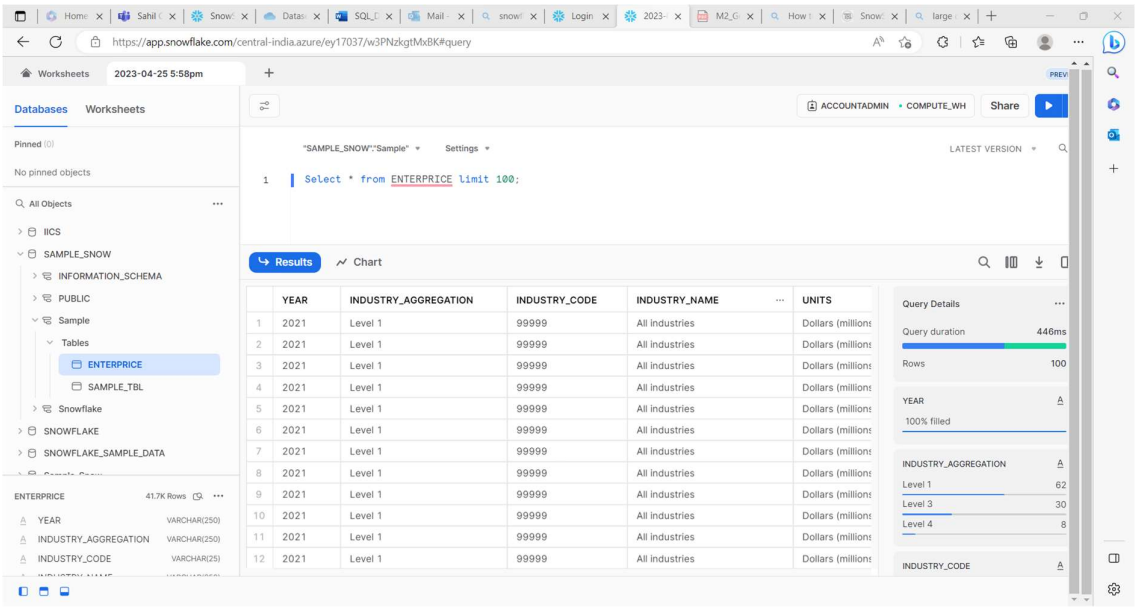
11. Now our csv data is loaded into the target table.

12. Go to Snowflake and refresh the database then check whether our table is created or not.



13. Now to display the records use the select command in the worksheet and give a limit of 100 as shown below.

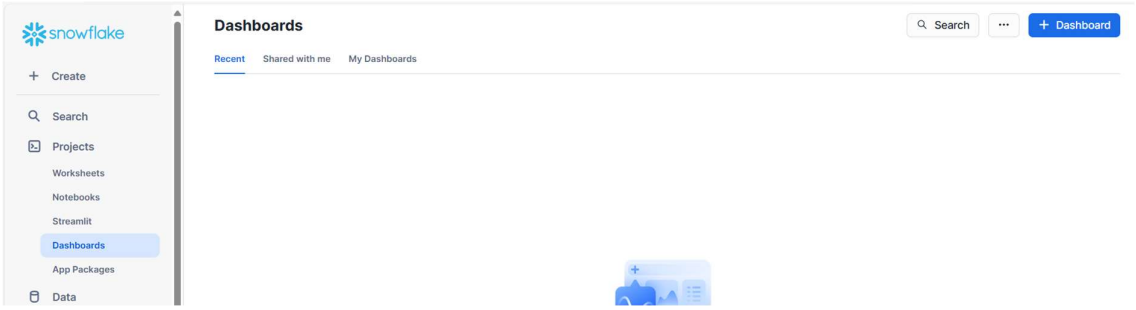
Query: select * from Enterprice limit 100;



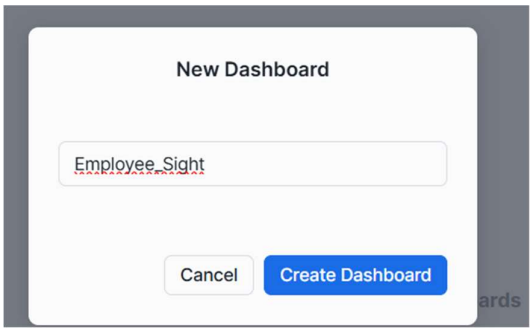
Snowsight

Snowflake Snowsight is a web-based interface that lets you query, visualize, and share data in Snowflake. Snowsight simplifies data tasks for users of all data roles, by providing an intuitive graphical interface, built-in data visualization capabilities, simplified data access and security controls—and collaboration features. Snowsight is accessible via any modern browser and can help you perform advanced data analysis and reporting more efficiently and effectively, without requiring extensive coding skills or complex/lengthy setup processes.

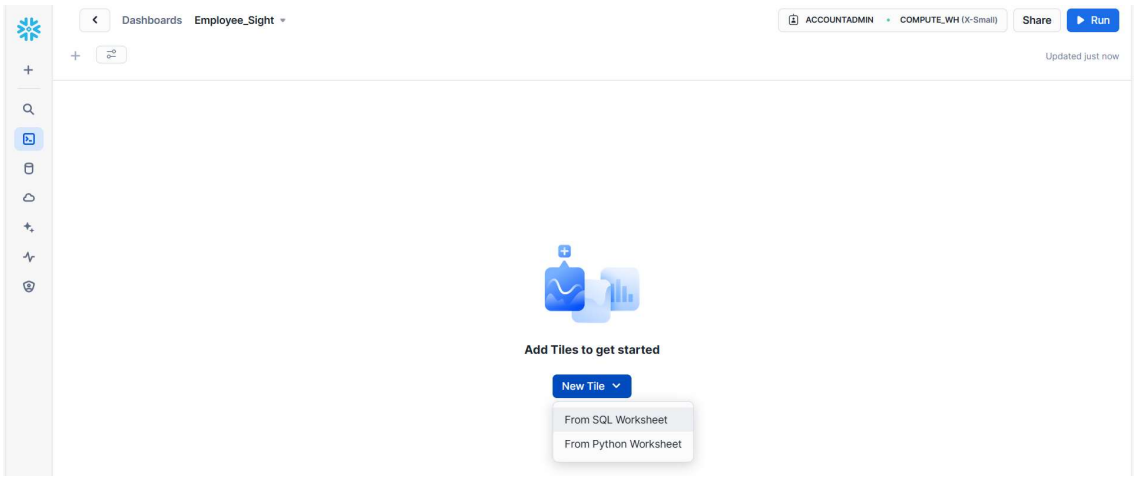
1. In this example we are using the Employee Details table that we created before.
2. Go to the Home page and go to Dashboard click on the plus symbol.



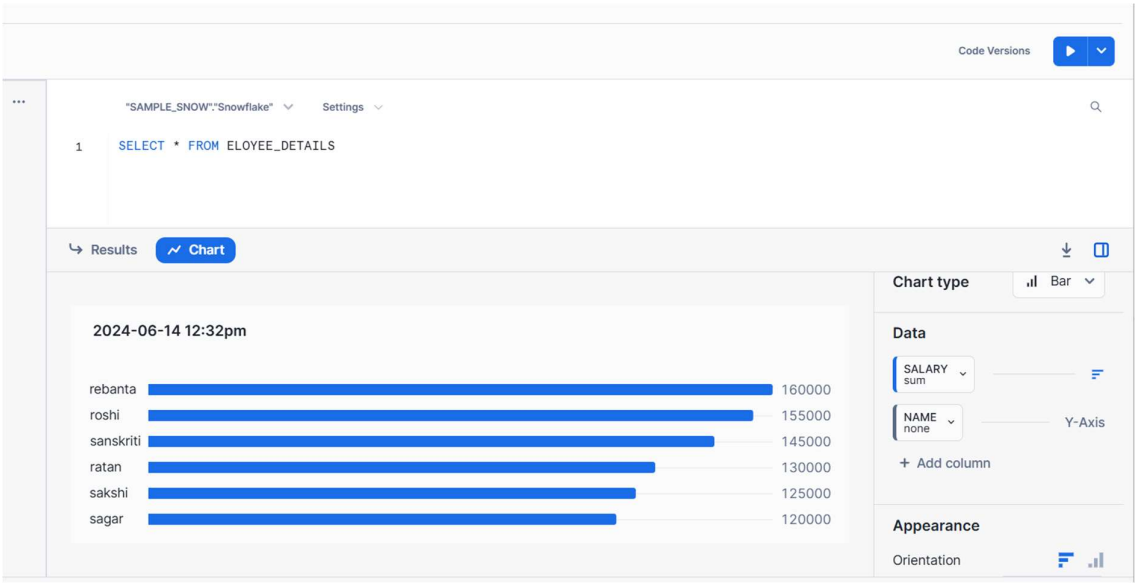
3. Give a name and click on Create.



4. Select the warehouse and click on Select from SQL Worksheet as shown below.



5. Give a query create a column chart with the below properties and click on Run.



6. Click on Return to Employee sight and you will see a chart like below.

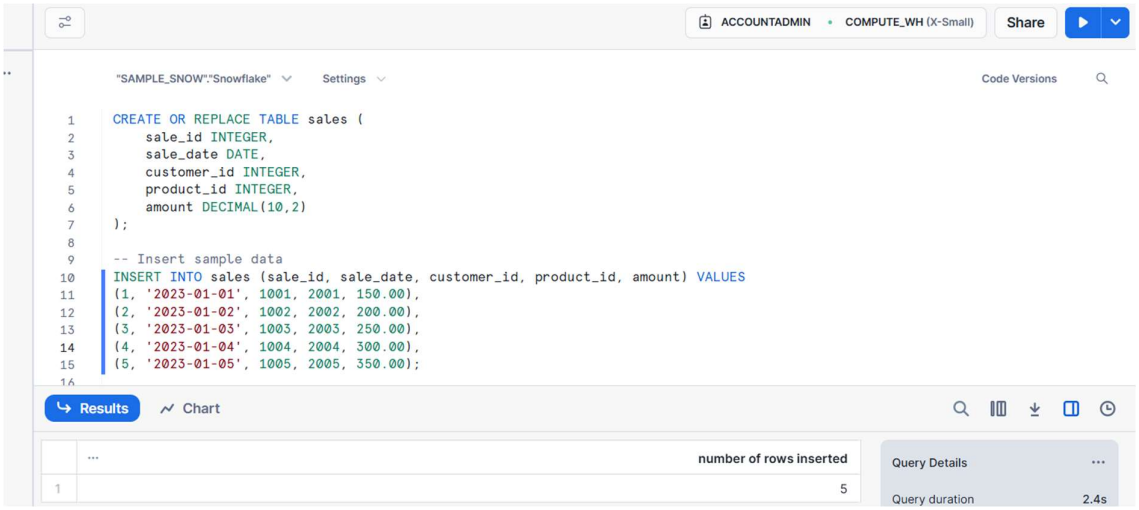
Micro-partitions

Micro-partitions are the foundational unit of data storage in Snowflake. Each micro-partition holds between 50 MB and 500 MB of compressed data in a columnar format, enabling efficient storage and rapid query performance. When data is loaded into a Snowflake table, it is automatically divided into micro-partitions based on the natural order of the incoming data. Each micro-partition includes metadata, such as min and max values for each column, null counts, and the number of distinct values, which aids in optimizing query performance by reducing the amount of data scanned during queries.

1. Create a table and load data by using the below query.

Query: CREATE OR REPLACE TABLE sales (
  sale_id INTEGER,
  sale_date DATE,
  customer_id INTEGER,
  product_id INTEGER,
  amount DECIMAL(10,2)
);

-- Insert sample data
INSERT INTO sales (sale_id, sale_date, customer_id, product_id, amount) VALUES
(1, '2023-01-01', 1001, 2001, 150.00),
(2, '2023-01-02', 1002, 2002, 200.00),
(3, '2023-01-03', 1003, 2003, 250.00),
(4, '2023-01-04', 1004, 2004, 300.00),
(5, '2023-01-05', 1005, 2005, 350.00);

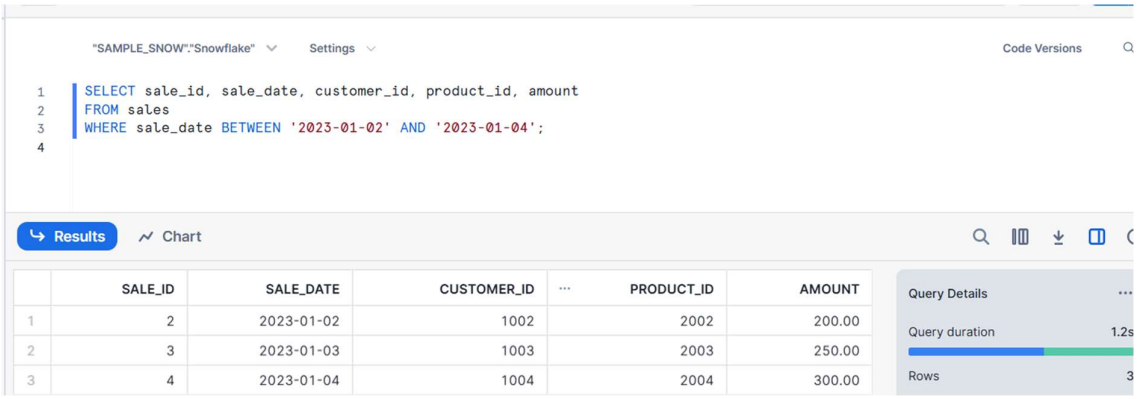


2. Let's run a query to retrieve sales data for a specific date range. Snowflake will use the metadata stored in the micro-partitions to efficiently scan only the relevant partitions.

Query: SELECT sale_id, sale_date, customer_id, product_id, amount

FROM sales

WHERE sale_date BETWEEN '2023-01-02' AND '2023-01-04';



How Micro-partitions Optimize the Query:

1. **Metadata Pruning:** Snowflake examines the metadata of each micro-partition to determine if the sale_date range falls within the min and maximum values of the partition. Only the relevant micro-partitions are scanned.
2. **Columnar Access:** Since the data is stored in a columnar format, Snowflake only reads the columns specified in the query (sale_id, sale_date, customer_id, product_id, amount), reducing I/O and improving performance.
3. **Compression:** The compressed data within the micro-partitions is decompressed on the fly, minimizing storage requirements and speeding up data retrieval.

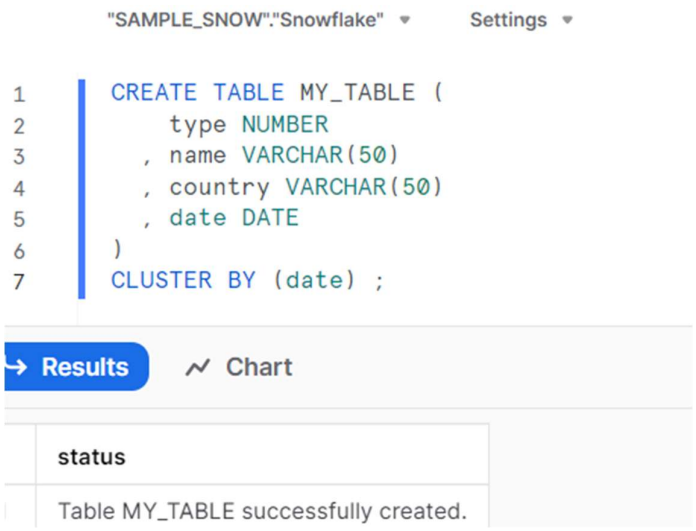
Data Clustering

Clustering is a technique used to organize data storage to better accommodate anticipated queries. The key objective is to increase query performance while lowering the amount of system resources needed to run queries.

Create Clustered Tables

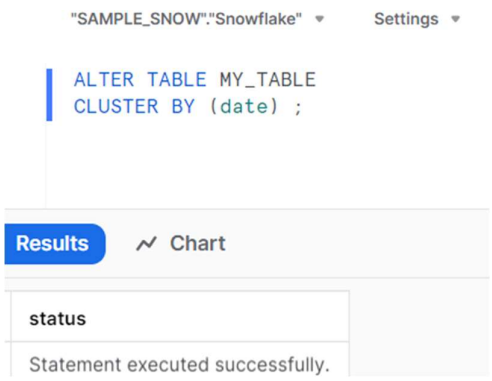
1. A clustering key can be defined when a table is created by appending a CLUSTER BY clause to CREATE TABLE as shown below.

Query: CREATE TABLE MY_TABLE (
type NUMBER
, name VARCHAR(50)
, country VARCHAR(50)
, date DATE
)
CLUSTER BY (date) ;



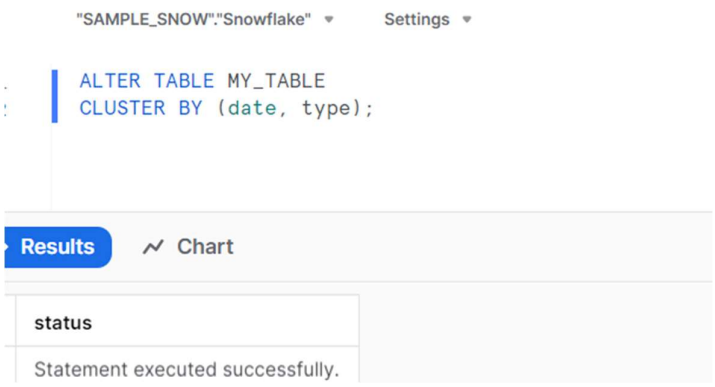
2. In the above example we have created a table with clustered key.
3. However you can also cluster a table by specifying a CLUSTER KEY at a later point in time using ALTER TABLE as shown below.

Query: ALTER TABLE MY_TABLE CLUSTER BY (date) ;



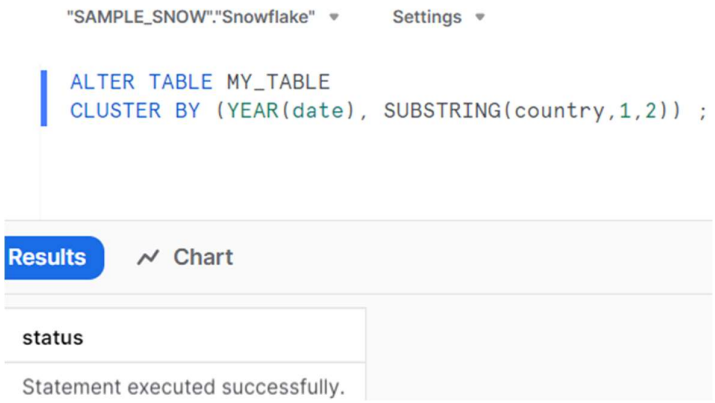
4. Although we have just used one field as a clustering key in our example, this is not the only choice.
5. A clustering key can be defined using many fields if desired as shown below.

Query: ALTER TABLE MY_TABLE CLUSTER BY (date, type) ;



6. Another advantage is that Snowflake also supports expressions on fields in a cluster key.
7. The below example shows the table is clustered using the year of the date field and the first two letters of the country field.

Query: ALTER TABLE MY_TABLE CLUSTER BY (YEAR(date), SUBSTRING(country,1,2)) ;

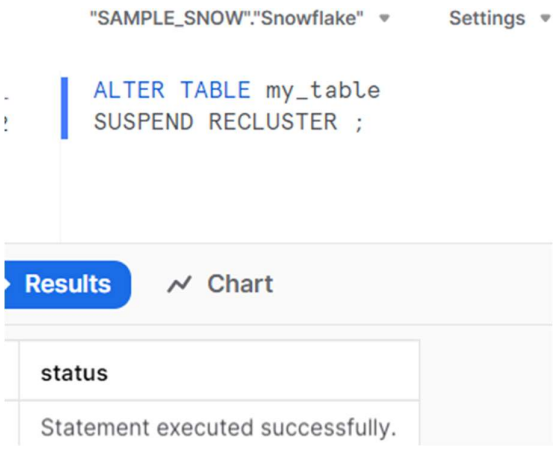


Automatic Clustering

Automatic Clustering is the Snowflake service that seamlessly and continually manages all reclustering, as needed, of clustered tables. Note that Automatic Clustering consumes Snowflake credits, but does not require you to provide a virtual warehouse. Instead, Snowflake internally manages and achieves efficient resource utilization for reclustering the tables.

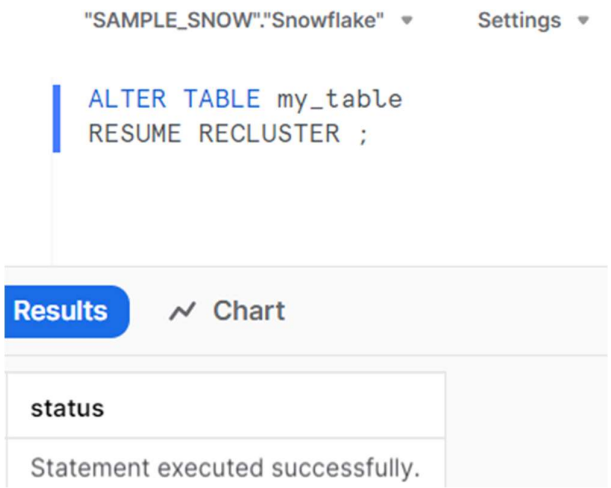
1. To suspend Automatic Clustering for a table, use the ALTER TABLE command with a SUSPEND RECLUSTER clause.

Query: ALTER TABLE my_table SUSPEND RECLUSTER ;



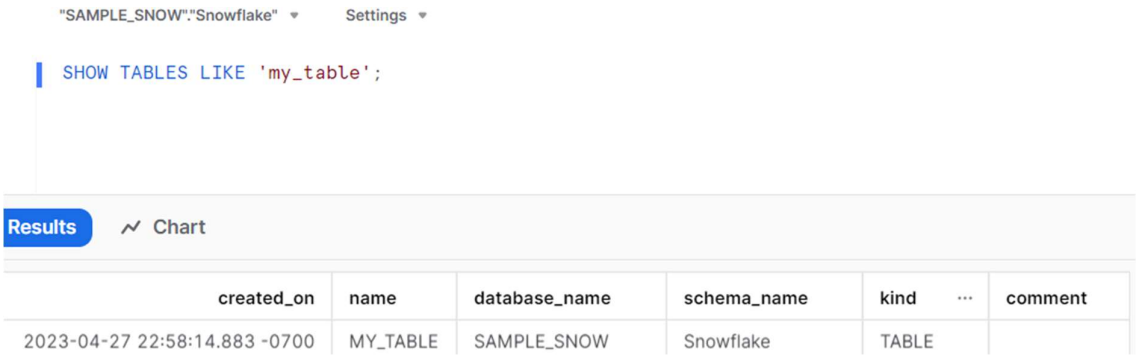
2. To resume Automatic Clustering for a clustered table, use the ALTER TABLE command with a RESUME RECLUSTER clause.

Query: ALTER TABLE my_table RESUME RECLUSTER ;



3. Verify if Automatic Clustering is enabled for a table using the SHOW TABLES command.

Query: SHOW TABLES LIKE 'my_table';



Sequence

Sequences are a common way to generate unique, sequential numbers. They can be used to allocate primary keys, auto-increment values, or other types of unique identifiers.

Sequences in Snowflake are also used to generate unique numbers with auto-increment functionality based on a defined interval. They guarantee uniqueness across multiple statements and sessions, ensuring that each execution of the sequence produces a distinct value.

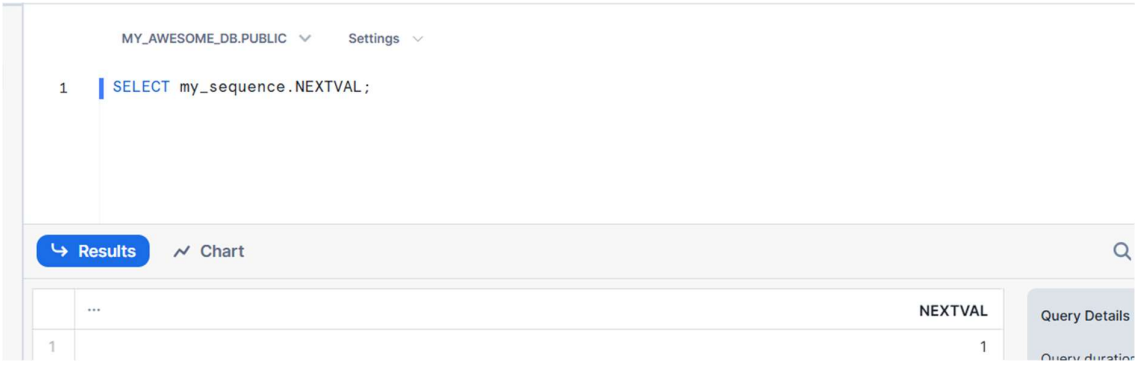
1. We need to create a database to house the objects for this sequence generation.
2. Execute the following commands either in Snowflake WebUI or SnowSQL command-line client.

Query: CREATE DATABASE my_awesome_db;

CREATE SEQUENCE my_sequence;

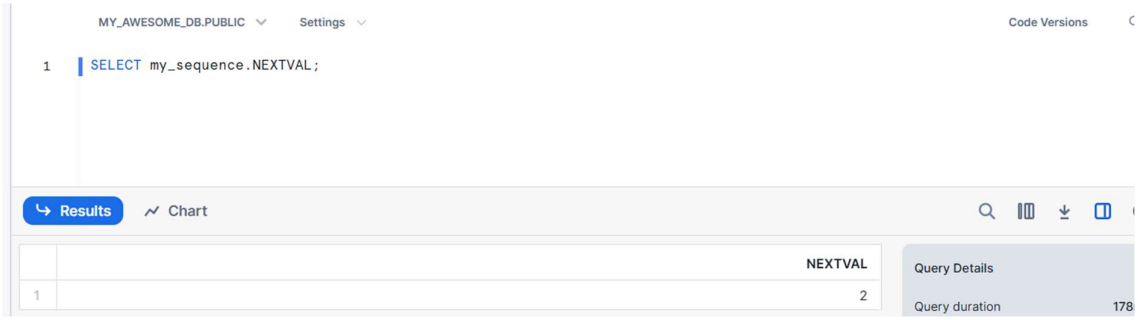
3. Now, let's retrieve a value from the newly created sequence. You can now use the NEXTVAL function with sequence to generate unique numbers.
4. For example, the following statement will return the next value in the my_sequence sequence.

Query: SELECT my_sequence.NEXTVAL;



5. After the sequence has been incremented, running the same SELECT statement again will provide the next value in the sequence.

Query: SELECT my_sequence.NEXTVAL;



6. This time, the query will return the next value, which should be 2.
7. To validate that the sequence always returns unique values, even within the same statement, execute the following SQL.

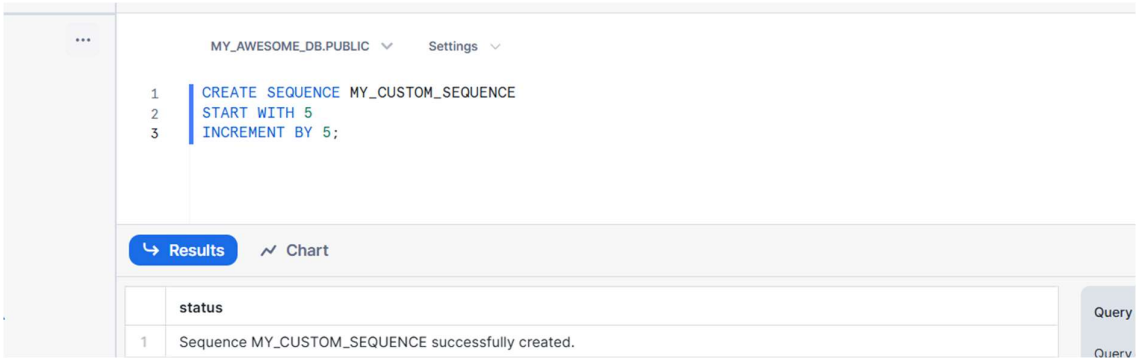
Query: SELECT my_sequence.NEXTVAL, my_sequence.NEXTVAL, my_sequence.NEXTVAL, my_sequence.NEXTVAL, my_sequence.NEXTVAL, my_sequence.NEXTVAL;

8. Sequences can also be used to populate columns in a table.
9. Let's create a custom sequence with custom start and increment values.
10. The following example shows how to create a sequence & table and populate a column with values from a sequence.

Query: CREATE SEQUENCE MY_CUSTOM_SEQUENCE

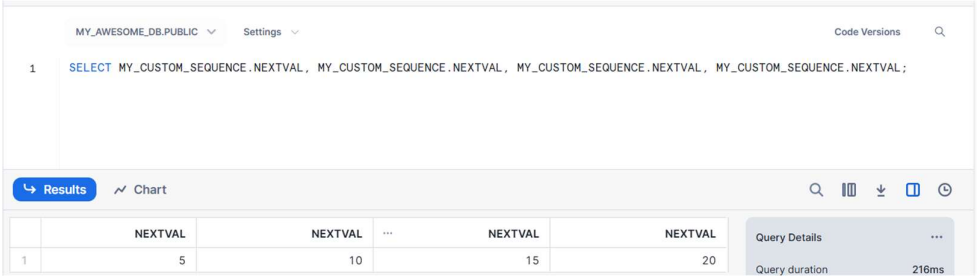
START WITH 5

INCREMENT BY 5;



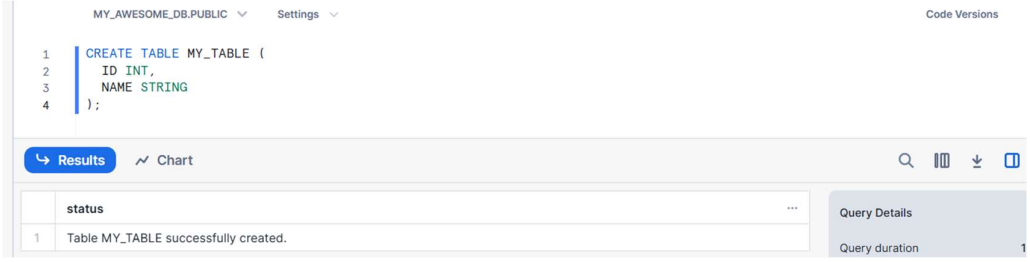
11. To test the sequence with the custom start and increment values, execute the following SQL.

Query: SELECT MY_CUSTOM_SEQUENCE.NEXTVAL, MY_CUSTOM_SEQUENCE.NEXTVAL, MY_CUSTOM_SEQUENCE.NEXTVAL, MY_CUSTOM_SEQUENCE.NEXTVAL;



12. Sequences can be used to populate incremental values in table columns. Let's create a new table and insert data into one of its columns using a sequence.
13. Execute the following SQL statements.

Query: CREATE TABLE MY_TABLE (
ID INT,
NAME STRING
);



14. Let's write a command that inserts some rows into the MY_TABLE table.
15. We will populate the ID column using the MY_CUSTOM_SEQUENCE sequence, which ensures that each row receives a unique and incremental identifier.
16. Also, we will fill the NAME column with custom string values. You can either add your own custom strings or use random strings if desired.

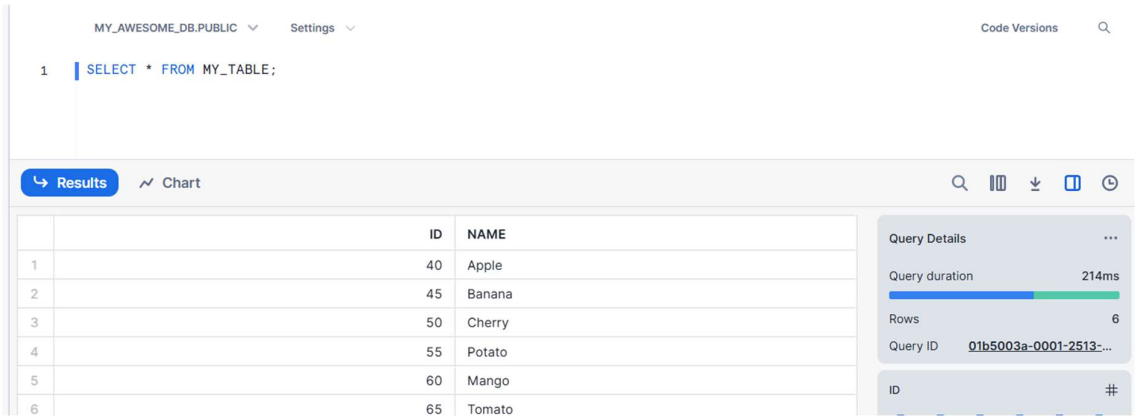
Query: INSERT INTO MY_TABLE (ID, NAME)

SELECT MY_CUSTOM_SEQUENCE.NEXTVAL, t.NAME

FROM (VALUES ('Apple'), ('Banana'), ('Cherry'), ('Potato'), ('Mango'), ('Tomato')) AS t(NAME);

17. To view the data inserted into the table, execute the following SQL.

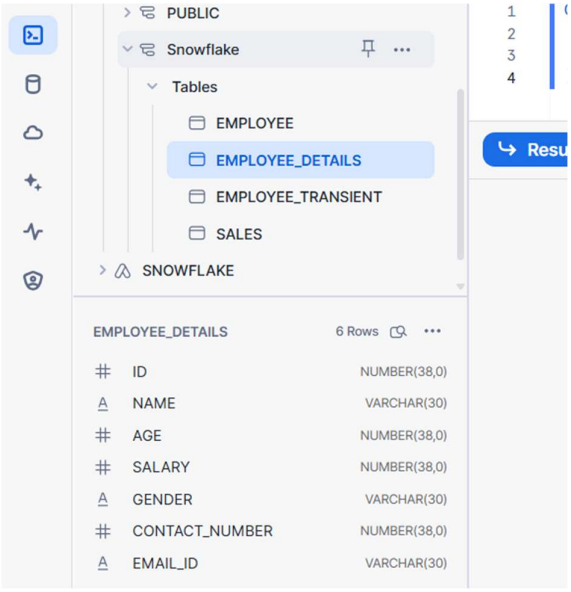
Query: SELECT * FROM MY_TABLE;



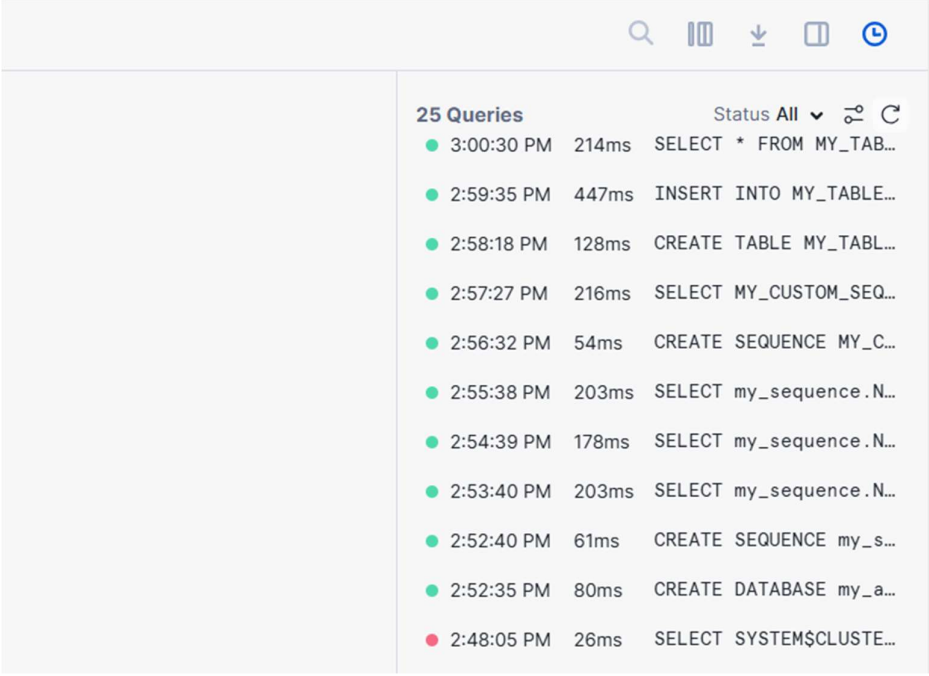
18. As you can see, the output displays the inserted data, with the ID column showing auto-incremented values.

Monitor Usage and Storage

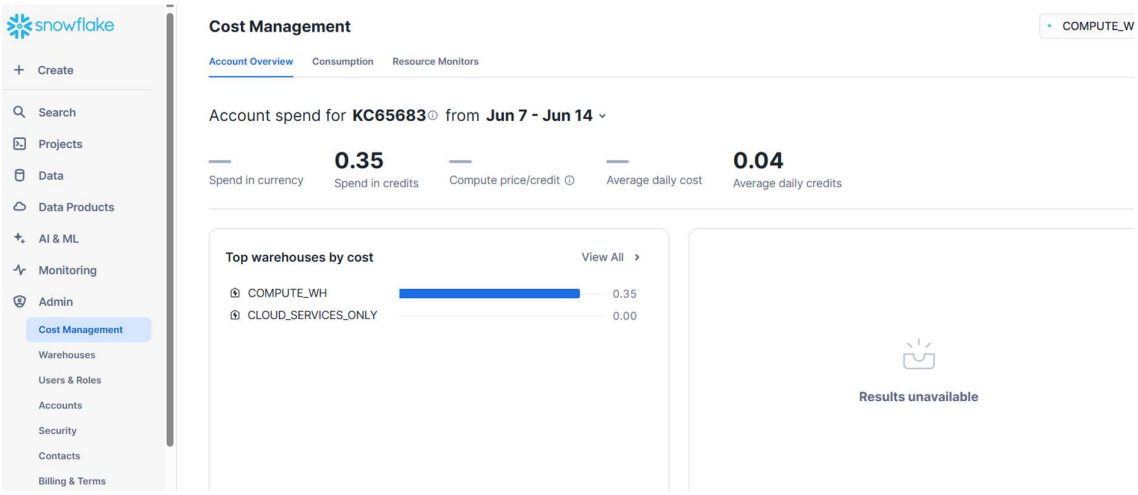
1. The user can check how many rows have individual tables as well as data size. If the user has access to the table, he/she can view these details by just selecting a table.
2. In the left-down panel, the user can see the Table Name then number of rows, and the data storage size. After that, it shows the column's definition of the table.



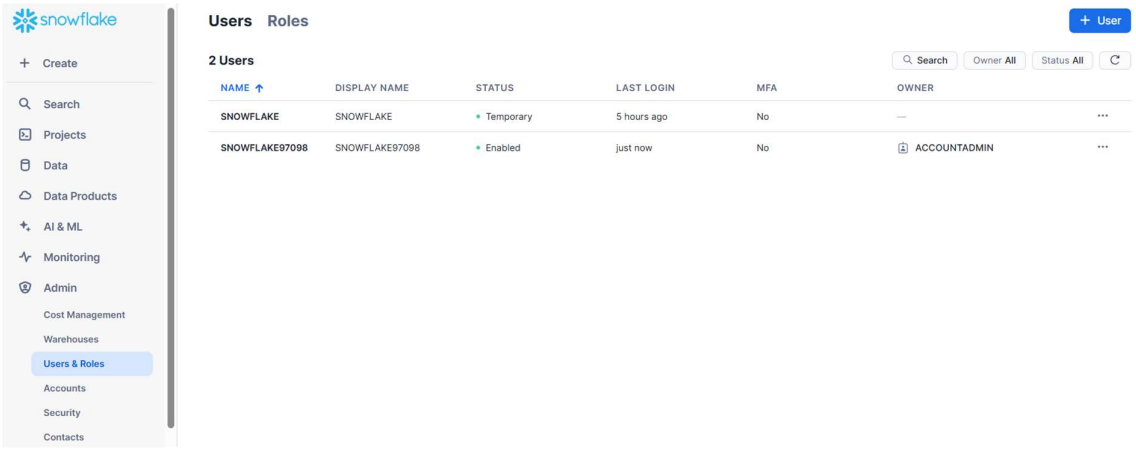
3. In this section, users can check their activities in a snowflake like what queries they are using, the current status of the query, how much time it took to run, etc.
4. To view the history, click the History tab present at the bottom Ribbon. It will show the user's history.



5. To perform account-level monitoring, the user must be logged in as an ACCOUNTADMIN role.
6. It will show Account Usage by default. Users can see the number of warehouses created, how much credit has been used, average storage used (it means that how much data we scanned while running the query vs overall storage), and how much data transferred.



7. Click the tab USERS. It displays the names of all the users present in the account.



8. A new role can be created here by clicking the Create button present at the top of the role list. Selecting a role, gives the option to enable or delete the role as well, by clicking the Edit button and Drop respectively.