

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?

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, showing various columns like invoice_id, branch, city, customer_type, gender, product_line, unit_price, quantity, tax_pct, total, date, time, payment, and cogs. A context menu is open over the 'product_line' column, with the option 'Toggle autocommit mode...' highlighted.

The main pane contains the following SQL query:

```
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 'Toggle autocommit mode...' option is displayed, stating: "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 the output of the query:

| count(DISTINCT product_li...) |
|-------------------------------|
| 6 |

The status bar at the bottom indicates "Result 6" and "Read Only".

The "Action Output" section shows a history of previous queries:

| 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' 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 |

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;
```

Result Grid

| 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 | 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 |
| | 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 |

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;
```

Result Grid

| month | cogs |
|----------|-----------|
| February | 92589.88 |
| March | 104243.34 |
| January | 110754.16 |

Result 11

| Action Output | Time | Action |
|---------------|--------------|---|
| | 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 |
| | 125 21:07:42 | SELECT month_name AS month, SUM(cogs) AS cogs FROM sales GROUP BY month_name ORDER BY cogs |

Query Completed

Q What product line had the largest revenue?

Local instance 3306 (salesdatawalmart) - Warning - not supported | Migration | Administration | Schemas

SQL_queries | SQL File 4* | SQL File 5* | SQL File 6* | SQL File 7* | SQL File 8* | SQL File 9*

Filter objects

```
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;
10
11
```

Result Grid | Filter Rows: Search | Export:

| 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 |

Result 1

Q What is the city with the largest revenue?

Local instance 3306 (pizzassales) - Warning - not supported | Local instance 3306 (salesdatawalmart) - Warning - not supported | Migration | Administration | Schemas

SQL_queries | SQL File 4* | SQL File 5* | SQL File 6* | SQL File 7* | SQL File 8* | SQL File 9*

Filter objects

```
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;
20
```

Result Grid | Filter Rows: Search | Export:

| b city | total_revenue |
|-------------|---------------|
| B Mandalay | 106197.6720 |
| A Yangon | 106200.3705 |
| C Naypyitaw | 110568.7065 |

Result 2

| Action Output | Time | Action | Response | Table: sales | 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 | Columns: | | ✓ 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 | invoice_id | varchar(30) | ✓ 118 | 20:50:14 | SELECT DISTINCT city FROM sales |
| ✓ 118 | 20:50:14 | SELECT DISTINCT city FROM sales | 3 row(s) returned | branch | varchar(5) | ✓ 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 | city | varchar(30) | ✓ 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 | customer_type | 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 | gender | 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 | product_line | varchar(100) | ✓ 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 | unit_price | decimal(10,2) | ✓ 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 | quantity | int | ✓ 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 | tax_pct | float | ✓ 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 | total | decimal(12,4) | ✓ 127 | 21:12:22 | SELECT branch, city, SUM(total) AS total_revenue FROM sales GROUP BY city, branch ORDER BY total_revenue |
| | | | | date | datetime | | | |

Query Completed

Q What product line had the largest VAT?

```
3306 (pizzassales) - Warning - not supported | Local instance 3306 (salesdatawalmart) - Warning - not supported | Migration | Home | Local instance 3306 (pizzassales) - Warning - not supported | Local instance 3306 (salesdatawalmart) - Warning - not supported | Migration |  
SQL_queries | SQL File 4* | SQL File 5* | SQL File 6* | SQL File 7* | SQL File 8* | SQL File 9* | Administration | Schemas |  
File | Database | Table | Column | Index | Constraint | Procedure | Function | Trigger | View | Filter objects |  
Don't Limit |  
1 -- What product line had the largest VAT?  
2  
3  
4 * select product_line ,avg(tax_pct) as vat from sales group by product_line order by vat desc;  
5 * SELECT  
6 *   product_line,  
7 *     AVG(tax_pct) as avg_tax  
8 * FROM sales  
9 * GROUP BY product_line  
10 * ORDER BY avg_tax DESC;  
11  
12  
100% 1:12 | Result Grid | Filter Rows: Search | Export:  
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 |
```

```
3306 (pizzassales) - Warning - not supported | Local instance 3306 (salesdatawalmart) - Warning - not supported | Migration | Home | Local instance 3306 (pizzassales) - Warning - not supported | Local instance 3306 (salesdatawalmart) - Warning - not supported | Migration |  
SQL_queries | SQL File 4* | SQL File 5* | SQL File 6* | SQL File 7* | SQL File 8* | SQL File 9* | Administration | Schemas |  
File | Database | Table | Column | Index | Constraint | Procedure | Function | Trigger | View | Filter objects |  
Don't Limit |  
15  
16  
17  
18  
19 -- Which branch sold more products than average product sold?  
20  
21 * select * from sales;  
22 * select branch , sum(quantity) from sales group by branch having sum(quantity) > (select avg(q  
23 *  
24  
100% 2:23 | Result Grid | Filter Rows: Search | Export:  
branch | sum(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 |

| Action Output | Time | Action | Table: sales | Session |
|---------------|----------|---|---------------|-------------------|
| ✓ 120 | 20:53:31 | SELECT count(DISTINCT product_line) FROM sales | Columns: | |
| ✓ 121 | 20:57:23 | select payment , count(payment) as cnt from sales group by payment order by cnt desc | invoice_id | varchar(30) PK |
| ✓ 122 | 20:58:07 | select payment , count(payment) as cnt from sales group by payment order by cnt desc | branch | varchar(5) |
| ✓ 123 | 21:02:04 | SELECT SUM(quantity) as qty, product_line FROM sales GROUP BY product_line ORDER BY qty DESC | city | varchar(30) |
| ✓ 124 | 21:05:31 | Select sum(total), month_name from sales group by month_name | customer_type | varchar(30) |
| ✓ 125 | 21:07:42 | SELECT month_name AS month, SUM(cogs) AS cogs FROM sales GROUP BY month_name ORDER BY cogs | gender | varchar(30) |
| ✓ 126 | 21:11:42 | SELECT product_line, SUM(total) as total_revenue FROM sales GROUP BY product_line ORDER BY total_revenue DESC | product_line | varchar(100) |
| ✓ 127 | 21:12:22 | SELECT branch, city, SUM(total) AS total_revenue FROM sales GROUP BY city, branch ORDER BY total_revenue | unit_price | decimal(10,2) |
| ✓ 128 | 21:15:46 | select product_line ,avg(tax_pct) as vat from sales group by product_line order by vat desc | quantity | int |
| ✓ 129 | 21:15:46 | SELECT product_line, AVG(tax_pct) as avg_tax FROM sales GROUP BY product_line ORDER BY avg_tax DESC | tax_pct | float |
| ✓ 130 | 21:17:46 | select branch , sum(quantity) from sales group by branch having sum(quantity) > (select avg(quantity) from sales) | total | decimal(12,4) |
| | | | date | datetime |

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 a SQL development environment with two main panes. The left pane contains a script editor with two queries. The first query calculates the average quantity per product line. The second query uses a CASE statement to categorize each product line as 'Good' if the average quantity is greater than 6, and 'Bad' otherwise. The right pane shows the results of a third query, which finds the most common product line by gender.

Script Editor (Left):

```
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
37 -- average sales
38
39 • SELECT
40     AVG(quantity) AS avg_qnty
41     FROM sales;
42
43 • SELECT
44     product_line,
45     CASE
46         WHEN AVG(quantity) > 6 THEN "Good"
47         ELSE "Bad"
48     END AS remark
49     FROM sales
50     GROUP BY product_line;
```

Object Explorer (Center):

- SCHEMAS
 - ecommerce
 - salesdatawalmart
 - Tables
 - sales
 - Columns
 - invoice_id
 - branch
 - city
 - customer_ty...
 - gender
 - product_line
 - unit_price
 - quantity
 - tax_pct
 - total
 - date
 - time
 - payment
 - cogs

Results Grid (Right):

| 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 (Bottom):

| 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 and a main query editor area.

Sidebar:

- Filter objects search bar
- Object 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

Table Information:

 - 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)

Query Editor:

```
-- What is 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;
```

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 | Response |
|--------------|---|-------------------|
| 130 21:17:46 | select branch , sum(quantity) from sales group by branch having sum(quantity) > (select avg(quantity) from sales) | 3 row(s) returned |

