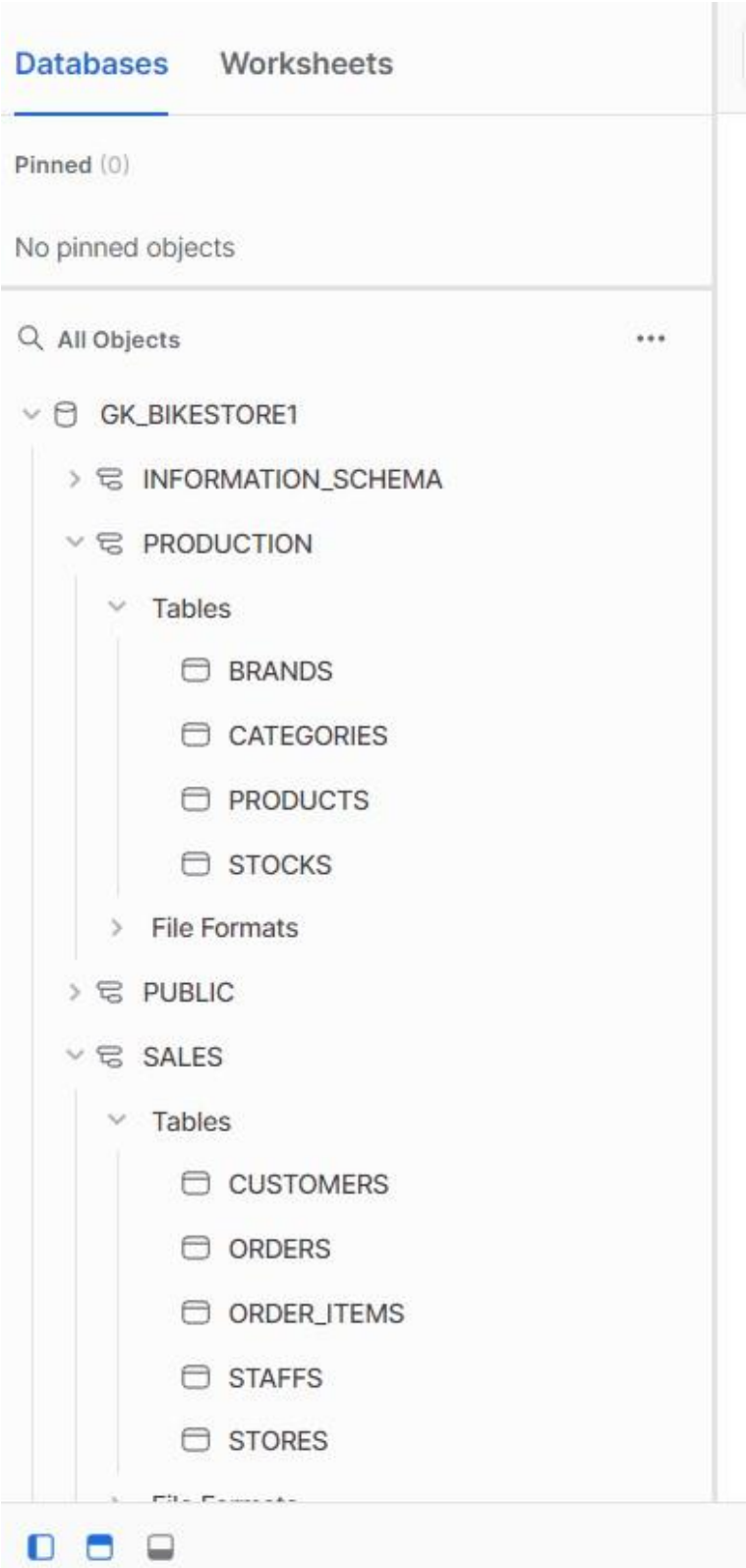
1.Design the complete database + schema + tables for the diagram shown above using appropriate data type for every column along with any contraints (checks + PK) mentioned in the task description and load the below data into the requisite tables.



EER DIAGRAM

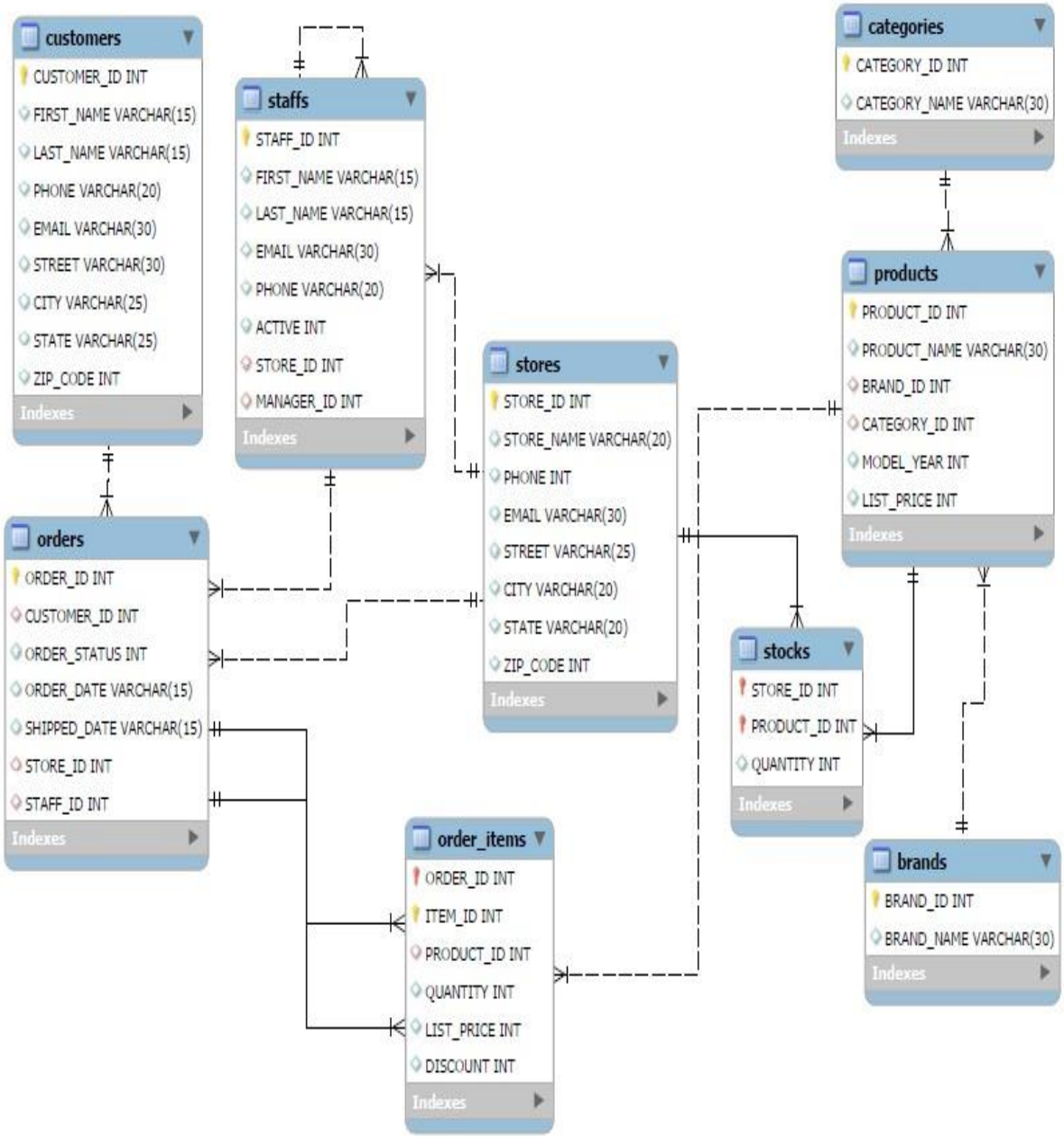

customers
CUSTOMER_ID INT
FIRST_NAME VARCHAR(15)
LAST_NAME VARCHAR(15)
PHONE VARCHAR(20)
EMAIL VARCHAR(30)
STREET VARCHAR(30)
CITY VARCHAR(25)
STATE VARCHAR(25)
ZIP_CODE INT
Indexes
staffs
STAFF_ID INT
FIRST_NAME VARCHAR(15)
LAST_NAME VARCHAR(15)
EMAIL VARCHAR(30)
PHONE VARCHAR(20)
ACTIVE INT
STORE_ID INT
MANAGER_ID INT
Indexes
categories
CATEGORY_ID INT
CATEGORY_NAME VARCHAR(30)
Indexes
products
PRODUCT_ID INT
PRODUCT_NAME VARCHAR(30)
BRAND_ID INT
CATEGORY_ID INT
MODEL_YEAR INT
LIST_PRICE INT
Indexes
stores
STORE_ID INT
STORE_NAME VARCHAR(20)
PHONE INT
EMAIL VARCHAR(30)
STREET VARCHAR(25)
CITY VARCHAR(20)
STATE VARCHAR(20)
ZIP_CODE INT
Indexes
orders
ORDER_ID INT
CUSTOMER_ID INT
ORDER_STATUS INT
ORDER_DATE VARCHAR(15)
SHIPPED_DATE VARCHAR(15)
STORE_ID INT
STAFF_ID INT
Indexes
stocks
STORE_ID INT
PRODUCT_ID INT
QUANTITY INT
Indexes
order_items
ORDER_ID INT
ITEM_ID INT
PRODUCT_ID INT
QUANTITY INT
LIST_PRICE INT
DISCOUNT INT
Indexes
brands
BRAND_ID INT
BRAND_NAME VARCHAR(30)
Indexes

TABLE CREATION ON SNOWFLAKE SCRIPT

```
CREATE DATABASE GK_BikeStore1;

CREATE SCHEMA SALES;

CREATE SCHEMA PRODUCTION;

USE GK_BIKESTORE1;

-------CREATING CUSTOMERS TABLE UNDER   THE SALES SCHEMA-------------------

CREATE OR REPLACE TABLE SALES.CUSTOMERS(
CUSTOMER_ID INT AUTOINCREMENT PRIMARY KEY,
FIRST_NAME VARCHAR(15),
LAST_NAME VARCHAR(15),
PHONE VARCHAR(20),
EMAIL VARCHAR(100),
```

```
STREET VARCHAR(80),
CITY VARCHAR(25),
STATE VARCHAR(25),
ZIP_CODE NUMBER(10,0) );

select * from SALES.CUSTOMERS;

---CREATING STAFFS TABLE UNDER THE   SALES SCHEMA-
------------------------

CREATE OR REPLACE TABLE SALES.STAFFS(
STAFF_ID INT PRIMARY KEY,
FIRST_NAME VARCHAR(30),
LAST_NAME VARCHAR(30),
EMAIL VARCHAR(100),
PHONE VARCHAR(20),
ACTIVE INT,
STORE_ID INT,
MANAGER_ID INT );
```

```
SELECT * FROM SALES.STAFFS;

----------CREATING ORDERS TABLE UNDER THE SALES SCHEMA----------

CREATE OR REPLACE TABLE SALES.ORDERS (
ORDER_ID NUMBER(4,0) PRIMARY KEY,
CUSTOMER_ID INT,
ORDER_STATUS NUMBER(3,0),
ORDER_DATE DATE,
REQUIRED_DATE DATE,
SHIPPED_DATE VARCHAR(10),
STORE_ID NUMBER(2,0),
STAFF_ID NUMBER(2,0)
);
```

----------CREATING STORES TABLE UNDER THE SALES SCHEMA----------

```
CREATE OR REPLACE TABLE SALES.STORES(
STORE_ID NUMBER(2,0) AUTOINCREMENT PRIMARY KEY,
STORE_NAME VARCHAR(20),
PHONE varchar(20),
EMAIL VARCHAR(100),
STREET VARCHAR(25),
CITY VARCHAR(20),
STATE VARCHAR(20),
ZIP_CODE INT );
```

----------CREATING ORDER_ITEMS TABLE UNDER THE SALES SCHEMA----------

```
CREATE OR REPLACE TABLE SALES.ORDER_ITEMS(
ORDER_ID INT ,
ITEM_ID INT ,
PRODUCT_ID INT,
QUANTITY INT,
LIST_PRICE INT,
DISCOUNT INT,
PRIMARY KEY (ORDER_ID,ITEM_ID));
```

----------CREATING CATEGORIES TABLE UNDER THE PRODUCTION SCHEMA----------

```
CREATE OR REPLACE TABLE PRODUCTION.CATEGORIES(
CATEGORY_ID INT PRIMARY KEY,
CATEGORY_NAME VARCHAR(30) );
```

----------CREATING PRODUCTS TABLE UNDER THE PRODUCTION SCHEMA----------

```
CREATE OR REPLACE TABLE PRODUCTION.PRODUCTS(
PRODUCT_ID INT PRIMARY KEY,
PRODUCT_NAME VARCHAR(80),
BRAND_ID INT,
CATEGORY_ID INT,
MODEL_YEAR INT,
LIST_PRICE INT);
```

----------CREATING STOCKS TABLE UNDER THE PRODUCTION SCHEMA----------

```
CREATE OR REPLACE TABLE PRODUCTION.STOCKS(
STORE_ID INT ,
PRODUCT_ID INT,
QUANTITY INT,
PRIMARY KEY (STORE_ID,PRODUCT_ID));
```

----------CREATING BRANDS TABLE UNDER THE PRODUCTION SCHEMA----------

```
CREATE OR REPLACE TABLE PRODUCTION.BRANDS(
BRAND_ID INT primary KEY,
BRAND_NAME VARCHAR(30));
```

----------CREATING CSV FILE FORMAT FOR BULK DATA UPLOAD ----------

```
create or replace file format CSV_FILE_FORMAT
   type = 'csv'
   compression = 'none'
   field_delimiter = ','
   field_optionally_enclosed_by = 'none'
   skip_header = 1 ;
```

2. Once the table has got created , there is a requirement of FOREIGN KEY implementation coming into picture where one needs to add(ALTER TABLE COMMAND) below foreign key on the table mentioned pointing to another table (READ ABOUT FOREIGN KEY) as :

------creating foreign key for both the sales and production schema using alter command----------

```
ALTER TABLE staffs
ADD FOREIGN KEY (STORE_ID) references stores(STORE_ID);

ALTER TABLE staffs
ADD FOREIGN KEY (manager_id) REFERENCES staffs(staff_id);
```

```
ALTER TABLE products
ADD  FOREIGN KEY (category_id)  REFERENCES
categories(category_id);

ALTER TABLE products
ADD  FOREIGN KEY (brand_id) REFERENCES
brands(brand_id);

ALTER TABLE ORDERS
ADD  FOREIGN KEY (customer_id) REFERENCES
customers(customer_id) ;

ALTER TABLE ORDERS
ADD  FOREIGN KEY (STORE_ID) REFERENCES
STORES(STORE_ID) ;

ALTER TABLE ORDERS
ADD  FOREIGN KEY (STAFF_ID) REFERENCES
STAFFS(STAFF_ID) ;
```

```
ALTER TABLE ORDER_ITEMS
ADD  FOREIGN KEY (ORDER_ID) REFERENCES
ORDERS(ORDER_ID) ;

ALTER TABLE ORDER_ITEMS
ADD FOREIGN KEy (PRODUCT_ID) REFERENCES
production.PRODUCTS(PRODUCT_ID) ;

ALTER TABLE STOCKS
ADD FOREIGN KEY (STORE_ID) REFERENCES
sales.STORES(STORE_ID) ;

ALTER TABLE STOCKS
ADD FOREIGN KEY (PRODUCT_ID) REFERENCES
PRODUCTS(PRODUCT_ID) ;
```

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

```
-----------FINDING NULL IN BRANDS TABLE --------
SELECT count(*)
FROM GK_BIKESTORE1.PRODUCTION.BRANDS
WHERE BRAND_ID = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.PRODUCTION.BRANDS
WHERE BRAND_NAME = 'NULL';

----------FINDING NULL IN CATEGORIES TABLE --------

SELECT count(*)
FROM GK_BIKESTORE1.PRODUCTION.CATEGORIES
WHERE CATEGORY_ID = 'NULL';
```

```
SELECT count(*)
FROM GK_BIKESTORE1.PRODUCTION.CATEGORIES
WHERE CATEGORY_NAME = 'NULL';
```

----------FINDING NULL IN PRODUCTS TABLE --------

```
SELECT count(*)
FROM GK_BIKESTORE1.PRODUCTION.PRODUCTS
WHERE 'PRODUCT_ID' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.PRODUCTION.PRODUCTS
WHERE PRODUCT_NAME = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.PRODUCTION.PRODUCTS
WHERE 'BRAND_ID' = 'NULL';
```

```
SELECT count(*)
FROM GK_BIKESTORE1.PRODUCTION.PRODUCTS
WHERE 'CATEGORY_ID' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.PRODUCTION.PRODUCTS
WHERE 'MODEL_YEAR' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.PRODUCTION.PRODUCTS
WHERE 'LIST_PRICE' = 'NULL';
```

----------FINDING NULL IN STOCKS TABLE --------

```
SELECT count(*)
FROM GK_BIKESTORE1.PRODUCTION.STOCKS
WHERE 'STORE_ID'= 'NULL';
```

```
SELECT count(*)
FROM GK_BIKESTORE1.PRODUCTION.STOCKS
WHERE 'PRODUCT_ID' = 'NULL';



SELECT count(*)
FROM GK_BIKESTORE1.PRODUCTION.STOCKS
WHERE 'QUANTITY' = 'NULL';

-----INSIDE THE PRODUCTION SCHEMA THERE NO NULL
VALUES-------------------------------------

--------FINDING NULL IN SALES SCHEMA OF CUSTOMERS
TABLE -----------------

SELECT count(*)
FROM GK_BIKESTORE1.SALES.CUSTOMERS
WHERE 'CUSTOMER_ID' = 'NULL';
```

```
SELECT count(*)
FROM GK_BIKESTORE1.SALES.CUSTOMERS
WHERE 'FIRST_NAME' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.CUSTOMERS
WHERE 'LAST_NAME' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.CUSTOMERS
WHERE 'PHONE' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.CUSTOMERS
WHERE 'EMAIL' = 'NULL';
```

```
SELECT count(*)
FROM GK_BIKESTORE1.SALES.CUSTOMERS
WHERE 'STREET' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.CUSTOMERS
WHERE 'CITY' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.CUSTOMERS
WHERE 'STATE' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.CUSTOMERS
WHERE 'ZIP_CODE' = 'NULL';
```

----------CHECKING NULL IN ORDERS TABLE ----------

```
SELECT count(*)
FROM GK_BIKESTORE1.SALES.ORDERS
WHERE 'ORDER_ID' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.ORDERS
WHERE 'CUSTOMER_ID' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.ORDERS
WHERE 'ORDER_STATUS' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.ORDERS
WHERE 'ORDER_DATE' = 'NULL';
```

```
SELECT count(*)
FROM GK_BIKESTORE1.SALES.ORDERS
WHERE 'REQUIRED_DATE' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.ORDERS
WHERE SHIPPED_DATE = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.ORDERS
WHERE 'STORE_ID' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.ORDERS
WHERE 'STAFF_ID' = 'NULL';
```

--------- IN THE ORDER TABLE 170 NULL VALUES IN SHIPPED_DATE COLUMN----------

----------CHECKING NULL IN ORDER_ITEMS ----------

```
SELECT count(*)
FROM GK_BIKESTORE1.SALES.ORDER_ITEMS
WHERE 'STAFF_ID' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.ORDER_ITEMS
WHERE 'ITEM_ID' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.ORDER_ITEMS
WHERE 'PRODUCT_ID' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.ORDER_ITEMS
WHERE 'QUANTITY' = 'NULL';
```

```
SELECT count(*)
FROM GK_BIKESTORE1.SALES.ORDER_ITEMS
WHERE 'LIST_PRICE' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.ORDER_ITEMS
WHERE 'DISCOUNT' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.ORDER_ITEMS
WHERE 'TOTAL_PRICE' = 'NULL';
```

----------NO NULL VALUES IN ORDER_ITEMS----------

----------CHECKING NULL IN STAFFS ----------

```
SELECT count(*)
FROM GK_BIKESTORE1.SALES.STAFFS
WHERE 'STAFF_ID' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.STAFFS
WHERE 'FIRST_NAME' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.STAFFS
WHERE 'LAST_NAME' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.STAFFS
WHERE 'EMAIL' = 'NULL';
```

```
SELECT count(*)
FROM GK_BIKESTORE1.SALES.STAFFS
WHERE 'PHONE' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.STAFFS
WHERE 'ACTIVE' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.STAFFS
WHERE 'STORE_ID' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.STAFFS
WHERE 'MANAGER_ID' = 'NULL';
```

-------- NO NULL VALUES INSIDE THE STAFFS----------

----------CHECKING NULL IN STORES ----------

```
SELECT count(*)
FROM GK_BIKESTORE1.SALES.STORES
WHERE 'STORE_ID' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.STORES
WHERE 'STORE_NAME' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.STORES
WHERE 'PHONE' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.STORES
WHERE 'EMAIL' = 'NULL';
```

```
SELECT count(*)
FROM GK_BIKESTORE1.SALES.STORES
WHERE 'STREET' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.STORES
WHERE 'CITY' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.STORES
WHERE 'STATE' = 'NULL';

SELECT count(*)
FROM GK_BIKESTORE1.SALES.STORES
WHERE 'ZIP_CODE' = 'NULL';
```

----THERE IS NO NULL VALUES IN THE STORES TABLES-----

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

```
SELECT
customer_id,first_name,last_name,phone,email,street,city,state,zip_code,COUNT(*)
FROM GK_BIKESTORE1.SALES.CUSTOMERS
GROUP BY 1,2,3,4,5,6,7,8,9
HAVING COUNT(*) > 1;
```

```
SELECT
order_id,customer_id,order_status,order_date,required_date,shipped_date,store_id,staff_id,count(*)
from GK_BIKESTORE1.SALES.ORDERS
GROUP BY 1,2,3,4,5,6,7,8
HAVING COUNT(*) >1;
```

```
SELECT
ORDER_ID,ITEM_ID,PRODUCT_ID,QUANTITY,LIST_PRICE,
DISCOUNT,TOTAL_PRICE,COUNT(*)
FROM GK_BIKESTORE1.SALES.ORDER_ITEMS
GROUP BY 1,2,3,4,5,6,7
HAVING COUNT(*) >1;

SELECT
STAFF_ID,FIRST_NAME,LAST_NAME,EMAIL,PHONE,ACTI
VE,STORE_ID,MANAGER_ID,COUNT(*)
FROM  GK_BIKESTORE1.SALES.STAFFS
GROUP BY 1,2,3,4,5,6,7,8
HAVING COUNT(*) > 1;

SELECT
STORE_ID,STORE_NAME,PHONE,EMAIL,STREET,CITY,STA
TE,ZIP_CODE,COUNT(*)
FROM GK_BIKESTORE1.SALES.STORES
GROUP BY 1,2,3,4,5,6,7,8
HAVING COUNT(*) >1 ;
```

----------THERE IS NO IDENTICAL ROW IN THE SALES SCHEMA----------

```
SELECT BRAND_ID,BRAND_NAME,COUNT(*)
FROM GK_BIKESTORE1.PRODUCTION.BRANDS
GROUP BY 1,2
HAVING COUNT(*) >1;

SELECT CATEGORY_ID,CATEGORY_NAME,COUNT(*)
FROM GK_BIKESTORE1.PRODUCTION.CATEGORIES
GROUP BY 1,2
HAVING COUNT(*) > 1;

SELECT PRODUCT_ID,PRODUCT_NAME,BRAND_ID,CATEGORY_ID,MODEL_YEAR,LIST_PRICE,COUNT(*)
FROM GK_BIKESTORE1.PRODUCTION.PRODUCTS
GROUP BY 1,2,3,4,5,6
HAVING COUNT(*) > 1;
```

```
SELECT STORE_ID,PRODUCT_ID,QUANTITY,COUNT(*)
FROM  GK_BIKESTORE1.PRODUCTION.STOCKS
GROUP BY 1,2,3
HAVING COUNT(*) > 1;
```

--------THERE IS NO DUPLICATE IN THE PRODUCTION SCHEMA----------

3.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)

----FINDING UNIQUE TABLES IN SALES SCHEMA----------

desc schema GK_BIKESTORE1.SALES;

---THERE IS FIVE UNIQUE TABLES UNDER SALES SCHEMA---------

	created_on	name	kind
1	2023-09-01 05:23:21.508 -0700	CUSTOMERS	TABLE
2	2023-09-01 05:32:36.241 -0700	ORDERS	TABLE
3	2023-09-01 02:01:14.011 -0700	ORDER_ITEMS	TABLE
4	2023-09-01 02:00:03.371 -0700	STAFFS	TABLE
5	2023-09-01 05:43:02.705 -0700	STORES	TABLE

select count(*) from GK_BIKESTORE1.SALES.CUSTOMERS;

// THERE ARE TOTAL 1445 RECORDS IN CUTOMERS TABLE---

	COUNT(*)
1	1,445

select count(*) from GK_BIKESTORE1.SALES.ORDERS;

// THERE ARE TOTAL 1615 RECORDS IN ORDERS TABLE---

	COUNT(*)
1	1,615

select count(*) from GK_BIKESTORE1.SALES.ORDER_ITEMS;

// THERE ARE TOTAL 4722 RECORDS IN ORDER_ITEMS TABLE ---

	COUNT(*)
1	4,722

select count(*) from GK_BIKESTORE1.SALES.STAFFS;

// THERE ARE TOTAL 10 RECORDS IN STAFFS TABLE----------

	COUNT(*)
1	10

select count(*) from GK_BIKESTORE1.SALES.STORES;

// THERE IS ONLY 3 RECORDS IN STORES TABLE------

	COUNT(*)
1	3

----FINDING UNIQUE TABLES IN PRODUCTION SCHEMA-----------

desc schema GK_BIKESTORE1.PRODUCTION;

//-------THERE IS FOUR UNIQUE COULMN IN PRODUCTION SCHEMA----------

	created_on	name	kind
1	2023-09-01 02:01:29.694 -0700	BRANDS	TABLE
2	2023-09-01 02:01:20.744 -0700	CATEGORIES	TABLE
3	2023-09-01 02:01:20.312 -0700	PRODUCTS	TABLE
4	2023-09-01 02:01:27.167 -0700	STOCKS	TABLE

select count(*) from GK_BIKESTORE1.PRODUCTION.BRANDS;

//THERE ARE TOTAL 9 RECORDS IN BRANDS TABLE—

	COUNT(*)
1	9

select count(*) from GK_BIKESTORE1.PRODUCTION.CATEGORIES;

//THERE ARE TOTAL 7 RECORDS IN CATEGORIES TABLE—

select count(*) from GK_BIKESTORE1.PRODUCTION.STOCKS;

//THERE ARE 939 RECORDS IN STOCKS TABLE---

	COUNT(*)
1	939

select count(*) from GK_BIKESTORE1.PRODUCTION.PRODUCTS;

//THERE ARE 321 RECORDS IN PRODUCTS TABLE—

	COUNT(*)
1	321

4.How many total serving customer BikeStore has ?

```
select count(distinct customer_id) as
total_SERV_CUSTOMERS
from GK_BIKESTORE1.SALES.ORDERS;
```

	TOTAL_SERV_CUSTOMERS
1	1,445

HENCE TOTAL SERVING CUSTOMER IS 1445;

5.How many total orders are there ?

```
select count(ORDER_ID) as total_orders
from  GK_BIKESTORE1.SALES.ORDERS;
```

	TOTAL_ORDERS
1	1,615

HENCE TOTAL ORDERS IS 1615;

6. Which store has the highest number of sales ?

```
select O1.STORE_ID AS STORE_ID,
SUM(OI2.TOTAL_PRICE) AS TOTAL_SALES
from GK_BIKESTORE1.SALES.ORDERS O1,
GK_BIKESTORE1.SALES.ORDER_ITEMS OI2
where O1.ORDER_ID = OI2.ORDER_ID
GROUP BY O1.STORE_ID
ORDER BY TOTAL_SALES ASC;
```

	STORE_ID	TOTAL_SALES
1	3	962,608
2	1	1,790,160
3	2	5,826,286

---HENCE STORE_ID 2 has the highest sales I.E. 5,826,286

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

```
select customer_id,count(order_id) as NO_OF_ORDERS
FROM GK_BIKESTORE1.SALES.ORDERS
GROUP BY 1
ORDER BY 2 DESC
LIMIT 10;
```

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

9. Which are the TOP 3 selling product ?

```
select P1.product_id, P1.product_name,
SUM(OI2.TOTAL_PRICE) AS TOTAL_SALES
from GK_BIKESTORE1.PRODUCTION.PRODUCTS P1,
GK_BIKESTORE1.SALES.ORDER_ITEMS OI2 where
P1.product_id = OI2.product_id
group by P1.product_id, P1.product_name
order by TOTAL_SALES desc
limit 3;
```

	PRODUCT_ID	PRODUCT_NAME	TOTAL_SALES
1	7	"Trek Slash 8 27.5 - 2016"	616,000
2	9	"Trek Conduit+ - 2016"	435,000
3	4	"Trek Fuel EX 8 29 - 2016"	414,700

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

ALTER TABLE GK_BIKESTORE1.SALES.ORDER_ITEMS ADD COLUMN TOTAL_PRICE INT;

	ORDER_ID	ITEM_ID	PRODUCT_ID	QUANTITY	LIST_PRICE	DISCOUNT	TOTAL_PRICE
1	1	1	20	1	600	0	600
2	1	2	8	2	1,800	0	3,600
3	1	3	10	2	1,549	0	3,098
4	1	4	16	2	600	0	1,200
5	1	5	4	1	2,900	0	2,900
6	2	1	20	1	600	0	600
7	2	2	16	2	600	0	1,200
8	3	1	3	1	1,000	0	1,000
9	3	2	20	1	600	0	600
10	4	1	2	2	750	0	1,500
11	5	1	10	2	1,549	0	3,098
12	5	2	17	1	429	0	429
13	5	3	26	1	600	0	600

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

UPDATE GK_BIKESTORE1.SALES.ORDER_ITEMS SET TOTAL_PRICE = QUANTITY * LIST_PRICE;

	ORDER_ID	ITEM_ID	PRODUCT_ID	QUANTITY	LIST_PRICE	DISCOUNT	TOTAL_PRICE
1	1	1	20	1	600	0	600
2	1	2	8	2	1,800	0	3,600
3	1	3	10	2	1,549	0	3,098
4	1	4	16	2	600	0	1,200
5	1	5	4	1	2,900	0	2,900
6	2	1	20	1	600	0	600
7	2	2	16	2	600	0	1,200
8	3	1	3	1	1,000	0	1,000
9	3	2	20	1	600	0	600
10	4	1	2	2	750	0	1,500
11	5	1	10	2	1,549	0	3,098
12	5	2	17	1	429	0	429
13	5	3	26	1	600	0	600

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

select sum(total_price) as Total_price from GK_BIKESTORE1.SALES.ORDER_ITEMS;

	TOTAL_PRICE
1	8,579,054

>>>>HENCE TOTAL PRICE PAID TO ALL SALES.ORDER_ITEM IS 8,579,054;