Walmart Sales Data Analysis

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 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

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 sold	VARCHAR(100)
unit_price	The price of each product	DECIMAL(10, 2)
quantity	The amount of the product sold	INT

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

RESULTS AND DISCUSSIONS

- 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.

- 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.
- Customer Analysis
- This analysis aims to uncover the different customers segments, purchase trends and the profitability of each customer segment.

Approach Used and Query

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.

Build a database

Create table and insert the data.

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.

```
CREATE DATABASE IF NOT EXISTS walmartSales1;


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(30) NOT NULL,
   product_line VARCHAR(100) NOT NULL,
    unit_price DECIMAL(10,2) NOT NULL,
    quantity INT NOT NULL,
    tax_pct FLOAT(6,4) NOT NULL,
    total DECIMAL(12, 4) NOT NULL,
    date DATETIME NOT NULL,
    time TIME NOT NULL,
    payment VARCHAR(15) NOT NULL,
    cogs DECIMAL(10,2) NOT NULL,
    gross_margin_pct FLOAT(11,9),
    gross_income DECIMAL(12, 4),
    rating FLOAT(2, 1)
);

SELECT
        *
FROM sales;
```

- Click table and click import and import the data walmart

<u>Feature Engineering</u>: This will help use generate some new columns from existing ones.

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.

```
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_day
FROM sales;

ALTER TABLE sales ADD COLUMN time_of_day VARCHAR(20);
```

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.

```
SELECT
        date,
        DAYNAME(date)
FROM sales;

ALTER TABLE sales ADD COLUMN day_name VARCHAR(10);

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

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.

```
SELECT
        date,
        MONTHNAME(date)
FROM sales;
```

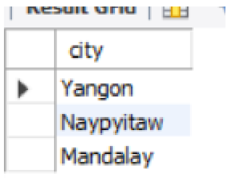
```
ALTER TABLE sales ADD COLUMN month_name VARCHAR(10);
```

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

Exploratory Data Analysis (EDA)

- unique cities does the data have

```
SELECT DISTINCT city FROM sales;
```



```
city
Yangon
Naypyitaw
Mandalay
```

- Branch in each city
- SELECT DISTINCT city,branch FROM sales;

- Number of unique product lines present in the data

- select distinct product_line from sales;

- most selling product line

```
SELECT SUM(quantity) as qty,product_line
FROM sales
GROUP BY product_line
ORDER BY qty DESC;
```

- 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 desc;
```

- The month with the largest COGS

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

- product line with the largest revenue

```
SELECT product_line,SUM(total) as total_revenue
FROM sales
GROUP BY product_line
ORDER BY total_revenue DESC;
```

- The city with the largest revenue

```
SELECT branch,city,SUM(total) AS total_revenue
FROM sales
GROUP BY city, branch
ORDER BY total_revenue;
```

- product line had the largest VAT

-
  ```
  SELECT product_line,AVG(tax_pct) as avg_tax
  FROM sales
  GROUP BY product_line
  ORDER BY avg_tax DESC;
  ```

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

-
  ```
  SELECT
       product_line,
       CASE
            WHEN AVG(quantity) > 6 THEN "Good"
      ELSE "Bad"
    END AS remark
  FROM sales
  GROUP BY product_line;
  ```

- The branch sold more products than average product sold
  ```
  SELECT branch, SUM(quantity) AS qnty
  FROM sales
  GROUP BY branch
  HAVING SUM(quantity) > (SELECT AVG(quantity) FROM sales);
  ```
- The most common product line by gender
  ```
  SELECT gender,product_line,COUNT(gender) AS total_cnt
  FROM sales
  GROUP BY gender, product_line
  ORDER BY total_cnt DESC;
  ```

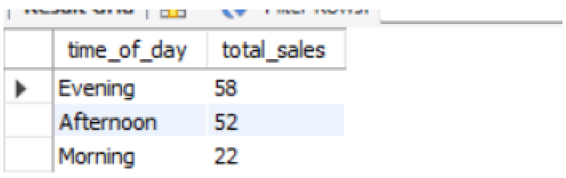
- The average rating of each product line
  ```
  SELECT ROUND(AVG(rating), 2) as avg_rating,product_line
  FROM sales
  GROUP BY product_line
  ```

ORDER BY avg_rating DESC;

Sales

- Number of sales made in each time of the day per weekday

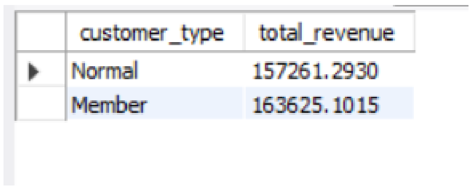
```
SELECT time_of_day,COUNT(*) AS total_sales
FROM sales
WHERE day_name = "Sunday"
GROUP BY time_of_day
ORDER BY total_sales DESC;
```



time_of_day	total_sales
Evening	58
Afternoon	52
Morning	22

- The customer types brings the most revenue

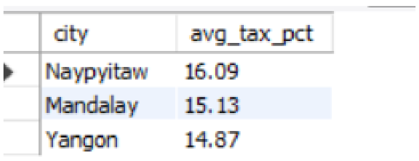
```
SELECT customer_type,SUM(total) AS total_revenue
FROM sales
GROUP BY customer_type
ORDER BY total_revenue;
```



customer_type	total_revenue
Normal	157261.2930
Member	163625.1015

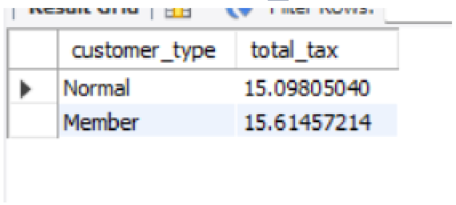
- city with largest tax percent/ VAT (Value Added Tax)

```
SELECT city,ROUND(AVG(tax_pct), 2) AS avg_tax_pct
FROM sales
GROUP BY city
ORDER BY avg_tax_pct DESC;
```



	city	avg_tax_pct
▶	Naypyitaw	16.09
	Mandalay	15.13
	Yangon	14.87

- The customer type pays the most in VAT
  SELECT customer_type,AVG(tax_pct) AS total_tax
  FROM sales
  GROUP BY customer_type
  ORDER BY total_tax;



	customer_type	total_tax
▶	Normal	15.09805040
	Member	15.61457214

- **Customer Analysis**

1. unique customer types in the data

- SELECT DISTINCT customer_type
  FROM sales;

2. unique payment methods in the data

- SELECT DISTINCT payment
  FROM sales;

3. Most common type of customer

- SELECT customer_type,count(*) as count
  FROM sales
  GROUP BY customer_type
  ORDER BY count DESC;

4. Type of customer who buys the most

- SELECTcustomer_type,COUNT(*)
  FROM sales
  GROUP BY customer_type;

5. The gender most of the customers belongs to

-
  ```
  SELECT gender,COUNT(*) as gender_cnt
  FROM sales
  GROUP BY gender
  ORDER BY gender_cnt DESC;
  ```

6. The gender distribution per branch

-
  ```
  SELECT gender,COUNT(*) as gender_cnt
  FROM sales
  WHERE branch = "C"
  GROUP BY gender
  ORDER BY gender_cnt DESC;
  ```

7. The time of the day,customers give most ratings

-
  ```
  SELECT time_of_day,AVG(rating) AS avg_rating
  FROM sales
  GROUP BY time_of_day
  ORDER BY avg_rating DESC;
  ```

8. Time of the day do customers give most ratings per branch

-
  ```
  SELECT time_of_day,AVG(rating) AS avg_rating
  FROM sales
  WHERE branch = "A"
  GROUP BY time_of_day
  ORDER BY avg_rating DESC;
  ```

9. Day fo the week with the best avg ratings

-
  ```
  SELECT day_name,AVG(rating) AS avg_rating
  FROM sales
  GROUP BY day_name
  ORDER BY avg_rating DESC;
  ```

10. Day of the week with the best average ratings per branch

-
  ```
  SELECT day_name,COUNT(day_name) total_sales
  FROM sales
  ```

```
WHERE branch = "C"
GROUP BY day_name
ORDER BY total_sales DESC;
```