

# **SQL-Mongo Project – Supermarket Sales**

## Contents

Relational Data Model .....	3
Assumptions/Notes About Data Entities and Relationships .....	3
Entity-Relationship Diagram .....	4
Physical MySQL Database .....	5
Assumptions/Notes About Data Set .....	5
Screen shot of Physical Database objects .....	6
Data in the Database .....	9
SQL Queries.....	10
SQL Query 1 .....	10
Question (Q1) .....	10
Notes/Comments About SQL Query and Results (Include # of Rows in Result) .....	10
Translation .....	10
Screen Shot of SQL Query and Results .....	11
SQL Query 2 .....	12
Question (Q.2) .....	12
Notes/Comments About SQL Query and Results (Include # of Rows in Result) .....	12
Translation .....	12
Screen Shot of SQL Query and Results .....	14
SQL Query 3 .....	15
Question (Q.3) .....	15
Notes/Comments About SQL Query and Results (Include # of Rows in Result) .....	15
Translation .....	15
Screen Shot of SQL Query and Results .....	17
SQL Query 4 .....	18
Question (Q.4) .....	18
Notes/Comments About SQL Query and Results (Include # of Rows in Result) .....	18
Translation .....	18
Screen Shot of SQL Query and Results .....	19
SQL Query 5 .....	20
Question (Q.5) .....	20
Notes/Comments About SQL Query and Results (Include # of Rows in Result) .....	20
Translation .....	20

Screen Shot of SQL Query and Results .....	22
Data Review for MongoDB .....	23
Assumptions/Notes About Data Collections, Attributes and Relationships between Collections .....	23
Physical Mongo Database.....	24
Assumptions/Notes About Data Set.....	24
Screen shot of Physical Database objects (Database, Collections and Attributes).....	25
Data in the Database .....	27
MongoDB Queries/Code.....	28
Mongo Query 1.....	28
Question (Q.1) .....	28
Notes/Comments About MongoDB Query/Code and Results (Include # of Documents in Result)...	28
Translation .....	28
Screen Shot of MongoDB Query/Code and Results .....	29
Mongo Query 2.....	30
Question (Q.2) .....	30
Notes/Comments About MongoDB Query/Code and Results (Include # of Documents in Result)...	30
Translation .....	30
Screen Shot of MongoDB Query/Code and Results .....	31
Mongo Query 3.....	32
Question (Q.4) .....	32
Notes/Comments About MongoDB Query/Code and Results (Include # of Documents in Result)...	32
Translation .....	32
Screen Shot of MongoDB Query/Code and Results .....	33

## Relational Data Model

### Assumptions/Notes About Data Entities and Relationships

Include assumptions about data entities and their relationships with each other.

#### Data Requirements:

1. Database needs to store details about each invoice being generated through transactions.
2. Database needs to store details about their branches.
3. Database needs to store every Invoice-ID and is assigned to one Unique Customer-ID.
4. In Every Product Line, unique unit price represents one unique Product –ID.
5. Database needs to store details about the customer (Customer ID, Gender)
6. Identify the type of the customer (Normal, Member) for each order.
7. Database needs to store details about the product line and identify each type of product line the product belongs to.
8. Database needs to identify the tax amount for each invoice total and store it with Invoice ID
9. Database needs to store date and time for each unique transaction.
10. Company needs to identify the payment type been used for each unique transaction and store it in the database.
11. Database needs to store the cost of goods sold for per invoice.
12. Company needs to recognize the gross margin of each transaction and stored in database.
13. Database needs to store satisfaction rating for each customer.

#### Data entities and their relationships:

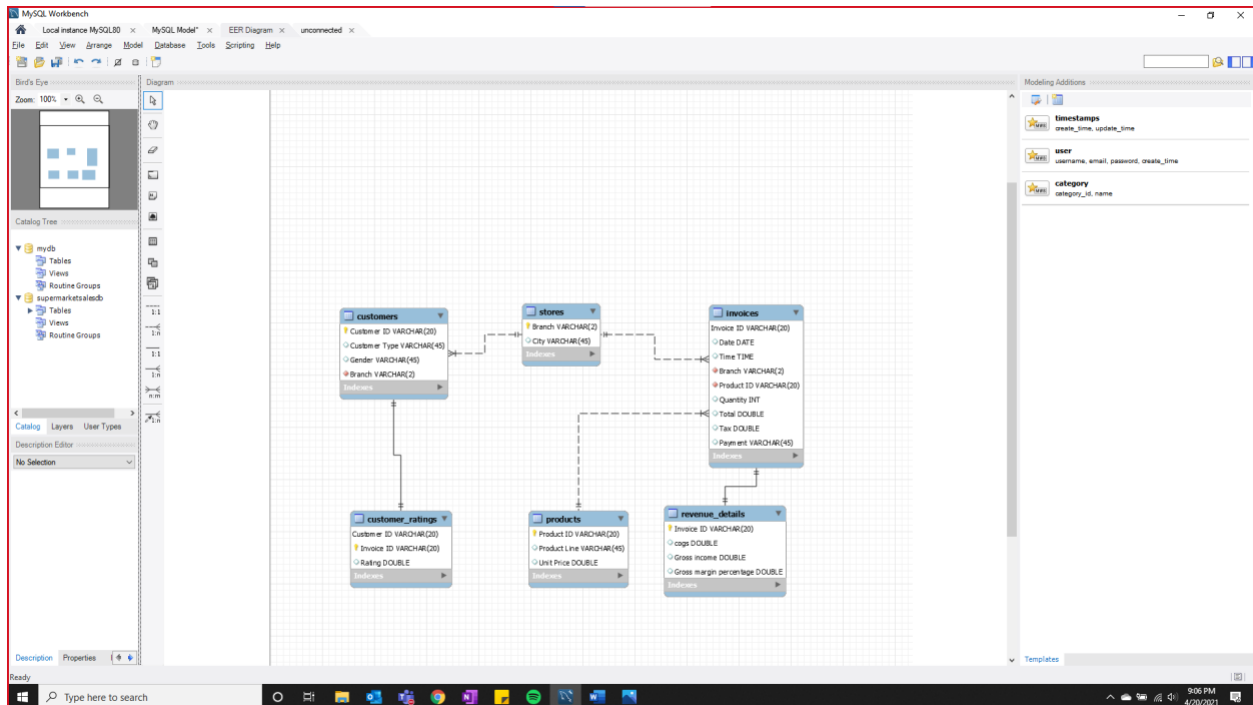
1. Stores have a 1:M non-identifying relationship with customers.
2. Stores have a 1:M non-identifying relationship with invoices.
3. Customers have 1:1 identifying relationship with customer\_ratings.
4. Products have a 1:M non-identifying relationship with invoices.
5. Revenue\_details have 1:1 identifying relationship with invoices.

#### Include reasons why the data model is in 3NF:

1. Every entity has unique and clear name.
2. Every attribute is directly related to the entity.

3. Every attribute has clearly understood name and is unique inside the entity.
4. Every attribute only holds one value that is indivisible.
5. Every entity has unique identifier (primary key/composite primary key).
6. All non-key columns have functional dependency on the entire primary key and only on the entire primary key.

## Entity-Relationship Diagram



## Physical MySQL Database

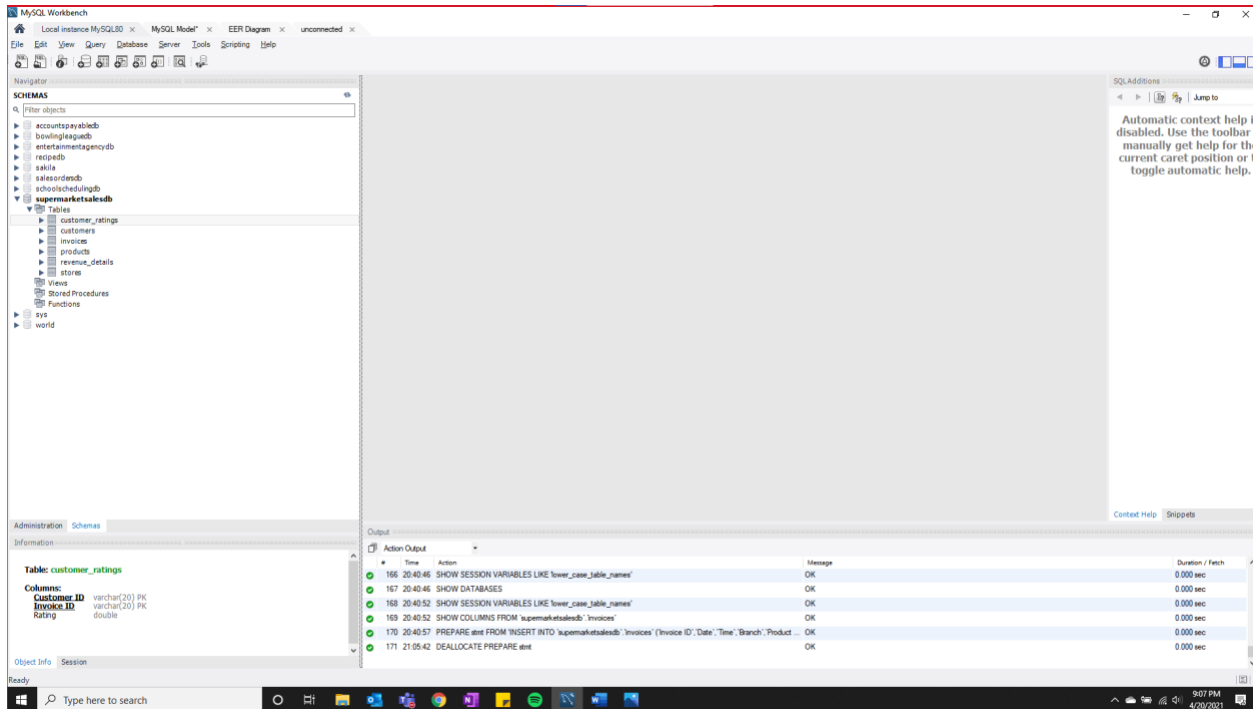
### Assumptions/Notes About Data Set

Include any assumptions made about data such as empty fields, sparse data, bad data, etc.

#### **Assumptions:**

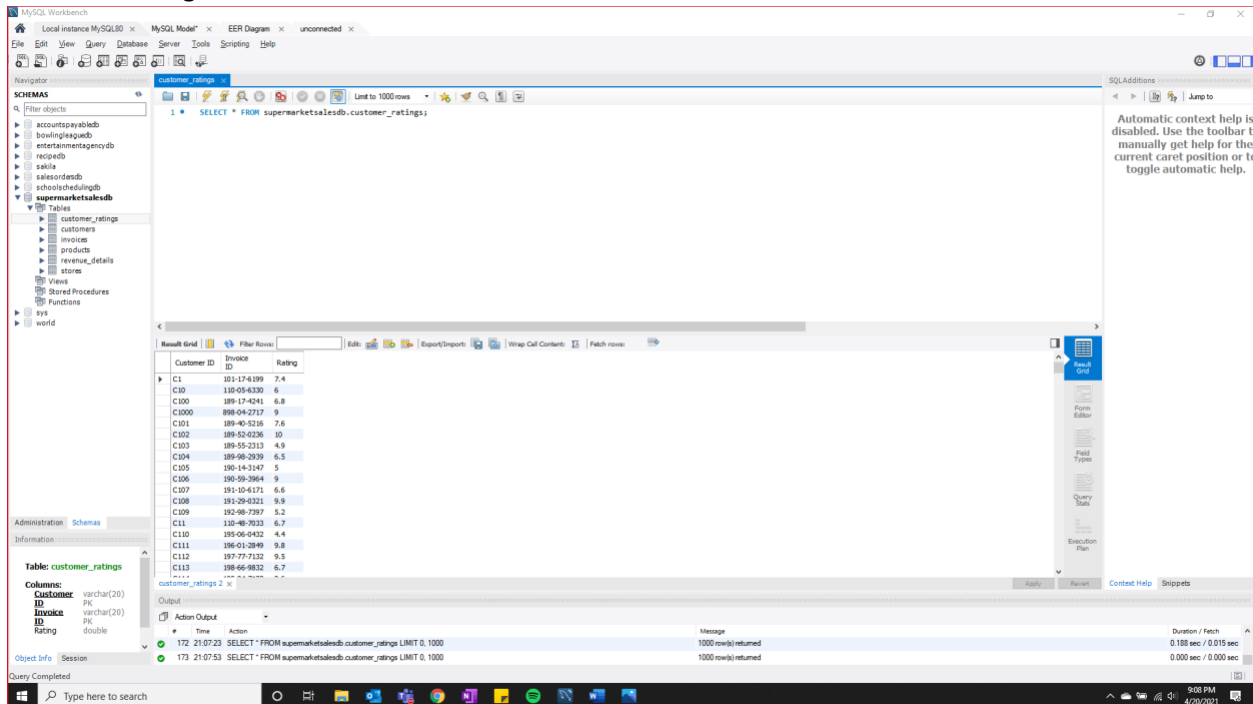
1. There is a unique customer for every invoice.
2. There is a unique invoice for every transaction.
3. A store can have many customers.
4. A store can have many invoices.
5. A product can have many invoices.
6. An invoice has one revenue detail.
7. A customer has one customer rating.
8. The value of tax is unrelated to total.
9. The value of total is unrelated to quantity and unit price.
10. The value of cogs is independent.
11. Gross margin percentage is an independent attribute.
12. Gross income is an independent attribute.

## Screen shot of Physical Database objects

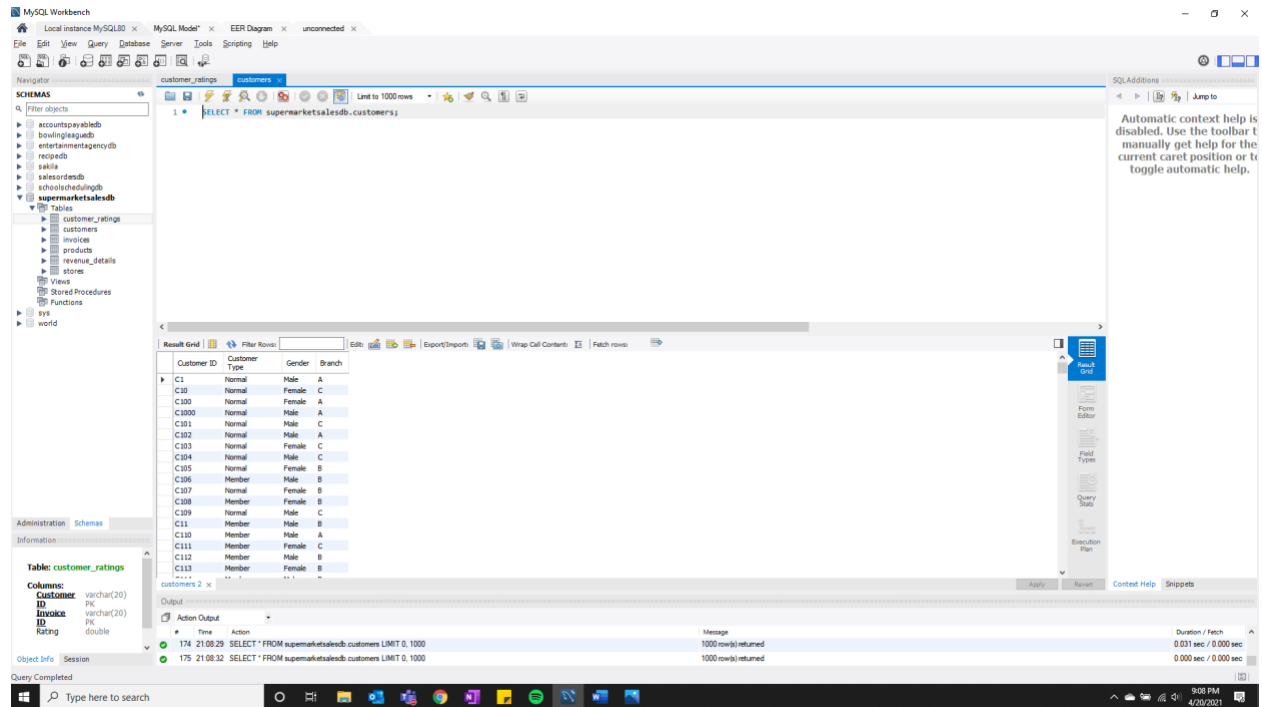


To check row counts of data loaded into each table:

customer ratings:



## customers:

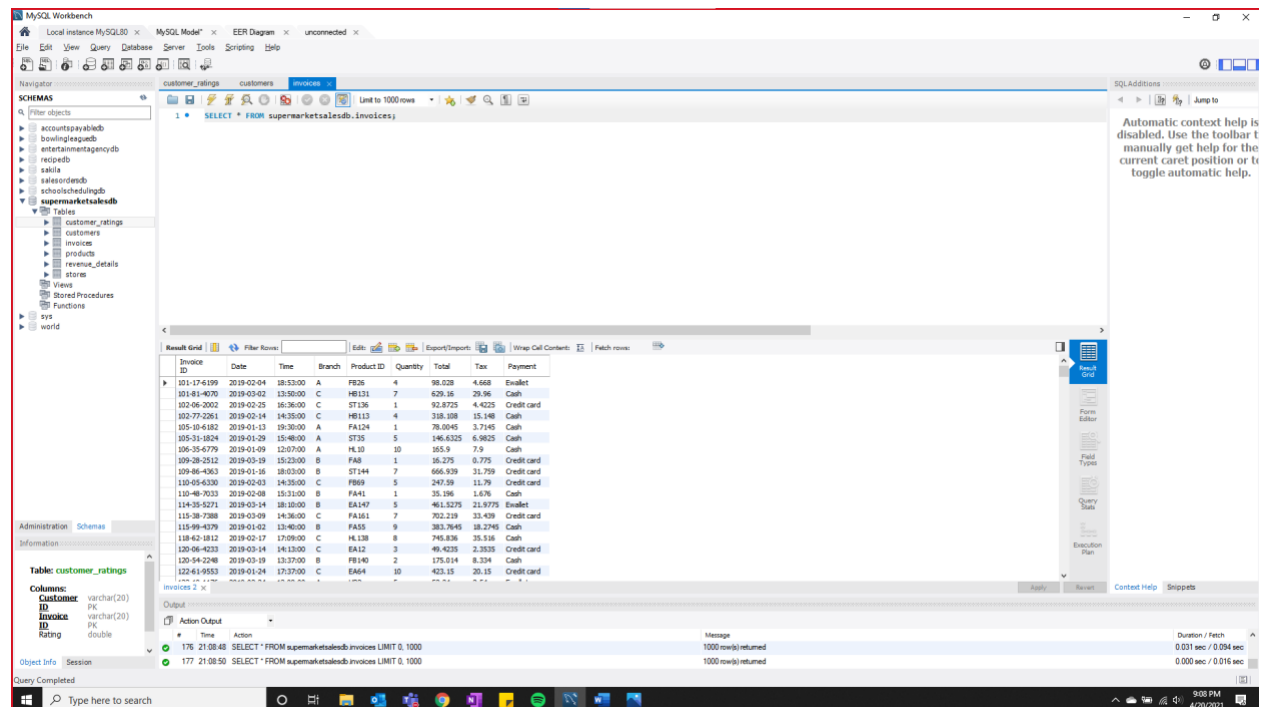


MySQL Workbench interface showing the 'customers' table in the 'supermarket\_sales' database. The table structure is as follows:

Customer ID	Customer Type	Gender	Branch
C1	Normal	Male	A
C10	Normal	Female	C
C100	Normal	Female	A
C1000	Normal	Male	A
C101	Normal	Male	C
C102	Normal	Male	C
C103	Normal	Female	C
C104	Normal	Male	C
C105	Normal	Female	B
C106	Member	Male	B
C107	Normal	Female	B
C108	Member	Female	B
C109	Normal	Male	C
C11	Member	Male	B
C110	Member	Male	A
C111	Member	Female	C
C112	Member	Male	B
C113	Member	Female	B

The query 'SELECT \* FROM supermarket\_salesdb.customers;' is executed, showing 175 rows. The output shows a list of customers with their IDs, types, genders, and branches.

## invoices:



MySQL Workbench interface showing the 'invoices' table in the 'supermarket\_sales' database. The table structure is as follows:

Invoice ID	Date	Time	Branch	Product ID	Quantity	Total	Tax	Payment
101-17-6199	2019-02-04	18:53:00	A	FE26	4	98.028	4.668	Enaliet
101-81-4070	2019-03-02	13:50:00	C	HB131	7	629.16	29.96	Cash
102-09-2002	2019-02-25	16:36:00	C	ST136	1	92.8725	4.4225	Credit card
102-77-0261	2019-02-14	14:35:00	C	HB113	4	318.198	15.148	Cash
105-10-6182	2019-01-13	19:30:00	A	FA124	1	78.0045	3.7145	Cash
105-31-1824	2019-01-29	15:48:00	A	ST35	5	146.6325	6.9825	Cash
106-16-6779	2019-01-09	12:07:00	A	HL30	10	165.9	7.9	Cash
109-28-2512	2019-03-19	15:23:00	B	FA8	1	16.275	0.775	Credit card
109-86-4363	2019-01-16	18:03:00	B	ST144	7	666.939	31.799	Credit card
110-05-6330	2019-02-03	14:35:00	C	FA88	5	247.59	11.79	Credit card
110-48-7033	2019-02-08	15:31:00	B	FA41	1	35.196	1.676	Cash
114-39-9271	2019-03-14	18:10:00	B	EA147	5	461.5275	21.9775	Enaliet
115-39-7988	2019-03-09	14:36:00	C	FA161	7	702.219	33.409	Credit card
115-99-4379	2019-01-02	13:40:00	B	FA35	9	383.7648	18.2748	Cash
118-62-1812	2019-02-17	17:09:00	C	HL138	8	745.836	35.516	Cash
120-06-4233	2019-03-14	14:13:00	C	EA12	3	49.4235	2.3035	Credit card
120-54-2248	2019-03-19	13:37:00	B	FE140	2	175.014	8.324	Cash
122-61-9553	2019-01-24	17:37:00	C	EA64	10	423.15	20.15	Credit card

The query 'SELECT \* FROM supermarket\_salesdb.invoices;' is executed, showing 177 rows. The output shows a list of invoices with their IDs, dates, times, branches, product IDs, quantities, totals, taxes, and payment methods.



## products:

MySQL Workbench interface showing the 'products' table in the 'supermarket\_sales' database. The table structure is as follows:

Product ID	Product Line	Unit	Price
EA1	Electronic accessories	35.56	
EA10	Electronic accessories	14.96	
EA100	Electronic accessories	62.48	
EA101	Electronic accessories	63.22	
EA102	Electronic accessories	64.44	
EA103	Electronic accessories	64.95	
EA104	Electronic accessories	65.94	
EA105	Electronic accessories	66.06	
EA106	Electronic accessories	66.25	
EA107	Electronic accessories	66.65	
EA108	Electronic accessories	68.84	
EA109	Electronic accessories	69.58	
EA110	Electronic accessories	71.89	
EA111	Electronic accessories	71.95	
EA112	Electronic accessories	72.13	
EA113	Electronic accessories	72.17	
EA114	Electronic accessories	72.2	

The query results show 14 rows of electronic accessories. The query executed was: `SELECT * FROM supermarket_sales.products`. The output shows 993 rows returned.

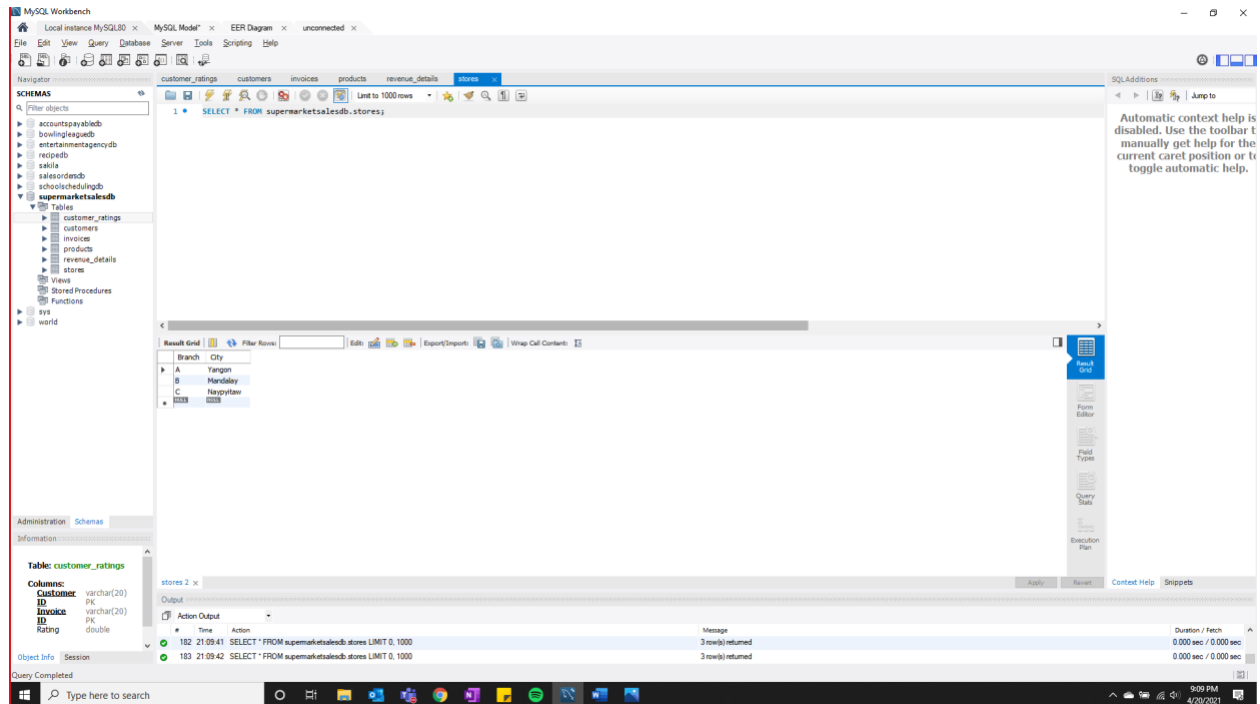
## revenue details:

MySQL Workbench interface showing the 'revenue\_details' table in the 'supermarket\_sales' database. The table structure is as follows:

Invoice ID	cogs	Gross income	Gross margin percentage
101-174199	93.36	4.668	4.761904762
101-814070	599.2	29.96	4.761904762
102-062012	88.45	4.4225	4.761904762
102-772261	302.96	15.148	4.761904762
105-104182	74.29	3.7145	4.761904762
105-31424	128.65	6.9025	4.761904762
106-354779	158	7.9	4.761904762
109-282512	15.5	0.775	4.761904762
109-864363	635.18	31.739	4.761904762
110-014330	235.8	11.79	4.761904762
110-487033	33.52	1.676	4.761904762
114-355271	439.35	21.9775	4.761904762
115-387388	660.78	33.439	4.761904762
115-994379	365.49	18.2745	4.761904762
118-621812	730.32	35.516	4.761904762
120-064233	47.07	2.3535	4.761904762
120-542248	166.68	8.334	4.761904762
122-619553	403	20.15	4.761904762

The query results show 180 rows of revenue details. The query executed was: `SELECT * FROM supermarket_sales.revenue_details`. The output shows 1000 rows returned.

stores:



## Data in the Database

Table Name	Primary Key	Foreign Key	# of Rows in Table
customer_ratings	Customer ID, Invoice ID		1000
customers	Customer ID	Branch	1000
invoices	Invoice ID	Branch, Product ID	1000
products	Product ID		993
revenue_details	Invoice ID		1000
stores	Branch		3

## SQL Queries

### SQL Query 1

#### Question (Q1)

Some retailers believe that there is more money to be made in selling fashion accessories to men than sports and travel to women. Is this true?

#### Notes/Comments About SQL Query and Results (Include # of Rows in Result)

##### Explanation:

First, we find the sum of Gross income for 2 different categories, i.e., 'fashion accessories sold to men' and 'sports and travel sold to women'. We then compare its values to see which is greater.

##### Assumptions:

We assume the value of 'gross income' denotes the money made on a purchase. Hence, summing the value of gross income for the 2 categories 'fashion accessories sold to men' and 'sports and travel sold to women' can be used to compare which category makes the most money.

##### # of Rows in Result: 1

##### Conclusion:

From the result of the query mentioned below, we can see that the sum of gross income for the category 'fashion accessories sold to men' is \$1136.595 compared 'sports and travel sold to women' which is \$1360.70. Hence the belief of some retailers that there is more money to be made in selling fashion accessories to men than sports and travel to women is FALSE.

#### Translation

##### Translation:

Select sum of Gross Income for the case when gender is male and Product line is Fashion Accessories and for the case when gender is female and Product line is Sports and Travel from Customer table joined with Customer Ratings table where Customer ID in Customer table matches Customer ID in Customer Ratings table joined with revenue details table where Invoice ID in revenue details table match with invoice in customer ratings table joined with invoices table where invoice ID in invoices table matches with invoice id in revenue details table joined with products table where product id in products table matches with product id in invoices table.

##### Cleanup:

```
select sum(case when c.gender = 'Male' and p.'Product line' = 'Fashion Accessories' then rd.'Gross income'
end ) as Men_Fashion_Accessories_Profit, sum(case when c.gender = 'Female' and p.'Product line' =
'Sports and Travel' then rd.'Gross income' end) as Women_Sports_and_Travel_Profit from Customers as
c join customer_ratings as cr on c.'Customer ID' = cr.'Customer ID' join revenue_details as rd on cr.'Invoice
ID' = rd.'Invoice ID' join invoices as i on rd.'Invoice ID' = i.'Invoice ID' join products as p on i.'Product ID' =
p.'Product ID'
```

##### Query:

```

select sum(case when c.gender = 'Male' and p.'Product line' = 'Fashion Accessories'
then rd.'Gross income' end ) as Men_Fashion_Accessories_Profit,
sum(case when c.gender = 'Female' and p.'Product line' = 'Sports and Travel'
then rd.'Gross income' end) as Women_Sports_and_Travel_Profit
from Customers as c
join customer_ratings as cr
on c.'Customer ID' = cr.'Customer ID'
join revenue_details as rd
on cr.'Invoice ID' = rd.'Invoice ID'
join invoices as i
on rd.'Invoice ID' = i.'Invoice ID'
join products as p
on i.'Product ID' = p.'Product ID';

```

## Screen Shot of SQL Query and Results

The screenshot displays the MySQL Workbench interface. The left sidebar shows the 'SCHEMAS' panel with a tree view of databases and tables. The main editor window contains the following SQL query:

```

###Query 1
#Some retailers believe that there is more money to be
# made in selling fashion accessories to men
#than sports and travel to women.Is this true?

select sum(case when c.gender = 'Male' and p.'Product line' = 'Fashion Accessories'
then rd.'Gross income' end ) as Men_Fashion_Accessories_Profit,
sum(case when c.gender = 'Female' and p.'Product line' = 'Sports and Travel'
then rd.'Gross income' end) as Women_Sports_and_Travel_Profit
from Customers as c
join customer_ratings as cr
on c.'Customer ID' = cr.'Customer ID'
join revenue_details as rd
on cr.'Invoice ID' = rd.'Invoice ID'
join invoices as i
on rd.'Invoice ID' = i.'Invoice ID'
join products as p
on i.'Product ID' = p.'Product ID';

```

Below the query editor, the 'Result Grid' shows the results of the query. The grid has two columns: 'Men\_Fashion\_Accessories\_Profit' and 'Women\_Sports\_and\_Travel\_Profit'. The first row shows values 1136.595 and 1360.7009999999999 respectively. The status bar at the bottom indicates 'Query Completed'.

Men_Fashion_Accessories_Profit	Women_Sports_and_Travel_Profit
1136.595	1360.7009999999999

## SQL Query 2

### Question (Q.2)

Some retailers believe that revenue in food and beverages can be increased amongst women by focusing on Ewallets, while others believe eWallets are more popular with men buying electronic accessories. Who is right?

### Notes/Comments About SQL Query and Results (Include # of Rows in Result)

#### Explanation:

First, we find the sum of 'Total' for the 2 categories compared i.e., 'food and beverages sold to women' and 'electronic accessories sold to men' grouped by the different payment types such as Ewallet, Credit card and cash. We can use this result to compare the revenue generated for 'food and beverages sold to women' and 'electronic accessories sold to men' amongst different payment types and also see if Ewallet is a popular choice.

#### Assumptions:

Revenue is the total amount of income generated by the sale of goods or services related to the company's primary operations. We assume the value of 'total' denotes the revenue of a purchase. Hence, summing the value of total for the 2 categories 'food and beverages sold to women' and 'electronic accessories sold to men' can be used to compare revenue generated by different payment types.

#### # of Rows in Result: 3

#### Conclusion:

From the results of the query mentioned below, we can see that the sum of total for 'food and beverages sold to women' is least for payments made using Ewallets compared to credit cards and cash.

Next, we can see that sum of total for 'electronic accessories sold to men' is maximum for payments made using Ewallets.

Hence, the claim made by some retailers that revenue in food and beverages can be increased amongst women by focusing on Ewallets is FALSE, while the claim that eWallets are more popular with men buying electronic accessories is TRUE.

### Translation

#### Translation:

select payment, sum of total for the case when Product line is Food and Beverages and gender is Female as Food\_and\_Beverages\_Women\_Revenue, sum of total for the case when Product line is Electronic Accessories and gender is Male as Electronic\_Accessories\_Men\_Revenue from Customers table joined with customer ratings table when Customer ID from customers matches Customer ID from customer ratings table joined with revenue details table where Invoice ID from customer ratings matches Invoice ID from revenue details joined with invoices table where Invoice ID from revenue details matches Invoice ID from invoices table joined with products table where Product ID from invoices matches Product ID from products grouped by Payment

**Cleanup:**

```
select i.payment Payment_Method, sum(case when p.'Product line' = 'Food and Beverages' and c.gender = 'Female' then i.Total end ) as Women_Food_and_Beverages_Revenue, sum(case when p.'Product line' = 'Electronic Accessories' and c.gender = 'Male' then i.Total end) as Men_Electronic_Accessories_Revenue
from Customers as c join customer_ratings as cr on c.'Customer ID' = cr.'Customer ID' join revenue_details as rd on cr.'Invoice ID' = rd.'Invoice ID' join invoices as i on rd.'Invoice ID' = i.'Invoice ID' join products as p on i.'Product ID' = p.'Product ID' group by i.Payment
```

**Query:**

```
select i.payment Payment_Method, sum(case when p.'Product line' = 'Food and Beverages'
and c.gender = 'Female' then i.Total end ) as Women_Food_and_Beverages_Revenue,
sum(case when p.'Product line' = 'Electronic Accessories'
and c.gender = 'Male' then i.Total end) as Men_Electronic_Accessories_Revenue
from Customers as c
join customer_ratings as cr
on c.'Customer ID' = cr.'Customer ID'
join revenue_details as rd
on cr.'Invoice ID' = rd.'Invoice ID'
join invoices as i
on rd.'Invoice ID' = i.'Invoice ID'
join products as p
on i.'Product ID' = p.'Product ID'
group by i.Payment
```

## Screen Shot of SQL Query and Results

The screenshot displays the MySQL Workbench interface. The left sidebar shows the 'SCHEMAS' panel with a tree view of databases and tables. The 'supermarketsalesdb' database is selected, showing tables like 'customer\_ratings', 'customers', 'invoices', 'products', 'revenue\_details', and 'stores'. The main editor window contains an SQL query labeled '###Query 2'. The query calculates revenue for 'Food and Beverages' and 'Electronic Accessories' for females and males. The 'Result Grid' at the bottom shows the query results with columns: 'Payment\_Method', 'Women\_Food\_and\_Beverages\_Revenue', and 'Men\_Electronic\_Accessories\_Revenue'. The results are as follows:

Payment_Method	Women_Food_and_Beverages_Revenue	Men_Electronic_Accessories_Revenue
Ewallet	9335.040	11304.5435
Credit card	11000.7435	6355.5975
Cash	12826.127999999999	9515.268

The 'Action Output' panel at the bottom shows the execution details: 1 row(s) returned, 0.0073 sec / 0.00001...

## SQL Query 3

### Question (Q.3)

Some retailers believe payment method is a bigger indicator of health and beauty purchases while other retailers believe gender is a bigger factor. Who is right?

### Notes/Comments About SQL Query and Results (Include # of Rows in Result)

#### Explanation:

For the Product line 'Health and Beauty', first we find the count of purchases made for different payment methods. Next, we find the count of purchases made for different gender. We can use this result to see which of the two (payment method, gender) play a bigger factor for health and beauty purchases.

#### Assumptions:

'Count' of purchases for different methods or gender can be used to determine which of the two plays a bigger factor in the purchase of health and beauty products.

#### # of Rows in Result: 6

#### Conclusion:

From the result of the query mentioned below, for the product line 'health and beauty' we can see that when we consider the factor as payment method 34.9% (53 of 152) of purchases are made with Ewallets, 32.2% (49 of 152) of purchases are made with cash and 32.9% (50 of 152) of purchases are made with credit cards.

When we consider the factor as gender 42.1% (64 of 152) of purchases are made by women, 57.9% (88 of 152) of purchases are by men.

We can conclude that GENDER is a bigger indicator of health and beauty purchases as we can clearly see that men make majority of purchases in this product line compared to women. We cannot say the same about payment types as all the payment methods have approximately the same share of purchases.

## Translation

### Translation:

select distinct payments as Factor, count of payments as Health\_and\_Beauty\_Count grouped by different payment types from invoices joined with products on Product ID from invoices matching Product ID from products table where Product Line is Health and Beauty union with select "--", "--" union with select distinct Gender as Factor, count of payments as Health\_and\_Beauty\_Count grouped by Gender from customers table joined with customer\_ratings table on Customer ID from customer matching Customer ID from customer ratings joined with invoices table on Invoice ID from customer ratings Invoice ID from invoices table joined with products table on Product ID from invoices table matching Product ID from products table where Product Line is Health and Beauty

### Cleanup:



```
select distinct i.payment as Factor, count(i.payment) as Healht_and_Beauty_Count from invoices as I join
products as p on i.'Product ID' = p.'Product ID' where p.'Product Line' = 'Health and Beauty' group by
'Payment' union select "--", "--" union select distinct c.Gender as Factor, count(c.Gender) as
Healht_and_Beauty_Count from customers as c join 'customer_ratings' as cr on c.'Customer ID' =
cr.'Customer ID' join invoices as i on cr.'Invoice ID' = i.'Invoice ID' join products as p on i.'Product ID' =
p.'Product ID' where p.'Product Line' = 'Health and Beauty' group by 'Gender'
```

**Query:**

```
select distinct i.payment as Factor, count(i.payment) as Healht_and_Beauty_Count
from invoices as i
join products as p
on i.'Product ID' = p.'Product ID'
where p.'Product Line' = 'Health and Beauty'
group by 'Payment'
union
select "--", "--"
union
select distinct c.Gender as Factor, count(c.Gender) as Health_and_Beauty_Count
from customers as c
join 'customer_ratings' as cr
on c.'Customer ID' = cr.'Customer ID'
join invoices as i
on cr.'Invoice ID' = i.'Invoice ID'
join products as p
on i.'Product ID' = p.'Product ID'
where p.'Product Line' = 'Health and Beauty'
group by 'Gender'
```

## Screen Shot of SQL Query and Results

The screenshot displays the MySQL Workbench interface. The left sidebar shows the 'SCHEMAS' panel with a tree view of databases including 'supermarketsalesdb'. The main editor window contains an SQL query. The query is as follows:

```
49 #indicator of health and beauty purchases while other retailers
50 # believe gender is a bigger factor. Who is right?
51 select distinct i.payment as Factor, count(i.payment) as Healht_and_Beauty_Count
52 from invoices as i
53 join products as p
54 on i.'Product ID' = p.'Product ID'
55 where p.'Product Line' = 'Health and Beauty'
56 group by 'Payment'
57 union
58 select "___", "___"
59 union
60 select distinct c.Gender as Factor, count(c.Gender) as Health_and_Beauty_Count
61 from customers as c
62 join 'customer_ratings' as cr
63 on c.'Customer ID' = cr.'Customer ID'
64 join invoices as i
65 on cr.'Invoice ID' = i.'Invoice ID'
66 join products as p
67 on i.'Product ID' = p.'Product ID'
68 where p.'Product Line' = 'Health and Beauty'
69 group by 'Gender' ;
70
```

The 'Result Grid' at the bottom shows the results of the query. It has two columns: 'Factor' and 'Healht\_and\_Beauty\_Count'. The results are as follows:

Factor	Healht_and_Beauty_Count
EWallet	53
Cash	49
Credit card	50
___	___
Female	54
Male	58

The 'Action Output' panel at the bottom shows the execution details of the query:

Time	Action	Response	Duration / Fetch Time
1	21:21:47	select distinct i.payment as Factor, count(i.payment) as Healht_and_Beauty_Count from invoices as i join products as p on...	6 row(s) returned 0.0057 sec / 0.00001...

The status bar at the bottom indicates 'Query Completed'.

## SQL Query 4

### Question (Q.4)

Some retailers believe that their members are spending more per purchase while members believe they are spending less per purchase. Who is right?

### Notes/Comments About SQL Query and Results (Include # of Rows in Result)

#### Explanation:

First, we find the average of total where customer type is Member. Next, we find the average of total where customer type is Normal. We then compare the averages to find if members or non-members are spending more per purchase

#### Assumptions:

Revenue is the total amount of income generated by the sale of goods or services related to the company's primary operations. We assume the value of 'total' denotes spending of a purchase. Hence, we find the average of total for members and non-members to compare their average spending per purchase.

#### # of Rows in Result: 1

#### Conclusion:

The result of the query shown below clearly shows that members spent on average \$327.8 per purchase and non-members spent on average \$318.12 per purchase. Hence, some retailers who believe that their members are spending more per purchase are right and members who believe they are spending less per purchase are wrong.

## Translation

### Translation:

select average of total for the case when Customer Type is Member as Avg\_Members\_Spending, average of total for the case when Customer Type is Normal as Avg\_Non\_Members\_Spending from Customers table joined with customer\_ratings on Customer ID from customer matching Customer ID from customer ratings joined with revenue\_details on Invoice ID from credit ratings matching Invoice ID from revenue details joining invoices on matching Invoice ID from revenue details matching Invoice ID from invoices joining products table on Product ID from invoices matching Product ID in products table.

### Cleanup:

```
select avg(case when c.'Customer Type' = 'Member' then i.Total end) as Avg_Members_Spending,
avg(case when c.'Customer Type' = 'Normal' then i.Total end) as Avg_Non_Members_Spending from
Customers as c join customer_ratings as cr on c.'Customer ID' = cr.'Customer ID' join revenue_details as
rd on cr.'Invoice ID' = rd.'Invoice ID' join invoices as i on rd.'Invoice ID' = i.'Invoice ID' join products as p
on i.'Product ID' = p.'Product ID'
```

### Query:

```

select avg(case when c.'Customer Type' = 'Member'
then i.Total end) as Avg_Members_Spending,
avg(case when c.'Customer Type' = 'Normal'
then i.Total end) as Avg_Non_Members_Spending
from Customers as c
join customer_ratings as cr
on c.'Customer ID' = cr.'Customer ID'
join revenue_details as rd
on cr.'Invoice ID' = rd.'Invoice ID'
join invoices as i
on rd.'Invoice ID' = i.'Invoice ID'
join products as p
on i.'Product ID' = p.'Product ID'

```

## Screen Shot of SQL Query and Results

The screenshot displays the MySQL Workbench interface. On the left, the 'SCHEMAS' pane shows a tree view of databases, with 'supermarketsalesdb' selected and its tables expanded. The main editor window contains the following SQL query:

```

###QUERY 4
#Some retailers believe that their members are spending more per
#purchase while members believe they are spending less per purchase.
#Who is right?
select avg(case when c.'Customer Type' = 'Member'
then i.Total end) as Avg_Members_Spending,
avg(case when c.'Customer Type' = 'Normal'
then i.Total end) as Avg_Non_Members_Spending
from Customers as c
join customer_ratings as cr
on c.'Customer ID' = cr.'Customer ID'
join revenue_details as rd
on cr.'Invoice ID' = rd.'Invoice ID'
join invoices as i
on rd.'Invoice ID' = i.'Invoice ID'
join products as p
on i.'Product ID' = p.'Product ID'

```

Below the query editor, the 'Result Grid' shows the execution results. The grid has two columns: 'Avg\_Members\_Spending' and 'Avg\_Non\_Members\_Spending'. The first row contains the values 327.7913053892215 and 318.1228557114231 respectively. The status bar at the bottom indicates 'Query Completed' and '1 row(s) returned'.

Avg_Members_Spending	Avg_Non_Members_Spending
327.7913053892215	318.1228557114231

## SQL Query 5

### Question (Q.5)

Some retailers believe that their male members are bringing in more overall revenue per purchase while others believe female non-members are bringing in more revenue per purchase of fashion accessories. Who is right?

### Notes/Comments About SQL Query and Results (Include # of Rows in Result)

#### Explanation:

First, we find the average of 'total' for the category 'fashion accessories sold to male members'.

Next, we find the average of 'total' for the category 'fashion accessories sold to female non-members'.

Then we compare these averages to find if male members are bringing in more overall revenue per purchase or female non-members are bringing in more revenue per purchase of fashion accessories.

#### Assumptions:

We assume that we are comparing the overall revenue per purchase for the fashion accessories sold to male members and fashion accessories sold to female non-members.

Revenue is the total amount of income generated by the sale of goods or services related to the company's primary operations. We assume the value of 'total' denotes the revenue of a purchase.

Hence, we find the average of total for the 2 categories 'fashion accessories sold to male members' and 'fashion accessories sold to female non-members' to find which group brings in more revenue per purchase of fashion accessories.

#### # of Rows in Result: 1

#### Conclusion:

The result of the query shown below clearly shows that the average of 'total' for the category 'fashion accessories sold to male members' is \$287.21 as compared to 'total' for the category 'fashion accessories sold to female non-members' which is \$312.55.

Hence, we can conclude that the claim made by some retailers that their male members are bringing in more overall revenue per purchase is FALSE and the claim that female non-members are bringing in more revenue per purchase of fashion accessories is TRUE.

### Translation

#### Translation:

select average of total for the case where Product line is Fashion Accessories and Customer Type is Member and gender is Male as Avg\_Male\_Member\_Fashion\_Accessories\_Revenue, average of total for the case where Product line is Fashion Accessories and Customer Type is Normal and gender is Female as Avg\_Female\_Non\_Member\_Fashion\_Accessories\_Revenue from Customers table joined with customer ratings table on Customer ID from customers matching Customer ID from customer ratings joined with revenue details table on Invoice ID from customer ratings matching Invoice ID from revenue details joined

with invoices table on Invoice ID from revenue details matching Invoice ID from invoices joined with products on Product ID from invoices matching Product ID from products

#### **Cleanup:**

```
select avg(case when p.'Product line' = 'Fashion Accessories' and c.'Customer Type' = 'Member' and c.gender = 'Male' then i.Total end ) as Avg_Male_Member_Fashion_Accessories_Revenue, avg(case when p.'Product line' = 'Fashion Accessories' and c.'Customer Type' = 'Normal' and c.gender = 'Female' then i.Total end ) as Avg_Female_Non_Member_Fashion_Accessories_Revenue from Customers as c join customer_ratings as cr on c.'Customer ID' = cr.'Customer ID' join revenue_details as rd on cr.'Invoice ID' = rd.'Invoice ID' join invoices as i on rd.'Invoice ID' = i.'Invoice ID' join products as p on i.'Product ID' = p.'Product ID'
```

#### **Query:**

```
select
```

```
avg(case when p.'Product line' = 'Fashion Accessories'
```

```
and c.'Customer Type' = 'Member' and c.gender = 'Male' then i.Total end ) as  
Avg_Male_Member_Fashion_Accessories_Revenue,
```

```
avg(case when p.'Product line' = 'Fashion Accessories'
```

```
and c.'Customer Type' = 'Normal' and c.gender = 'Female' then i.Total end ) as  
Avg_Female_Non_Member_Fashion_Accessories_Revenue
```

```
from Customers as c
```

```
join customer_ratings as cr
```

```
on c.'Customer ID' = cr.'Customer ID'
```

```
join revenue_details as rd
```

```
on cr.'Invoice ID' = rd.'Invoice ID'
```

```
join invoices as i
```

```
on rd.'Invoice ID' = i.'Invoice ID'
```

```
join products as p
```

```
on i.'Product ID' = p.'Product ID'
```

## Screen Shot of SQL Query and Results

The screenshot displays the MySQL Workbench interface. The left sidebar shows the 'SCHEMAS' panel with a tree view of databases and tables. The 'supermarketsalesdb' database is selected, showing tables like 'customer\_ratings', 'customers', 'invoices', 'products', 'revenue\_details', and 'stores'. The main editor window shows a SQL query (Query 5) with comments and a SELECT statement. The query calculates average revenue for male members and female non-members for 'Fashion Accessories'. The 'Result Grid' at the bottom shows the query results with two columns: 'Avg\_Male\_Member\_Fashion\_Accessories\_Revenue' and 'Avg\_Female\_Non\_Member\_Fashion\_Accessories\_Revenue'. The results are 287.21298153846148 and 312.5457867142857 respectively. The status bar at the bottom indicates 'Query Completed'.

```
###Query 5
#Some retailers believe that their male members
#are bringing in more overall revenue per purchase
#while others believe female non-members are bringing in more
#revenue per purchase of fashion accessories. Who is right?

select
avg(case when p.`Product line` = 'Fashion Accessories'
and c.`Customer Type` = 'Member' and c.gender = 'Male' then i.Total end ) as Avg_Male_Member_Fashion_Accessories_Revenue,
avg(case when p.`Product line` = 'Fashion Accessories'
and c.`Customer Type` = 'Normal' and c.gender = 'Female' then i.Total end ) as Avg_Female_Non_Member_Fashion_Accessories_Revenue
from Customers as c
join customer_ratings as cr
on c.`Customer ID` = cr.`Customer ID`
join revenue_details as rd
on cr.`Invoice ID` = rd.`Invoice ID`
join invoices as i
on rd.`Invoice ID` = i.`Invoice ID`
join products as p
on i.`Product ID` = p.`Product ID`;
```

Avg_Male_Member_Fashion_Accessories_Revenue	Avg_Female_Non_Member_Fashion_Accessories_Revenue
287.21298153846148	312.5457867142857

Result 53

Action Output

Time	Action	Response	Duration / Fetch Time
21:22:33	select avg(case when p.`Product line` = 'Fashion Accessories' and c.`Customer Type` = 'Member' and c.gender = 'Male'...	1 row(s) returned	0.0059 sec / 0.00001...

Query Completed

## Data Review for MongoDB

### Assumptions/Notes About Data Collections, Attributes and Relationships between Collections

Supermarketsales database has 1 collection supermarketsales which has 17 attributes.

List of attributes:

Invoice ID, Branch, City, Customer type, Gender, Product Line, Unit price, Quantity, Tax 5%, Total, Date, Time, Payment, cogs, gross margin percentage, gross income, Rating

Number of documents in supermarketsales collection: 1000

Since there is only one collection, there are no relationships.



# Physical Mongo Database

## Assumptions/Notes About Data Set

1. Database needs to store details about each invoice being generated through transactions.
2. Database needs to store details about their branches.
3. Database needs to store every Invoice-ID and is assigned to one Unique customer.
4. Database needs to store details about the customer (Customer ID, Gender)
5. Identify the type of the customer (Normal, Member) for each order.
6. Database needs to store details about the product line and identify each type of product line the product belongs to.
7. Database needs to identify the tax amount for each invoice total
8. Database needs to store date and time for each unique transaction.
9. Company needs to identify the payment type been used for each unique transaction and store it in the database.
10. Database needs to store the cost of goods sold for per invoice.
11. Company needs to recognize the gross margin of each transaction and stored in database.
12. Database needs to store satisfaction rating for each customer.

## Screen shot of Physical Database objects (Database, Collections and Attributes)

This screenshot shows the MongoDB Compass interface displaying the physical database structure. The left sidebar shows the hierarchy: Hosts, Cluster (Replica Set), Edition (MongoDB 4.4.5 Enterprise), and Collections. The 'supermarketsales' database is selected, showing collections: DBProjectQuery1, DBProjectQuery2, DBProjectQuery4, and supermarketsales. The main panel shows a table of collections with the following data:

Collection Name	Documents	Avg. Document Size	Total Document Size	Num. Indexes	Total Index Size	Properties
DBProjectQuery1 (view on: supermarketsales)	0	-	0.0 B	-	0.0 B	
DBProjectQuery2 (view on: supermarketsales)	0	-	0.0 B	-	0.0 B	
DBProjectQuery4 (view on: supermarketsales)	0	-	0.0 B	-	0.0 B	
<b>supermarketsales</b>	1,000	365.4 B	356.8 KB	1	24.0 KB	

This screenshot shows the MongoDB Compass interface displaying the document data for the 'supermarketsales' collection. The left sidebar shows the hierarchy: Hosts, Cluster (Replica Set), Edition (MongoDB 4.4.5 Enterprise), and Collections. The 'supermarketsales' database is selected, showing collections: DBProjectQuery1, DBProjectQuery2, DBProjectQuery4, and supermarketsales. The main panel shows a table of documents with the following data:

#	supermarketsales	_id ObjectId	Invoice ID String	Branch String	City String	Customer type String	Gen
1	609845dfea143dbdabee3cb	"758-67-8428"	"A"	"Yangon"	"Member"	"Fen"	
2	609845dfea143dbdabee3cc	"226-31-3881"	"C"	"Naypyitaw"	"Normal"	"Fen"	
3	609845dfea143dbdabee3cd	"631-41-3188"	"A"	"Yangon"	"Normal"	"Ha1"	
4	609845dfea143dbdabee3ce	"123-19-1176"	"A"	"Yangon"	"Member"	"Ha1"	
5	609845dfea143dbdabee3cf	"373-73-7918"	"A"	"Yangon"	"Normal"	"Ha1"	
6	609845dfea143dbdabee3d0	"699-14-3826"	"C"	"Naypyitaw"	"Normal"	"Ha1"	
7	609845dfea143dbdabee3d1	"355-53-5943"	"A"	"Yangon"	"Member"	"Fen"	
8	609845dfea143dbdabee3d2	"315-22-5665"	"C"	"Naypyitaw"	"Normal"	"Fen"	
9	609845dfea143dbdabee3d3	"665-32-9167"	"A"	"Yangon"	"Member"	"Fen"	
10	609845dfea143dbdabee3d4	"692-92-5582"	"B"	"Handalay"	"Member"	"Fen"	
11	609845dfea143dbdabee3d5	"351-62-0822"	"B"	"Handalay"	"Member"	"Fen"	
12	609845dfea143dbdabee3d6	"529-56-3974"	"B"	"Handalay"	"Member"	"Ha1"	
13	609845dfea143dbdabee3d7	"365-64-0515"	"A"	"Yangon"	"Normal"	"Fen"	
14	609845dfea143dbdabee3d8	"252-56-2699"	"A"	"Yangon"	"Normal"	"Ha1"	
15	609845dfea143dbdabee3d9	"829-34-3918"	"A"	"Yangon"	"Normal"	"Fen"	
16	609845dfea143dbdabee3da	"299-46-1885"	"B"	"Handalay"	"Member"	"Fen"	
17	609845dfea143dbdabee3db	"656-95-9349"	"A"	"Yangon"	"Member"	"Fen"	
18	609845dfea143dbdabee3dc	"765-26-6951"	"A"	"Yangon"	"Normal"	"Ha1"	
19	609845dfea143dbdabee3dd	"329-62-1586"	"A"	"Yangon"	"Normal"	"Ha1"	
20	609845dfea143dbdabee3de	"319-58-3348"	"B"	"Handalay"	"Normal"	"Fen"	

MongoDB Compass - cluster0.zkxsp.mongodb.net/supermarketsales.supermarketsales

DOCUMENTS 1k TOTAL SIZE 356.8KB AVG SIZE 365B INDEXES 1 TOTAL SIZE 24.0KB AVG SIZE 24.0KB

Documents Aggregations Schema Explain Plan Indexes Validation

Filter { field: 'value' }

ADD DATA VIEW

Displaying documents 1 - 20 of 1000

#	Gender String	Product line String	Unit price Double	Quantity Int32	Tax 5% Double	Total
1	"Female"	"Health and beauty"	74.69	7	26.1415	548.
2	"Female"	"Electronic accessories"	15.28	5	3.82	88.2
3	"Male"	"Home and lifestyle"	46.33	7	16.2155	340.
4	"Male"	"Health and beauty"	58.22	8	23.288	489.
5	"Male"	"Sports and travel"	86.31	7	30.2885	634.
6	"Male"	"Electronic accessories"	85.39	7	29.8865	627.
7	"Female"	"Electronic accessories"	68.84	6	20.652	433.
8	"Female"	"Home and lifestyle"	73.56	10	36.78	772.
9	"Female"	"Health and beauty"	36.26	2	3.626	76.1
10	"Female"	"Food and beverages"	54.84	3	8.226	172.
11	"Female"	"Fashion accessories"	14.48	4	2.896	68.8
12	"Male"	"Electronic accessories"	25.51	4	5.102	107.
13	"Female"	"Electronic accessories"	46.95	5	11.7375	246.
14	"Male"	"Food and beverages"	43.19	10	21.595	453.
15	"Female"	"Health and beauty"	71.38	10	35.69	749.
16	"Female"	"Sports and travel"	93.72	6	28.116	590.
17	"Female"	"Health and beauty"	68.93	7	24.1255	586.
18	"Male"	"Sports and travel"	72.61	6	21.783	457.
19	"Male"	"Food and beverages"	54.67	3	8.2005	172.
20	"Female"	"Home and lifestyle"	40.3	2	4.83	84.6

MongoDB Compass - cluster0.zkxsp.mongodb.net/supermarketsales.supermarketsales

DOCUMENTS 1k TOTAL SIZE 356.8KB AVG SIZE 365B INDEXES 1 TOTAL SIZE 24.0KB AVG SIZE 24.0KB

Documents Aggregations Schema Explain Plan Indexes Validation

Filter { field: 'value' }

ADD DATA VIEW

Displaying documents 1 - 20 of 1000

#	Total Double	Date Date	Time String	Payment String	cogs Double	gross
1	548.9715	2019-01-05T06:00:00.000+00:00	"13:08"	"Ewallet"	522.83	4.76
2	88.22	2019-03-08T06:00:00.000+00:00	"10:29"	"Cash"	76.4	4.76
3	340.5255	2019-03-03T06:00:00.000+00:00	"13:23"	"Credit card"	324.31	4.76
4	489.048	2019-01-27T06:00:00.000+00:00	"20:33"	"Ewallet"	465.76	4.76
5	634.3785	2019-02-08T06:00:00.000+00:00	"10:37"	"Ewallet"	604.17	4.76
6	627.6165	2019-03-25T05:00:00.000+00:00	"18:30"	"Ewallet"	597.73	4.76
7	433.692	2019-02-25T06:00:00.000+00:00	"14:36"	"Ewallet"	413.04	4.76
8	772.38	2019-02-24T06:00:00.000+00:00	"11:38"	"Ewallet"	735.6	4.76
9	76.146	2019-01-10T06:00:00.000+00:00	"17:15"	"Credit card"	72.52	4.76
10	172.746	2019-02-20T06:00:00.000+00:00	"13:27"	"Credit card"	164.52	4.76
11	68.816	2019-02-06T06:00:00.000+00:00	"18:07"	"Ewallet"	57.92	4.76
12	107.142	2019-03-09T06:00:00.000+00:00	"17:03"	"Cash"	102.04	4.76
13	246.4875	2019-02-12T06:00:00.000+00:00	"10:25"	"Ewallet"	234.75	4.76
14	453.495	2019-02-07T06:00:00.000+00:00	"16:48"	"Ewallet"	431.9	4.76
15	749.49	2019-03-29T05:00:00.000+00:00	"19:21"	"Cash"	713.8	4.76
16	590.436	2019-01-15T06:00:00.000+00:00	"16:19"	"Cash"	562.32	4.76
17	586.6355	2019-03-11T05:00:00.000+00:00	"11:03"	"Credit card"	482.51	4.76
18	457.443	2019-01-01T06:00:00.000+00:00	"10:39"	"Credit card"	435.66	4.76
19	172.2105	2019-01-21T06:00:00.000+00:00	"18:00"	"Credit card"	164.01	4.76
20	84.63	2019-03-11T05:00:00.000+00:00	"15:30"	"Ewallet"	80.6	4.76

**supermarketsales.supermarketsales**

DOCUMENTS 1k TOTAL SIZE 356.8KB AVG. SIZE 365B INDEXES 1 TOTAL SIZE 24.0KB AVG. SIZE 24.0KB

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' }

ADD DATA VIEW {}

Displaying documents 1 - 20 of 1000 REFRESH

	Payment String	cogs Double	gross margin percentage Double	gross income Double	Rating Double
1	"Ewallet"	522.83	4.761984762	26.1415	9.1
2	"Cash"	76.4	4.761984762	3.82	9.6
3	"Credit card"	324.31	4.761984762	16.2155	7.4
4	"Ewallet"	465.76	4.761984762	23.288	8.4
5	"Ewallet"	684.17	4.761984762	38.2885	5.3
6	"Ewallet"	597.73	4.761984762	29.8865	4.1
7	"Ewallet"	413.04	4.761984762	28.652	5.8
8	"Ewallet"	735.6	4.761984762	36.78	8
9	"Credit card"	72.52	4.761984762	3.626	7.2
10	"Credit card"	164.52	4.761984762	8.226	5.9
11	"Ewallet"	57.92	4.761984762	2.896	4.5
12	"Cash"	182.84	4.761984762	5.102	6.8
13	"Ewallet"	234.75	4.761984762	11.7375	7.1
14	"Ewallet"	431.9	4.761984762	21.595	8.2
15	"Cash"	713.8	4.761984762	35.69	5.7
16	"Cash"	562.32	4.761984762	28.116	4.5
17	"Credit card"	482.51	4.761984762	24.1255	4.6
18	"Credit card"	435.66	4.761984762	21.783	6.9
19	"Credit card"	164.81	4.761984762	8.2085	8.6
20	"Ewallet"	88.6	4.761984762	4.83	4.4

## Data in the Database

Collection Name	Relationships With Other Collections (if any)	# of Documents in Collection
supermarketsales	No relationship	1000

## MongoDB Queries/Code

Pick 3 SQL queries and write them in MongoDB

### Mongo Query 1

#### Question (Q.1)

Some retailers believe that there is more money to be made in selling fashion accessories to men than sports and travel to women. Is this true?

Notes/Comments About MongoDB Query/Code and Results (Include # of Documents in Result)

#### **Explanation:**

First, we filter the documents which have Product Line= fashion accessories, gender= Male and Product Line= sports and travel, gender= Female.

Next, we find the sum of Gross income for 2 different categories, i.e., 'fashion accessories sold to men' and 'sports and travel sold to women'. We then compare its values to see which is greater.

#### **Assumptions:**

We assume the value of 'gross income' denotes the money made on a purchase. Hence, summing the value of gross income for the 2 categories 'fashion accessories sold to men' and 'sports and travel sold to women' can be used to compare which category makes the most money.

**# of Documents in Result: 2**

#### **Conclusion:**

From the result of the query shown below, we can see that the sum of gross income for the category 'fashion accessories sold to men' is \$1136.595 compared 'sports and travel sold to women' which is \$1360.70. Hence the belief of some retailers that there is more money to be made in selling fashion accessories to men than sports and travel to women is FALSE.

### Translation

Using supermarketsales collection, we use \$match to filter documents for Product Line= fashion accessories, gender= Male and Product Line= sports and travel, gender= Female. \$group is used in the next stage to get sum of gross income grouped by product line and gender.

## Screen Shot of MongoDB Query/Code and Results

MongoDB Compass - cluster0.zkxsp.mongodb.net/supermarketsales.supermarketsales

DOCUMENTS 1k 356.8KB 365B INDEXES 1 24.0KB 24.0KB

Aggregations

COLLATION DBProjectQuery1 SAVE

Output after \$match stage (Sample of 20 documents)

```
1 {
2   "Sor": [
3     {
4       "Product line": "Sports and trav
5       "Gender": "Female"
6     }
7   ],
8   {
9     "Product line": "Fashion accesso
10    "Gender": "Male"
11  }
12 ]
13 }
14 }
15 }
16 }
17 }
```

Output after \$group stage (Sample of 2 documents)

```
1 {
2   _id: {gender: '$Gender', pr: '$Product line'},
3   Gender: {first: '$Gender'},
4   Productline: {first: '$Product line'},
5   TotalAmount: {sum: '$gross income'}
6 }
```

ADD STAGE

MongoDB Compass - cluster0.zkxsp.mongodb.net/supermarketsales.DBProjectQuery1

supermarketsales.DBProjectQuery1 (view on: supermarketsales.supermarketsales) MODIFY SOURCE Read Only

Documents

VIEW { } { }

Displaying documents 1 - 2 of 2 REFRESH

_id Object	Gender String	Productline String	TotalAmount Double
{ } 2 fields	"Female"	"Sports and travel"	1360.701
{ } 2 fields	"Male"	"Fashion accessories"	1136.595

## Mongo Query 2

### Question (Q.2)

Some retailers believe that revenue in food and beverages can be increased amongst women by focusing on Ewallets, while others believe eWallets are more popular with men buying electronic accessories. Who is right?

### Notes/Comments About MongoDB Query/Code and Results (Include # of Documents in Result)

#### Explanation:

First, we filter the documents which have Product Line= food and beverages, gender= Female and Product Line= electronic accessories, gender= Male.

Next, we find the sum of 'Total' for the 2 categories compared i.e., 'food and beverages sold to women' and 'electronic accessories sold to men' grouped by the different payment types such as Ewallet, Credit card and cash. We can use this result to compare the revenue generated for 'food and beverages sold to women' and 'electronic accessories sold to men' amongst different payment types and also see if Ewallet is a popular choice.

#### Assumptions:

Revenue is the total amount of income generated by the sale of goods or services related to the company's primary operations. We assume the value of 'total' denotes the revenue of a purchase. Hence, summing the value of total for the 2 categories 'food and beverages sold to women' and 'electronic accessories sold to men' can be used to compare revenue generated by different payment types.

#### # of Documents in Result: 6

#### Conclusion:

From the results of the query shown below, we can see that the sum of total for 'food and beverages sold to women' is least for payments made using Ewallets compared to credit cards and cash.

Next, we can see that sum of total for 'electronic accessories sold to men' is maximum for payments made using Ewallets.

Hence, the claim made by some retailers that revenue in food and beverages can be increased amongst women by focusing on Ewallets is FALSE, while the claim that eWallets are more popular with men buying electronic accessories is TRUE.

## Translation

Using supermarketsales collection, we use \$match to filter documents for Product Line= food and beverages, gender= Female and Product Line= electronic accessories, gender= Male. \$group is used in the next stage to get sum of total grouped by product line and payment.



## Screen Shot of MongoDB Query/Code and Results

The screenshot shows the MongoDB Compass interface for the `supermarketsales.supermarketsales` collection. The aggregation pipeline is defined as follows:

```
1 {
2   $match: {
3     $and: [
4       {"Product line": "Food and beverages"},
5       {"Gender": "Female"}
6     ]
7   },
8   $group: {
9     _id: {
10      "Product line": "Electronic accessories",
11      "Gender": "Male"
12    }
13  }
14 }
```

The output after the `$match` stage (Sample of 20 documents) shows three documents with detailed metadata and product information.

The output after the `$group` stage (Sample of 6 documents) shows three documents with aggregated data:

```
1 {
2   _id: {
3     "Product line": "Electronic accessories",
4     "Gender": "Male"
5   },
6   TotalRevenue: { $sum: '$Total' }
7 }
```

ADD STAGE

The screenshot shows the MongoDB Compass interface for the `supermarketsales.DBProjectQuery2` collection. The results of the aggregation pipeline are displayed in a table view:

_id Object	Gender String	Productline String	PaymentType String	TotalRevenue Double
{ 2 fields }	"Female"	"Food and beverages"	"Cash"	12826.128
{ 2 fields }	"Male"	"Electronic accessories"	"Credit card"	6355.5975
{ 2 fields }	"Male"	"Electronic accessories"	"Ewallet"	11364.6435
{ 2 fields }	"Female"	"Food and beverages"	"Credit card"	11009.7435
{ 2 fields }	"Male"	"Electronic accessories"	"Cash"	9515.268
{ 2 fields }	"Female"	"Food and beverages"	"Ewallet"	9335.046



## Mongo Query 3

### Question (Q.4)

Some retailers believe that their members are spending more per purchase while members believe they are spending less per purchase. Who is right?

Notes/Comments About MongoDB Query/Code and Results (Include # of Documents in Result)

#### **Explanation:**

First, we find the average of total where customer type is Member. Next, we find the average of total where customer type is Normal. We then compare the averages to find if members or non-members are spending more per purchase

#### **Assumptions:**

Revenue is the total amount of income generated by the sale of goods or services related to the company's primary operations. We assume the value of 'total' denotes spending of a purchase. Hence, we find the average of total for members and non-members to compare their average spending per purchase.

#### **# of Documents in Result: 2**

#### **Conclusion:**

The result of the query shown below clearly shows that members spent on average \$327.8 per purchase and non-members spent on average \$318.12 per purchase. Hence, some retailers who believe that their members are spending more per purchase are right and members who believe they are spending less per purchase are wrong.

## Translation

Using supermarketsales collection, we use \$group to get the average of total grouped by customer type to compare the averages to find if members or non-members are spending more per purchase.

## Screen Shot of MongoDB Query/Code and Results

MongoDB Compass - cluster0.zkxsp.mongodb.net/supermarketsales.supermarketsales

DOCUMENTS 1k 356.8KB 365B INDEXES 1 24.0KB 24.0KB

Documents Aggregations Schema Explain Plan Indexes Validation

COLLATION DBProjectQuery4 SAVE SAMPLE MODE AUTO PREVIEW

1000 Documents in the Collection Preview of Documents in the Collection

Select an operator to construct expressions used in the aggregation pipeline stages. [Learn more](#)

Output after \$group stage (Sample of 2 documents)

```
1- {
2  _id: { $first: '$Customer type' },
3  CustomerType: { $first: '$Customer type' },
4  AverageSpending: { $avg: '$Total' }
5 }
6 }
```

ADD STAGE

MongoDB Compass - cluster0.zkxsp.mongodb.net/supermarketsales.DBProjectQuery4

supermarketsales.DBProjectQuery4 (view on: supermarketsales.supermarketsales) MODIFY SOURCE Read Only

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' } OPTIONS FIND RESET REFRESH

Displaying documents 1 - 2 of 2

	_id Object	CustomerType String	AverageSpending Double
1	{ 1 fields	"Member"	327.7913853892215
2	{ 1 fields	"Normal"	318.12285571142286