Snowflake Time Travel, Cloning and Cache Assignment

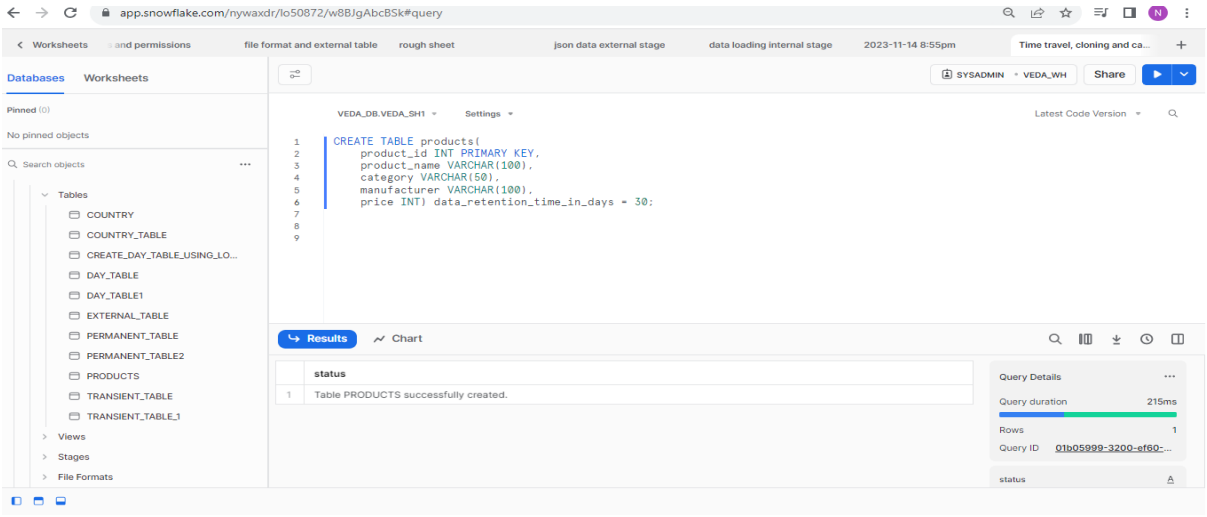
1. Time Travel :-

- Time travel is Retention time that means the time we can travel back into the past in snowflake this depends on the edition we use.
- This is basically the property for objects, for example if we take table, this is specifying for how much time we can travel back for this object.
- So, the default value here is usually set to one.
- So, if the table has its retention period property, set to one then we can travel back for 24 hours for this table.

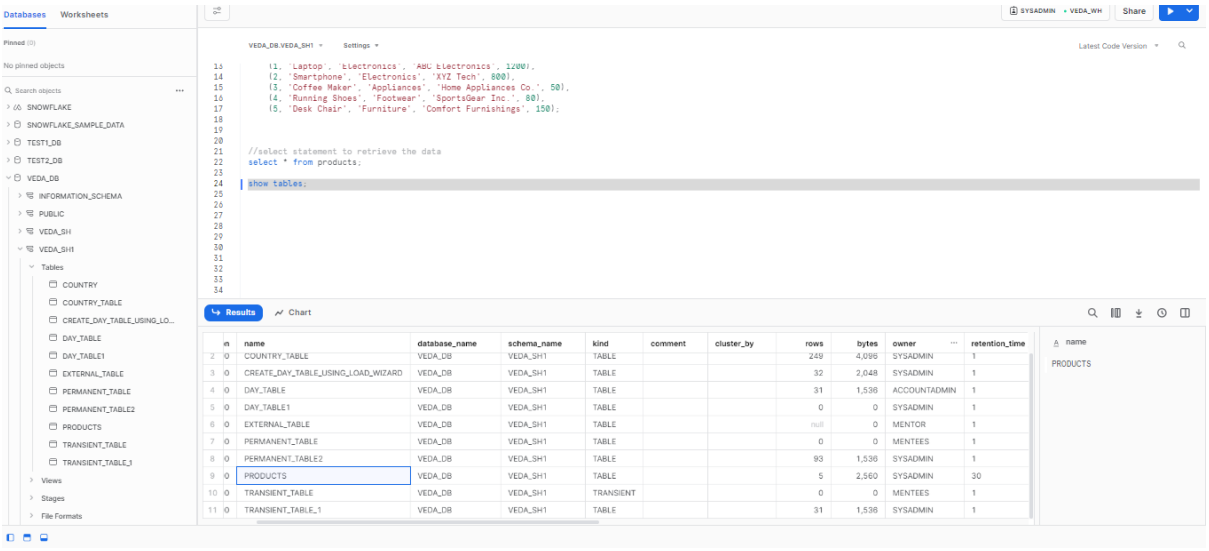
Now I am taking a sample table called Products :-

Creating a table called products mentioning retention time as 30 days.

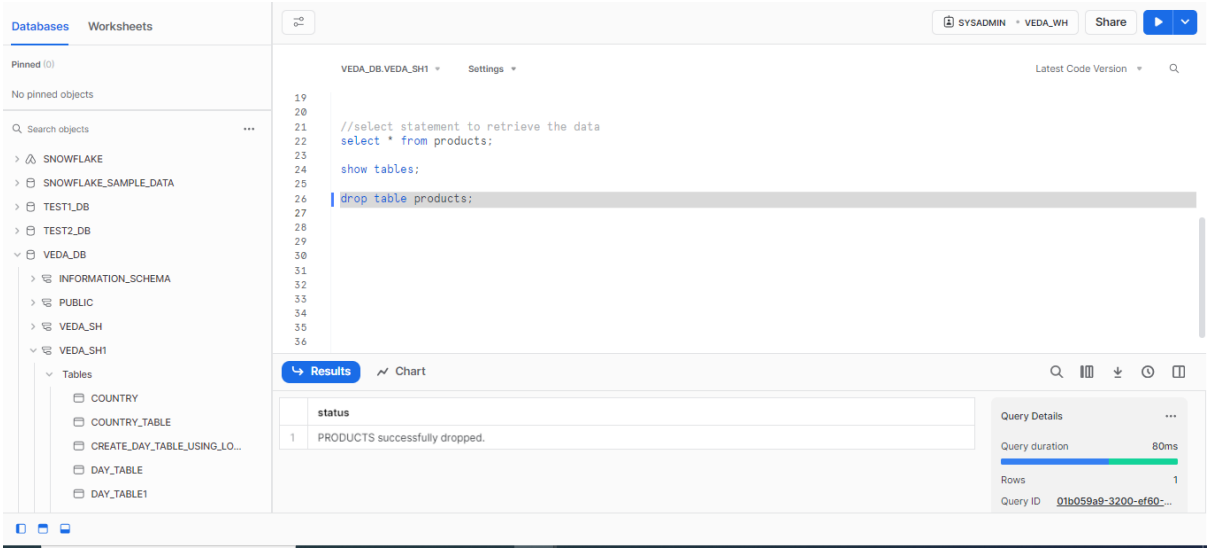
```
CREATE TABLE products (
product_id INT PRIMARY KEY,
product_name VARCHAR(100),
category VARCHAR(50),
manufacturer VARCHAR(100),
price INT) data_retention_time_in_days = 30;
```



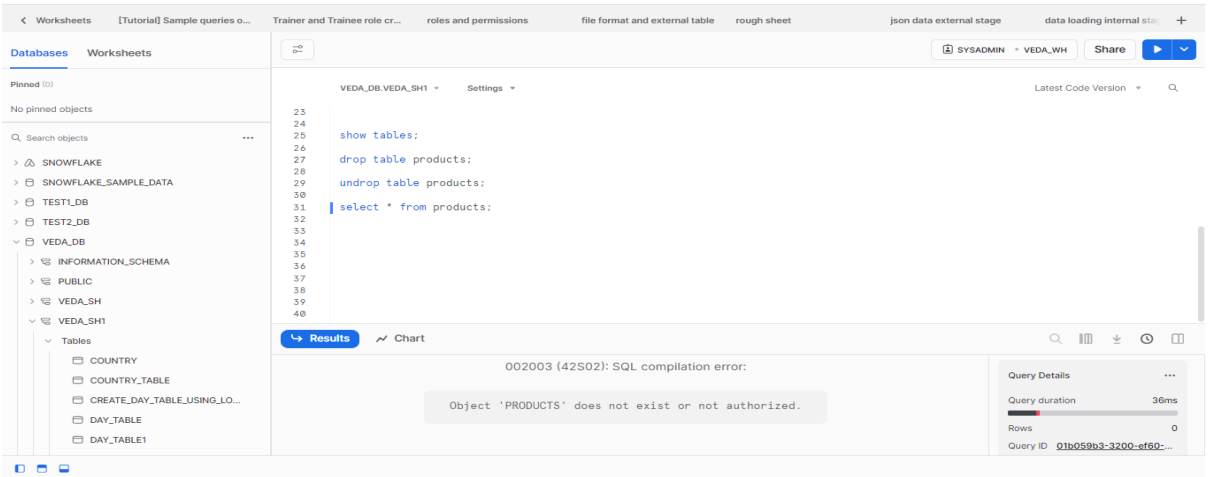
Now I have inserted the data into the table with 5 records and now I am using show tables query and the retention time can be seen here



Now I am dropping the table products using drop table command



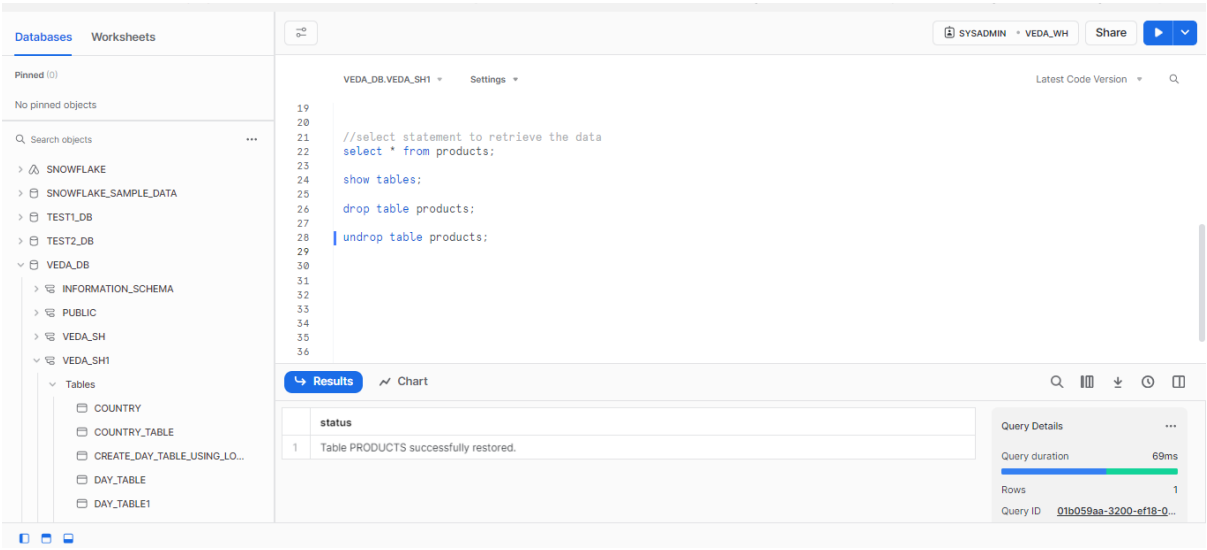
Checking whether the table is dropped or not



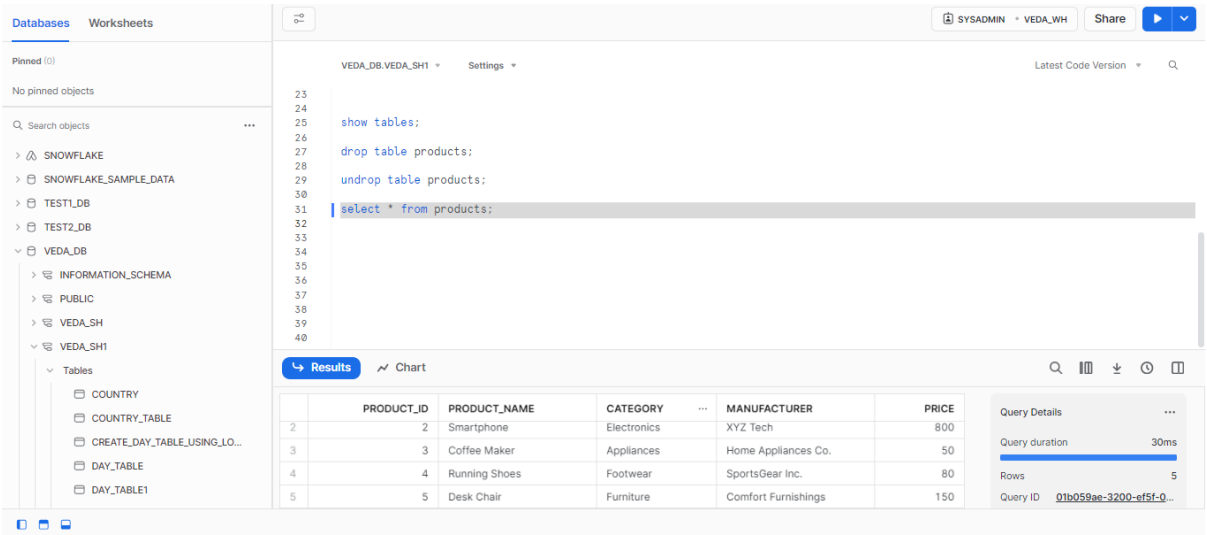
Now I am trying to undrop the table as the retention time is set to 30 days so the table is restored back.

Query :-

```
drop table products;

undrop table products;
```

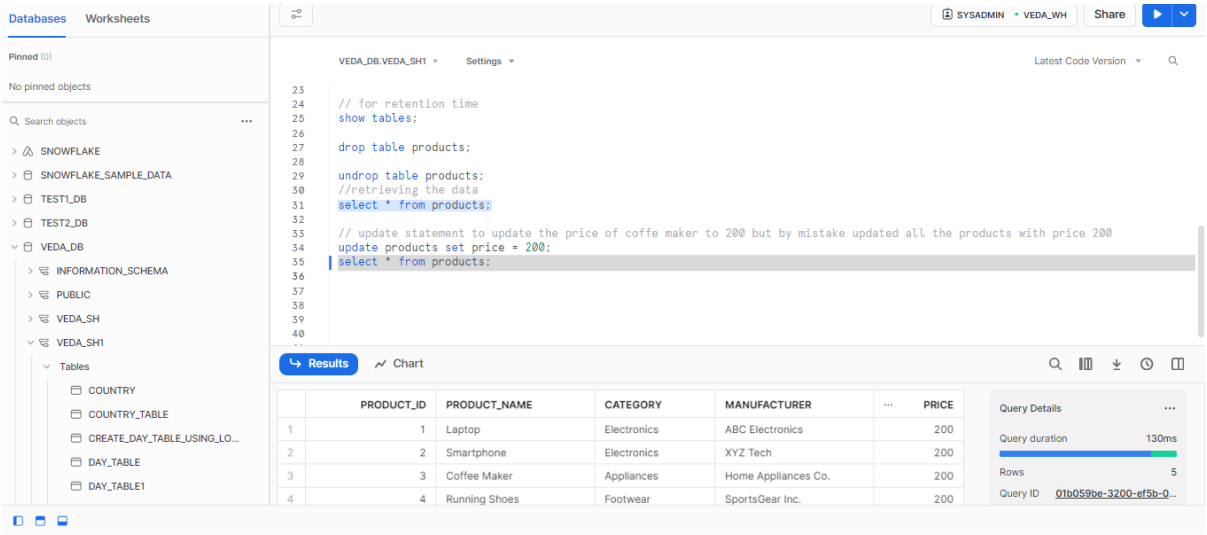


As I have used undrop the data is restored back and I have listed the data here



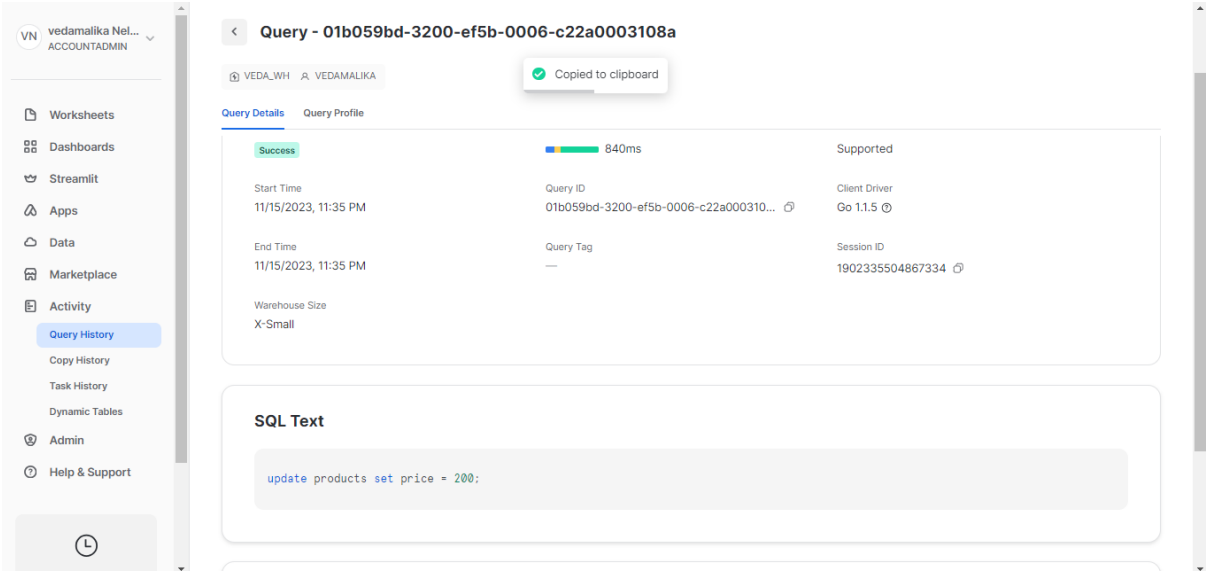
Even if by mistake if we have updated something wrong then also, we can roll back the previous data.

Here I am updating the product table so in product table I want to change the price of coffee maker to 200 but instead of this I have updated price of all products to 200 by mistake so all the rows have been updated with price 200



- **Here I can travel back using before**

I have gone to query history and got the query id from there



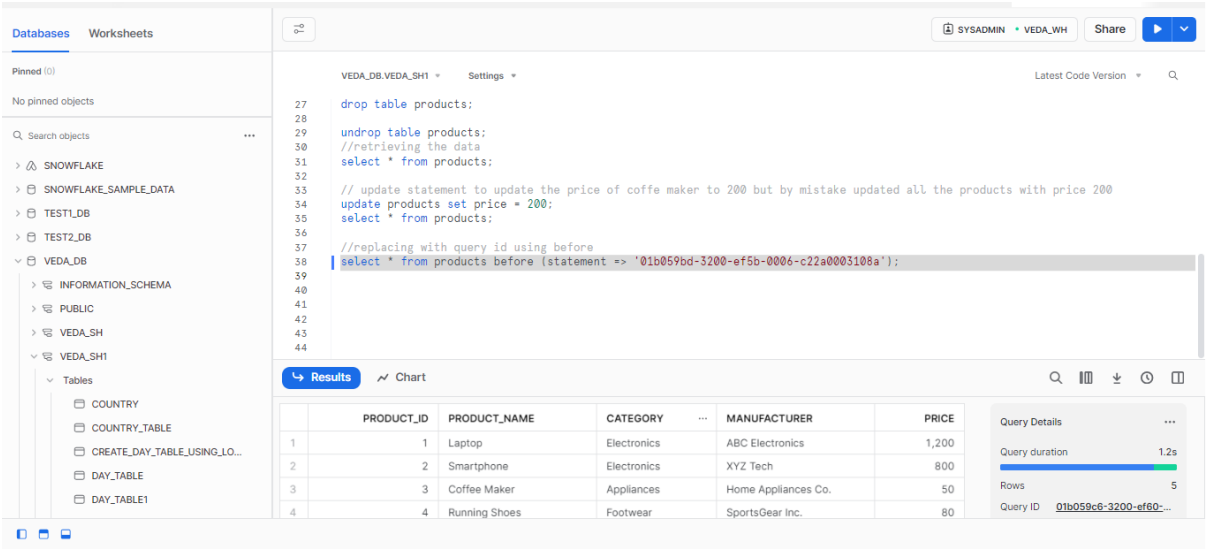
Now in the syntax of time travel I have placed the query id there

we can use before and at commands.

Using before command so it means before this statement whatever the data present in the table, so that need to be retrieved

Query :-

```
select * from products before (statement => '01b059bd-3200-ef5b-0006-c22a0003108a');
```



- **Now in other scenario I am using timestamp in which we will be able to access the time travel data.**

  For this I need to give the exact timestamp.

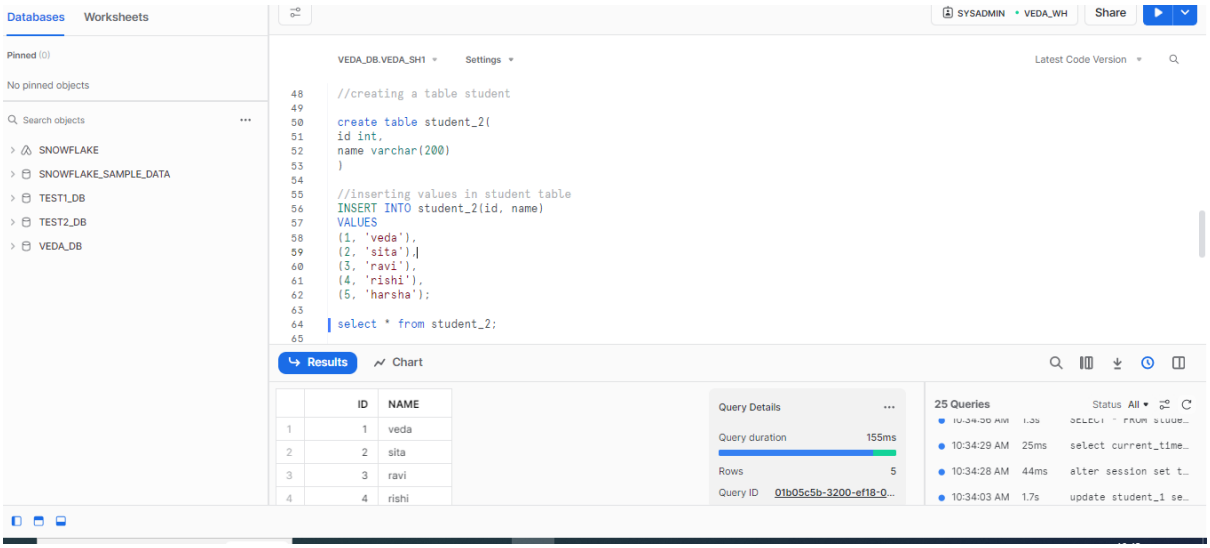
  Like I want to access the data present in table before that timestamp

Select records from the 'student' table before a specific timestamp

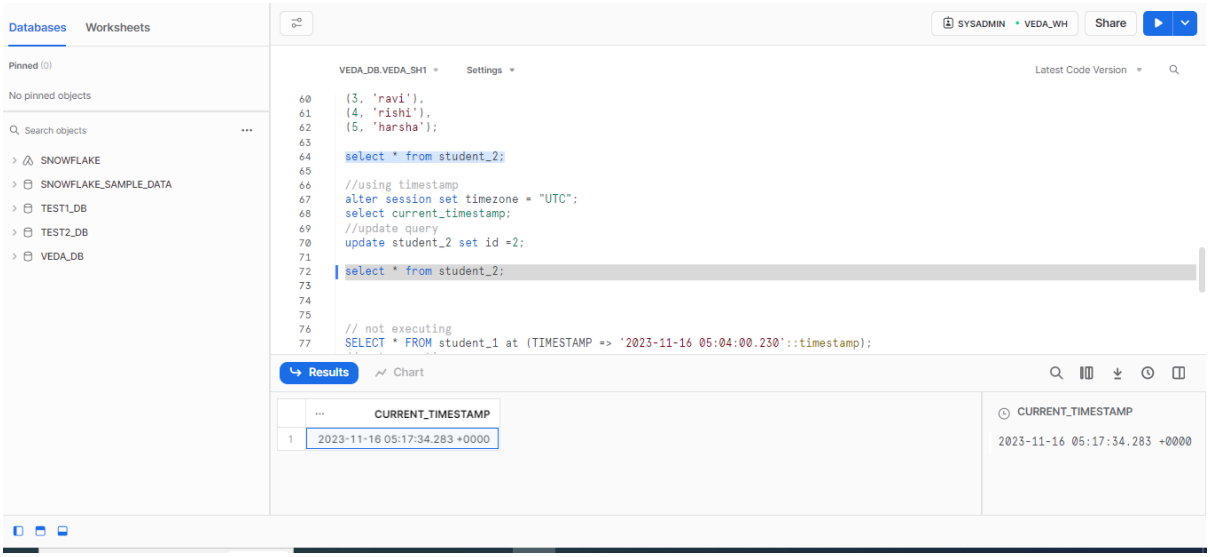
1. First here I am using the query so that it sets the session time zone to UTC (Coordinated Universal Time).
   Query :-
   alter session set timezone = "UTC";
2. This query retrieves the current timestamp in the session's time zone.
   Query :-
   select current_timestamp;
3. This query retrieves records from the 'student' table as they existed before the specified timestamp. “before and at” both are used for time travel.
   Query :-
   SELECT * FROM student_1 at (TIMESTAMP => '2023-11-16 05:04:00.230'::timestamp);
   SELECT * FROM student_1 before (TIMESTAMP => '2023-11-16 05:04:00.230'::timestamp);

Result is :-

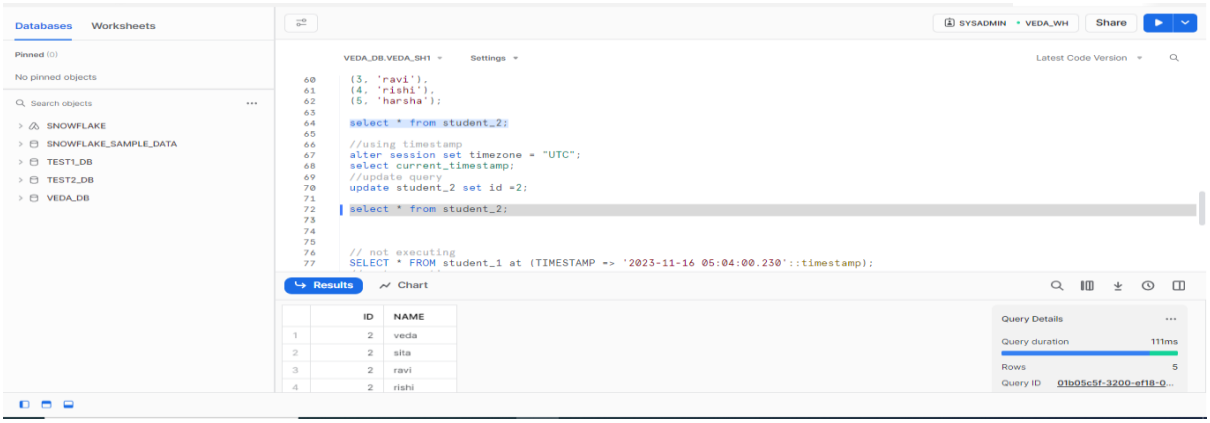
1. I have created a student table and inserted 5 records and retrieved the data.



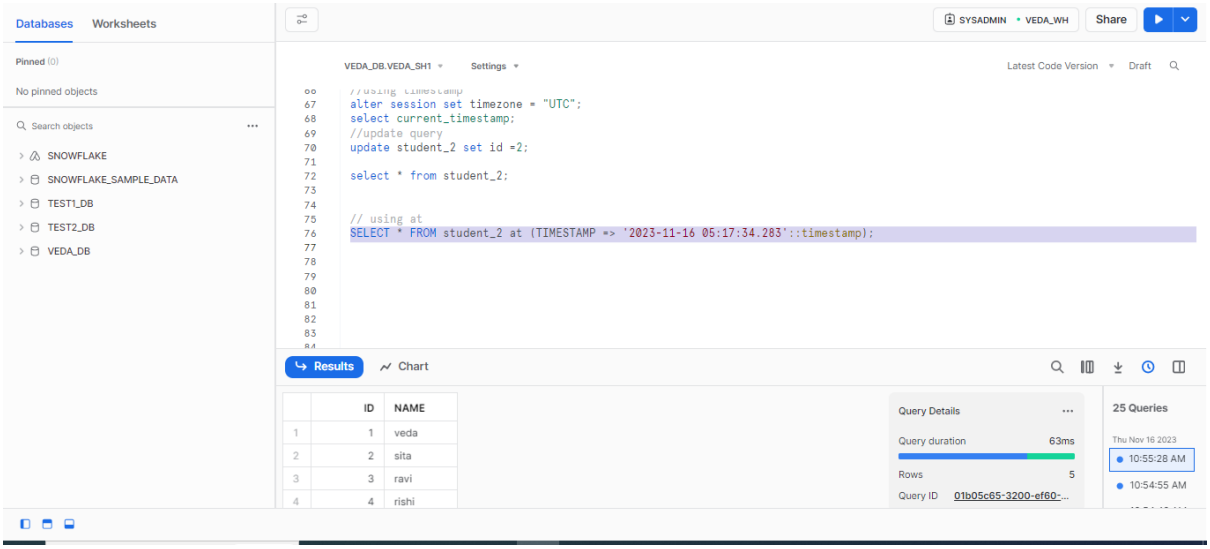
2. I have got the current timestamp now



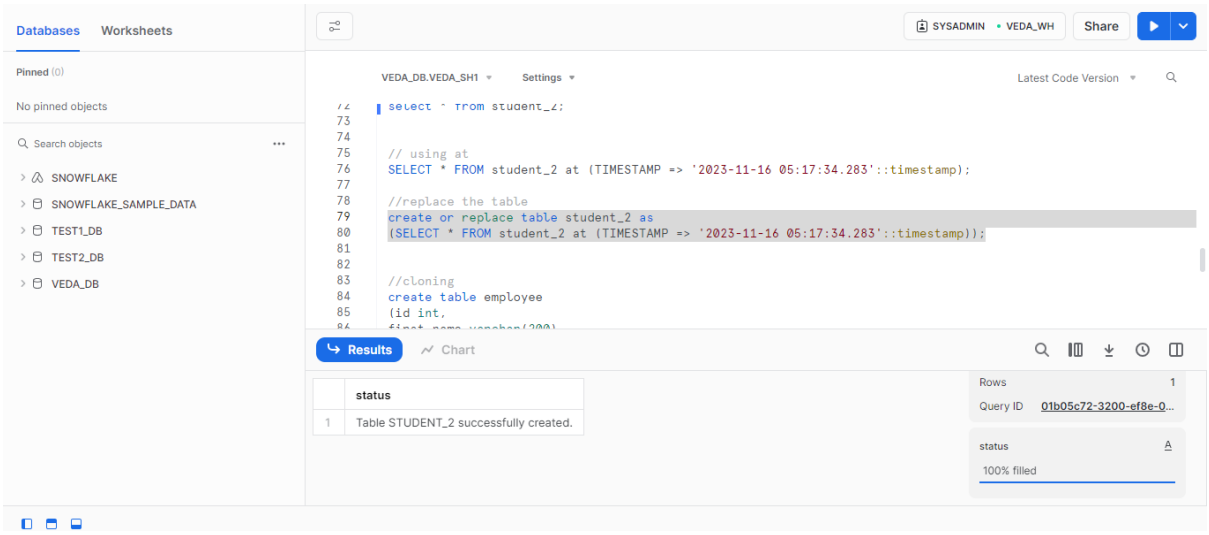
3. Now I am updating all the rows with id is 2



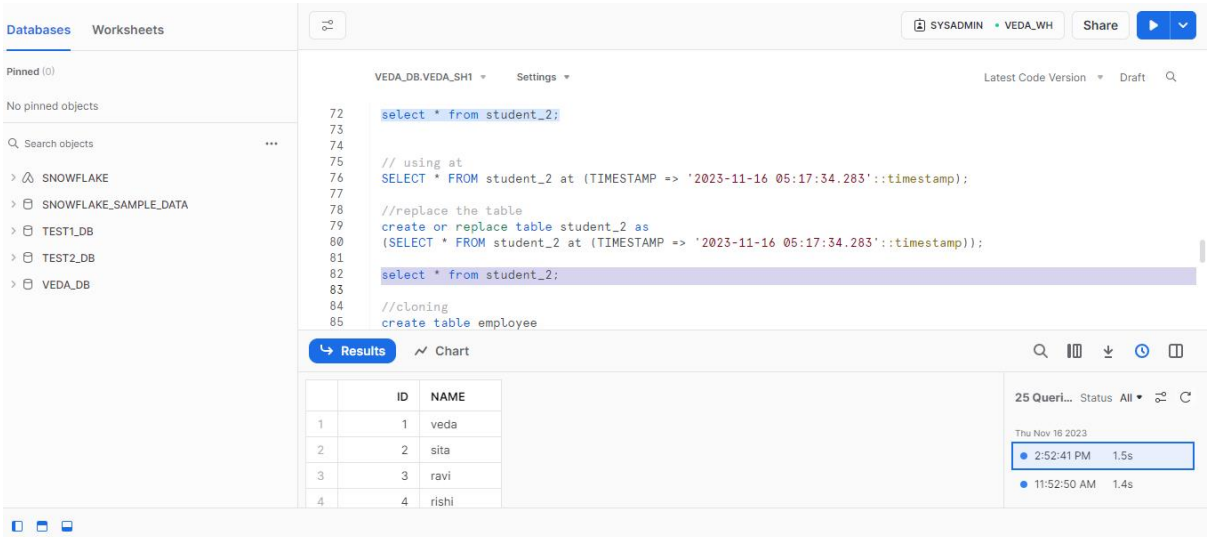
4. Now I am using at to retrieve the data back



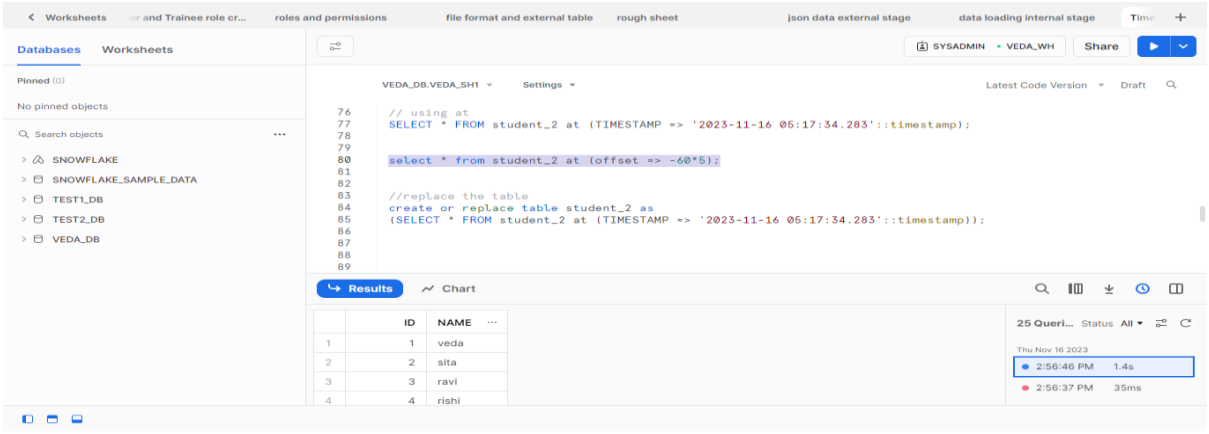
5. Now we need to replace the table



6. I got the previous data now in the student table

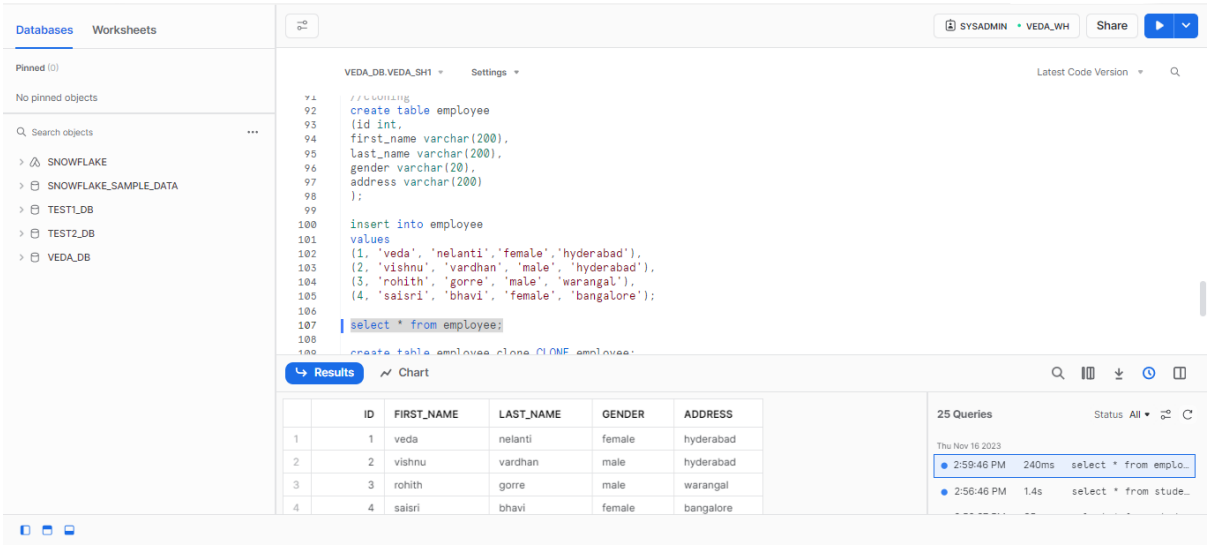


We can use offset also now

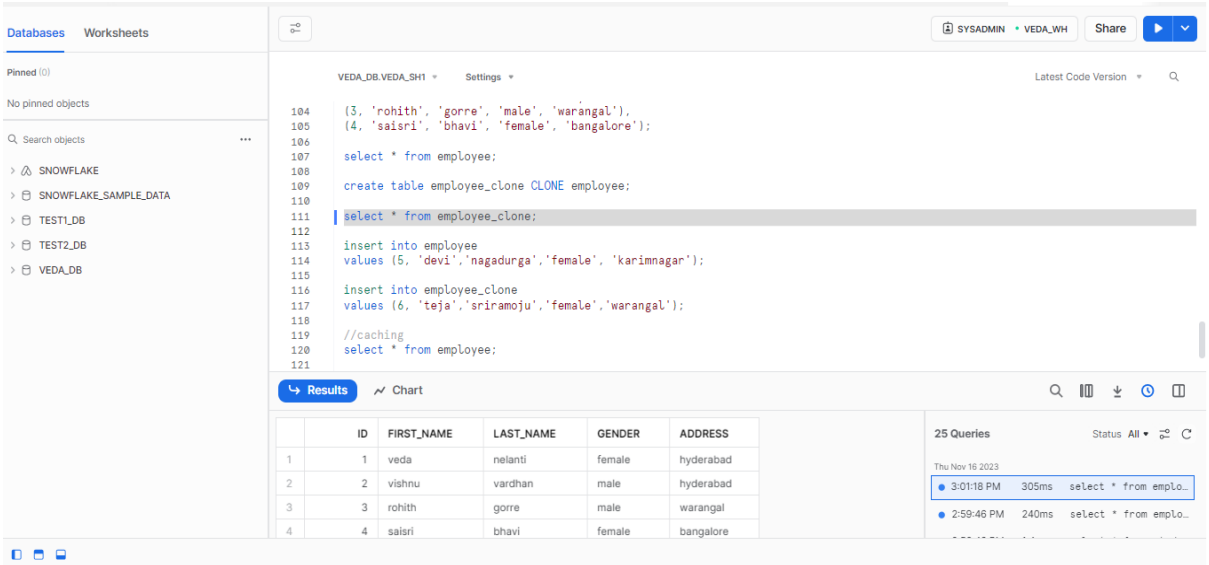


2. Cloning :-

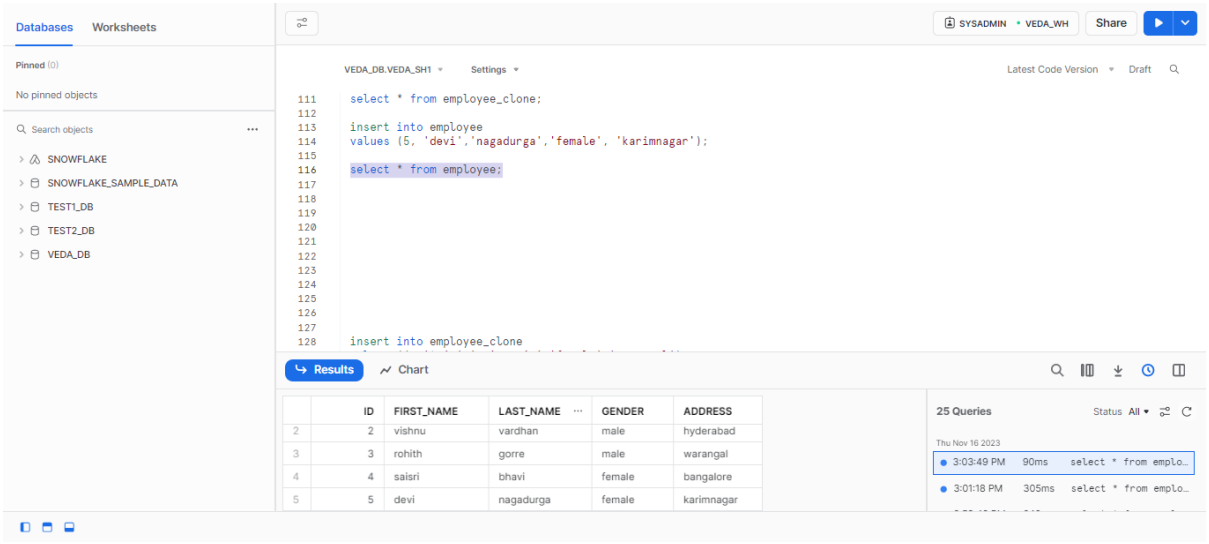
I have created a table called employee and inserted the values



Now I have created employee_clone table



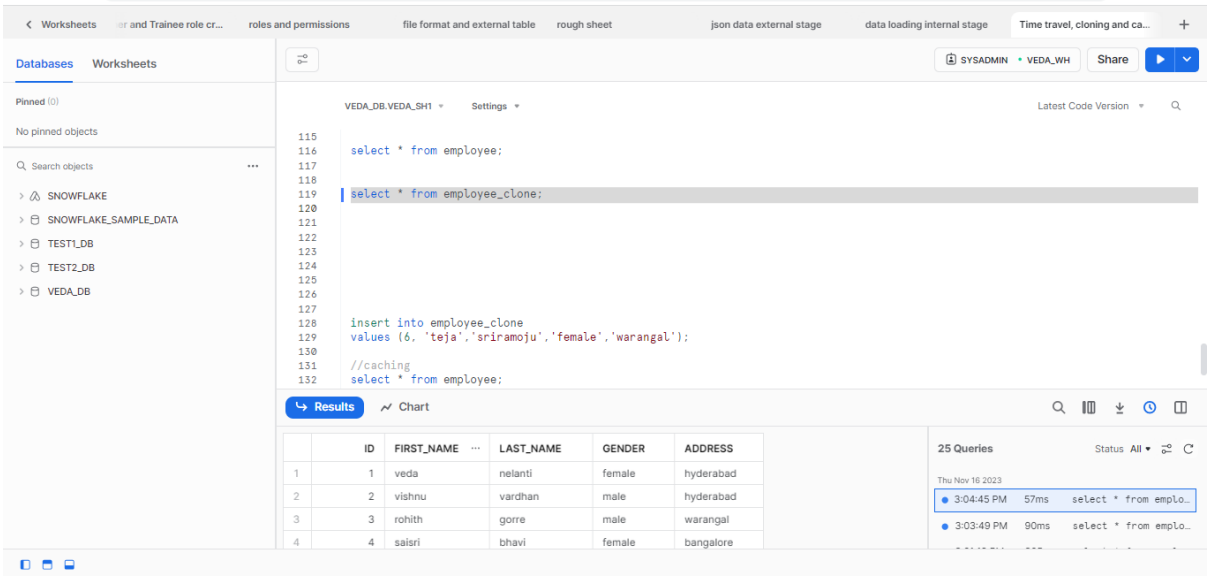
Here I have added a other record in employee table



```
select * from employee_clone;

insert into employee
values (5, 'devi','nagadurga','female', 'karimnagar');

select * from employee;

insert into employee_clone
```

	ID	FIRST_NAME	LAST_NAME	GENDER	ADDRESS
2	2	vishnu	vardhan	male	hyderabad
3	3	rohith	gorre	male	warangal
4	4	saisri	bhavi	female	bangalore
5	5	devi	nagadurga	female	karimnagar

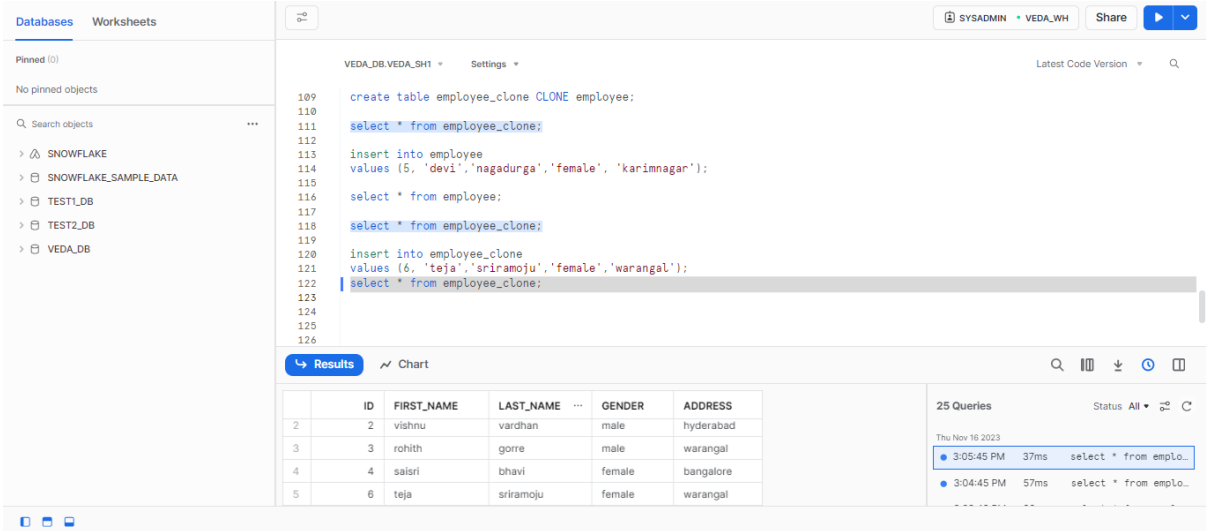
The record is not added in clone table



```
select * from employee;

select * from employee_clone;

insert into employee_clone
values (6, 'teja','sriramoju','female','warangal');

//caching
select * from employee;
```

	ID	FIRST_NAME	LAST_NAME	GENDER	ADDRESS
1	1	veda	nelanti	female	hyderabad
2	2	vishnu	vardhan	male	hyderabad
3	3	rohith	gorre	male	warangal
4	4	saisri	bhavi	female	bangalore

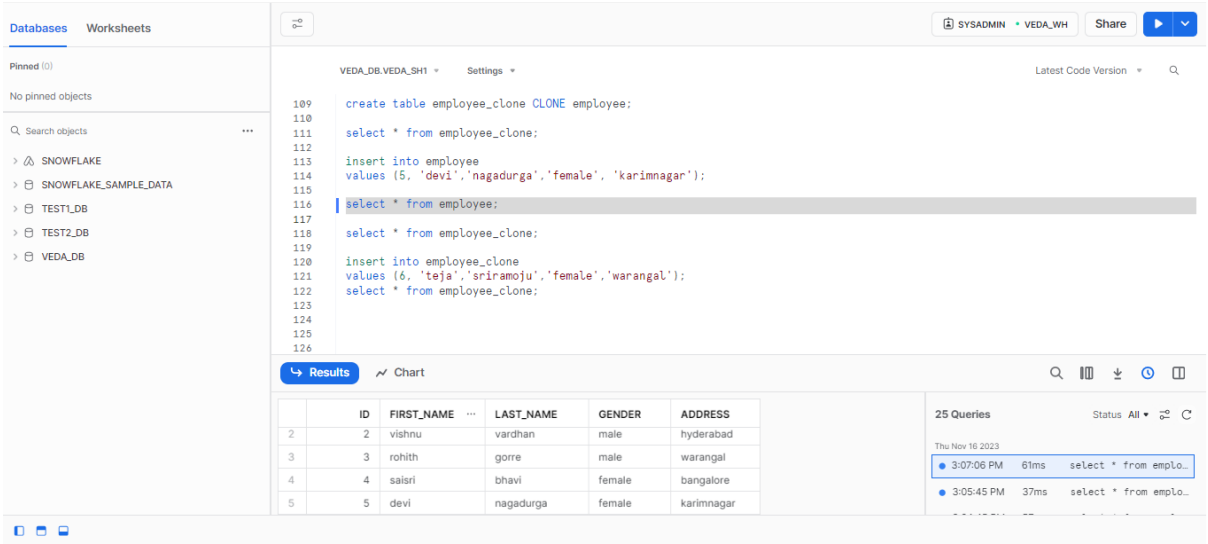
Now I have added the one record in clone table



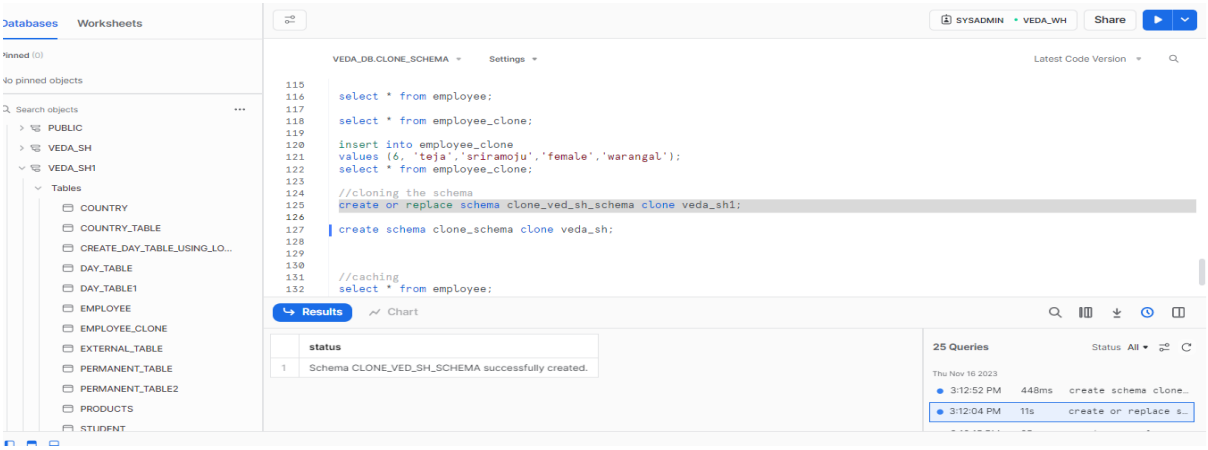
```
create table employee_clone CLONE employee;

select * from employee_clone;

insert into employee
values (5, 'devi','nagadurga','female', 'karimnagar');

select * from employee;

select * from employee_clone;

insert into employee_clone
values (6, 'teja','sriramoju','female','warangal');
select * from employee_clone;
```

	ID	FIRST_NAME	LAST_NAME	GENDER	ADDRESS
2	2	vishnu	vardhan	male	hyderabad
3	3	rohith	gorre	male	warangal
4	4	saisri	bhavi	female	bangalore
5	6	teja	sriramoju	female	warangal

When I query to get the data from employee table then the record which has been added in employee_clone table is not there in employee so main table and clone table is not dependant on each other



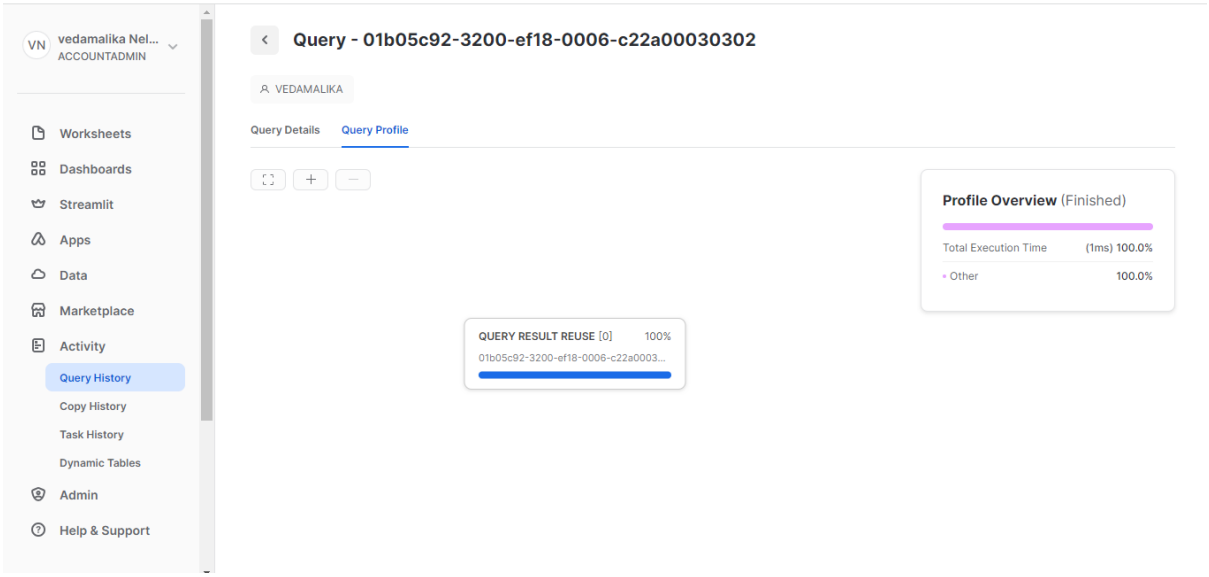
Now we can also clone the schema



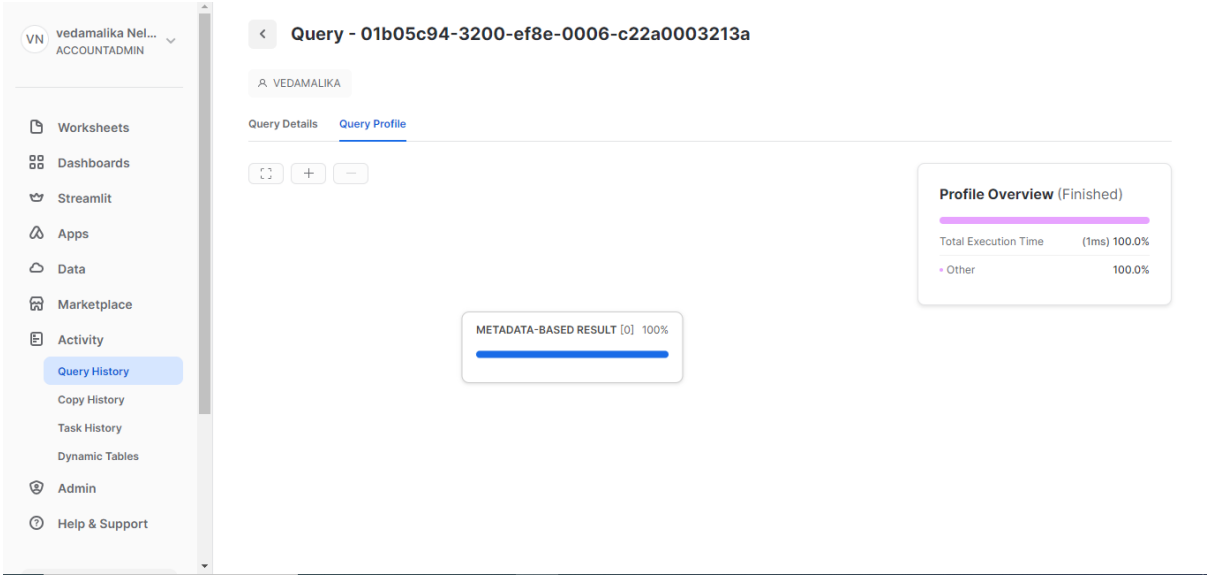
we can clone the database too

3. Cache :-

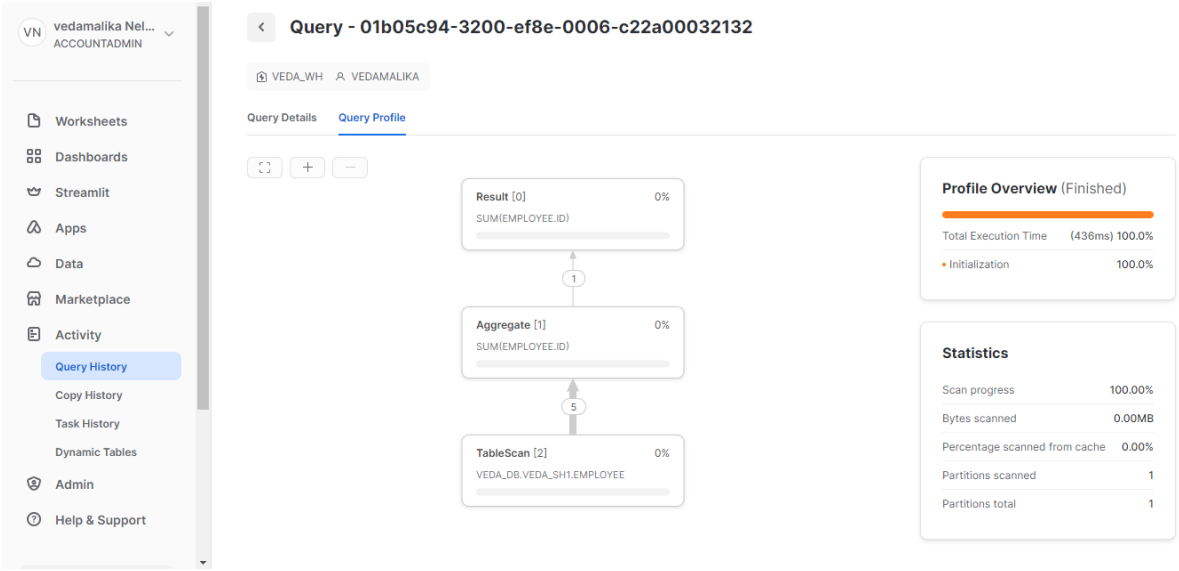
We have result cache



We have meta data cache



For sum aggregate function from the table scan the result has been achieved



From table scan we have got the result