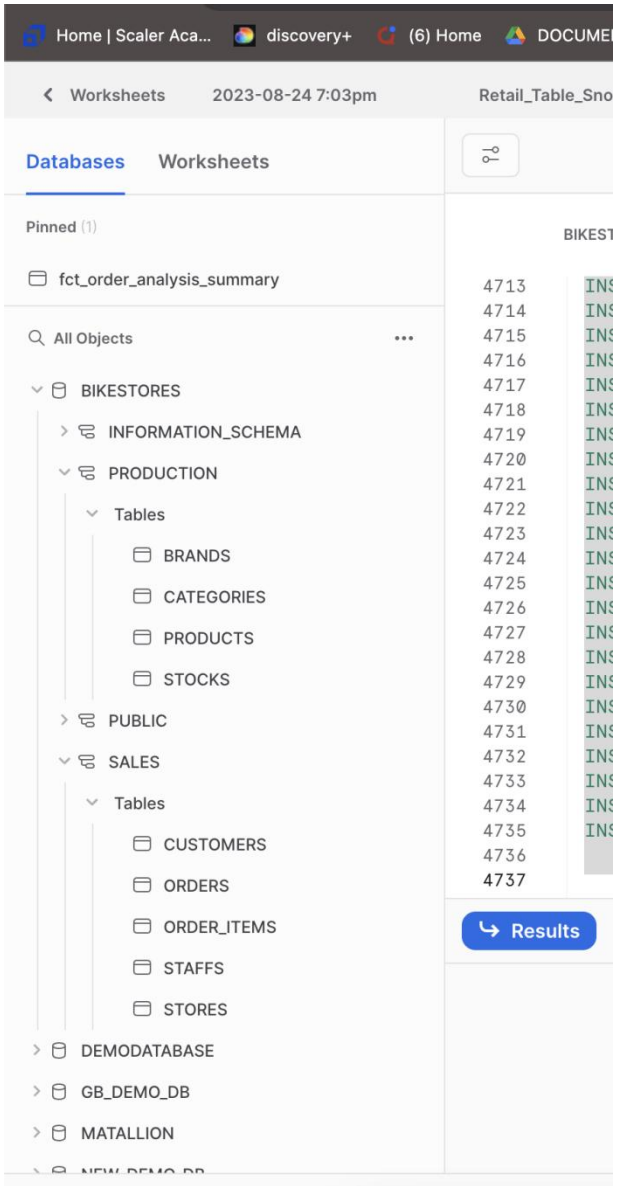
1) Designed the completed database, schema and tables. Included all the constraints while creating the table.

```
create database bikestore;

Create schema bikestore.sales;

Create schema bikestore.production;

CREATE TABLE production.categories (
	category_id INT IDENTITY (1, 1) PRIMARY KEY,
	category_name VARCHAR (255) NOT NULL
);

CREATE TABLE production.brands (
	brand_id INT IDENTITY (1, 1) PRIMARY KEY,
	brand_name VARCHAR (255) NOT NULL
);

CREATE TABLE production.products (
	product_id INT IDENTITY (1, 1) PRIMARY KEY,
	product_name VARCHAR (255) NOT NULL,
	brand_id INT NOT NULL,
	category_id INT NOT NULL,
	model_year SMALLINT NOT NULL,
	list_price DECIMAL (10, 2) NOT NULL
);

CREATE TABLE production.stocks (
	store_id INT,
	product_id INT,
	quantity INT,
	PRIMARY KEY (store_id, product_id)
);

CREATE TABLE sales.stores (
	store_id INT IDENTITY (1, 1) PRIMARY KEY,
	store_name VARCHAR (255) NOT NULL,
	phone VARCHAR (25),
	email VARCHAR (255),
	street VARCHAR (255),
	city VARCHAR (255),
	state VARCHAR (10),
	zip_code VARCHAR (5)
```

```
);

CREATE TABLE sales.staffs (
	staff_id INT	PRIMARY KEY,
	first_name VARCHAR (50) NOT NULL,
	last_name VARCHAR (50) NOT NULL,
	email VARCHAR (255) NOT NULL UNIQUE,
	phone VARCHAR (25),
	active INT NOT NULL,
	store_id INT NOT NULL,
	manager_id INT
);

CREATE TABLE sales.orders (
	order_id INT IDENTITY (1, 1) PRIMARY KEY,
	customer_id INT,
	order_status tinyint NOT NULL,
	order_date varchar(20),
	required_date varchar(20),
	shipped_date varchar(20),
	store_id INT NOT NULL,
	staff_id INT NOT NULL
);

CREATE TABLE order_items (
	order_id INT,
	item_id INT,
	product_id INT NOT NULL,
	quantity INT NOT NULL,
	list_price DECIMAL (10, 2) NOT NULL,
	discount DECIMAL (10, 2) NOT NULL,
	PRIMARY KEY (order_id, item_id)
);

CREATE TABLE sales.customers (
	customer_id INT IDENTITY (1, 1) PRIMARY KEY,
	first_name VARCHAR (255) NOT NULL,
	last_name VARCHAR (255) NOT NULL,
	phone VARCHAR (25),
	email VARCHAR (255) NOT NULL,
	street VARCHAR (255),
	city VARCHAR (50),
	state VARCHAR (25),
	zip_code VARCHAR (5)
```

);


Home | Scaler Aca...
discovery+
(6) Home
DOCUME
Worksheets
2023-08-24 7:03pm
Retail_Table_Sno
Databases
Worksheets
Pinned (1)
fct_order_analysis_summary
All Objects
BIKESTORES
INFORMATION_SCHEMA
PRODUCTION
Tables
BRANDS
CATEGORIES
PRODUCTS
STOCKS
PUBLIC
SALES
Tables
CUSTOMERS
ORDERS
ORDER_ITEMS
STAFFS
STORES
DEMODATABASE
GB_DEMO_DB
MATALLION
BIKEST
4713
4714
4715
4716
4717
4718
4719
4720
4721
4722
4723
4724
4725
4726
4727
4728
4729
4730
4731
4732
4733
4734
4735
4736
4737
Results

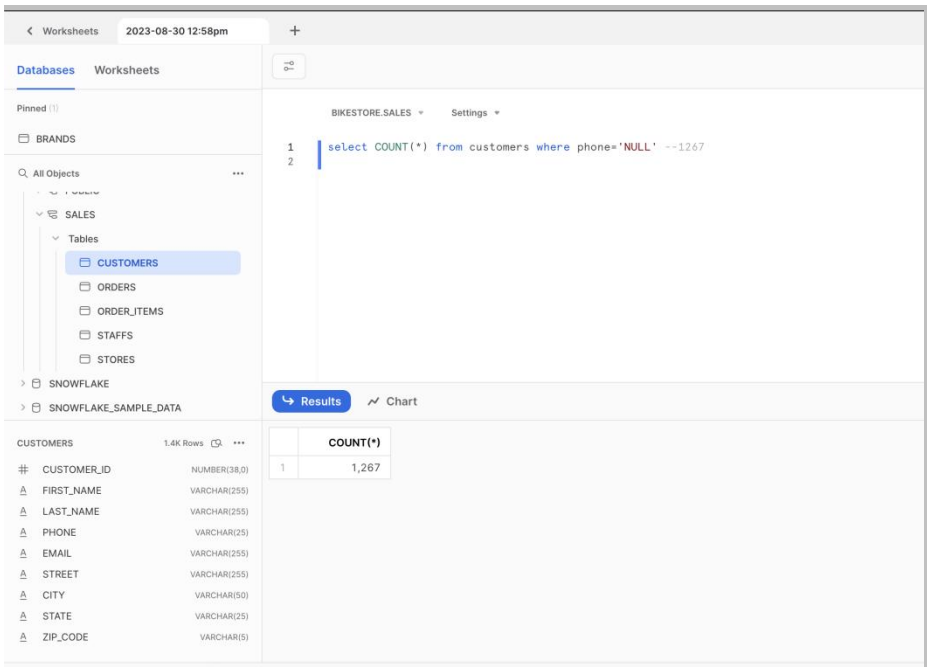
2) Implemented the foreign key constaint using alter table command for the require tables. Below are the changes

```
alter table staffs
add CONSTRAINT store_id
FOREIGN KEY (store_id) REFERENCES stores(store_id);

alter table staffs
add CONSTRAINT manager_id
FOREIGN KEY (manager_id) REFERENCES stores(store_id);

---

alter table order_items
add CONSTRAINT order_id
FOREIGN KEY (order_id) REFERENCES orders(order_id);

alter table order_items
add CONSTRAINT product_id
FOREIGNKEY(product_id)REFERENCES
BIKESTORE.PRODUCTION.PRODUCTS(product_id);

----
alter table orders
add CONSTRAINT customer_id
FOREIGN KEY (customer_id) REFERENCES customers(customer_id);

alter table orders
add CONSTRAINT store_id
FOREIGN KEY (store_id) REFERENCES stores(store_id);

alter table orders
add CONSTRAINT staff_id
FOREIGN KEY (staff_id) REFERENCES staffs(staff_id);

----

alter table stocks
add CONSTRAINT store_id
FOREIGN KEY (store_id) REFERENCES BIKESTORE.SALES.STORES(store_id);

alter table stocks
add CONSTRAINT product_id
FOREIGN KEY (product_id) REFERENCES products(product_id);
```

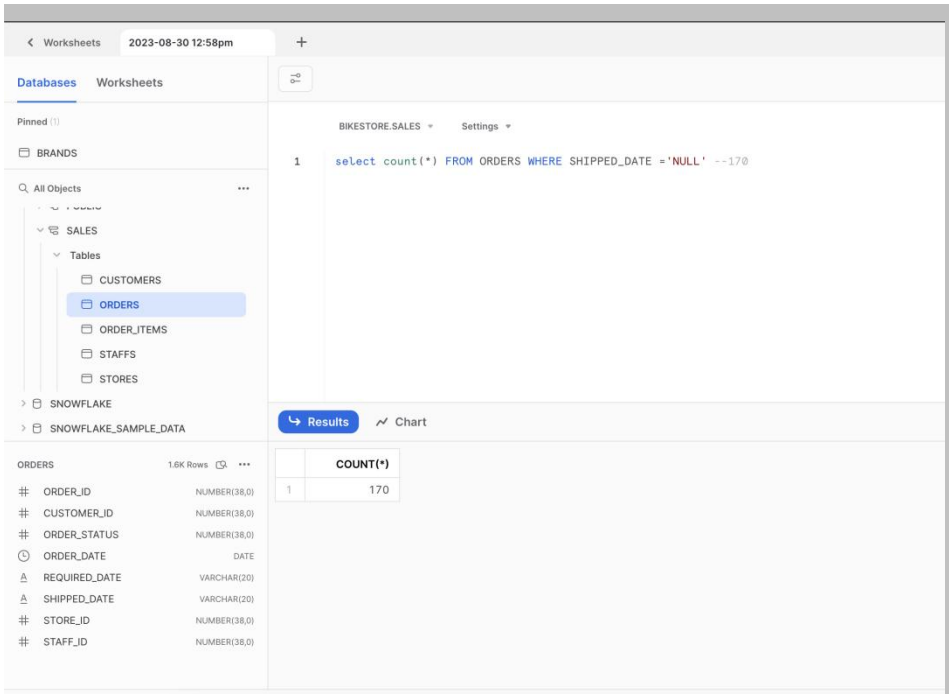
```
alter table products
add CONSTRAINT category_id
FOREIGN KEY (category_id) REFERENCES categories(category_id);

alter table products
add CONSTRAINT brand_id
FOREIGN KEY (brand_id) REFERENCES brands(brand_id);
```

3) Does any of the table has missing or NULL value ? If yes which are those and what are their counts ?

```
select *
from table_name
where column_name IS NULL or column_name IS NULL ...
/* This is general query which will give us the record where column is null */

select COUNT(*) from customers where phone='NULL' --1267
```




```
select count(*) FROM ORDERS WHERE SHIPPED_DATE ='NULL' --170
```



4) Does the datasets has any DUPLICATE(identical rows) ? If yes – can you just keep the first record and remove all rest if its possible without using any JOINS or WINDOW function

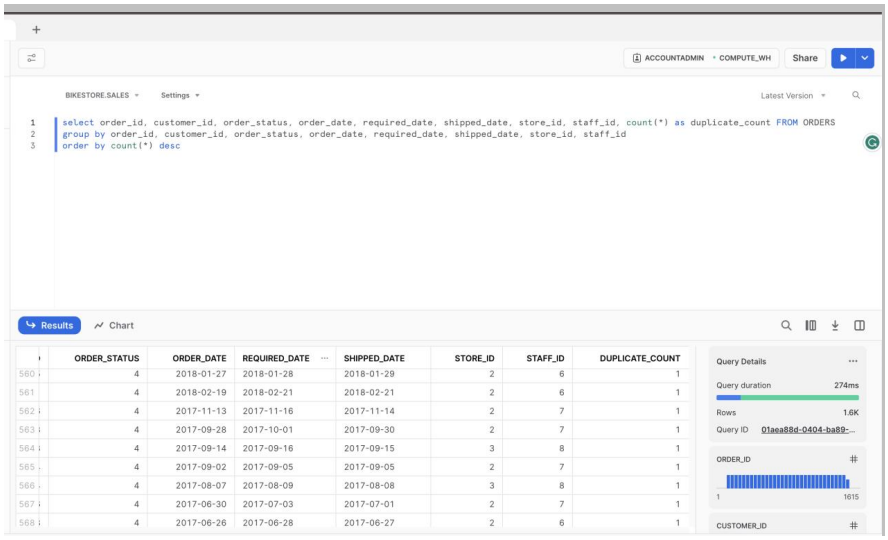
No the dataset doesn't contain any duplicate rows. I checked by grouping all the columns and taking count of it.

Example - select order_id, customer_id, order_status, order_date, required_date, shipped_date, store_id, staff_id, count(*) as duplicate_count

FROM ORDERS

group by order_id, customer_id, order_status, order_date, required_date, shipped_date, store_id, staff_id

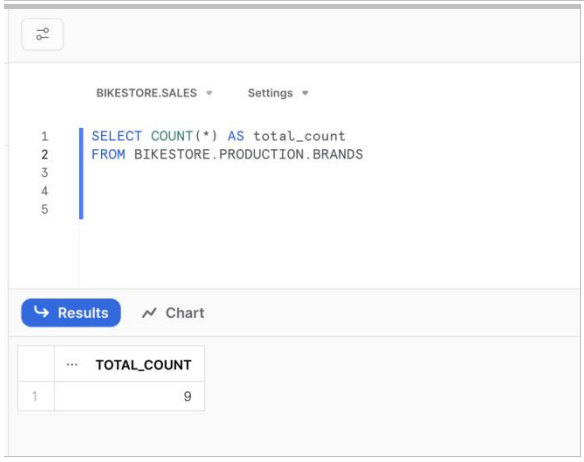
order by count(*) desc



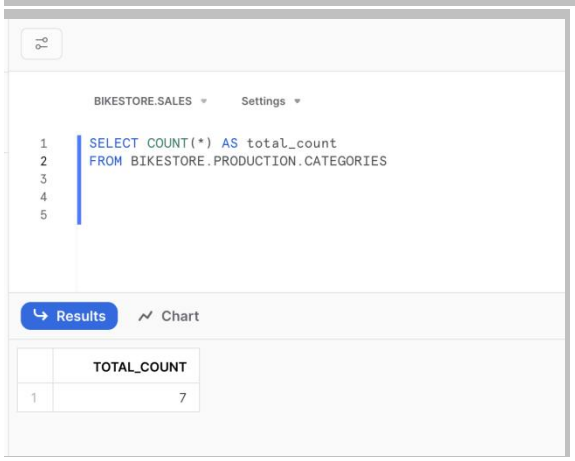
5) How many unique tables are present in each schema and under each table how many records are we having ? (Write SQL Script for the same – I don't need answer like 3/5/4 etc)

There are 4 unique tables present in production schema and 5 tables present in sales scehma.

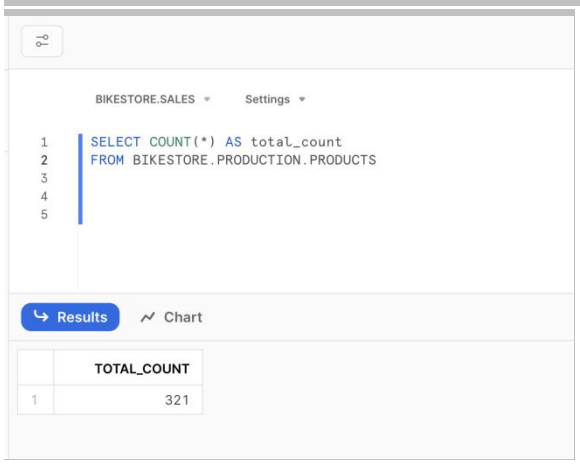
BIKESTORE.PRODUCTION.BRANDS - 9 records



BIKESTORE.PRODUCTION.CATEGORIES - 7 records



BIKESTORE.PRODUCTION.PRODUCTS - 321 records



BIKESTORE.PRODUCTION.STOCKS-939records

BIKESTORE.SALES Settings

```
SELECT COUNT(*) AS total_count
FROM BIKESTORE.PRODUCTION.STOCKS
```

Results Chart

	TOTAL_COUNT
1	939

BIKESTORE.SALES.CUSTOMERS - 1445 records

BIKESTORE.SALES Settings

```
SELECT COUNT(*) AS total_count
FROM BIKESTORE.SALES.CUSTOMERS
```

Results Chart

	TOTAL_COUNT
1	1,445

BIKESTORE.SALES.STORES - 3 records

BIKESTORE.SALES Settings

```
SELECT COUNT(*) AS total_count
FROM BIKESTORE.SALES.STORES
```

Results Chart

	TOTAL_COUNT
1	3

BIKESTORE.SALES.STAFFS - 10 records

BIKESTORE.SALES Settings

```
SELECT COUNT(*) AS total_count
FROM BIKESTORE.SALES.STAFFS
```

Results Chart

	TOTAL_COUNT
1	10

BIKESTORE.SALES.ORDER_ITEMS - 4722 records

BIKESTORE.SALES Settings

```
SELECT COUNT(*) AS total_count
FROM BIKESTORE.SALES.ORDER_ITEMS
```

Results Chart

	TOTAL_COUNT
1	4,722

BIKESTORE.SALES.ORDERS - 1615 records

BIKESTORE.SALES Settings

```
SELECT COUNT(*) AS total_count
FROM ORDERS
```

Results Chart

	TOTAL_COUNT
1	1,615

6) How many total serving customer BikeStore has ?

select count(distinct customer_id) as total_serving_customer from customers -- 1445

BIKESTORE.SALES Settings

```
select count(distinct customer_id) as total_serving_customer from customers
```

Results Chart

	TOTAL_SERVING_CUSTOMER
1	1,445

7) How many total orders are there ?

select count(*) as total_orders from orders - 1615

BIKESTORE.SALES Settings

```
select count(*) as total_orders from orders
```

Results Chart

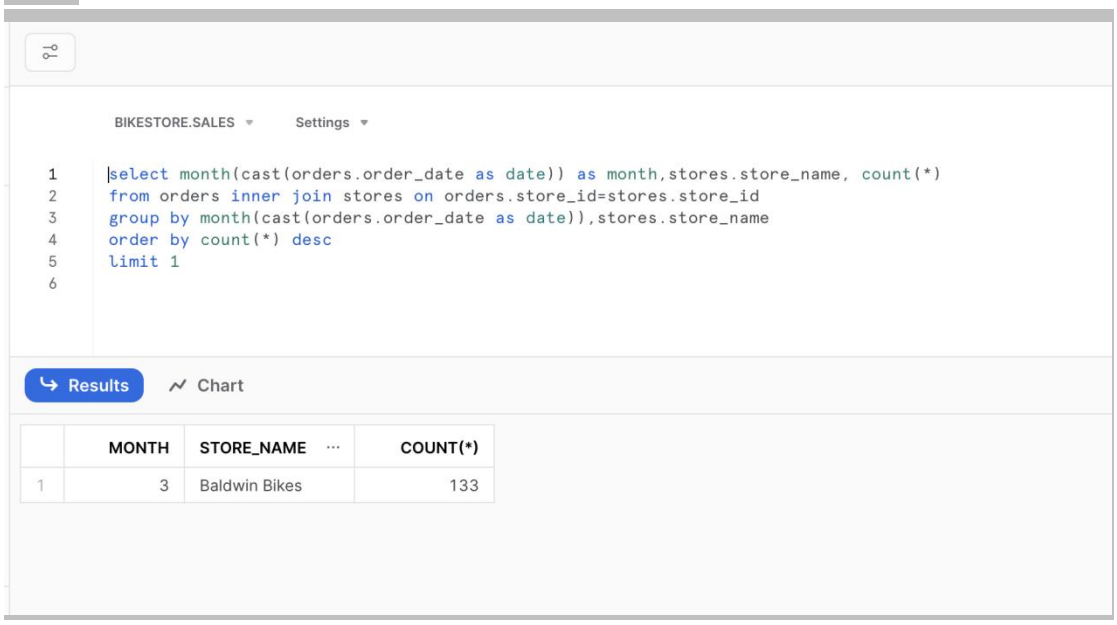
	TOTAL_ORDERS
1	1,615

8) Which store has the highest number of sales ?

```
select o.store_id,sum((oi.list_price*oi.quantity)-oi.discount) as total_sales, s.store_name
From orders as o join order_items as oi on oi.order_id=o.order_id
join stores as s on o.store_id=s.store_id
group by o.store_id,s.store_name
order by total_sales desc
Limit
1
```

9) Which month the sales was highest and for which store ?

```
select month(cast(orders.order_date as date)) as month,stores.store_name, count(*)
from orders inner join stores on orders.store_id=stores.store_id
group by month(cast(orders.order_date as date)),stores.store_name
order by count(*) desc
limit 1
```



10) How many orders each customer has placed (give me top 10 customers)

```
SELECT CUSTOMER_ID FROM ORDERS
GROUP BY CUSTOMER_ID
ORDER BY COUNT(ORDER_ID) DESC LIMIT 10
```

BIKESTORE.SALES Settings

```
SELECT CUSTOMER_ID
FROM ORDERS
GROUP BY 1
ORDER BY COUNT(ORDER_ID)  DESC
LIMIT 10
```

Results Chart

	CUSTOMER_ID
1	13
2	9
3	12
4	31
5	24
6	43
7	7
8	66
9	46
10	50

11) Which are the TOP 3 selling product ?

```
select p.product_id,p.product_name,sum((oi.list_price*oi.quantity)-oi.discount) as total_sales
From ORDER_ITEMS as oi join BIKESTORE.PRODUCTION.PRODUCTS as p on oi.product_id=p.product_id
group by p.product_id,p.product_name
order by total_sales desc
limit 3
```

ACCOUNTADMIN

BIKESTORE.SALES Settings

```
select p.product_id,p.product_name,sum((oi.list_price*oi.quantity)-oi.discount) as total_sales
From ORDER_ITEMS as oi join BIKESTORE.PRODUCTION.PRODUCTS as p on oi.product_id=p.product_id
group by p.product_id,p.product_name
order by total_sales desc
limit 3
```

Results Chart

	PRODUCT_ID	PRODUCT_NAME	TOTAL_SALES
1	7	"Trek Slash 8 27.5 - 2016"	615,988.44
2	9	"Trek Conduit+ - 2016"	434,988.03
3	4	"Trek Fuel EX 8 29 - 2016"	414,687.65

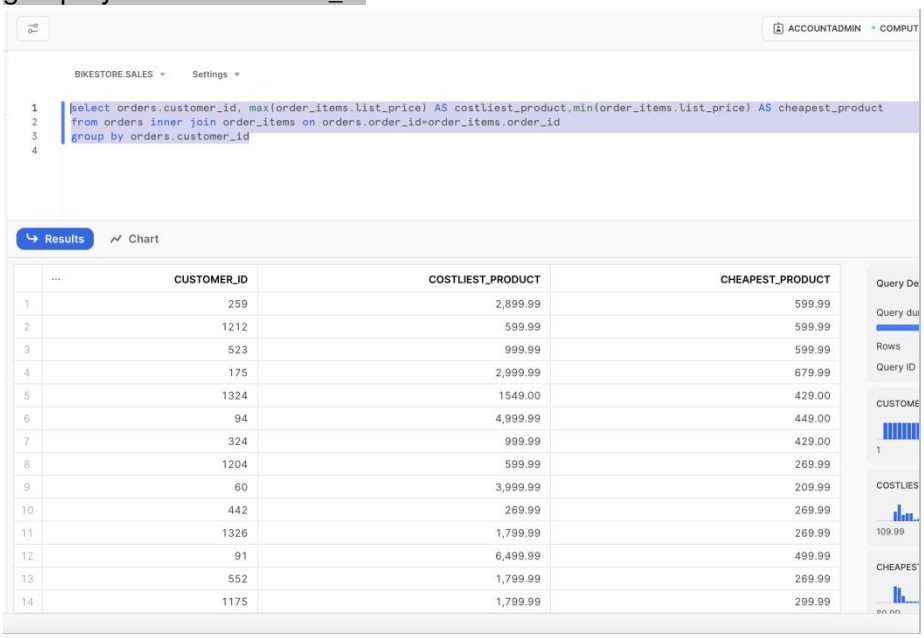
12) Which was the first and last order placed by the customer who has placed maximum number of orders ?

select customer_id, min(order_date) as first_order_date, max(order_date) as last_order_date, count(order_id)
from orders
group by 1
order by count(order_id) desc, min(order_date)
limit 1

	CUSTOMER_ID	FIRST_ORDER_DATE	LAST_ORDER_DATE	...	COUNT(ORDER_ID)
1	50	2016-02-11	2018-04-12		3

13) For every customer , which is the cheapest product and the costliest product which the customer has bought.

Select orders.customer_id, max(order_items.list_price) AS costliest_product, min(order_items.list_price) AS cheapest_product
from orders inner join order_items on orders.order_id=order_items.order_id
group by orders.customer_id



	...	CUSTOMER_ID	COSTLIEST_PRODUCT	CHEAPEST_PRODUCT
1		259	2,899.99	599.99
2		1212	599.99	599.99
3		523	999.99	599.99
4		175	2,999.99	679.99
5		1324	1549.00	429.00
6		94	4,999.99	449.00
7		324	999.99	429.00
8		1204	599.99	269.99
9		60	3,999.99	209.99
10		442	269.99	269.99
11		1326	1,799.99	269.99
12		91	6,499.99	499.99
13		552	1,799.99	269.99
14		1175	1,799.99	299.99

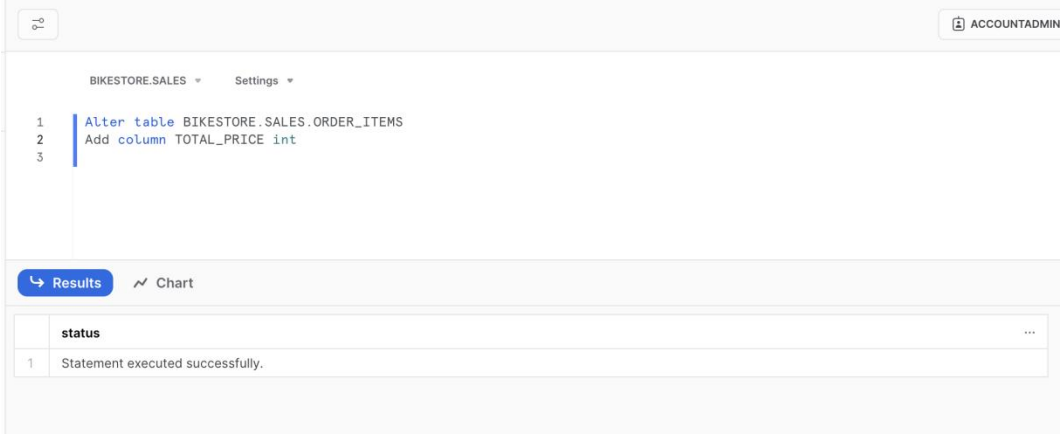
14) Which product has orders more than 200 ?

```
select product_id, count(*) as number_of_orders
from order_items
group by product_id
having count(*)>200
```

There are no product whose orders are more than 200

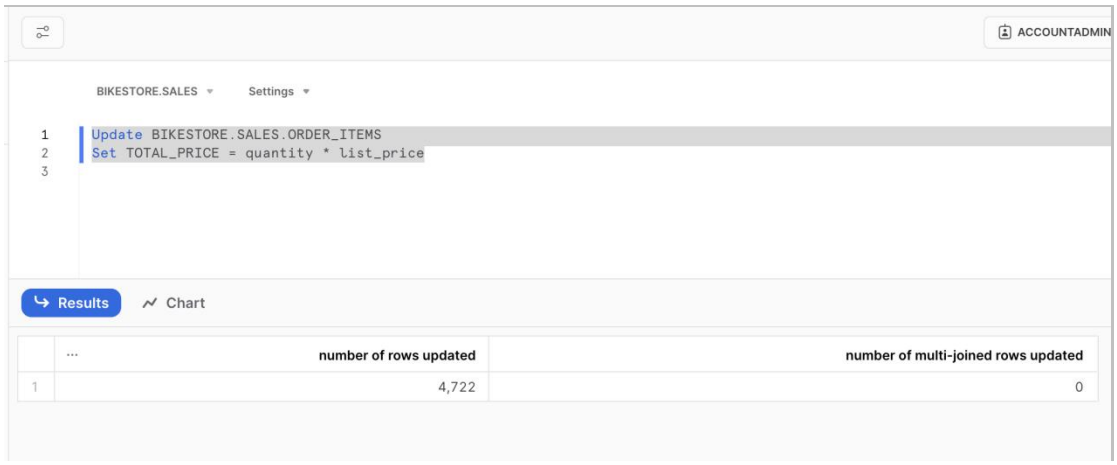
15) Add a column TOTAL_PRICE with appropriate data type into the sales.order_items

```
Alter table sales.order_items
Add column TOTAL_PRICE int
```



16) Calculate TOTAL_PRICE = quantity * list price and Update the value for all rows in the sales.order_items table.

Update sales.order_items
Set TOTAL_PRICE = quantity * list



17) What is the value of the TOTAL_PRICE paid for all the sales.order_items ?

Select sum(TOTAL_PRICE) from sales.order_items