CREATE OR REPLACE DATABASE SALES\_DB;

USE DATABASE SALES\_DB;

CREATE OR REPLACE SCHEMA SALES\_SCHEMA;

USE SCHEMA SALES\_SCHEMA;

CREATE OR REPLACE STORAGE INTEGRATION gcs\_integration

TYPE = EXTERNAL\_STAGE

STORAGE\_PROVIDER = 'GCS'

ENABLED = TRUE

STORAGE\_ALLOWED\_LOCATIONS = ('gcs://vinaydw/');

CREATE OR REPLACE STAGE SALES\_DB.SALES\_SCHEMA.SALES\_STAGE

STORAGE\_INTEGRATION = gcs\_integration

URL = 'gcs://vinaydw/';

LIST @SALES\_DB.SALES\_SCHEMA.SALES\_STAGE;

--CREATING STAGING TABLES

CREATE OR REPLACE TABLE SALES\_DB.SALES\_SCHEMA.staging\_customers (

customer\_id STRING,

gender STRING,

age INT,

city STRING,

state STRING,

loyalty\_tier STRING

);

CREATE OR REPLACE TABLE SALES\_DB.SALES\_SCHEMA.staging\_products (

product\_id STRING,

product\_category STRING,

price\_per\_unit DECIMAL(10,2),

product\_name STRING,

brand STRING

);

CREATE OR REPLACE TABLE SALES\_DB.SALES\_SCHEMA.staging\_sales (

sale\_id STRING,

customer\_id STRING,

product\_id STRING,

quantity INT,

price\_per\_unit DECIMAL(10,2),

total\_amount DECIMAL(10,2),

sale\_date STRING -- Storing as STRING first, then converting to DATE

);

--LOADING DATA INTO STAGING TABLES

CREATE OR REPLACE FILE FORMAT SALES\_DB.SALES\_SCHEMA.MY\_CSV\_FORMAT

TYPE = 'CSV'

SKIP\_HEADER = 1

FIELD\_OPTIONALLY\_ENCLOSED\_BY = '"'

DATE\_FORMAT = 'YYYY-MM-DD';

-- Load Customers

COPY INTO SALES\_DB.SALES\_SCHEMA.staging\_customers

FROM @SALES\_DB.SALES\_SCHEMA.SALES\_STAGE/customers\_synthetic.csv

FILE\_FORMAT = (FORMAT\_NAME = 'SALES\_DB.SALES\_SCHEMA.MY\_CSV\_FORMAT')

ON\_ERROR = 'CONTINUE';

-- Load Products

COPY INTO SALES\_DB.SALES\_SCHEMA.staging\_products

FROM @SALES\_DB.SALES\_SCHEMA.SALES\_STAGE/products\_synthetic.csv

FILE\_FORMAT = (FORMAT\_NAME = 'SALES\_DB.SALES\_SCHEMA.MY\_CSV\_FORMAT')

ON\_ERROR = 'CONTINUE';

-- Load Sales (Sale Date as STRING)

COPY INTO SALES\_DB.SALES\_SCHEMA.staging\_sales

FROM @SALES\_DB.SALES\_SCHEMA.SALES\_STAGE/sales\_synthetic.csv

FILE\_FORMAT = (FORMAT\_NAME = 'SALES\_DB.SALES\_SCHEMA.MY\_CSV\_FORMAT')

ON\_ERROR = 'CONTINUE';

--CONVERT SALE DATE FROM STRING TO DATE

ALTER TABLE SALES\_DB.SALES\_SCHEMA.staging\_sales ADD COLUMN sale\_date\_converted DATE;

UPDATE SALES\_DB.SALES\_SCHEMA.staging\_sales

SET sale\_date\_converted = TRY\_TO\_DATE(sale\_date, 'YYYY-MM-DD');

ALTER TABLE SALES\_DB.SALES\_SCHEMA.staging\_sales DROP COLUMN sale\_date;

ALTER TABLE SALES\_DB.SALES\_SCHEMA.staging\_sales RENAME COLUMN sale\_date\_converted TO sale\_date;

--VERIFYING LOADED DATA

SELECT COUNT(\*) FROM SALES\_DB.SALES\_SCHEMA.staging\_customers;

SELECT COUNT(\*) FROM SALES\_DB.SALES\_SCHEMA.staging\_products;

SELECT COUNT(\*) FROM SALES\_DB.SALES\_SCHEMA.staging\_sales;

SELECT DISTINCT sale\_date FROM SALES\_DB.SALES\_SCHEMA.staging\_sales LIMIT 10;

--CREATING NORMALIZED TABLES

CREATE OR REPLACE TABLE SALES\_DB.SALES\_SCHEMA.customers (

customer\_id STRING PRIMARY KEY,

gender STRING,

age INT,

city STRING,

state STRING,

loyalty\_tier STRING

);

CREATE OR REPLACE TABLE SALES\_DB.SALES\_SCHEMA.products (

product\_id STRING PRIMARY KEY,

product\_category STRING,

price\_per\_unit DECIMAL(10,2),

product\_name STRING,

brand STRING

);

CREATE OR REPLACE TABLE SALES\_DB.SALES\_SCHEMA.sales (

sale\_id STRING PRIMARY KEY,

customer\_id STRING REFERENCES SALES\_DB.SALES\_SCHEMA.customers(customer\_id),

product\_id STRING REFERENCES SALES\_DB.SALES\_SCHEMA.products(product\_id),

quantity INT,

total\_amount DECIMAL(10,2),

sale\_date DATE

);

--LOADING DATA INTO NORMALIZED TABLES

INSERT INTO SALES\_DB.SALES\_SCHEMA.customers

SELECT DISTINCT \* FROM SALES\_DB.SALES\_SCHEMA.staging\_customers;

INSERT INTO SALES\_DB.SALES\_SCHEMA.products

SELECT DISTINCT \* FROM SALES\_DB.SALES\_SCHEMA.staging\_products;

DESC TABLE SALES\_DB.SALES\_SCHEMA.staging\_sales;

DESC TABLE SALES\_DB.SALES\_SCHEMA.sales;

INSERT INTO SALES\_DB.SALES\_SCHEMA.sales (sale\_id, customer\_id, product\_id, quantity, total\_amount, sale\_date)

SELECT DISTINCT sale\_id, customer\_id, product\_id, quantity, total\_amount, sale\_date

FROM SALES\_DB.SALES\_SCHEMA.staging\_sales;

SELECT COUNT(\*) FROM SALES\_DB.SALES\_SCHEMA.sales;

SELECT \* FROM SALES\_DB.SALES\_SCHEMA.sales LIMIT 10;

--Create Dimension Tables

CREATE OR REPLACE TABLE SALES\_DB.SALES\_SCHEMA.dim\_customers (

customer\_key INT AUTOINCREMENT PRIMARY KEY,

customer\_id STRING UNIQUE,

gender STRING,

age INT,

city STRING,

state STRING,

loyalty\_tier STRING

);

CREATE OR REPLACE TABLE SALES\_DB.SALES\_SCHEMA.dim\_products (

product\_key INT AUTOINCREMENT PRIMARY KEY,

product\_id STRING UNIQUE,

product\_category STRING,

price\_per\_unit DECIMAL(10,2),

product\_name STRING,

brand STRING

);

CREATE OR REPLACE TABLE SALES\_DB.SALES\_SCHEMA.dim\_date (

date\_key INT PRIMARY KEY,

sale\_date DATE,

year INT,

month INT,

day INT,

day\_of\_week STRING

);

--LOADING INTO dim-customer

INSERT INTO SALES\_DB.SALES\_SCHEMA.dim\_customers (customer\_id, gender, age, city, state, loyalty\_tier)

SELECT DISTINCT customer\_id, gender, age, city, state, loyalty\_tier

FROM SALES\_DB.SALES\_SCHEMA.customers;

--LOADING INTO dim-product

INSERT INTO SALES\_DB.SALES\_SCHEMA.dim\_products (product\_id, product\_category, price\_per\_unit, product\_name, brand)

SELECT DISTINCT product\_id, product\_category, price\_per\_unit, product\_name, brand

FROM SALES\_DB.SALES\_SCHEMA.products;

--LOADING INTO dim-date

INSERT INTO SALES\_DB.SALES\_SCHEMA.dim\_date (date\_key, sale\_date, year, month, day, day\_of\_week)

SELECT

YEAR(sale\_date) \* 10000 + MONTH(sale\_date) \* 100 + DAY(sale\_date) AS date\_key,

sale\_date,

YEAR(sale\_date),

MONTH(sale\_date),

DAY(sale\_date),

DAYNAME(sale\_date)

FROM SALES\_DB.SALES\_SCHEMA.sales

WHERE sale\_date IS NOT NULL

GROUP BY sale\_date;

--fact table

CREATE OR REPLACE TABLE SALES\_DB.SALES\_SCHEMA.fact\_sales (

sales\_key INT AUTOINCREMENT PRIMARY KEY,

customer\_key INT REFERENCES SALES\_DB.SALES\_SCHEMA.dim\_customers(customer\_key),

product\_key INT REFERENCES SALES\_DB.SALES\_SCHEMA.dim\_products(product\_key),

date\_key INT REFERENCES SALES\_DB.SALES\_SCHEMA.dim\_date(date\_key),

quantity INT,

total\_amount DECIMAL(10,2)

);

--Populating the fact Table

INSERT INTO SALES\_DB.SALES\_SCHEMA.fact\_sales (customer\_key, product\_key, date\_key, quantity, total\_amount)

SELECT

c.customer\_key,

p.product\_key,

d.date\_key,

s.quantity,

s.total\_amount

FROM SALES\_DB.SALES\_SCHEMA.sales AS s

JOIN SALES\_DB.SALES\_SCHEMA.dim\_customers c ON s.customer\_id = c.customer\_id

JOIN SALES\_DB.SALES\_SCHEMA.dim\_products p ON s.product\_id = p.product\_id

JOIN SALES\_DB.SALES\_SCHEMA.dim\_date d ON s.sale\_date = d.sale\_date;

--Analytical Queries

--Total Sales Revenue by Product Category

SELECT p.product\_category, SUM(f.total\_amount) AS total\_sales

FROM SALES\_DB.SALES\_SCHEMA.fact\_sales f

JOIN SALES\_DB.SALES\_SCHEMA.dim\_products p ON f.product\_key = p.product\_key

GROUP BY p.product\_category

ORDER BY total\_sales DESC;

--Top 5 Customers by Total Spend

SELECT c.customer\_id, c.city, c.state, SUM(f.total\_amount) AS total\_spent

FROM SALES\_DB.SALES\_SCHEMA.fact\_sales f

JOIN SALES\_DB.SALES\_SCHEMA.dim\_customers c ON f.customer\_key = c.customer\_key

GROUP BY c.customer\_id, c.city, c.state

ORDER BY total\_spent DESC

LIMIT 5;

--Monthly Sales Trend

SELECT d.year, d.month, SUM(f.total\_amount) AS monthly\_sales

FROM SALES\_DB.SALES\_SCHEMA.fact\_sales f

JOIN SALES\_DB.SALES\_SCHEMA.dim\_date d ON f.date\_key = d.date\_key

GROUP BY d.year, d.month

ORDER BY d.year, d.month;

--Best-Selling Products by Quantity Sold

SELECT p.product\_name, SUM(f.quantity) AS total\_sold

FROM SALES\_DB.SALES\_SCHEMA.fact\_sales f

JOIN SALES\_DB.SALES\_SCHEMA.dim\_products p ON f.product\_key = p.product\_key

GROUP BY p.product\_name

ORDER BY total\_sold DESC

LIMIT 5;

--Sales Performance by Day of the Week

SELECT d.day\_of\_week, SUM(f.total\_amount) AS total\_sales

FROM SALES\_DB.SALES\_SCHEMA.fact\_sales f

JOIN SALES\_DB.SALES\_SCHEMA.dim\_date d ON f.date\_key = d.date\_key

GROUP BY d.day\_of\_week

ORDER BY total\_sales DESC;