Walmart Sales Data Analysis

About

This project aims to explore the Walmart Sales data to understand top performing branches and products, sales trend of of different products, customer behaviour. The aims is to study how sales strategies can be improved and optimized. The dataset was obtained from the Kaggle Walmart Sales Forecasting Competition (https://www.kaggle.com/c/walmart-recruiting-store-sales-forecasting).

"In this recruiting competition, job-seekers are provided with historical sales data for 45 Walmart stores located in different regions. Each store contains many departments, and participants must project the sales for each department in each store. To add to the challenge, selected holiday markdown events are included in the dataset. These markdowns are known to affect sales, but it is challenging to predict which departments are affected and the extent of the impact." source (https://www.kaggle.com/c/walmart-recruiting-store-sales-forecasting)

Purposes Of The Project

The major aim of thie project is to gain insight into the sales data of Walmart to understand the different factors that affect sales of the different branches.

About Data

The dataset was obtained from the Kaggle Walmart Sales Forecasting Competition (https://www.kaggle.com/c/walmart-recruiting-store-sales-forecasting). This dataset contains sales transactions from a three different branches of Walmart, respectively located in Mandalay, Yangon and Naypyitaw. The data contains 17 columns and 1000 rows:

Column	Description	Data Type
invoice_id	Invoice of the sales made	VARCHAR(30)
branch	Branch at which sales were made	VARCHAR(5)
city	The location of the branch	VARCHAR(30)
customer_type	The type of the customer	VARCHAR(30)
gender	Gender of the customer making purchase	VARCHAR(10)
product_line	Product line of the product solf	VARCHAR(100)
unit_price	The price of each product	DECIMAL(10, 2)
quantity	The amount of the product sold	INT
VAT	The amount of tax on the purchase	FLOAT(6, 4)
total	The total cost of the purchase	DECIMAL(10, 2)
date	The date on which the purchase was made	DATE
time	The time at which the purchase was made	TIMESTAMP
payment_method	The total amount paid	DECIMAL(10, 2)
cogs	Cost Of Goods sold	DECIMAL(10, 2)

Column	Description	Data Type
gross_margin_percentage	Gross margin percentage	FLOAT(11, 9)
gross_income	Gross Income	DECIMAL(10, 2)
rating	Rating	FLOAT(2, 1)

Analysis List

1. Product Analysis

> *Conduct analysis on the data to understand the different product lines, the products lines performing best and the product lines that need to be improved.*

2. Sales Analysis

> *This analysis aims to answer the question of the sales trends of product. The result of this can help use measure the effectiveness of each sales strategy the business applies and what modificatoins are needed to gain more sales.*

3. Customer Analysis

> *This analysis aims to uncover the different customers segments, purchase trends and the profitability of each customer segment.*

Approach Used

1. **Data Wrangling:** This is the first step where inspection of data is done to make sure **NULL** values and missing values are detected and data replacement methods are used to replace, missing or **NULL** values.

> 1. *Build a database*
> 2. *Create table and insert the data.*
> 3. *Select columns with null values in them. There are no null values in our database as in creating the tables, we set **NOT NULL** for each field, hence null values are filtered out.*

2. **Feature Engineering:** This will help use generate some new columns from existing ones.

1. *Add a new column named `time_of_day` to give insight of sales in the Morning, Afternoon and Evening. This will help answer the question on which part of the day most sales are made.*

2. *Add a new column named `day_name` that contains the extracted days of the week on which the given transaction took place (Mon, Tue, Wed, Thur, Fri). This will help answer the question on which week of the day each branch is busiest.*

3. *Add a new column named `month_name` that contains the extracted months of the year on which the given transaction took place (Jan, Feb, Mar). Help determine which month of the year has the most sales and profit.*

2. **Exploratory Data Analysis (EDA):** Exploratory data analysis is done to answer the listed questions and aims of this project.

3. **Conclusion:**

Business Questions To Answer

Generic Question

1. How many unique cities does the data have?
2. In which city is each branch?

Product

1. How many unique product lines does the data have?
2. What is the most common payment method?
3. What is the most selling product line?
4. What is the total revenue by month?
5. What month had the largest COGS?
6. What product line had the largest revenue?
7. What is the city with the largest revenue?
8. What product line had the largest VAT?
9. Fetch each product line and add a column to those product line showing "Good", "Bad". Good if its greater than average sales
10. Which branch sold more products than average product sold?
11. What is the most common product line by gender?
12. What is the average rating of each product line?

Sales

1. Number of sales made in each time of the day per weekday
2. Which of the customer types brings the most revenue?
3. Which city has the largest tax percent/ VAT (**Value Added Tax**)?
4. Which customer type pays the most in VAT?

Customer

1. How many unique customer types does the data have?
2. How many unique payment methods does the data have?
3. What is the most common customer type?
4. Which customer type buys the most?
5. What is the gender of most of the customers?
6. What is the gender distribution per branch?
7. Which time of the day do customers give most ratings?
8. Which time of the day do customers give most ratings per branch?
9. Which day fo the week has the best avg ratings?
10. Which day of the week has the best average ratings per branch?

```
create database SalesData_walmart;
Create table SalesData_walmart.Sales (
invoice_id Varchar(30) not null primary key,
branch varchar(50) not null,
city  varchar (30) not null,
custoemer_type   varchar(30) not null,
 gender varchar(10) 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(10,4) not null,
 date datetime not null,
 time time not null,
 payment_method varchar (15) not null ,
 cogs Decimal(10,2)  not null,
 gross_margin_percentage float (11,9),
 gross_income decimal( 12,4) not null,
 ratings float (2,1)
 );

-- ---------------------------------------------------------------------
---------------------------------------------------------------
-- ------------------------------------------------FEATURE ENGINEERING-----
---------------------------------------------------------------
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_the_day
from sales

Alter table sales Add column time_of_the_day varchar(20);
UPDATE sales
SET time_of_the_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
)


-- ---day_name
SELECT date, Dayname(date) AS day_name
FROM sales;

ALTER TABLE sales ADD COLUMN day_nanme VARCHAR(20) ;

 UPDATE sales
 SET day_name = DAYNAME(date) ;

 -- ---month_name
```

```
SELECT  date, monthname(date)
FROM sales

 ALTER TABLE sales ADD COLUMN month_name VARCHAR(20);

UPDATE sales
SET month_name =  monthname (date) ;

-- ---------------------------------------------------------------------
-- ---------------------------- Generic ------------------------------

 -- How many unique cities does the data have?
 Select Distinct city from sales

 -- In which city is each branch?
 Select Distinct city, branch
 from sales;

 -- ---------------------------------------------------------------------
-- ---------------------------- Product ------------------------------

-- How many unique product lines does the data have?
SELECT distinct product_line
from sales


-- What is the most selling product line
SELECT product_line , sum(quantity) as qty
from sales
group by product_line
ORDER BY qty DESC;


-- What is the total revenue by month?--
SELECT month_name as month , sum(total) as total_revenue
from sales
group by month_name
order by total_revenue;

 -- What month had the largest COGS?
 select month_name as month, sum(cogs) largest_cogs
 from sales
 group by month_name
 order by largest_cogs ;


-- What product line had the largest revenue?
Select product_line, sum(total) as largest_revenue
from sales
group by product_line
order by largest_revenue
```

```sql
-- What is the city with the largest revenue?

Select city , Sum(total) as total_revenue
from sales
group by city
order by total_revenue;

-- What product line had the largest VAT?
Select product_line, Sum(vat) as total_vat
from sales
group by product_line
order by total_vat desc;

      select product_line,
      AVG(vat) as avg_tax
FROM sales
GROUP BY product_line
ORDER BY avg_tax DESC;


-- Which branch sold more products than average product sold?
Select  branch,
sum(quantity) as qty
from sales
group by  branch
Having sum(quantity) >  (select avg(quantity) from sales)

-- What is the most common product line by gender
Select product_line,count(gender) as total_count
from sales
group by product_line, gender
order by  total_count

-- What is the average rating of each product line
Select round( Avg (ratings)), product_line
from sales
group by product_line


-- ------------------------- Customers -------------------------------
-- -------------------------------------------------------------------



-- How many unique customer types does the data have?
select distinct custoemer_type
from sales

-- How many unique payment methods does the data have?
Select distinct payment_method
from sales

-- What is the most common customer type?
Select custoemer_type,count(custoemer_type)
```

```
from sales
group by custoemer_type

-- Which customer type buys the most?
Select custoemer_type, count(custoemer_type)
from sales
group by custoemer_type


-- What is the gender of most of the customers?

SELECT gender, COUNT(gender) FROM sales GROUP BY gender

-- Number of sales made in each time of the day per weekday
 Select time_of_the_day,
 count(*) as cnt
 from sales
 Where day_name = 'sunday'
 group by time_of_the_day




-- Which of the customer types brings the most revenue?
Select custoemer_type , Sum(total) as total_revenue
from sales
group by custoemer_type
order by total_revenue DESC;

-- Which city has the largest tax percent/ VAT (Value Added Tax)?
Select city, Avg(VAt) as VAT
from sales
group by city
order by vat desc;



-- Which customer type pays the most in VAT?
Select custoemer_type , Avg(VAT) as VAT
from sales
group by custoemer_type
order by VAT desc;

### Customer

 -- How many unique customer types does the data have?
 Select distinct custoemer_type
 from sales ;



 -- How many unique payment methods does the data have?
 Select Distinct payment_method
 from sales;
```

```
-- What is the most common customer type?


Select custoemer_type ,
count(*) as no_of_customer
from sales
group by custoemer_type
order by no_of_customer desc;


 -- Which customer type buys the most?
 Select custoemer_type,
 Sum(quantity) as total_buys
 From sales
 group by custoemer_type
 order by total_buys desc;

-- What is the gender of most of the customers?
Select gender ,
count(gender) as no_of_gender
from sales
group by gender
order by no_of_gender desc;
-- What is the gender distribution per branch?
 Select gender , branch,
count(*) as no_of_gender
from sales
group by branch,gender

-- Which time of the day do customers give most ratings?
Select time_of_the_day,
COUNT(ratings) as number_of_ratings
from sales
group by time_of_the_day
order by number_of_ratings desc;
 -- Which time of the day do customers give most ratings per branch?
  Select time_of_the_day, branch,
COUNT(ratings) as number_of_ratings
from sales
group by time_of_the_day,branch
order by number_of_ratings desc;
 -- Which day fo the week has the best avg ratings?
    Select dayname(date), AVG(ratings) as avg_ratings
    from sales
    group by  dayname(date)
    order by avg_ratings desc;

 -- Which day of the week has the best average ratings per branch?

  Select dayname(date), AVG(ratings) as avg_ratings, branch
    from sales
    group by  dayname(date),branch
    order by avg_ratings desc;
```

```sql
 -- Create the database and sales table
CREATE DATABASE IF NOT EXISTS SalesData_walmart;
USE SalesData_walmart;

CREATE TABLE IF NOT EXISTS Sales (
    invoice_id VARCHAR(30) NOT NULL PRIMARY KEY,
    branch VARCHAR(50) NOT NULL,
    city VARCHAR(30) NOT NULL,
    customer_type VARCHAR(30) NOT NULL,
    gender VARCHAR(10) 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(10, 4) NOT NULL,
    date DATETIME NOT NULL,
    time TIME NOT NULL,
    payment_method VARCHAR(15) NOT NULL,
    cogs DECIMAL(10, 2) NOT NULL,
    gross_margin_percentage FLOAT(11, 9),
    gross_income DECIMAL(12, 4) NOT NULL,
    ratings FLOAT(2, 1)
);

-- Feature Engineering
-- Determine time of the day
```

```sql
ALTER TABLE Sales ADD COLUMN time_of_the_day VARCHAR(20);

UPDATE Sales
SET time_of_the_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
);

-- Determine day name
ALTER TABLE Sales ADD COLUMN day_name VARCHAR(20);

UPDATE Sales
SET day_name = DAYNAME(date);

-- Determine month name
ALTER TABLE Sales ADD COLUMN month_name VARCHAR(20);

UPDATE Sales
SET month_name = MONTHNAME(date);

-- Queries
-- Generic
-- Unique cities
SELECT DISTINCT city FROM Sales;

-- Branches with their cities
SELECT DISTINCT city, branch FROM Sales;

-- Product
-- Unique product lines
SELECT DISTINCT product_line FROM Sales;

-- Most selling product line
SELECT product_line, SUM(quantity) AS total_quantity_sold
FROM Sales
GROUP BY product_line
ORDER BY total_quantity_sold DESC;

-- Total revenue by month
SELECT month_name, SUM(total) AS total_revenue
FROM Sales
GROUP BY month_name
ORDER BY total_revenue;

-- Month with the largest COGS
SELECT month_name, SUM(cogs) AS largest_cogs
FROM Sales
GROUP BY month_name
ORDER BY largest_cogs DESC
LIMIT 1;
```

```sql
-- Product line with the largest revenue
SELECT product_line, SUM(total) AS largest_revenue
FROM Sales
GROUP BY product_line
ORDER BY largest_revenue DESC
LIMIT 1;

-- City with the largest revenue
SELECT city, SUM(total) AS total_revenue
FROM Sales
GROUP BY city
ORDER BY total_revenue DESC
LIMIT 1;

-- Product line with the largest VAT
SELECT product_line, SUM(VAT) AS total_VAT
FROM Sales
GROUP BY product_line
ORDER BY total_VAT DESC
LIMIT 1;

-- Customer
-- Unique customer types
SELECT DISTINCT customer_type FROM Sales;

-- Unique payment methods
SELECT DISTINCT payment_method FROM Sales;

-- Most common customer type
SELECT customer_type, COUNT(*) AS total_customers
FROM Sales
GROUP BY customer_type
ORDER BY total_customers DESC
LIMIT 1;

-- Customer type with the most purchases
SELECT customer_type, SUM(quantity) AS total_purchases
FROM Sales
GROUP BY customer_type
ORDER BY total_purchases DESC
LIMIT 1;

-- Gender distribution of customers
SELECT gender, COUNT(*) AS total_customers
FROM Sales
GROUP BY gender
ORDER BY total_customers DESC;

-- Gender distribution per branch
SELECT gender, branch, COUNT(*) AS total_customers
FROM Sales
GROUP BY branch, gender;

-- Time of day with most ratings
```

```sql
SELECT time_of_the_day, COUNT(ratings) AS total_ratings
FROM Sales
GROUP BY time_of_the_day
ORDER BY total_ratings DESC
LIMIT 1;

-- Day of the week with the best average ratings
SELECT DAYNAME(date) AS day_of_week, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date)
ORDER BY avg_ratings DESC
LIMIT 1;

-- Which day of the week has the best average ratings per branch?
SELECT DAYNAME(date) AS day_of_week, branch, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date), branch
ORDER BY avg_ratings DESC;
-- Create the database and sales table
CREATE DATABASE IF NOT EXISTS SalesData_walmart;
USE SalesData_walmart;

CREATE TABLE IF NOT EXISTS Sales (
    invoice_id VARCHAR(30) NOT NULL PRIMARY KEY,
    branch VARCHAR(50) NOT NULL,
    city VARCHAR(30) NOT NULL,
    customer_type VARCHAR(30) NOT NULL,
    gender VARCHAR(10) 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(10, 4) NOT NULL,
    date DATETIME NOT NULL,
    time TIME NOT NULL,
    payment_method VARCHAR(15) NOT NULL,
    cogs DECIMAL(10, 2) NOT NULL,
    gross_margin_percentage FLOAT(11, 9),
    gross_income DECIMAL(12, 4) NOT NULL,
    ratings FLOAT(2, 1)
);

-- Feature Engineering
-- Determine time of the day
ALTER TABLE Sales ADD COLUMN time_of_the_day VARCHAR(20);

UPDATE Sales
SET time_of_the_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
);
```

```sql
-- Determine day name
ALTER TABLE Sales ADD COLUMN day_name VARCHAR(20);

UPDATE Sales
SET day_name = DAYNAME(date);

-- Determine month name
ALTER TABLE Sales ADD COLUMN month_name VARCHAR(20);

UPDATE Sales
SET month_name = MONTHNAME(date);

-- Queries
-- Generic
-- Unique cities
SELECT DISTINCT city FROM Sales;

-- Branches with their cities
SELECT DISTINCT city, branch FROM Sales;

-- Product
-- Unique product lines
SELECT DISTINCT product_line FROM Sales;

-- Most selling product line
SELECT product_line, SUM(quantity) AS total_quantity_sold
FROM Sales
GROUP BY product_line
ORDER BY total_quantity_sold DESC;

-- Total revenue by month
SELECT month_name, SUM(total) AS total_revenue
FROM Sales
GROUP BY month_name
ORDER BY total_revenue;

-- Month with the largest COGS
SELECT month_name, SUM(cogs) AS largest_cogs
FROM Sales
GROUP BY month_name
ORDER BY largest_cogs DESC
LIMIT 1;

-- Product line with the largest revenue
SELECT product_line, SUM(total) AS largest_revenue
FROM Sales
GROUP BY product_line
ORDER BY largest_revenue DESC
LIMIT 1;

-- City with the largest revenue
SELECT city, SUM(total) AS total_revenue
FROM Sales
```

```
GROUP BY city
ORDER BY total_revenue DESC
LIMIT 1;

-- Product line with the largest VAT
SELECT product_line, SUM(VAT) AS total_VAT
FROM Sales
GROUP BY product_line
ORDER BY total_VAT DESC
LIMIT 1;

-- Customer
-- Unique customer types
SELECT DISTINCT customer_type FROM Sales;

-- Unique payment methods
SELECT DISTINCT payment_method FROM Sales;

-- Most common customer type
SELECT customer_type, COUNT(*) AS total_customers
FROM Sales
GROUP BY customer_type
ORDER BY total_customers DESC
LIMIT 1;

-- Customer type with the most purchases
SELECT customer_type, SUM(quantity) AS total_purchases
FROM Sales
GROUP BY customer_type
ORDER BY total_purchases DESC
LIMIT 1;

-- Gender distribution of customers
SELECT gender, COUNT(*) AS total_customers
FROM Sales
GROUP BY gender
ORDER BY total_customers DESC;

-- Gender distribution per branch
SELECT gender, branch, COUNT(*) AS total_customers
FROM Sales
GROUP BY branch, gender;

-- Time of day with most ratings
SELECT time_of_the_day, COUNT(ratings) AS total_ratings
FROM Sales
GROUP BY time_of_the_day
ORDER BY total_ratings DESC
LIMIT 1;

-- Day of the week with the best average ratings
SELECT DAYNAME(date) AS day_of_week, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date)
```

```
ORDER BY avg_ratings DESC
LIMIT 1;

-- Which day of the week has the best average ratings per branch?
SELECT DAYNAME(date) AS day_of_week, branch, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date), branch
ORDER BY avg_ratings DESC;
-- Create the database and sales table
CREATE DATABASE IF NOT EXISTS SalesData_walmart;
USE SalesData_walmart;

CREATE TABLE IF NOT EXISTS Sales (
    invoice_id VARCHAR(30) NOT NULL PRIMARY KEY,
    branch VARCHAR(50) NOT NULL,
    city VARCHAR(30) NOT NULL,
    customer_type VARCHAR(30) NOT NULL,
    gender VARCHAR(10) 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(10, 4) NOT NULL,
    date DATETIME NOT NULL,
    time TIME NOT NULL,
    payment_method VARCHAR(15) NOT NULL,
    cogs DECIMAL(10, 2) NOT NULL,
    gross_margin_percentage FLOAT(11, 9),
    gross_income DECIMAL(12, 4) NOT NULL,
    ratings FLOAT(2, 1)
);

-- Feature Engineering
-- Determine time of the day
ALTER TABLE Sales ADD COLUMN time_of_the_day VARCHAR(20);

UPDATE Sales
SET time_of_the_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
);

-- Determine day name
ALTER TABLE Sales ADD COLUMN day_name VARCHAR(20);

UPDATE Sales
SET day_name = DAYNAME(date);

-- Determine month name
ALTER TABLE Sales ADD COLUMN month_name VARCHAR(20);
```

```
UPDATE Sales
SET month_name = MONTHNAME(date);

-- Queries
-- Generic
-- Unique cities
SELECT DISTINCT city FROM Sales;

-- Branches with their cities
SELECT DISTINCT city, branch FROM Sales;

-- Product
-- Unique product lines
SELECT DISTINCT product_line FROM Sales;

-- Most selling product line
SELECT product_line, SUM(quantity) AS total_quantity_sold
FROM Sales
GROUP BY product_line
ORDER BY total_quantity_sold DESC;

-- Total revenue by month
SELECT month_name, SUM(total) AS total_revenue
FROM Sales
GROUP BY month_name
ORDER BY total_revenue;

-- Month with the largest COGS
SELECT month_name, SUM(cogs) AS largest_cogs
FROM Sales
GROUP BY month_name
ORDER BY largest_cogs DESC
LIMIT 1;

-- Product line with the largest revenue
SELECT product_line, SUM(total) AS largest_revenue
FROM Sales
GROUP BY product_line
ORDER BY largest_revenue DESC
LIMIT 1;

-- City with the largest revenue
SELECT city, SUM(total) AS total_revenue
FROM Sales
GROUP BY city
ORDER BY total_revenue DESC
LIMIT 1;

-- Product line with the largest VAT
SELECT product_line, SUM(VAT) AS total_VAT
FROM Sales
GROUP BY product_line
ORDER BY total_VAT DESC
LIMIT 1;
```

```sql
-- Customer
-- Unique customer types
SELECT DISTINCT customer_type FROM Sales;

-- Unique payment methods
SELECT DISTINCT payment_method FROM Sales;

-- Most common customer type
SELECT customer_type, COUNT(*) AS total_customers
FROM Sales
GROUP BY customer_type
ORDER BY total_customers DESC
LIMIT 1;

-- Customer type with the most purchases
SELECT customer_type, SUM(quantity) AS total_purchases
FROM Sales
GROUP BY customer_type
ORDER BY total_purchases DESC
LIMIT 1;

-- Gender distribution of customers
SELECT gender, COUNT(*) AS total_customers
FROM Sales
GROUP BY gender
ORDER BY total_customers DESC;

-- Gender distribution per branch
SELECT gender, branch, COUNT(*) AS total_customers
FROM Sales
GROUP BY branch, gender;

-- Time of day with most ratings
SELECT time_of_the_day, COUNT(ratings) AS total_ratings
FROM Sales
GROUP BY time_of_the_day
ORDER BY total_ratings DESC
LIMIT 1;

-- Day of the week with the best average ratings
SELECT DAYNAME(date) AS day_of_week, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date)
ORDER BY avg_ratings DESC
LIMIT 1;

-- Which day of the week has the best average ratings per branch?
SELECT DAYNAME(date) AS day_of_week, branch, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date), branch
ORDER BY avg_ratings DESC;
-- Create the database and sales table
CREATE DATABASE IF NOT EXISTS SalesData_walmart;
```

```sql
USE SalesData_walmart;

CREATE TABLE IF NOT EXISTS Sales (
    invoice_id VARCHAR(30) NOT NULL PRIMARY KEY,
    branch VARCHAR(50) NOT NULL,
    city VARCHAR(30) NOT NULL,
    customer_type VARCHAR(30) NOT NULL,
    gender VARCHAR(10) 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(10, 4) NOT NULL,
    date DATETIME NOT NULL,
    time TIME NOT NULL,
    payment_method VARCHAR(15) NOT NULL,
    cogs DECIMAL(10, 2) NOT NULL,
    gross_margin_percentage FLOAT(11, 9),
    gross_income DECIMAL(12, 4) NOT NULL,
    ratings FLOAT(2, 1)
);

-- Feature Engineering
-- Determine time of the day
ALTER TABLE Sales ADD COLUMN time_of_the_day VARCHAR(20);

UPDATE Sales
SET time_of_the_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
);

-- Determine day name
ALTER TABLE Sales ADD COLUMN day_name VARCHAR(20);

UPDATE Sales
SET day_name = DAYNAME(date);

-- Determine month name
ALTER TABLE Sales ADD COLUMN month_name VARCHAR(20);

UPDATE Sales
SET month_name = MONTHNAME(date);

-- Queries
-- Generic
-- Unique cities
SELECT DISTINCT city FROM Sales;

-- Branches with their cities
SELECT DISTINCT city, branch FROM Sales;
```

```sql
-- Product
-- Unique product lines
SELECT DISTINCT product_line FROM Sales;

-- Most selling product line
SELECT product_line, SUM(quantity) AS total_quantity_sold
FROM Sales
GROUP BY product_line
ORDER BY total_quantity_sold DESC;

-- Total revenue by month
SELECT month_name, SUM(total) AS total_revenue
FROM Sales
GROUP BY month_name
ORDER BY total_revenue;

-- Month with the largest COGS
SELECT month_name, SUM(cogs) AS largest_cogs
FROM Sales
GROUP BY month_name
ORDER BY largest_cogs DESC
LIMIT 1;

-- Product line with the largest revenue
SELECT product_line, SUM(total) AS largest_revenue
FROM Sales
GROUP BY product_line
ORDER BY largest_revenue DESC
LIMIT 1;

-- City with the largest revenue
SELECT city, SUM(total) AS total_revenue
FROM Sales
GROUP BY city
ORDER BY total_revenue DESC
LIMIT 1;

-- Product line with the largest VAT
SELECT product_line, SUM(VAT) AS total_VAT
FROM Sales
GROUP BY product_line
ORDER BY total_VAT DESC
LIMIT 1;

-- Customer
-- Unique customer types
SELECT DISTINCT customer_type FROM Sales;

-- Unique payment methods
SELECT DISTINCT payment_method FROM Sales;

-- Most common customer type
SELECT customer_type, COUNT(*) AS total_customers
```

```
FROM Sales
GROUP BY customer_type
ORDER BY total_customers DESC
LIMIT 1;

-- Customer type with the most purchases
SELECT customer_type, SUM(quantity) AS total_purchases
FROM Sales
GROUP BY customer_type
ORDER BY total_purchases DESC
LIMIT 1;

-- Gender distribution of customers
SELECT gender, COUNT(*) AS total_customers
FROM Sales
GROUP BY gender
ORDER BY total_customers DESC;

-- Gender distribution per branch
SELECT gender, branch, COUNT(*) AS total_customers
FROM Sales
GROUP BY branch, gender;

-- Time of day with most ratings
SELECT time_of_the_day, COUNT(ratings) AS total_ratings
FROM Sales
GROUP BY time_of_the_day
ORDER BY total_ratings DESC
LIMIT 1;

-- Day of the week with the best average ratings
SELECT DAYNAME(date) AS day_of_week, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date)
ORDER BY avg_ratings DESC
LIMIT 1;

-- Which day of the week has the best average ratings per branch?
SELECT DAYNAME(date) AS day_of_week, branch, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date), branch
ORDER BY avg_ratings DESC;
-- Create the database and sales table
CREATE DATABASE IF NOT EXISTS SalesData_walmart;
USE SalesData_walmart;

CREATE TABLE IF NOT EXISTS Sales (
    invoice_id VARCHAR(30) NOT NULL PRIMARY KEY,
    branch VARCHAR(50) NOT NULL,
    city VARCHAR(30) NOT NULL,
    customer_type VARCHAR(30) NOT NULL,
    gender VARCHAR(10) 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(10, 4) NOT NULL,
    date DATETIME NOT NULL,
    time TIME NOT NULL,
    payment_method VARCHAR(15) NOT NULL,
    cogs DECIMAL(10, 2) NOT NULL,
    gross_margin_percentage FLOAT(11, 9),
    gross_income DECIMAL(12, 4) NOT NULL,
    ratings FLOAT(2, 1)
);

-- Feature Engineering
-- Determine time of the day
ALTER TABLE Sales ADD COLUMN time_of_the_day VARCHAR(20);

UPDATE Sales
SET time_of_the_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
);

-- Determine day name
ALTER TABLE Sales ADD COLUMN day_name VARCHAR(20);

UPDATE Sales
SET day_name = DAYNAME(date);

-- Determine month name
ALTER TABLE Sales ADD COLUMN month_name VARCHAR(20);

UPDATE Sales
SET month_name = MONTHNAME(date);

-- Queries
-- Generic
-- Unique cities
SELECT DISTINCT city FROM Sales;

-- Branches with their cities
SELECT DISTINCT city, branch FROM Sales;

-- Product
-- Unique product lines
SELECT DISTINCT product_line FROM Sales;

-- Most selling product line
SELECT product_line, SUM(quantity) AS total_quantity_sold
FROM Sales
GROUP BY product_line
ORDER BY total_quantity_sold DESC;
```

```sql
-- Total revenue by month
SELECT month_name, SUM(total) AS total_revenue
FROM Sales
GROUP BY month_name
ORDER BY total_revenue;

-- Month with the largest COGS
SELECT month_name, SUM(cogs) AS largest_cogs
FROM Sales
GROUP BY month_name
ORDER BY largest_cogs DESC
LIMIT 1;

-- Product line with the largest revenue
SELECT product_line, SUM(total) AS largest_revenue
FROM Sales
GROUP BY product_line
ORDER BY largest_revenue DESC
LIMIT 1;

-- City with the largest revenue
SELECT city, SUM(total) AS total_revenue
FROM Sales
GROUP BY city
ORDER BY total_revenue DESC
LIMIT 1;

-- Product line with the largest VAT
SELECT product_line, SUM(VAT) AS total_VAT
FROM Sales
GROUP BY product_line
ORDER BY total_VAT DESC
LIMIT 1;

-- Customer
-- Unique customer types
SELECT DISTINCT customer_type FROM Sales;

-- Unique payment methods
SELECT DISTINCT payment_method FROM Sales;

-- Most common customer type
SELECT customer_type, COUNT(*) AS total_customers
FROM Sales
GROUP BY customer_type
ORDER BY total_customers DESC
LIMIT 1;

-- Customer type with the most purchases
SELECT customer_type, SUM(quantity) AS total_purchases
FROM Sales
GROUP BY customer_type
ORDER BY total_purchases DESC
```

```
LIMIT 1;

-- Gender distribution of customers
SELECT gender, COUNT(*) AS total_customers
FROM Sales
GROUP BY gender
ORDER BY total_customers DESC;

-- Gender distribution per branch
SELECT gender, branch, COUNT(*) AS total_customers
FROM Sales
GROUP BY branch, gender;

-- Time of day with most ratings
SELECT time_of_the_day, COUNT(ratings) AS total_ratings
FROM Sales
GROUP BY time_of_the_day
ORDER BY total_ratings DESC
LIMIT 1;

-- Day of the week with the best average ratings
SELECT DAYNAME(date) AS day_of_week, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date)
ORDER BY avg_ratings DESC
LIMIT 1;

-- Which day of the week has the best average ratings per branch?
SELECT DAYNAME(date) AS day_of_week, branch, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date), branch
ORDER BY avg_ratings DESC;
-- Create the database and sales table
CREATE DATABASE IF NOT EXISTS SalesData_walmart;
USE SalesData_walmart;

CREATE TABLE IF NOT EXISTS Sales (
    invoice_id VARCHAR(30) NOT NULL PRIMARY KEY,
    branch VARCHAR(50) NOT NULL,
    city VARCHAR(30) NOT NULL,
    customer_type VARCHAR(30) NOT NULL,
    gender VARCHAR(10) 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(10, 4) NOT NULL,
    date DATETIME NOT NULL,
    time TIME NOT NULL,
    payment_method VARCHAR(15) NOT NULL,
    cogs DECIMAL(10, 2) NOT NULL,
    gross_margin_percentage FLOAT(11, 9),
    gross_income DECIMAL(12, 4) NOT NULL,
    ratings FLOAT(2, 1)
```

```
);

-- Feature Engineering
-- Determine time of the day
ALTER TABLE Sales ADD COLUMN time_of_the_day VARCHAR(20);

UPDATE Sales
SET time_of_the_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
);

-- Determine day name
ALTER TABLE Sales ADD COLUMN day_name VARCHAR(20);

UPDATE Sales
SET day_name = DAYNAME(date);

-- Determine month name
ALTER TABLE Sales ADD COLUMN month_name VARCHAR(20);

UPDATE Sales
SET month_name = MONTHNAME(date);

-- Queries
-- Generic
-- Unique cities
SELECT DISTINCT city FROM Sales;

-- Branches with their cities
SELECT DISTINCT city, branch FROM Sales;

-- Product
-- Unique product lines
SELECT DISTINCT product_line FROM Sales;

-- Most selling product line
SELECT product_line, SUM(quantity) AS total_quantity_sold
FROM Sales
GROUP BY product_line
ORDER BY total_quantity_sold DESC;

-- Total revenue by month
SELECT month_name, SUM(total) AS total_revenue
FROM Sales
GROUP BY month_name
ORDER BY total_revenue;

-- Month with the largest COGS
SELECT month_name, SUM(cogs) AS largest_cogs
FROM Sales
```

```
GROUP BY month_name
ORDER BY largest_cogs DESC
LIMIT 1;

-- Product line with the largest revenue
SELECT product_line, SUM(total) AS largest_revenue
FROM Sales
GROUP BY product_line
ORDER BY largest_revenue DESC
LIMIT 1;

-- City with the largest revenue
SELECT city, SUM(total) AS total_revenue
FROM Sales
GROUP BY city
ORDER BY total_revenue DESC
LIMIT 1;

-- Product line with the largest VAT
SELECT product_line, SUM(VAT) AS total_VAT
FROM Sales
GROUP BY product_line
ORDER BY total_VAT DESC
LIMIT 1;

-- Customer
-- Unique customer types
SELECT DISTINCT customer_type FROM Sales;

-- Unique payment methods
SELECT DISTINCT payment_method FROM Sales;

-- Most common customer type
SELECT customer_type, COUNT(*) AS total_customers
FROM Sales
GROUP BY customer_type
ORDER BY total_customers DESC
LIMIT 1;

-- Customer type with the most purchases
SELECT customer_type, SUM(quantity) AS total_purchases
FROM Sales
GROUP BY customer_type
ORDER BY total_purchases DESC
LIMIT 1;

-- Gender distribution of customers
SELECT gender, COUNT(*) AS total_customers
FROM Sales
GROUP BY gender
ORDER BY total_customers DESC;

-- Gender distribution per branch
SELECT gender, branch, COUNT(*) AS total_customers
```

```
FROM Sales
GROUP BY branch, gender;

-- Time of day with most ratings
SELECT time_of_the_day, COUNT(ratings) AS total_ratings
FROM Sales
GROUP BY time_of_the_day
ORDER BY total_ratings DESC
LIMIT 1;

-- Day of the week with the best average ratings
SELECT DAYNAME(date) AS day_of_week, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date)
ORDER BY avg_ratings DESC
LIMIT 1;

-- Which day of the week has the best average ratings per branch?
SELECT DAYNAME(date) AS day_of_week, branch, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date), branch
ORDER BY avg_ratings DESC;
-- Create the database and sales table
CREATE DATABASE IF NOT EXISTS SalesData_walmart;
USE SalesData_walmart;

CREATE TABLE IF NOT EXISTS Sales (
    invoice_id VARCHAR(30) NOT NULL PRIMARY KEY,
    branch VARCHAR(50) NOT NULL,
    city VARCHAR(30) NOT NULL,
    customer_type VARCHAR(30) NOT NULL,
    gender VARCHAR(10) 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(10, 4) NOT NULL,
    date DATETIME NOT NULL,
    time TIME NOT NULL,
    payment_method VARCHAR(15) NOT NULL,
    cogs DECIMAL(10, 2) NOT NULL,
    gross_margin_percentage FLOAT(11, 9),
    gross_income DECIMAL(12, 4) NOT NULL,
    ratings FLOAT(2, 1)
);

-- Feature Engineering
-- Determine time of the day
ALTER TABLE Sales ADD COLUMN time_of_the_day VARCHAR(20);

UPDATE Sales
SET time_of_the_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
);

-- Determine day name
ALTER TABLE Sales ADD COLUMN day_name VARCHAR(20);

UPDATE Sales
SET day_name = DAYNAME(date);

-- Determine month name
ALTER TABLE Sales ADD COLUMN month_name VARCHAR(20);

UPDATE Sales
SET month_name = MONTHNAME(date);

-- Queries
-- Generic
-- Unique cities
SELECT DISTINCT city FROM Sales;

-- Branches with their cities
SELECT DISTINCT city, branch FROM Sales;

-- Product
-- Unique product lines
SELECT DISTINCT product_line FROM Sales;

-- Most selling product line
SELECT product_line, SUM(quantity) AS total_quantity_sold
FROM Sales
GROUP BY product_line
ORDER BY total_quantity_sold DESC;

-- Total revenue by month
SELECT month_name, SUM(total) AS total_revenue
FROM Sales
GROUP BY month_name
ORDER BY total_revenue;

-- Month with the largest COGS
SELECT month_name, SUM(cogs) AS largest_cogs
FROM Sales
GROUP BY month_name
ORDER BY largest_cogs DESC
LIMIT 1;

-- Product line with the largest revenue
SELECT product_line, SUM(total) AS largest_revenue
FROM Sales
GROUP BY product_line
ORDER BY largest_revenue DESC
LIMIT 1;
```

```sql
-- City with the largest revenue
SELECT city, SUM(total) AS total_revenue
FROM Sales
GROUP BY city
ORDER BY total_revenue DESC
LIMIT 1;

-- Product line with the largest VAT
SELECT product_line, SUM(VAT) AS total_VAT
FROM Sales
GROUP BY product_line
ORDER BY total_VAT DESC
LIMIT 1;

-- Customer
-- Unique customer types
SELECT DISTINCT customer_type FROM Sales;

-- Unique payment methods
SELECT DISTINCT payment_method FROM Sales;

-- Most common customer type
SELECT customer_type, COUNT(*) AS total_customers
FROM Sales
GROUP BY customer_type
ORDER BY total_customers DESC
LIMIT 1;

-- Customer type with the most purchases
SELECT customer_type, SUM(quantity) AS total_purchases
FROM Sales
GROUP BY customer_type
ORDER BY total_purchases DESC
LIMIT 1;

-- Gender distribution of customers
SELECT gender, COUNT(*) AS total_customers
FROM Sales
GROUP BY gender
ORDER BY total_customers DESC;

-- Gender distribution per branch
SELECT gender, branch, COUNT(*) AS total_customers
FROM Sales
GROUP BY branch, gender;

-- Time of day with most ratings
SELECT time_of_the_day, COUNT(ratings) AS total_ratings
FROM Sales
GROUP BY time_of_the_day
ORDER BY total_ratings DESC
LIMIT 1;
```

```sql
-- Day of the week with the best average ratings
SELECT DAYNAME(date) AS day_of_week, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date)
ORDER BY avg_ratings DESC
LIMIT 1;

-- Which day of the week has the best average ratings per branch?
SELECT DAYNAME(date) AS day_of_week, branch, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date), branch
ORDER BY avg_ratings DESC;
-- Create the database and sales table
CREATE DATABASE IF NOT EXISTS SalesData_walmart;
USE SalesData_walmart;

CREATE TABLE IF NOT EXISTS Sales (
    invoice_id VARCHAR(30) NOT NULL PRIMARY KEY,
    branch VARCHAR(50) NOT NULL,
    city VARCHAR(30) NOT NULL,
    customer_type VARCHAR(30) NOT NULL,
    gender VARCHAR(10) 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(10, 4) NOT NULL,
    date DATETIME NOT NULL,
    time TIME NOT NULL,
    payment_method VARCHAR(15) NOT NULL,
    cogs DECIMAL(10, 2) NOT NULL,
    gross_margin_percentage FLOAT(11, 9),
    gross_income DECIMAL(12, 4) NOT NULL,
    ratings FLOAT(2, 1)
);

-- Feature Engineering
-- Determine time of the day
ALTER TABLE Sales ADD COLUMN time_of_the_day VARCHAR(20);

UPDATE Sales
SET time_of_the_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
);

-- Determine day name
ALTER TABLE Sales ADD COLUMN day_name VARCHAR(20);

UPDATE Sales
SET day_name = DAYNAME(date);
```

```sql
-- Determine month name
ALTER TABLE Sales ADD COLUMN month_name VARCHAR(20);

UPDATE Sales
SET month_name = MONTHNAME(date);

-- Queries
-- Generic
-- Unique cities
SELECT DISTINCT city FROM Sales;

-- Branches with their cities
SELECT DISTINCT city, branch FROM Sales;

-- Product
-- Unique product lines
SELECT DISTINCT product_line FROM Sales;

-- Most selling product line
SELECT product_line, SUM(quantity) AS total_quantity_sold
FROM Sales
GROUP BY product_line
ORDER BY total_quantity_sold DESC;

-- Total revenue by month
SELECT month_name, SUM(total) AS total_revenue
FROM Sales
GROUP BY month_name
ORDER BY total_revenue;

-- Month with the largest COGS
SELECT month_name, SUM(cogs) AS largest_cogs
FROM Sales
GROUP BY month_name
ORDER BY largest_cogs DESC
LIMIT 1;

-- Product line with the largest revenue
SELECT product_line, SUM(total) AS largest_revenue
FROM Sales
GROUP BY product_line
ORDER BY largest_revenue DESC
LIMIT 1;

-- City with the largest revenue
SELECT city, SUM(total) AS total_revenue
FROM Sales
GROUP BY city
ORDER BY total_revenue DESC
LIMIT 1;

-- Product line with the largest VAT
SELECT product_line, SUM(VAT) AS total_VAT
```

```
FROM Sales
GROUP BY product_line
ORDER BY total_VAT DESC
LIMIT 1;

-- Customer
-- Unique customer types
SELECT DISTINCT customer_type FROM Sales;

-- Unique payment methods
SELECT DISTINCT payment_method FROM Sales;

-- Most common customer type
SELECT customer_type, COUNT(*) AS total_customers
FROM Sales
GROUP BY customer_type
ORDER BY total_customers DESC
LIMIT 1;

-- Customer type with the most purchases
SELECT customer_type, SUM(quantity) AS total_purchases
FROM Sales
GROUP BY customer_type
ORDER BY total_purchases DESC
LIMIT 1;

-- Gender distribution of customers
SELECT gender, COUNT(*) AS total_customers
FROM Sales
GROUP BY gender
ORDER BY total_customers DESC;

-- Gender distribution per branch
SELECT gender, branch, COUNT(*) AS total_customers
FROM Sales
GROUP BY branch, gender;

-- Time of day with most ratings
SELECT time_of_the_day, COUNT(ratings) AS total_ratings
FROM Sales
GROUP BY time_of_the_day
ORDER BY total_ratings DESC
LIMIT 1;

-- Day of the week with the best average ratings
SELECT DAYNAME(date) AS day_of_week, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date)
ORDER BY avg_ratings DESC
LIMIT 1;

-- Which day of the week has the best average ratings per branch?
SELECT DAYNAME(date) AS day_of_week, branch, AVG(ratings) AS avg_ratings
FROM Sales
```

```
GROUP BY DAYOFWEEK(date), branch
ORDER BY avg_ratings DESC;
-- Create the database and sales table
CREATE DATABASE IF NOT EXISTS SalesData_walmart;
USE SalesData_walmart;

CREATE TABLE IF NOT EXISTS Sales (
    invoice_id VARCHAR(30) NOT NULL PRIMARY KEY,
    branch VARCHAR(50) NOT NULL,
    city VARCHAR(30) NOT NULL,
    customer_type VARCHAR(30) NOT NULL,
    gender VARCHAR(10) 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(10, 4) NOT NULL,
    date DATETIME NOT NULL,
    time TIME NOT NULL,
    payment_method VARCHAR(15) NOT NULL,
    cogs DECIMAL(10, 2) NOT NULL,
    gross_margin_percentage FLOAT(11, 9),
    gross_income DECIMAL(12, 4) NOT NULL,
    ratings FLOAT(2, 1)
);

-- Feature Engineering
-- Determine time of the day
ALTER TABLE Sales ADD COLUMN time_of_the_day VARCHAR(20);

UPDATE Sales
SET time_of_the_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
);

-- Determine day name
ALTER TABLE Sales ADD COLUMN day_name VARCHAR(20);

UPDATE Sales
SET day_name = DAYNAME(date);

-- Determine month name
ALTER TABLE Sales ADD COLUMN month_name VARCHAR(20);

UPDATE Sales
SET month_name = MONTHNAME(date);

-- Queries
-- Generic
-- Unique cities
```

```
SELECT DISTINCT city FROM Sales;

-- Branches with their cities
SELECT DISTINCT city, branch FROM Sales;

-- Product
-- Unique product lines
SELECT DISTINCT product_line FROM Sales;

-- Most selling product line
SELECT product_line, SUM(quantity) AS total_quantity_sold
FROM Sales
GROUP BY product_line
ORDER BY total_quantity_sold DESC;

-- Total revenue by month
SELECT month_name, SUM(total) AS total_revenue
FROM Sales
GROUP BY month_name
ORDER BY total_revenue;

-- Month with the largest COGS
SELECT month_name, SUM(cogs) AS largest_cogs
FROM Sales
GROUP BY month_name
ORDER BY largest_cogs DESC
LIMIT 1;

-- Product line with the largest revenue
SELECT product_line, SUM(total) AS largest_revenue
FROM Sales
GROUP BY product_line
ORDER BY largest_revenue DESC
LIMIT 1;

-- City with the largest revenue
SELECT city, SUM(total) AS total_revenue
FROM Sales
GROUP BY city
ORDER BY total_revenue DESC
LIMIT 1;

-- Product line with the largest VAT
SELECT product_line, SUM(VAT) AS total_VAT
FROM Sales
GROUP BY product_line
ORDER BY total_VAT DESC
LIMIT 1;

-- Customer
-- Unique customer types
SELECT DISTINCT customer_type FROM Sales;

-- Unique payment methods
```

```sql
SELECT DISTINCT payment_method FROM Sales;

-- Most common customer type
SELECT customer_type, COUNT(*) AS total_customers
FROM Sales
GROUP BY customer_type
ORDER BY total_customers DESC
LIMIT 1;

-- Customer type with the most purchases
SELECT customer_type, SUM(quantity) AS total_purchases
FROM Sales
GROUP BY customer_type
ORDER BY total_purchases DESC
LIMIT 1;

-- Gender distribution of customers
SELECT gender, COUNT(*) AS total_customers
FROM Sales
GROUP BY gender
ORDER BY total_customers DESC;

-- Gender distribution per branch
SELECT gender, branch, COUNT(*) AS total_customers
FROM Sales
GROUP BY branch, gender;

-- Time of day with most ratings
SELECT time_of_the_day, COUNT(ratings) AS total_ratings
FROM Sales
GROUP BY time_of_the_day
ORDER BY total_ratings DESC
LIMIT 1;

-- Day of the week with the best average ratings
SELECT DAYNAME(date) AS day_of_week, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date)
ORDER BY avg_ratings DESC
LIMIT 1;

-- Which day of the week has the best average ratings per branch?
SELECT DAYNAME(date) AS day_of_week, branch, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date), branch
ORDER BY avg_ratings DESC;
-- Create the database and sales table
CREATE DATABASE IF NOT EXISTS SalesData_walmart;
USE SalesData_walmart;

CREATE TABLE IF NOT EXISTS Sales (
    invoice_id VARCHAR(30) NOT NULL PRIMARY KEY,
    branch VARCHAR(50) NOT NULL,
    city VARCHAR(30) NOT NULL,
```

```
    customer_type VARCHAR(30) NOT NULL,
    gender VARCHAR(10) 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(10, 4) NOT NULL,
    date DATETIME NOT NULL,
    time TIME NOT NULL,
    payment_method VARCHAR(15) NOT NULL,
    cogs DECIMAL(10, 2) NOT NULL,
    gross_margin_percentage FLOAT(11, 9),
    gross_income DECIMAL(12, 4) NOT NULL,
    ratings FLOAT(2, 1)
);

-- Feature Engineering
-- Determine time of the day
ALTER TABLE Sales ADD COLUMN time_of_the_day VARCHAR(20);

UPDATE Sales
SET time_of_the_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
);

-- Determine day name
ALTER TABLE Sales ADD COLUMN day_name VARCHAR(20);

UPDATE Sales
SET day_name = DAYNAME(date);

-- Determine month name
ALTER TABLE Sales ADD COLUMN month_name VARCHAR(20);

UPDATE Sales
SET month_name = MONTHNAME(date);

-- Queries
-- Generic
-- Unique cities
SELECT DISTINCT city FROM Sales;

-- Branches with their cities
SELECT DISTINCT city, branch FROM Sales;

-- Product
-- Unique product lines
SELECT DISTINCT product_line FROM Sales;

-- Most selling product line
```

```sql
SELECT product_line, SUM(quantity) AS total_quantity_sold
FROM Sales
GROUP BY product_line
ORDER BY total_quantity_sold DESC;

-- Total revenue by month
SELECT month_name, SUM(total) AS total_revenue
FROM Sales
GROUP BY month_name
ORDER BY total_revenue;

-- Month with the largest COGS
SELECT month_name, SUM(cogs) AS largest_cogs
FROM Sales
GROUP BY month_name
ORDER BY largest_cogs DESC
LIMIT 1;

-- Product line with the largest revenue
SELECT product_line, SUM(total) AS largest_revenue
FROM Sales
GROUP BY product_line
ORDER BY largest_revenue DESC
LIMIT 1;

-- City with the largest revenue
SELECT city, SUM(total) AS total_revenue
FROM Sales
GROUP BY city
ORDER BY total_revenue DESC
LIMIT 1;

-- Product line with the largest VAT
SELECT product_line, SUM(VAT) AS total_VAT
FROM Sales
GROUP BY product_line
ORDER BY total_VAT DESC
LIMIT 1;

-- Customer
-- Unique customer types
SELECT DISTINCT customer_type FROM Sales;

-- Unique payment methods
SELECT DISTINCT payment_method FROM Sales;

-- Most common customer type
SELECT customer_type, COUNT(*) AS total_customers
FROM Sales
GROUP BY customer_type
ORDER BY total_customers DESC
LIMIT 1;

-- Customer type with the most purchases
```

```sql
SELECT customer_type, SUM(quantity) AS total_purchases
FROM Sales
GROUP BY customer_type
ORDER BY total_purchases DESC
LIMIT 1;

-- Gender distribution of customers
SELECT gender, COUNT(*) AS total_customers
FROM Sales
GROUP BY gender
ORDER BY total_customers DESC;

-- Gender distribution per branch
SELECT gender, branch, COUNT(*) AS total_customers
FROM Sales
GROUP BY branch, gender;

-- Time of day with most ratings
SELECT time_of_the_day, COUNT(ratings) AS total_ratings
FROM Sales
GROUP BY time_of_the_day
ORDER BY total_ratings DESC
LIMIT 1;

-- Day of the week with the best average ratings
SELECT DAYNAME(date) AS day_of_week, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date)
ORDER BY avg_ratings DESC
LIMIT 1;

-- Which day of the week has the best average ratings per branch?
SELECT DAYNAME(date) AS day_of_week, branch, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date), branch
ORDER BY avg_ratings DESC;
-- Create the database and sales table
CREATE DATABASE IF NOT EXISTS SalesData_walmart;
USE SalesData_walmart;

CREATE TABLE IF NOT EXISTS Sales (
    invoice_id VARCHAR(30) NOT NULL PRIMARY KEY,
    branch VARCHAR(50) NOT NULL,
    city VARCHAR(30) NOT NULL,
    customer_type VARCHAR(30) NOT NULL,
    gender VARCHAR(10) 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(10, 4) NOT NULL,
    date DATETIME NOT NULL,
    time TIME NOT NULL,
    payment_method VARCHAR(15) NOT NULL,
```

```
    cogs DECIMAL(10, 2) NOT NULL,
    gross_margin_percentage FLOAT(11, 9),
    gross_income DECIMAL(12, 4) NOT NULL,
    ratings FLOAT(2, 1)
);

-- Feature Engineering
-- Determine time of the day
ALTER TABLE Sales ADD COLUMN time_of_the_day VARCHAR(20);

UPDATE Sales
SET time_of_the_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
);

-- Determine day name
ALTER TABLE Sales ADD COLUMN day_name VARCHAR(20);

UPDATE Sales
SET day_name = DAYNAME(date);

-- Determine month name
ALTER TABLE Sales ADD COLUMN month_name VARCHAR(20);

UPDATE Sales
SET month_name = MONTHNAME(date);

-- Queries
-- Generic
-- Unique cities
SELECT DISTINCT city FROM Sales;

-- Branches with their cities
SELECT DISTINCT city, branch FROM Sales;

-- Product
-- Unique product lines
SELECT DISTINCT product_line FROM Sales;

-- Most selling product line
SELECT product_line, SUM(quantity) AS total_quantity_sold
FROM Sales
GROUP BY product_line
ORDER BY total_quantity_sold DESC;

-- Total revenue by month
SELECT month_name, SUM(total) AS total_revenue
FROM Sales
GROUP BY month_name
ORDER BY total_revenue;
```

```sql
-- Month with the largest COGS
SELECT month_name, SUM(cogs) AS largest_cogs
FROM Sales
GROUP BY month_name
ORDER BY largest_cogs DESC
LIMIT 1;

-- Product line with the largest revenue
SELECT product_line, SUM(total) AS largest_revenue
FROM Sales
GROUP BY product_line
ORDER BY largest_revenue DESC
LIMIT 1;

-- City with the largest revenue
SELECT city, SUM(total) AS total_revenue
FROM Sales
GROUP BY city
ORDER BY total_revenue DESC
LIMIT 1;

-- Product line with the largest VAT
SELECT product_line, SUM(VAT) AS total_VAT
FROM Sales
GROUP BY product_line
ORDER BY total_VAT DESC
LIMIT 1;

-- Customer
-- Unique customer types
SELECT DISTINCT customer_type FROM Sales;

-- Unique payment methods
SELECT DISTINCT payment_method FROM Sales;

-- Most common customer type
SELECT customer_type, COUNT(*) AS total_customers
FROM Sales
GROUP BY customer_type
ORDER BY total_customers DESC
LIMIT 1;

-- Customer type with the most purchases
SELECT customer_type, SUM(quantity) AS total_purchases
FROM Sales
GROUP BY customer_type
ORDER BY total_purchases DESC
LIMIT 1;

-- Gender distribution of customers
SELECT gender, COUNT(*) AS total_customers
FROM Sales
GROUP BY gender
```

```
ORDER BY total_customers DESC;

-- Gender distribution per branch
SELECT gender, branch, COUNT(*) AS total_customers
FROM Sales
GROUP BY branch, gender;

-- Time of day with most ratings
SELECT time_of_the_day, COUNT(ratings) AS total_ratings
FROM Sales
GROUP BY time_of_the_day
ORDER BY total_ratings DESC
LIMIT 1;

-- Day of the week with the best average ratings
SELECT DAYNAME(date) AS day_of_week, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date)
ORDER BY avg_ratings DESC
LIMIT 1;

-- Which day of the week has the best average ratings per branch?
SELECT DAYNAME(date) AS day_of_week, branch, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date), branch
ORDER BY avg_ratings DESC;
-- Create the database and sales table
CREATE DATABASE IF NOT EXISTS SalesData_walmart;
USE SalesData_walmart;

CREATE TABLE IF NOT EXISTS Sales (
    invoice_id VARCHAR(30) NOT NULL PRIMARY KEY,
    branch VARCHAR(50) NOT NULL,
    city VARCHAR(30) NOT NULL,
    customer_type VARCHAR(30) NOT NULL,
    gender VARCHAR(10) 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(10, 4) NOT NULL,
    date DATETIME NOT NULL,
    time TIME NOT NULL,
    payment_method VARCHAR(15) NOT NULL,
    cogs DECIMAL(10, 2) NOT NULL,
    gross_margin_percentage FLOAT(11, 9),
    gross_income DECIMAL(12, 4) NOT NULL,
    ratings FLOAT(2, 1)
);

-- Feature Engineering
-- Determine time of the day
ALTER TABLE Sales ADD COLUMN time_of_the_day VARCHAR(20);
```

```
UPDATE Sales
SET time_of_the_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
);

-- Determine day name
ALTER TABLE Sales ADD COLUMN day_name VARCHAR(20);

UPDATE Sales
SET day_name = DAYNAME(date);

-- Determine month name
ALTER TABLE Sales ADD COLUMN month_name VARCHAR(20);

UPDATE Sales
SET month_name = MONTHNAME(date);

-- Queries
-- Generic
-- Unique cities
SELECT DISTINCT city FROM Sales;

-- Branches with their cities
SELECT DISTINCT city, branch FROM Sales;

-- Product
-- Unique product lines
SELECT DISTINCT product_line FROM Sales;

-- Most selling product line
SELECT product_line, SUM(quantity) AS total_quantity_sold
FROM Sales
GROUP BY product_line
ORDER BY total_quantity_sold DESC;

-- Total revenue by month
SELECT month_name, SUM(total) AS total_revenue
FROM Sales
GROUP BY month_name
ORDER BY total_revenue;

-- Month with the largest COGS
SELECT month_name, SUM(cogs) AS largest_cogs
FROM Sales
GROUP BY month_name
ORDER BY largest_cogs DESC
LIMIT 1;

-- Product line with the largest revenue
SELECT product_line, SUM(total) AS largest_revenue
```

```
FROM Sales
GROUP BY product_line
ORDER BY largest_revenue DESC
LIMIT 1;

-- City with the largest revenue
SELECT city, SUM(total) AS total_revenue
FROM Sales
GROUP BY city
ORDER BY total_revenue DESC
LIMIT 1;

-- Product line with the largest VAT
SELECT product_line, SUM(VAT) AS total_VAT
FROM Sales
GROUP BY product_line
ORDER BY total_VAT DESC
LIMIT 1;

-- Customer
-- Unique customer types
SELECT DISTINCT customer_type FROM Sales;

-- Unique payment methods
SELECT DISTINCT payment_method FROM Sales;

-- Most common customer type
SELECT customer_type, COUNT(*) AS total_customers
FROM Sales
GROUP BY customer_type
ORDER BY total_customers DESC
LIMIT 1;

-- Customer type with the most purchases
SELECT customer_type, SUM(quantity) AS total_purchases
FROM Sales
GROUP BY customer_type
ORDER BY total_purchases DESC
LIMIT 1;

-- Gender distribution of customers
SELECT gender, COUNT(*) AS total_customers
FROM Sales
GROUP BY gender
ORDER BY total_customers DESC;

-- Gender distribution per branch
SELECT gender, branch, COUNT(*) AS total_customers
FROM Sales
GROUP BY branch, gender;

-- Time of day with most ratings
SELECT time_of_the_day, COUNT(ratings) AS total_ratings
FROM Sales
```

```
GROUP BY time_of_the_day
ORDER BY total_ratings DESC
LIMIT 1;

-- Day of the week with the best average ratings
SELECT DAYNAME(date) AS day_of_week, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date)
ORDER BY avg_ratings DESC
LIMIT 1;

-- Which day of the week has the best average ratings per branch?
SELECT DAYNAME(date) AS day_of_week, branch, AVG(ratings) AS avg_ratings
FROM Sales
GROUP BY DAYOFWEEK(date), branch
ORDER BY avg_ratings DESC;
```