Let's make a mini-project using the Snowflake Data Platform [Analytics in the Cloud].

I used the TPCH_SF1 schema in the SNOWFLAKE_SAMPLE_DATA database along with the COMPUTE_WH warehouse on the Snowflake Platform.

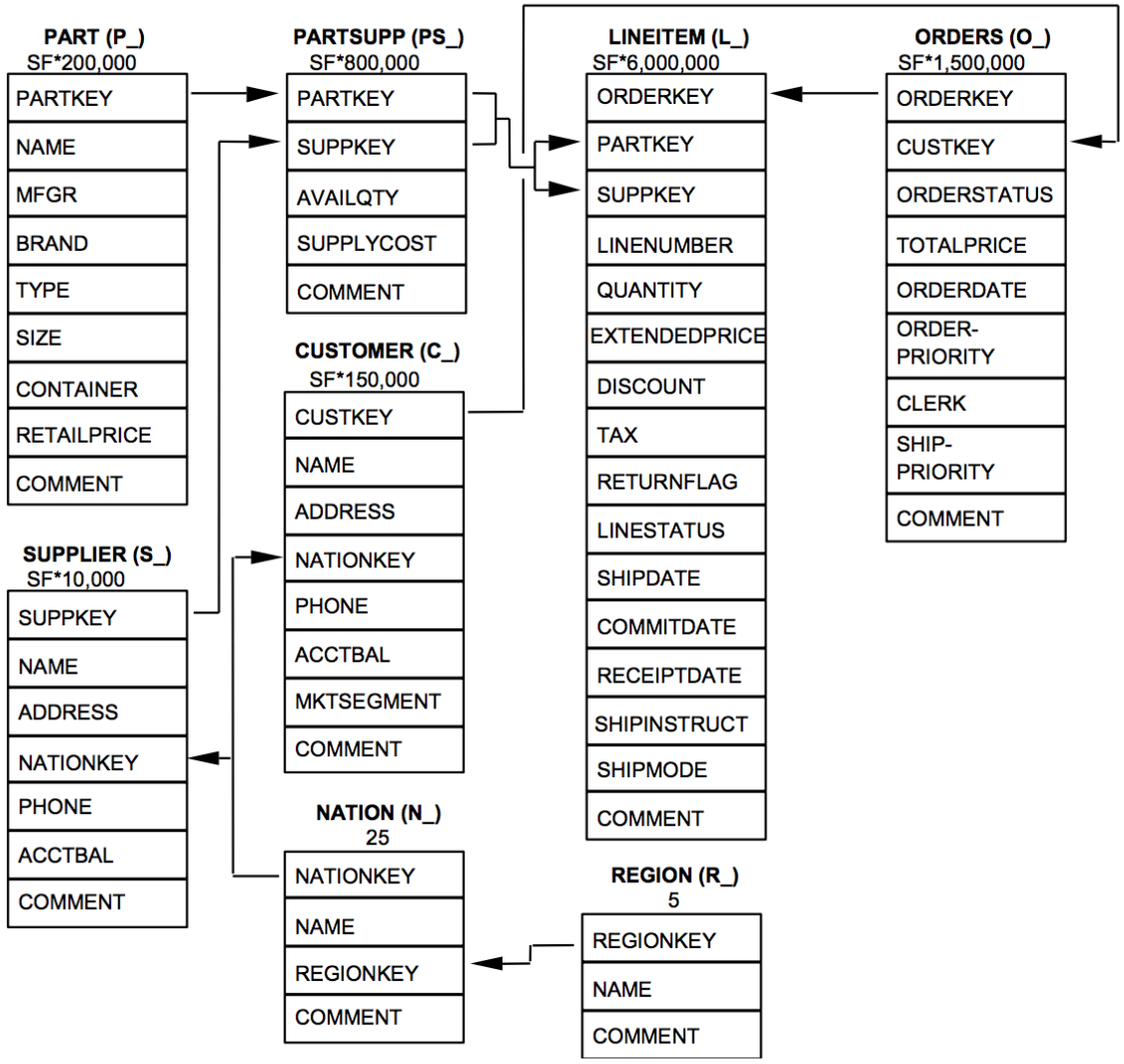
```
USE DATABASE SNOWFLAKE_SAMPLE_DATA;
USE SCHEMA TPCH_SF1;
 show tables;
```

Results Chart

created_on	name	database_name	schema_name	kind	comment
2021-11-09 20:08:26.847 -0800	CUSTOMER	SNOWFLAKE_SAMPLE_DATA	TPCH_SF1	TABLE	Customer data
2021-11-09 20:08:27.147 -0800	LINEITEM	SNOWFLAKE_SAMPLE_DATA	TPCH_SF1	TABLE	Lineitem data a
2021-11-09 20:08:27.115 -0800	NATION	SNOWFLAKE_SAMPLE_DATA	TPCH_SF1	TABLE	Nation data as
2021-11-09 20:08:27.095 -0800	ORDERS	SNOWFLAKE_SAMPLE_DATA	TPCH_SF1	TABLE	Orders data as
2021-11-09 20:08:27.095 -0800	PART	SNOWFLAKE_SAMPLE_DATA	TPCH_SF1	TABLE	Part data as de
2021-11-09 20:08:27.113 -0800	PARTSUPP	SNOWFLAKE_SAMPLE_DATA	TPCH_SF1	TABLE	Partsupp data
2021-11-09 20:08:27.967 -0800	REGION	SNOWFLAKE_SAMPLE_DATA	TPCH_SF1	TABLE	Region data as
2021-11-09 20:08:27.987 -0800	SUPPLIER	SNOWFLAKE_SAMPLE_DATA	TPCH_SF1	TABLE	Supplier data a

ER Model for TPCH_SF1 schema :

Figure 2: The TPC-H Schema



[**Source:** https://docs.snowflake.com/en/user-guide/sample-data-tpch]

Checking if the default warehouse is in the current session or not :

```
86    SELECT CURRENT_WAREHOUSE();
```

Results Chart

	CURRENT_WAREHOUSE()
1	COMPUTE_WH

SQL QUERIES:

1) Find the total sales for each customer -

```sql
SELECT
    C_CUSTKEY,
    C_NAME,
    SUM(L_EXTENDEDPRICE * (1 - L_DISCOUNT) * (1 + L_TAX)) AS TOTAL_SALES
FROM
    CUSTOMER
    JOIN ORDERS ON C_CUSTKEY = O_CUSTKEY
    JOIN LINEITEM ON O_ORDERKEY = L_ORDERKEY
GROUP BY
    C_CUSTKEY, C_NAME
ORDER BY TOTAL_SALES DESC;
```

Results | Chart

C_CUSTKEY	C_NAME	...	TOTAL_SALES
143500	Customer#000143500		7012698.003578
95257	Customer#000095257		6563512.766573
87115	Customer#000087115		6457527.640114
131113	Customer#000131113		6311430.273259
103834	Customer#000103834		6306525.501132
134380	Customer#000134380		6291611.633353

Explanation:

- This query performs joins on the CUSTOMER, ORDERS, and LINE ITEM tables and calculates the total sales for each customer by multiplying the extended price by (1 - discount) and (1 + tax), then summing them up.
- The GROUP BY clause groups the results by customer.
- The ORDER BY clause displays the results in descending order of Total Sales.

Potential Insights:

- Identify high-value customers based on their total sales.
- Analyze customer segments for targeted marketing in the future.

2) Find the top-selling products:

```sql
SELECT
    P_NAME,
    SUM(L_QUANTITY) AS TOTAL_QUANTITY_SOLD
FROM
    PART
    JOIN LINEITEM ON P_PARTKEY = L_PARTKEY
GROUP BY
    P_NAME
ORDER BY
    TOTAL_QUANTITY_SOLD DESC
LIMIT 5;
```

Results Chart

P_NAME	TOTAL_QUANTITY_SOLD
blush burlywood red blanched olive	1720.00
azure lime rosy peru powder	1677.00
dodger sienna orange khaki lace	1642.00
yellow snow bisque dodger drab	1562.00
rose red purple salmon sandy	1553.00

Explanation:

- This query joins the PART and LINEITEM tables.
- It calculates the total quantity sold for each product.
- The results are in descending order, and only the top 5 are selected using LIMIT.

Potential Insights:

- Identify the best-selling products.
- Manage inventory based on top-selling items.

3) Find the average order value for each region.

```sql
SELECT
    R_NAME AS REGION,
    AVG(O_TOTALPRICE) AS AVG_ORDER_VALUE
FROM
    REGION
    JOIN NATION ON R_REGIONKEY = N_REGIONKEY
    JOIN CUSTOMER ON N_NATIONKEY = C_NATIONKEY
    JOIN ORDERS ON C_CUSTKEY = O_CUSTKEY
GROUP BY
    R_NAME;
```

Results Chart

REGION	AVG_ORDER_VALUE
ASIA	151167.94274064
AMERICA	151476.05759625
EUROPE	150990.37034255
AFRICA	151274.68745935
MIDDLE EAST	151192.10578037

Explanation:

- This query joins the REGION, NATION, CUSTOMER, and ORDERS tables.
- It calculates the average order value for each region.
- The GROUP BY clause groups the results by region.

Potential Insights:

- Understand the average transaction value in different regions.
- Identify regions with high or low average order values for targeted marketing.

4) Finding How many orders were on time and delayed:

```sql
SELECT
    L_LINESTATUS,
    COUNT(*) AS ORDER_COUNT
FROM
    LINEITEM
    JOIN ORDERS ON L_ORDERKEY = O_ORDERKEY
GROUP BY
    L_LINESTATUS;
```

Results Chart

L_LINESTATUS	ORDER_COUNT
O	3004998
F	2996217

Explanation:

- It counts the number of orders based on their line status (shipped on time or delayed).
- The GROUP BY clause groups the results by line status.
- This query joins the LINEITEM and ORDERS tables.

Potential Insights:

- Assessing the efficiency of the shipping process by comparison.
- Identify trends in delayed shipments for improvement.

5) Find top 5 suppliers with highest supply cost:

```sql
SELECT
    S_NAME,
    SUM(PS_SUPPLYCOST) AS TOTAL_SUPPLY_COST
FROM
    SUPPLIER
    JOIN PARTSUPP ON S_SUPPKEY = PS_SUPPKEY
GROUP BY
    S_NAME
ORDER BY
    TOTAL_SUPPLY_COST DESC
LIMIT 5;
```

Results Chart

S_NAME	TOTAL_SUPPLY_COST
Supplier#000006248	50138.72
Supplier#000005464	49091.31
Supplier#000000537	48286.46
Supplier#000005182	48156.99
Supplier#000001528	47957.16

Explanation:

- This query joins the SUPPLIER and PARTSUPP tables.
- It calculates the total supply cost for each supplier by summing up the supply costs of all their parts.
- The results are ordered in descending order, and only the top 5 suppliers are selected using LIMIT.

Potential Insights:

- Identify suppliers with the highest supply costs to negotiate better deals with them or explore alternative suppliers to reduce dependency on 1 supplier.

Link to Snowflake Worksheet Results:

https://app.snowflake.com/yeocnri/lh82281/w1sOx4rsPitf/query