```
-- MELCHIZEDEK ACKAH-BLAY
-- 7 FEBRUARY 2024
-- WALMART DATASET PROJECT USING MYSQL WORKBENCH

CREATE DATABASE IF NOT EXISTS walmart;


-- CREATING THE TABLE I AM GOING TO WORK WITH

CREATE TABLE IF NOT EXISTS sales(
   invoice_id VARCHAR(30) NOT NULL PRIMARY KEY,
        branch VARCHAR(5) NOT NULL,
        city VARCHAR(30) NOT NULL,
        customer_type VARCHAR(30) NOT NULL,
        gender VARCHAR(15) NOT NULL,
        product_line VARCHAR(100) NOT NULL,
        unit_price DECIMAL(10, 2) NOT NULL,
        quantity INT NOT NULL,
        VAT float(6, 4) NOT NULL,
        total DECIMAL(12, 4) NOT NULL,
        date DATETIME NOT NULL,
        time TIME NOT NULL,
        payment_method VARCHAR(20) NOT NULL,
        cogs DECIMAL(10, 2),
        gross_margin_pct FLOAT(11, 9),
        gross_income DECIMAL(12, 4) NOT NULL,
        rating FLOAT(2.1)
);

-- RUNNING THE TABLE TO VERIFY THAT ALL REQUIRED COLUMNS ARE THERE
SELECT * FROM sales;

-- RE RUN AFTER IMPORTING DATA
SELECT * FROM sales;

-- ------------------------------ FEATURE ENGINEERING ------------------------------------------

-- TIME OF DAY --
SELECT time,
(CASE
   WHEN time BETWEEN "00:00:00" AND "12:00:00" THEN "Morning"
        WHEN time BETWEEN "12:01:00" AND "16:00:00" THEN "Afternoon"
        ELSE "Evening"
        END
```

```
) AS time_of_date
FROM sales;


--  INSERT THE TIME INTO THE SALES TABLE

ALTER TABLE sales ADD COLUMN time_of_day VARCHAR(20);


-- UPDATING QUERY

UPDATE sales
SET time_of_day = (
CASE
   WHEN time BETWEEN "00:00:00" AND "12:00:00" THEN "Morning"
         WHEN time BETWEEN "12:01:00" AND "16:00:00" THEN "Afternoon"
         ELSE "Evening"
         END
         );

SELECT * FROM sales;



-- INSERTING THE DAY AN INVOICE OCCURED BY CREATING A COLUMN FOR DAY NAME

SELECT date, DAYNAME(date)
FROM sales;

ALTER TABLE sales ADD COLUMN day_of_invoice VARCHAR(15);

UPDATE sales
SET day_of_invoice = DAYNAME(date);

-- RETRIEVING THE MONTH THE INVOICE OCCURED

SELECT date, MONTHNAME(date)
FROM sales;

ALTER TABLE sales ADD COLUMN month_of_invoice VARCHAR(15);

UPDATE sales
SET month_of_invoice = MONTHNAME(date);
```

```
-- ---------------------------------- EXPLORATORY DATA ANALYSIS
-----------------------------------------

-- 1. How many unique cities does the data have?
SELECT DISTINCT city
FROM sales;

-- 2. In which city is each branch?
SELECT DISTINCT city, branch
FROM sales;

-- How many unique product lines does the data have?
SELECT COUNT(DISTINCT product_line) AS 'unique products'
FROM sales;

--          -------- OR ---------
SELECT DISTINCT product_line
FROM sales;


-- What is the most common payment method?
SELECT payment_method, COUNT(payment_method) AS count
FROM sales
GROUP BY payment_method
ORDER BY count DESC;


-- What is the most selling product line
SELECT product_line, COUNT(product_line) AS count
FROM sales
GROUP BY product_line
ORDER BY count DESC;


-- What is the total revenue by month
SELECT month_of_invoice AS month, SUM(total) AS total_revenue
FROM sales
GROUP BY month_of_invoice
ORDER BY total_revenue DESC;


-- What month had the largest COGS
```

```
SELECT month_of_invoice AS month, SUM(cogs) AS cogs
FROM sales
GROUP BY month_of_invoice
ORDER BY cogs DESC;


-- What product line had the largest revenue
SELECT product_line AS product, SUM(total) AS total
FROM sales
GROUP BY product
ORDER BY total DESC;

-- What is the city with the largest revenue
SELECT branch, city, SUM(total) AS total
FROM sales
GROUP BY branch, city
ORDER BY total DESC;


-- What product line had the largest VAT
SELECT product_line, AVG(VAT) AS VAT
FROM sales
GROUP BY product_line
ORDER BY VAT DESC;

-- Fetch each product line and add a column to those product line showing "Good", "Bad".
-- Good if its greater than the average sales



-- Which branch sold more products than the average sold
SELECT branch,
(CASE
   WHEN SUM(total) > (SELECT AVG(total) FROM sales) THEN "sold MORE than average"
        ELSE "sold LESS than average"
        END
        ) AS Cases
FROM sales
GROUP BY branch;


-- What is the most common product line by gender
SELECT gender, product_line AS product, COUNT(product_line) AS product_quantity
FROM sales
```

```
GROUP BY gender, product
ORDER BY product_quantity DESC;


-- What is the average rating of each product line
SELECT product_line AS product, ROUND(AVG(rating), 2) AS rating
FROM sales
GROUP BY product
ORDER BY rating DESC;


-- ----------------------------------SALES-----------------------------------


-- Which of the customer types brings in the most revenue
SELECT customer_type, SUM(total) AS revenue
FROM sales
GROUP BY customer_type
ORDER BY revenue DESC;


-- Which city has the largest tax percent/VAT(Value Added Tax)
SELECT city, ROUND(AVG(VAT), 2) as VAT
FROM sales
GROUP BY city
ORDER BY VAT DESC;


--  ---------------------------------------- CUSTOMER ANALYSIS
-------------------------------------------------------------

-- How many unique customer types does the data have?
SELECT DISTINCT customer_type
FROM sales;

-- How many people in each customer class?
SELECT customer_type, COUNT(customer_type) AS customer
FROM sales
GROUP BY customer_type;


-- How many unique payment methods does the data have?
SELECT DISTINCT payment_method
FROM sales;
```

```
-- How many people use each payment method?
SELECT payment_method, COUNT(payment_method) as pay
FROM sales
GROUP BY payment_method;



-- Which customer type buys the most?
SELECT customer_type, SUM(total) AS expenditure
FROM sales
GROUP BY customer_type
ORDER BY customer_type DESC;


-- What is the gender of most of the customers?
SELECT gender, COUNT(gender) AS gender_count
FROM sales
GROUP BY gender
ORDER BY gender;

-- What is the gender distribution of each branch?
SELECT branch, COUNT(gender) as gender_count
FROM sales
WHERE branch = "A"
GROUP BY branch;

SELECT branch, COUNT(gender) as gender_count
FROM sales
WHERE branch = "B"
GROUP BY branch;

SELECT branch, COUNT(gender) as gender_count
FROM sales
WHERE branch = "C"
GROUP BY branch;


-- Which time of the day do customers give most ratings?
SELECT time_of_day, ROUND(AVG(rating), 2) as average_rating
FROM sales
GROUP BY time_of_day
ORDER BY average_rating DESC;
```

```
-- Which time of the day do customers give the most ratings per branch?
SELECT branch, time_of_day, ROUND(AVG(rating), 2) AS average_rating
FROM sales
GROUP BY branch, time_of_day
ORDER BY average_rating DESC;


-- Which day of the week has the best average ratings?
SELECT day_of_invoice, ROUND(AVG(rating),2) AS average_rating
FROM sales
GROUP BY day_of_invoice
ORDER BY average_rating DESC;

-- Which day of the week has the best average ratings per branch?
SELECT branch, ROUND(AVG(rating),2) AS average_rating
FROM sales
GROUP BY branch
ORDER BY average_rating DESC;
```