SQL
FINAL PROJECT

by Valentcia Angelica

CONTENT

SQL

Structured Query Language (SQL) is a programming language that used for manipulating and processing Relational Database Management System (RDBMS).

What can SQL do?

- Create database
- Create a new table in the database
- Insert, and update records in the database
- Create views in the database
- Execute queries in the database
- Many else...

RDBMS uses SQL queries for manipulating the database. In order to execute the queries, there is a lot of software that can be used, for instance, MySQL, Oracle, IBM DB2, Microsoft Access, etc.

MySQL on XAMPP

MySQL is an open source RDBMS that is widely used for executing SQL queries, which is know for its realibility and easy to use.

In this case, I used XAMPP as the web server, Apache as the local server for running MySQL.

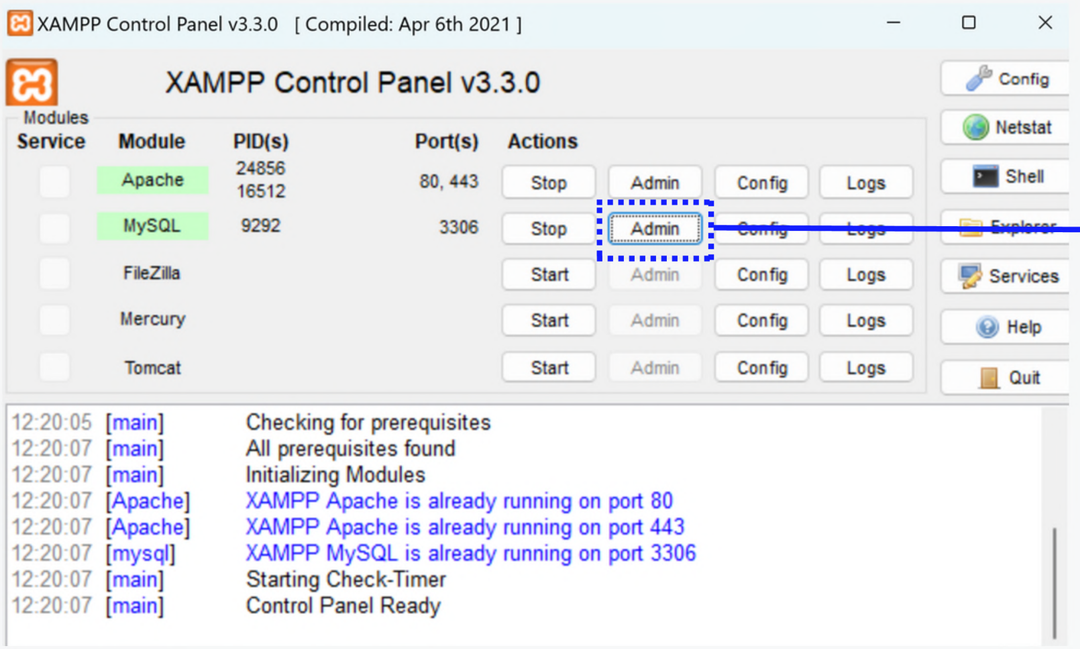
What makes MySQL popular?

- Easy to use
- Open source
- Efficiency
- Industry-standard
- Flexible RDBMS
- Cost-effective

MySQL on XAMPP

How to use it?

1. Start the Apache as the local server and then MySQL.

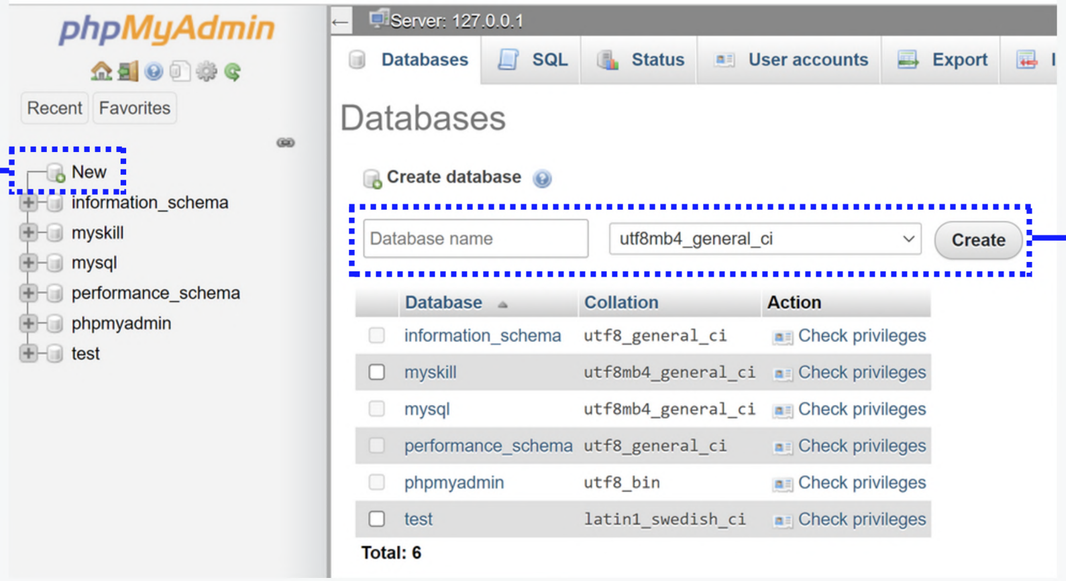


2. Click the Admin button on the MySQL line to direct to the phpMyAdmins.

MySQL on XAMPP

3. Create the database for storing the dataset / table.

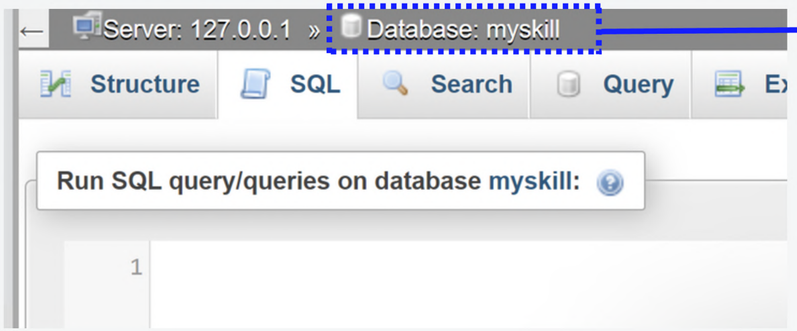
Click ‘New’ button to be directed to the ‘Create Database’ page.



Input the database name and click the ‘Create’ button to confirm making the database.

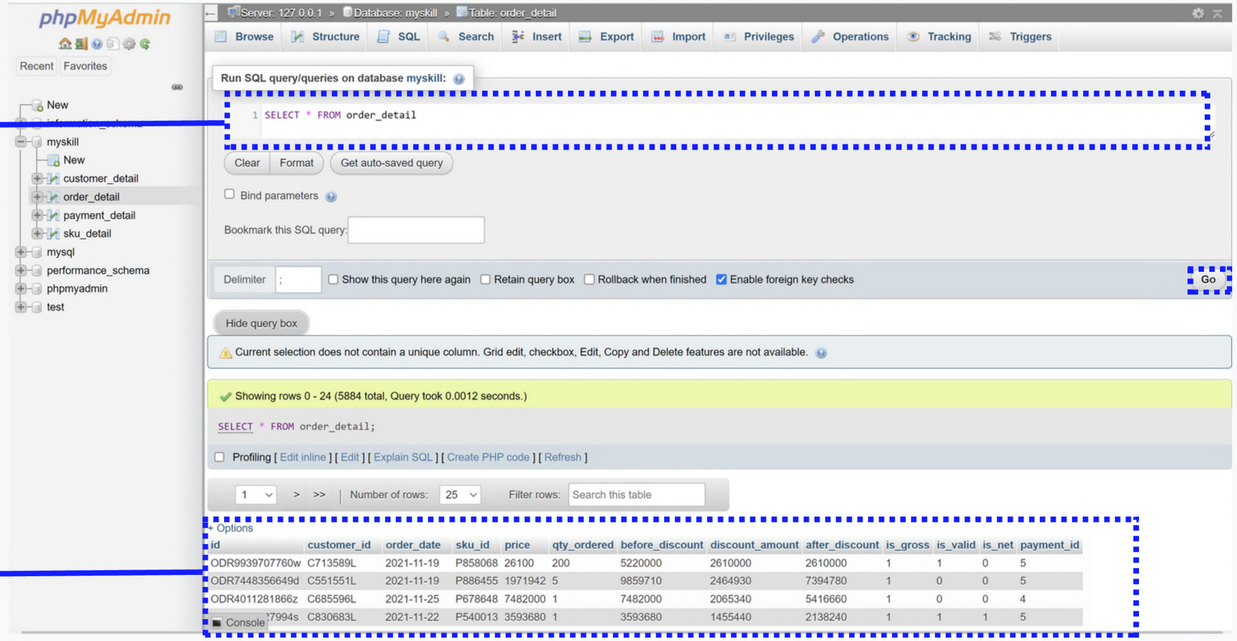
MySQL on XAMPP

4. Run SQL query on the SQL page.



Make sure that the chosen database is correct.

The written queries



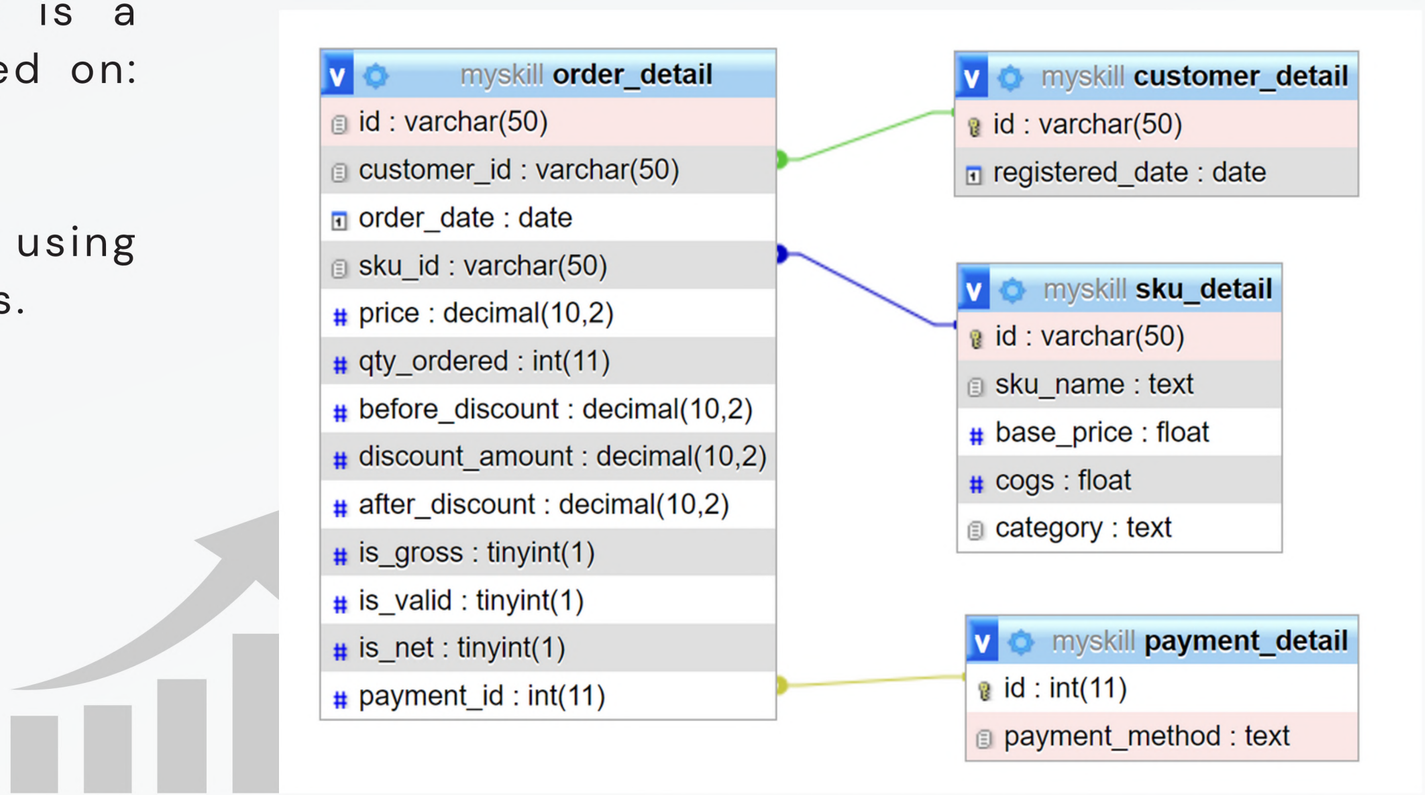
The output of the queries

‘Go’ button to run the queries

DATASET

Using a sales dataset that includes transactional details of sales. This is a dummy dataset and can be accessed on: httpp//

The given dataset will be processed using queries in order to make certain goals.



DATA ANALYSIS

A process with systematical statistics and logic techniques to evaluate, describe, and transform the data.

Case Study

01

All transactions that happened in 2021 with the highest total transaction (*after_discount*). Validate with is_valid = 1.

02

All transactions that happened in 2022 with the highest total transaction (*after_discount*), based on the category. Validate with is_valid = 1.

03

Compare the total transaction in 2021 and 2022, based on the category, and categorized it based on increasing or decreasing. Validate with is_valid = 1.

04

Show five popular *payment_method* in 2022 with its unique order. Validate with is_valid = 1.

05

List product:
1. Samsung
2. Apple
3. Sony
4. Huawei
5.Lenovo
based on their *total_transaction*. Validate with is_valid = 1.

01

Queries

```sql
SELECT SUM(after_discount) as total_after_discount,
EXTRACT(MONTH FROM order_date) as 'Month',
EXTRACT(YEAR FROM order_date) as 'Year'
FROM order_detail
WHERE YEAR(order_date) = 2021
AND is_valid = 1
GROUP BY EXTRACT(MONTH FROM order_date)
ORDER BY total_after_discount DESC
LIMIT 1;
```

Total transactions (*after_discount*)

Get the Month and Year from *order_date*

Group by Month because we want to show based on per-Month performance

Get the highest (no 1)

01

Result

With LIMIT 1

total_after_discount	Month	Year
227862744	8	2021

Without LIMIT 1

total_after_discount ▼ 1	Month	Year
227862744	8	2021
217309963	12	2021
207603259.875	10	2021
180396009.703125	11	2021
148007735.00156212	7	2021
145943335.24414062	9	2021
43154552	6	2021
36822126.5	1	2021
35611797	2	2021
34163856	5	2021
23643062	3	2021
22208472.59375	4	2021

With this result, we can get a business insight into the transactions that happened in 2021, which shows that August 2021 is the highest nominal of total transactions.

This data can be used to analyze the reason why August got the highest transaction. For instance, in August there was a holiday that increased the sales.

The other purpose with this data is to make a comparison between each month of sales, and can be used to improve each month sales performance.

02

Queries

```sql
SELECT sd.category, SUM(od.after_discount) as
total_after_discount
FROM sku_detail sd
JOIN order_detail od
ON sd.id = od.sku_id
WHERE YEAR(od.order_date) = 2022
AND od.is_valid = 1
GROUP BY sd.category
ORDER BY total_after_discount DESC;
```

Total transactions (*after_discount*)

Group by category because we want to show based on the category.

Order by total_after_discount to list the table in DESCENDING order.

02

Result

category	total_after_discount
Mobiles & Tablets	918451576
Entertainment	365344151
Appliances	316358100
Computing	214028543.375
Men Fashion	135588253
Women Fashion	93014970.625
Home & Living	79483716.1953125
Health & Sports	54235579.5859375
Beauty & Grooming	46211019.1875
Superstore	32643266.52734375
Kids & Baby	25931276.83984375
Others	21744646.021484375
Soghaat	17658332
School & Education	17362465.301635742
Books	6792519.203125

This data shows the total transactions based on their category. With this result, the Mobile and Tablets category got the highest total transaction, followed by other categories.

We can use this data to analyze the business side to improve the sales of other categories.

03

Queries

```sql
SELECT category,
COALESCE(total_2021, 0) total_2021,
COALESCE(total_2022, 0) total_2022,
CASE
WHEN COALESCE(total_2021, 0) > COALESCE(total_2022,0) THEN 'Decrease'
WHEN COALESCE(total_2021, 0) < COALESCE(total_2022,0) THEN 'Increase'
END AS status
FROM(
    SELECT sd.category,
    SUM(CASE WHEN YEAR(od.order_date) = 2021
        THEN od.after_discount ELSE 0 END) AS total_2021,
    SUM(CASE WHEN YEAR(od.order_date) = 2022
        THEN  od.after_discount ELSE 0 END) as total_2022
    FROM order_detail od
    JOIN sku_detail sd
    ON od.sku_id = sd.id
    WHERE YEAR(od.order_date) IN (2021, 2022)
    AND od.is_valid = 1
    GROUP BY sd.category
    ) AS totals;
```

To handle NULL value.

CASE WHEN ... THEN to perform logic queries.

Subquery to calculate the total transaction with the given year.

03

Result

category	total_2021	total_2022	status
Appliances	218550177	316358100	Increase
Beauty & Grooming	46047360	46211019.1875	Increase
Books	10124596	6792519.203125	Decrease
Computing	172878860	214028543.375	Increase
Entertainment	162326357.375	365344151	Increase
Health & Sports	33837965.6015625	54235579.5859375	Increase
Home & Living	45797873	79483716.1953125	Increase
Kids & Baby	23971057.796875	25931276.83984375	Increase
Men Fashion	58628198	135588253	Increase
Mobiles & Tablets	370606718	918451576	Increase
Others	40468515.75	21744646.021484375	Decrease
School & Education	11558982.392187119	17362465.301635742	Increase
Soghaat	15056202.603515625	17658332	Increase
Superstore	28828088	32643266.52734375	Increase
Women Fashion	84045961.3984375	93014970.625	Increase

The result shows the total_transactions in each category in 2021 and 2022. This data can be broken down to analyze the sales for each category to find the finest solution if there is a decreasing transaction.

04

Queries

```sql
SELECT pd.payment_method as payment_method_name,
COUNT(DISTINCT od.id) as total
FROM order_detail od
JOIN payment_detail pd
ON od.payment_id = pd.id
WHERE YEAR(od.order_date) = 2022
AND od.is_valid = 1
GROUP BY payment_method_name
ORDER BY total DESC
LIMIT 5;
```

To handle duplicate value.

GROUP BY payment_method_name to show the table based on the payment_method_name.

To show the top 5 from the row.

04

Result

payment_method_name	total 1
cod	1809
Payaxis	181
customercredit	75
Easypay	69
jazzwallet	26

The result shows the top 5 payment methods from unique transactions.

The data can be used to analyze the performance of each payment method and describe the trend of the methods among customers.

05

Queries

```sql
SELECT sum(total) as total_after_discount, product
FROM
(SELECT
 CASE
 WHEN sd.sku_name LIKE '%Samsung%' THEN 'Samsung'
    WHEN sd.sku_name LIKE '%Apple%' THEN 'Apple'
    WHEN sd.sku_name LIKE '%Sony%' THEN 'Sony'
    WHEN sd.sku_name LIKE '%Huawei%' THEN 'Huawei'
    WHEN sd.sku_name LIKE '%Lenovo%' THEN 'Lenovo'
 END AS product,
od.after_discount as total
 FROM order_detail od
 JOIN sku_detail sd
 ON od.sku_id = sd.id
 WHERE (sd.sku_name LIKE '%Samsung%'
         OR sd.sku_name LIKE '%Apple%'
         OR sd.sku_name LIKE '%Sony%'
         OR sd.sku_name LIKE '%Huawei%'
         OR sd.sku_name LIKE '%Lenovo%')
        AND od.is_valid = 1
 ) AS brand
GROUP BY product
ORDER BY total_after_discount DESC;
```

Subquery to handle the logic expressions and to categorise it into the given brands.

05

Result

total_after_discount ▼ 1	product
588764150	Samsung
213310080	Apple
63960718	Sony
63160260	Huawei
62379800.375	Lenovo

The result shows the total transactions for five brands.

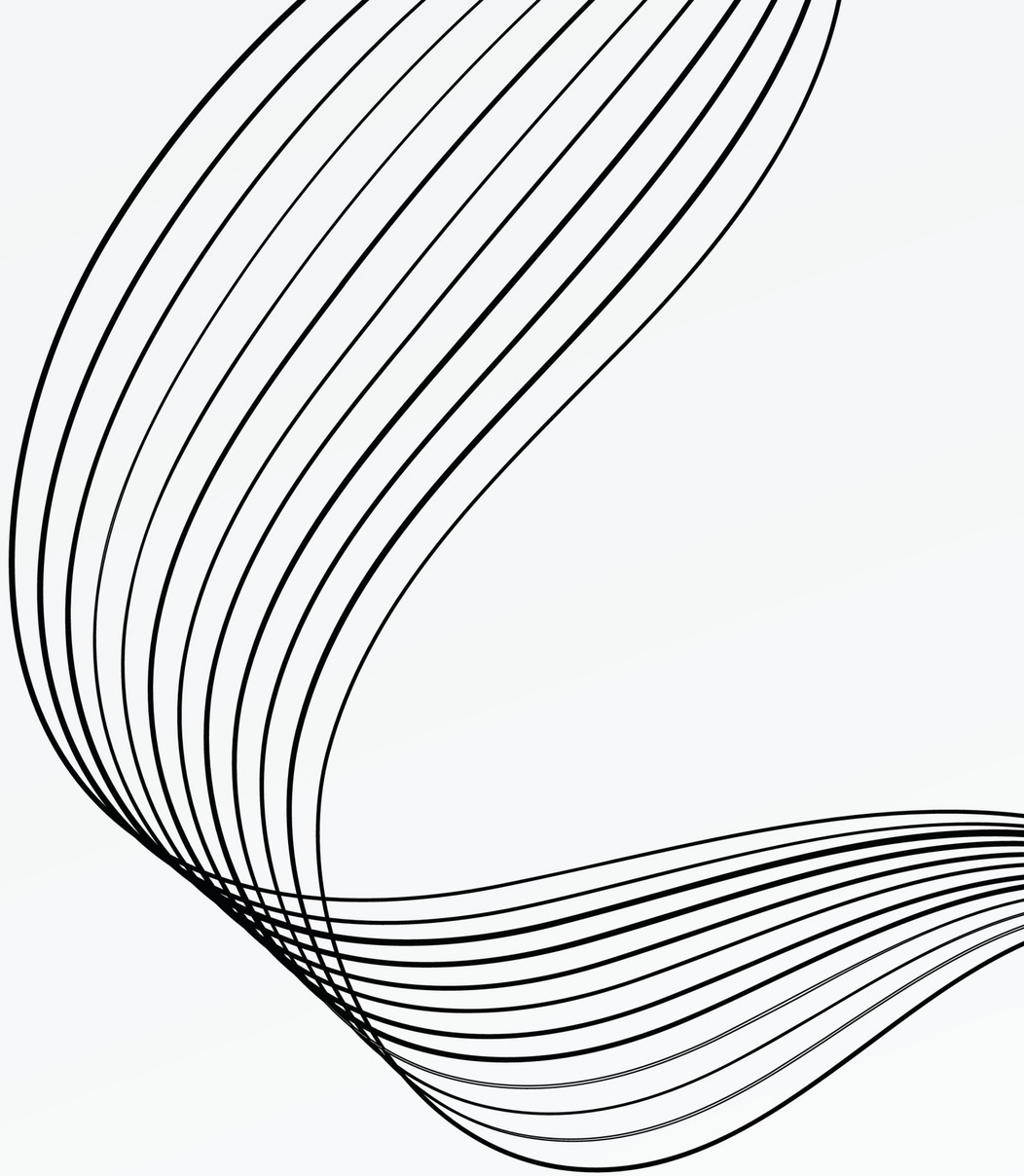
This indicates that the brand Samsung leads the sales by achieving the highest total transactions. This data can also be used to see the market trend among customers.

CONCLUSION

SQL can be used to analyze data on a huge scale and make the task easier to solve.

For future work, I want to try another kind of dataset to improve the exploration of the queries.

THANK YOU



Valentcia Angelica