

Walmart





Purposes Of The Project

This project aims to explore the Walmart Sales data to understand top performing branches and products, sales trend of different products, customer behaviour. The aim is to study how sales strategies can be improved and optimized

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 modifications 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

Data Wrangling

Elaborate on yoThis 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.
ur
second goal here.

Feature Engineering

This will help use generate some new columns from existing ones.

Exploratory Data Analysis (EDA)

Elaborate on your first goal here & last conclusion.

Business Questions To Answer

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?

Business Questions To Answer

13. Number of sales made in each time of the day per weekday.
14. Which of the customer types brings the most revenue?
15. Which city has the largest tax percent/ VAT (Value Added Tax)?
16. Which customer type pays the most in VAT?
17. How many unique customer types does the data have?
18. How many unique payment methods does the data have?
19. What is the most common customer type?
20. Which customer type buys the most?
21. What is the gender of most of the customers?
22. What is the gender distribution per branch?
23. Which time of the day do customers give most ratings?
24. Which time of the day do customers give most ratings per branch?
25. Which day fo the week has the best avg ratings?

Q How many unique product lines does the data have?

The screenshot shows a MySQL Workbench interface. On the left, the schema browser displays a database named 'salesdatawalmart' with a single table 'sales'. The 'Columns' section is expanded, listing various columns like invoice_id, branch, city, customer_type, gender, product_line, unit_price, quantity, tax_pct, total, date, time, payment, and cogs. A query editor window is open with the following SQL code:

```
12  
13 -- Q How many unique product lines does the data have?  
14  
15  
16 * SELECT count(DISTINCT product_line) FROM sales;  
17
```

A tooltip for the autocommit button (a blue circle with a white upward arrow) provides information about autocommit mode:

Toggle autocommit mode. When enabled, each statement will be committed immediately. NOTE: all query tabs in the same connection share the same transaction. To have independent transactions, you must open a new connection.

The results grid shows a single row with the value '6' under the column 'count(DISTINCT product_li...')'. Below the results grid, the 'Action Output' section lists several previous queries with their status (green checkmark for successful, red X for failed), timestamp, and details. The last successful query (ID 105) is highlighted.

Action	Time	Response
select * from sales	12:52:31	1000 row(s) returned
select count(product_line), gender from sales group by product_line	12:55:37	Error Code: 1055. Expression #2 of SELECT list is not in GROUP BY clause and contains nonaggregated column 'gender' which is not grouped by
select count(product_line), gender from sales order by count(product_line) group by gender	12:57:46	Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'select count(product_line), gender from sales order by count(product_line) group by gender' at line 1
select count(product_line), gender from sales group by gender	12:58:23	2 row(s) returned
select * from sales	12:58:55	1000 row(s) returned
select product_line,count(product_line), gender from sales group by gender	12:59:17	Error Code: 1055. Expression #1 of SELECT list is not in GROUP BY clause and contains nonaggregated column 'product_line' which is not grouped by
select product_line,count(product_line), gender from sales group by product_line	13:00:06	Error Code: 1055. Expression #3 of SELECT list is not in GROUP BY clause and contains nonaggregated column 'product_line' which is not grouped by
select product_line, count(gender) from sales group by product_line order by gender	13:01:06	Error Code: 1055. Expression #1 of ORDER BY clause is not in GROUP BY clause and contains nonaggregated column 'product_line' which is not grouped by
select product_line, count(gender) from sales group by product_line	13:01:16	6 row(s) returned

At the bottom, the 'Object Info' and 'Session' tabs are visible, along with the current table 'sales' and columns listed.

Q What is the most common payment method?

The screenshot shows a MySQL Workbench session titled "Local instance 3306 (salesdatawalmart) - Warning - not supported". The query window contains the following SQL code:

```
1 -- WHAT IS THE MOST COMMON PAYMENT METHOD
2 select * from sales;
3 select payment , count(payment) as cnt from sales group by payment order by cnt desc;
```

The result grid displays the following data:

payment	cnt
Ewallet	345
Cash	344
Credit card	311

The session history shows the following actions:

Action Output	Time	Action
	104 12:58:23	select count(product_line), gender from sales group by gender
	105 12:58:55	select * from sales
	106 12:59:17	select product_line, count(product_line), gender from sales group by gender
	107 13:00:06	select product_line, count(product_line), gender from sales group by product_line
	108 13:01:06	select product_line, count(gender) from sales group by product_line order by gender
	109 13:01:16	select product_line, count(gender) from sales group by product_line
	110 13:01:56	select product_line, count(gender) from sales group by product_line ,gender
	111 13:02:42	select product_line, count(gender) from sales group by product_line ,gender order by count(gender)
	112 13:02:50	SELECT gender, product_line, COUNT(gender) AS total_cnt FROM sales GROUP BY gender, product_line ORDER BY total_cnt
	113 13:04:06	select product_line, count(gender) from sales group by product_line ,gender order by count(gender) desc
	114 13:04:31	select product_line, count(gender) ,gender from sales group by product_line ,gender order by count(gender) desc
	115 13:25:51	select * from sales
	116 13:28:42	select avg(rating) , product_line from sales group by product_line order by avg(rating) desc
	117 13:29:23	select round(avg(rating),2) , product_line from sales group by product_line order by avg(rating) desc
	118 20:50:14	SELECT DISTINCT city FROM sales
	119 20:52:10	SELECT count(DISTINCT product_line) FROM sales
	120 20:53:31	SELECT count(DISTINCT product_line) FROM sales
	121 20:57:23	select payment , count(payment) as cnt from sales group by payment order by cnt desc

Q What is the most selling product line?

The screenshot shows a MySQL Workbench session titled "Local instance 3306 (salesdatawalmart) - Warning - not supported". The query window contains the following SQL code:

```
9
10
11
12
13
14 -- What is the most selling product line
15 SELECT SUM(quantity) as qty, product_line FROM sales
16 GROUP BY product_line
17 ORDER BY qty DESC;
```

The result grid displays the following data:

qty	product_line
971	Electronic accessories
952	Food and beverages
920	Sports and travel
911	Home and lifestyle
902	Fashion accessories
854	Health and beauty

The session history shows the following actions:

Action Output	Time	Action	Response
	113 13:04:06	select product_line, count(gender) from sales group by product_line ,gender order by count(gender) desc	12 row(s) returned
	114 13:04:31	select product_line, count(gender) ,gender from sales group by product_line ,gender order by count(gender) desc	12 row(s) returned
	115 13:25:51	select * from sales	1000 row(s) returned
	116 13:28:42	select avg(rating) , product_line from sales group by product_line order by avg(rating) desc	6 row(s) returned
	117 13:29:23	select round(avg(rating),2) , product_line from sales group by product_line order by avg(rating) desc	6 row(s) returned
	118 20:50:14	SELECT DISTINCT city FROM sales	3 row(s) returned
	119 20:52:10	SELECT count(DISTINCT product_line) FROM sales	1 row(s) returned
	120 20:53:31	SELECT count(DISTINCT product_line) FROM sales	1 row(s) returned
	121 20:57:23	select payment , count(payment) as cnt from sales group by payment order by cnt desc	3 row(s) returned
	122 20:58:07	select payment , count(payment) as cnt from sales group by payment order by cnt desc	3 row(s) returned

Q What is the total revenue by month?

```
-- 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;
```

sum(total)	month_name
109455.5070	March
116291.8680	January
97219.3740	February

Result 10

Action Output	Time	Action	Response
	114	13:04:31	select product_line, count(gender) ,gender from sales group by product_line ,gender order by count(gender) desc
	115	13:25:51	select * from sales
	116	13:28:42	select avg(rating) , product_line from sales group by product_line order by avg(rating) desc
	117	13:29:23	select round(avg(rating),2) , product_line from sales group by product_line order by avg(rating) desc
	118	20:50:14	SELECT DISTINCT city FROM sales
	119	20:52:10	SELECT count(DISTINCT product_line) FROM sales
	120	20:53:31	SELECT count(DISTINCT product_line) FROM sales
	121	20:57:23	select payment , count(payment) as cnt from sales group by payment order by cnt desc
	122	20:58:07	select payment , count(payment) as cnt from sales group by payment order by cnt desc
	123	21:02:04	SELECT SUM(quantity) as qty, product_line FROM sales GROUP BY product_line ORDER BY qty DESC
	124	21:05:31	Select sum(total), month_name from sales group by month_name
			3 row(s) returned

Q What month had the largest COGS?

```
-- What month had the largest COGS?
SELECT
    month_name AS month,
    SUM(cogs) AS cogs
FROM sales
GROUP BY month_name
ORDER BY cogs;
```

month	cogs
February	92589.88
March	104243.34
January	110754.16

Result 11

Action Output	Time	Action
	115	13:25:51
	116	13:28:42
	117	13:29:23
	118	20:50:14
	119	20:52:10
	120	20:53:31
	121	20:57:23
	122	20:58:07
	123	21:02:04
	124	21:05:31
	125	21:07:42

Object Info

Table: sales

Columns:

- invoice_id
- branch
- city
- customer_type
- gender
- product_line
- unit_price
- quantity
- tax_pct
- total
- date
- time
- payment
- cogs

Session

Table: sales

Columns:

- invoice_id
- branch
- city
- customer_type
- gender
- product_line
- unit_price
- quantity
- tax_pct
- total
- date

Query Completed

Q What product line had the largest revenue?

The screenshot shows a SQL IDE interface with two panes. The left pane contains the SQL query:

```
1
2 -- What product line had the largest revenue?
3
4 • SELECT
5   product_line,
6   SUM(total) as total_revenue
7 FROM sales
8 GROUP BY product_line
9 ORDER BY total_revenue DESC;
```

The right pane shows the results of the query:

product_line	total_revenue
Food and beverages	56144.8440
Sports and travel	55122.8265
Electronic accessories	54337.5315
Fashion accessories	54305.8950
Home and lifestyle	53861.9130
Health and beauty	49193.7390

Below the results, there is a table of recent actions and a detailed view of the 'sales' table columns.

Q What is the city with the largest revenue?

The screenshot shows a SQL IDE interface with two panes. The left pane contains the SQL query:

```
10
11
12 -- What is the city with the largest revenue?
13 • SELECT
14   branch,
15   city,
16   SUM(total) AS total_revenue
17 FROM sales
18 GROUP BY city, branch
19 ORDER BY total_revenue;
```

The right pane shows the results of the query:

branch	city	total_revenue
B	Mandalay	106197.6720
A	Yangon	106200.3705
C	Naypyitaw	110568.7065

Below the results, there is a table of recent actions and a detailed view of the 'sales' table columns.

Action Output	Time	Action	Response	Object Info	Session	Action Output	Time	Action
✓ 116	13:28:42	select avg(rating) , product_line from sales group by product_line order by avg(rating) desc	6 row(s) returned	Table: sales		✓ 117	13:29:23	select round(avg(rating),2) , product_line from sales group by product_line order by avg(rating) desc
✓ 117	13:29:23	select round(avg(rating),2) , product_line from sales group by product_line order by avg(rating) desc	6 row(s) returned	Columns:		✓ 118	20:50:14	SELECT DISTINCT city FROM sales
✓ 118	20:50:14	SELECT DISTINCT city FROM sales	3 row(s) returned	invoice_id	varchar(30)	✓ 119	20:52:10	SELECT count(DISTINCT product_line) FROM sales
✓ 119	20:52:10	SELECT count(DISTINCT product_line) FROM sales	1 row(s) returned	branch	varchar(5)	✓ 120	20:53:31	SELECT count(DISTINCT product_line) FROM sales
✓ 120	20:53:31	SELECT count(DISTINCT product_line) FROM sales	1 row(s) returned	city	varchar(30)	✓ 121	20:57:23	select payment , count(payment) as cnt from sales group by payment order by cnt desc
✓ 121	20:57:23	select payment , count(payment) as cnt from sales group by payment order by cnt desc	3 row(s) returned	customer_type	varchar(30)	✓ 122	20:58:07	select payment , count(payment) as cnt from sales group by payment order by cnt desc
✓ 122	20:58:07	select payment , count(payment) as cnt from sales group by payment order by cnt desc	3 row(s) returned	gender	varchar(30)	✓ 123	21:02:04	SELECT SUM(quantity) as qty, product_line FROM sales GROUP BY product_line ORDER BY qty DESC
✓ 123	21:02:04	SELECT SUM(quantity) as qty, product_line FROM sales GROUP BY product_line ORDER BY qty DESC	6 row(s) returned	product_line	varchar(100)	✓ 124	21:05:31	Select sum(total), month_name from sales group by month_name
✓ 124	21:05:31	Select sum(total), month_name from sales group by month_name	3 row(s) returned	unit_price	decimal(10,2)	✓ 125	21:07:42	SELECT month_name AS month, SUM(cogs) AS cogs FROM sales GROUP BY month_name ORDER BY cogs
✓ 125	21:07:42	SELECT month_name AS month, SUM(cogs) AS cogs FROM sales GROUP BY month_name ORDER BY cogs	3 row(s) returned	quantity	int	✓ 126	21:11:42	SELECT product_line, SUM(total) as total_revenue FROM sales GROUP BY product_line ORDER BY total_revenue DESC
✓ 126	21:11:42	SELECT product_line, SUM(total) as total_revenue FROM sales GROUP BY product_line ORDER BY total_revenue DESC	6 row(s) returned	tax_pct	float	✓ 127	21:12:22	SELECT branch, city, SUM(total) AS total_revenue FROM sales GROUP BY city, branch ORDER BY total_revenue
				total	decimal(12,4)			
				date	datetime			

Query Completed

Q What product line had the largest VAT?

Q Which branch sold more products than average product sold?

-- What product line had the largest VAT?

```
1 select product_line ,avg(tax_pct) as vat from sales group by product_line order by vat desc;
```

15

16

17

18

-- Which branch sold more products than average product sold?

```
19 select * from sales;
```

```
20 select branch , sum(quantity) from sales group by branch having sum(quantity) > (select avg(quantity) from sales);
```

Result Grid

product_line	avg_tax
Home and lifestyle	16.030331237986683
Sports and travel	15.812629552131677
Health and beauty	15.411572383030466
Food and beverages	15.365310291449228
Electronic accessories	15.220597051872927
Fashion accessories	14.528061806485894

Result 14 Result 15

Result Grid

bisum(quantity)
A 1859
C 1831
B 1820

Result 16

Action Output

Time	Action	Response
119 20:52:10	SELECT count(DISTINCT product_line) FROM sales	1 row(s) returned
120 20:53:31	SELECT count(DISTINCT product_line) FROM sales	1 row(s) returned
121 20:57:23	select payment , count(payment) as cnt from sales group by payment order by cnt desc	3 row(s) returned
122 20:58:07	select payment , count(payment) as cnt from sales group by payment order by cnt desc	3 row(s) returned
123 21:02:04	SELECT SUM(quantity) as qty, product_line FROM sales GROUP BY product_line ORDER BY qty DESC	6 row(s) returned
124 21:05:31	Select sum(total), month_name from sales group by month_name	3 row(s) returned
125 21:07:42	SELECT month_name AS month, SUM(cogs) AS cogs FROM sales GROUP BY month_name ORDER BY cogs	3 row(s) returned
126 21:11:42	SELECT product_line, SUM(total) as total_revenue FROM sales GROUP BY product_line ORDER BY total_revenue DESC	6 row(s) returned
127 21:12:22	SELECT branch, city, SUM(total) AS total_revenue FROM sales GROUP BY city, branch ORDER BY total_revenue	3 row(s) returned
128 21:15:46	select product_line ,avg(tax_pct) as vat from sales group by product_line order by vat desc	6 row(s) returned
129 21:15:46	SELECT product_line, AVG(tax_pct) as avg_tax FROM sales GROUP BY product_line ORDER BY avg_tax DESC	6 row(s) returned

Object Info

Table: sales

Columns:

- invoice_id varchar(30) PK
- branch varchar(5)
- city varchar(30)
- customer_type varchar(30)
- gender varchar(30)
- product_line varchar(100)
- unit_price decimal(10,2)
- quantity int
- tax_pct float
- total decimal(12,4)
- date datetime

Action Output

Time	Action
120 20:53:31	SELECT count(DISTINCT product_line) FROM sales
121 20:57:23	select payment , count(payment) as cnt from sales group by payment order by cnt desc
122 20:58:07	select payment , count(payment) as cnt from sales group by payment order by cnt desc
123 21:02:04	SELECT SUM(quantity) as qty, product_line FROM sales GROUP BY product_line ORDER BY qty DESC
124 21:05:31	Select sum(total), month_name from sales group by month_name
125 21:07:42	SELECT month_name AS month, SUM(cogs) AS cogs FROM sales GROUP BY month_name ORDER BY cogs
126 21:11:42	SELECT product_line, SUM(total) as total_revenue FROM sales GROUP BY product_line ORDER BY total_revenue DESC
127 21:12:22	SELECT branch, city, SUM(total) AS total_revenue FROM sales GROUP BY city, branch ORDER BY total_revenue
128 21:15:46	select product_line ,avg(tax_pct) as vat from sales group by product_line order by vat desc
129 21:15:46	SELECT product_line, AVG(tax_pct) as avg_tax FROM sales GROUP BY product_line ORDER BY avg_tax DESC
130 21:17:46	select branch , sum(quantity) from sales group by branch having sum(quantity) > (select avg(quantity) from sales)

Session

Query Completed

Q Fetch each product line and add a column to those product

Q line showing "Good", "Bad". Good if its greater than average sales

The screenshot shows the MySQL Workbench interface with two queries and their results.

Query 1: Fetch each product line and add a column to those product line showing "Good", "Bad". Good if its greater than average sales

```
32
33
34
35 -- Fetch each product line and add a column to those product
36 -- line showing "Good", "Bad". Good if its greater than average sales
37
38 • SELECT
39     AVG(quantity) AS avg_qnty
40     FROM sales;
41
42 • SELECT
43     product_line,
44     CASE
45         WHEN AVG(quantity) > 6 THEN "Good"
46         ELSE "Bad"
47     END AS remark
48     FROM sales
49     GROUP BY product_line;
```

Result 17 (Table):

product_line	remark
Food and beverages	Bad
Health and beauty	Bad
Sports and travel	Bad
Fashion accessories	Bad
Home and lifestyle	Bad
Electronic accessories	Bad

Query 2: What is the most common product line by gender

```
1 -- What is the most common product line by gender
2 • SELECT
3     gender,
4     product_line,
5     COUNT(gender) AS total_cnt
6     FROM sales
7     GROUP BY gender, product_line
8     ORDER BY total_cnt DESC;
```

Result 13 (Table):

gender	product_line	total_cnt
Female	Fashion accessories	96
Female	Food and beverages	90
Male	Health and beauty	88
Female	Sports and travel	88
Male	Electronic accessories	86
Male	Food and beverages	84
Female	Electronic accessories	84
Male	Fashion accessories	82
Male	Home and lifestyle	81
Female	Home and lifestyle	79
Male	Sports and travel	78
Female	Health and beauty	64

Action Output:

Time	Action
129 21:15:46	SELECT product_line, AVG(tax_pct) as avg_tax FROM sales GROUP BY product_line ORDER BY avg_tax DESC
130 21:17:46	select branch , sum(quantity) from sales group by branch having sum(quantity) > (select avg(quantity) from sales)
131 21:19:48	SELECT AVG(quantity) AS avg_qnty FROM sales
132 21:19:48	SELECT product_line, CASE WHEN AVG(quantity) > 6 THEN "Good" ELSE "Bad" END AS remark FROM sales GROUP...
133 21:22:00	SELECT gender, product_line, COUNT(gender) AS total_cnt FROM sales GROUP BY gender, product_line ORDER BY total...

Q What is the average rating of each product line

The screenshot shows a database interface with a sidebar on the left and a main query and results area on the right.

Left Sidebar:

- Filter objects search bar
- Navigation tree:
 - > ecommerce
 - > salesdatawalmart
 - Tables
 - sales
 - Columns
 - invoice_id
 - branch
 - city
 - customer_ty...
 - gender
 - product_line
 - unit_price
 - quantity
 - tax_pct
 - total
 - date
 - time
 - payment
 - cogs

Object Info: Table: sales

Columns:

Column	Type
invoice_id	varchar(30) PK
branch	varchar(5)
city	varchar(30)
customer_type	varchar(30)
gender	varchar(30)
product_line	varchar(100)

Session:

Query Area:

```
-- What is the average rating of each product line
SELECT
 20    ROUND(AVG(rating), 2) as avg_rating,
21      product_line
22  FROM sales
23 GROUP BY product_line
24 ORDER BY avg_rating DESC;
```

Execution status: 100% | 26:24

Result Grid:

avg_rating	product_line
7.11	Food and beverages
7.03	Fashion accessories
7	Health and beauty
6.92	Sports and travel
6.92	Electronic accessories
6.84	Home and lifestyle

Action Output:

Time	Action
130 21:17:46	select branch , sum(quantity) from sales group by branch having sum(quantity) > (select avg(quantity) from sales)

Response: 3 row(s) returned

Right Sidebar:

 - Result Grid
 - Form Editor
 - Field Types
 - Query Stats
 - Execution Plan

