

UCDF2106ICT(SE)

HAND OUT DATE: 1ST 2022

HAND IN DATE: 15th 2022

WEIGHTAGE:

MEMBER: Kong Jun Hinn TP064398

Koh Hong Kai TP064049

Lee Jia Heng TP063938

Louis Soin Keng Yong TP064103

INSTRUCTIONS TO CANDIDATES:

**1 Submit your assignment at the administrative counter.**

**2 Students are advised to underpin their answers with the use of references (cited using the Harvard Name System of Referencing).**

**3 Late submission will be awarded zero (0) unless Extenuating Circumstances (EC) are upheld.**

**4 Cases of plagiarism will be penalized.**

**5 The assignment should be bound in an appropriate style (comb bound or stapled).**

**6 Where the assignment should be submitted in both hardcopy and softcopy, the softcopy of the written assignment and source code (where appropriate) should be on a CD in an envelope / CD cover and attached to the hardcopy.**

**7 You must obtain 50% overall to pass this module.**

Contents

[1.0 Database Schema 3](#_Toc111465343)

[2.0 Database Diagram 4](#_Toc111465344)

[3.0 Data Dictionary 5](#_Toc111465345)

[4.0 SQL – Data Definition Language (DDL) 7](#_Toc111465346)

[4.1 Create Database 7](#_Toc111465347)

[4.2 Product 7](#_Toc111465348)

[4.3 Member 8](#_Toc111465349)

[4.4 Supplier 9](#_Toc111465350)

[4.5 Invoice 10](#_Toc111465351)

[4.6 Stock Count 11](#_Toc111465352)

[4.7 Product Rating 12](#_Toc111465353)

[4.8 Member total expenses 13](#_Toc111465354)

[4.9 Member order 14](#_Toc111465355)

[4.10 Supplier Invoice 15](#_Toc111465356)

[4.11 Supplier Product 16](#_Toc111465357)

[5.0 SQL-Data Manipulation Language (DML) 18](#_Toc111465358)

[6.0 Workload Matrix 25](#_Toc111465359)

# Diagram, schematic Description automatically generatedDatabase Schema

Chen’s Notation

Diagram

Description automatically generated

Crow’s Foot Notation

# 2.0 Database Diagram

Diagram

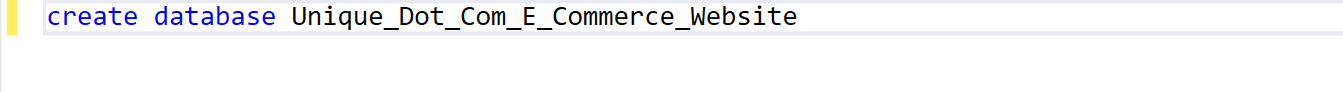
Description automatically generated

# 3.0 Data Dictionary

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Field Name | Data Type | Field Size for display | Description | Example |
| publisher\_ID | Nvarchar | 50 | Unique ID of each supplier | S01 |
| Supplier\_name | Nvarchar | 50 | Name of the supplier | AB Sdn.Bhd. |
| Supplier\_address | Nvarchar | 50 | Location of the supplier | No. 46, Jalan Puteri |
| Product\_ID | Nvarchar | 50 | Unique ID of each product | P01 |
| Product\_name | Nvarchar | 50 | Name of the product | Pencil |
| Product\_price | Decimal | 10,2 | Price of the product | 5.00 |
| Supply\_Quantity | integer | 10 | Quantity of product from supplier | 50 |
| Invoice\_num | nvarchar | 50 | Unique Invoice Num of each invoice | N01 |
| Invoice\_date | Date/Time | 10 | Date of invoice printed out | 5/2/2022 |
| Order\_quantity | integer | 10 | Quantity of product that ordered by Unique Dot Com | 50 |
| Pay\_amount | decimal | 10,2 | Amount of product price that need to bill by Unique Dot Com | 250.00 |
| Stock\_quantity | Integer | 10 | Quantity of stock in Unique Dot Com | 15 |
| Rating | integer | 10 | Rating of product from member  1= ‘terrible’,  5= ‘excellent’ | 3 |
| Num\_of\_comment | integer | 10 | Number of comment given to a product by a member | 20 |
| Member\_ID | nvarchar | 50 | Unique ID of the member | M01 |
| Member\_name | nvarchar | 50 | Name of the member | Ali |
| Gender | nvarchar | 50 | Gender of the member  M=Male,  F=Female | M |
| Address | nvarchar | 50 | Location of member | 53-49, Taman Bukit Serdang |
| Total\_expenses | decimal | 10,2 | Total expenses of a member in Unique Dot Com | 110.00 |
| Contact\_num | integer | 10 | Contact number of member | 011-2345 |
| Num\_of\_feedback | integer | 10 | Number of feedbacks to a product given by a member | 2 |
| quantity | integer | 10 | Quantity of products ordered by a member | 5 |
| Total\_price | Decimal | 10,2 | Total price of product ordered by a member | 5 |
| Date | Date/Time | 10 | Date of order made by a member | 23/6/2022 |
| Delivery\_status | nvarchar | 50 | Delivery status of product ordered by a member | Delivered |

# 4.0 SQL – Data Definition Language (DDL)

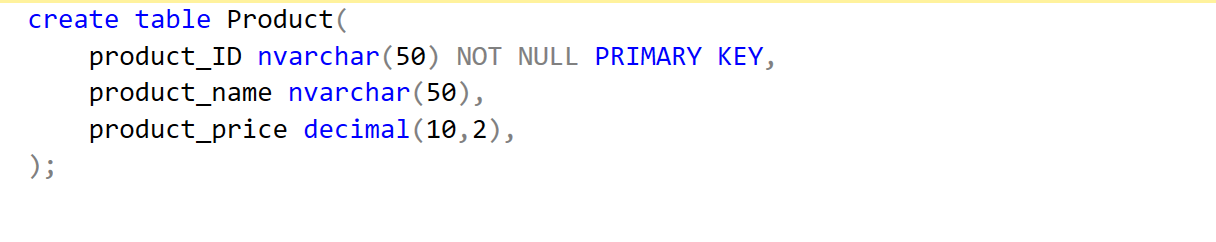
# 4.1 Create Database



***Figure 4.1.1:*** *Query of Creating Database Unique Dot Com E-Commerce Website*

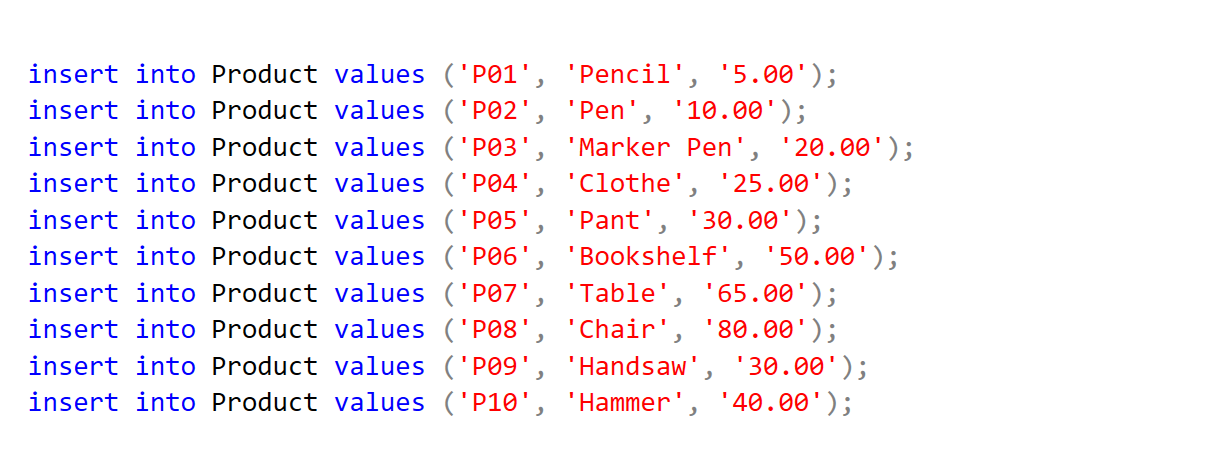
At first, we will construct a database called Unique Dot Com E-Commerce Website.

# 4.2 Product



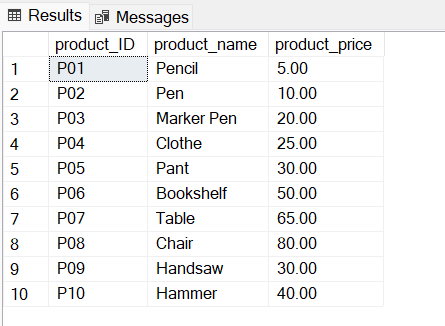
***Figure 4.2.1:*** *Query of Table Product*

After create a database, we start to add all query for generating table. First, we create a table product, which includes product id, product name and product price, is shown in the image above. Product ID is the primary key in this query, we make a check constraint which let the product ID start with ‘P’ and continues as number, for example the first product ID was P01. To make it easier, we used nvarchar (50) for each. This is because of it’s a string of varying length that can include letters, numbers, and special characters. Second column and third column which are product name and product price will be showing the different name and price of product according by product ID.



***Figure 4.2.2:*** *Query of Insert Value into Table Product*

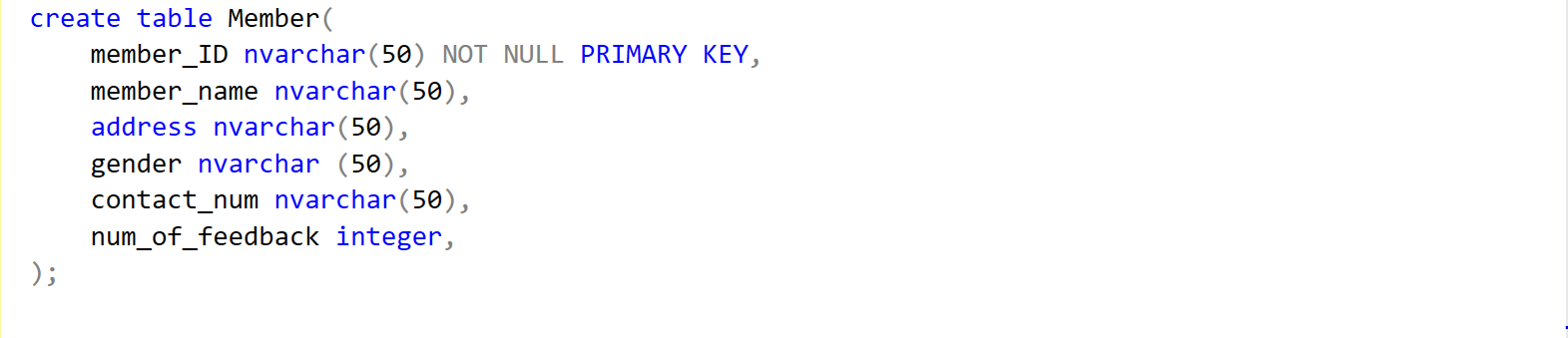
We begin inserting values into the product table when we finish constructing table members. The image above depicts the information that we enter and the necessity to double-check that it is correct.



***Figure 4.2.3:*** *Screenshot of output*

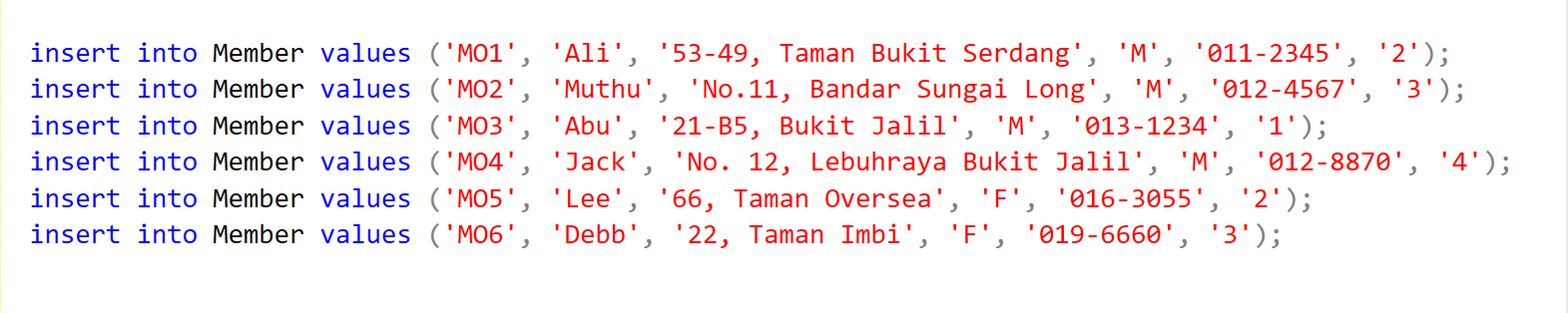
***Figure 4.2.3Screenshot of output***

# 4.3 Member



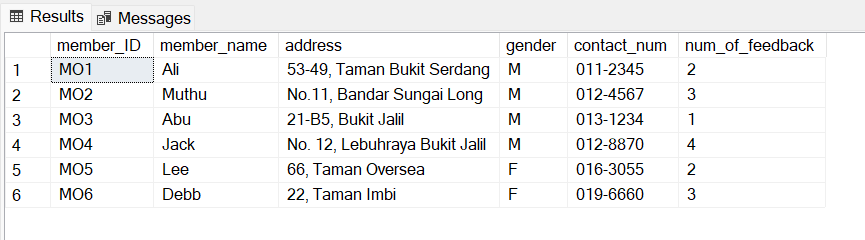
***Figure 4.3.1:*** *Query of Table Member*

We create and new table name as “Member” and enter the “member\_ID”, “member\_Name”, “address”, “gender”, “contact\_num”, “num\_of\_feedback”. The primary key will be the “member\_ID” and all the data type will be nvarchar (50) except “num\_of\_feedback” will be integer. This is to collect customer feedback by using number format. In member table, member ID is the primary key and cannot be null. To check constraint, all the member ID will start with ‘M’ and continues as number, for example the first customer ID was M01. Following by member name, address, gender and contact number will be shown up according by member ID.



***Figure 4.3.2:*** *Query of Insert Value into Table Member*

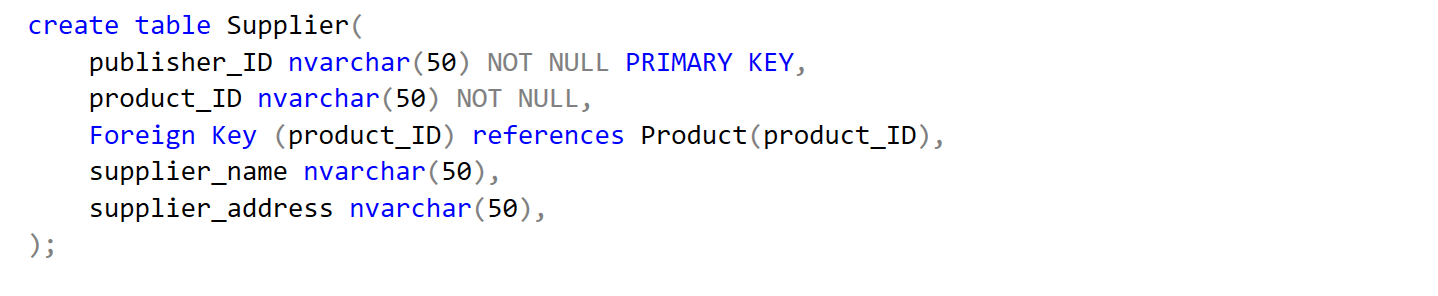
After finish creating the table, we insert data into the Member table. Based on the sequence of insert query of column name, key in the data accordingly, and put the correct data in the correct column.



***Figure 4.3.3:*** *Screenshot of output*

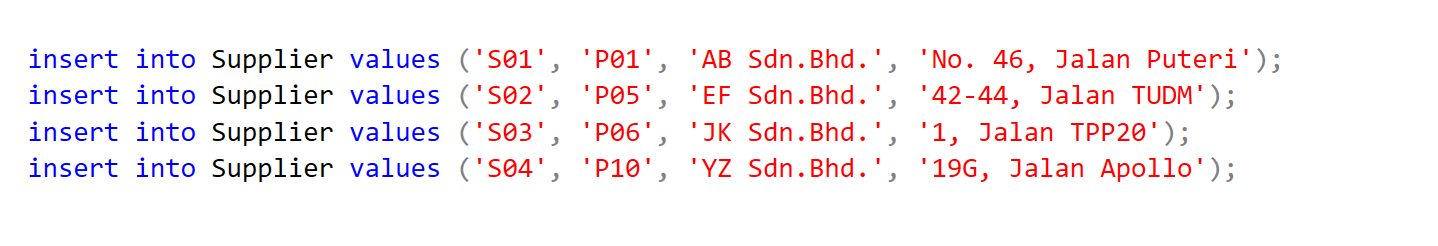
This is the outcome of Member table

# 4.4 Supplier



***Figure 4.4.1:*** *Query of Table Supplier*

After the member table is created, we create a new table name Supplier. The figure shows the query for creating the table which includes publisher\_ID, product\_ID, supplier\_name and supplier\_address. Publisher\_ID is the primary key of the table. The second column is product\_ID which is the identification of product and is also a foreign key. Moreover, we also create the third column supplier\_name as well as supplier\_address in forth column. Therefore, we can check the orders according to Supplier.



***Figure 4.4.2:*** *Query of Insert Value into Table supplier*

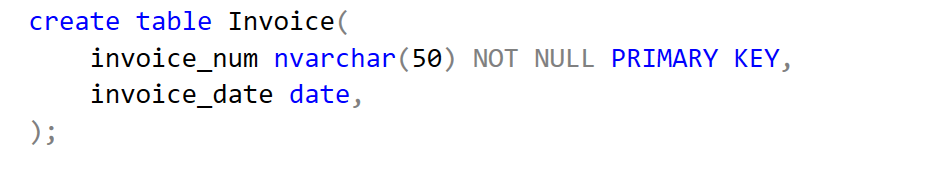
We inserted the data into these tables after it were created

*Text

Description automatically generated with medium confidence*

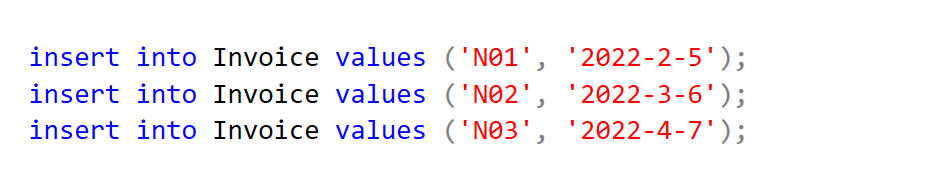
***Figure 4.4.3:*** *Screenshot of output*

# 4.5 Invoice



***Figure 4.5.1:*** *Query of Table Invoice*

The table that created after supplier is invoice. There are only 2 columns that have been created which are invoice\_num and invoice\_date. For invoice number, each invoice have their own serial number after the product has been sold, it is also the primary key of this table. The column invoice\_date is referring to the date that the invoice produced.

**

***Figure 4.5.2:*** *Query of Insert Value into Table Invoice*

We insert the relevant data into invoice table

Table

Description automatically generated

***Figure 4.5.3:*** *Screenshot of output*

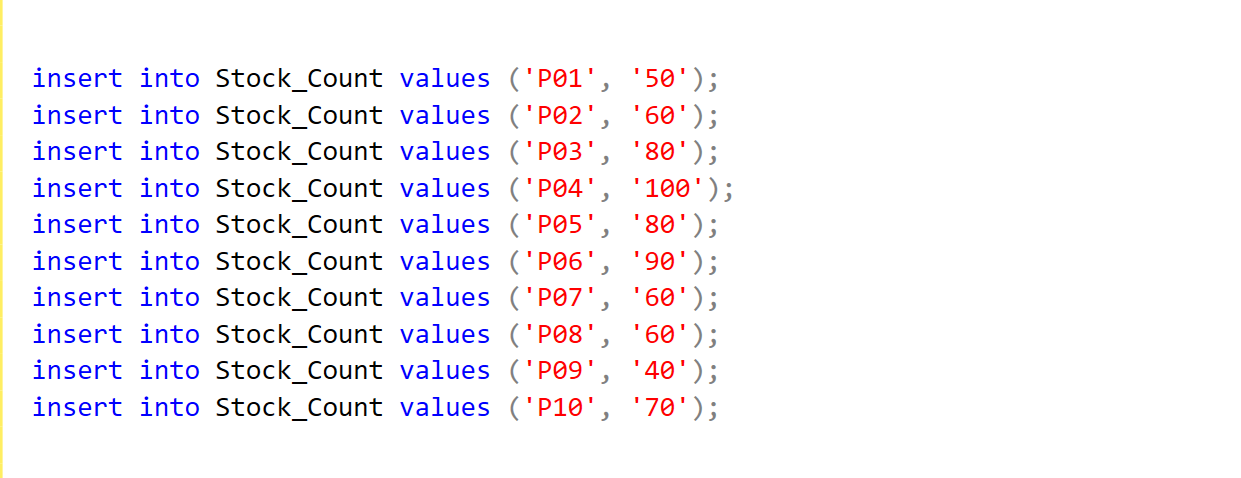
# 4.6 Stock Count

A picture containing graphical user interface

Description automatically generated

***Figure 4.6.1:*** *Query of Table Stock Count*

After creating tables invoice, we start creating table stock count. Figure 4.6.1 shows the query of creating table stock count which just include 2 column the product and stock quantity. The primary key in this table is product ID and is a foreign key from table product too. This table will help manager to track the quantity of product by tracking the product ID.



***Figure 4.6.2:*** *Query of Insert Value into Table stock count*

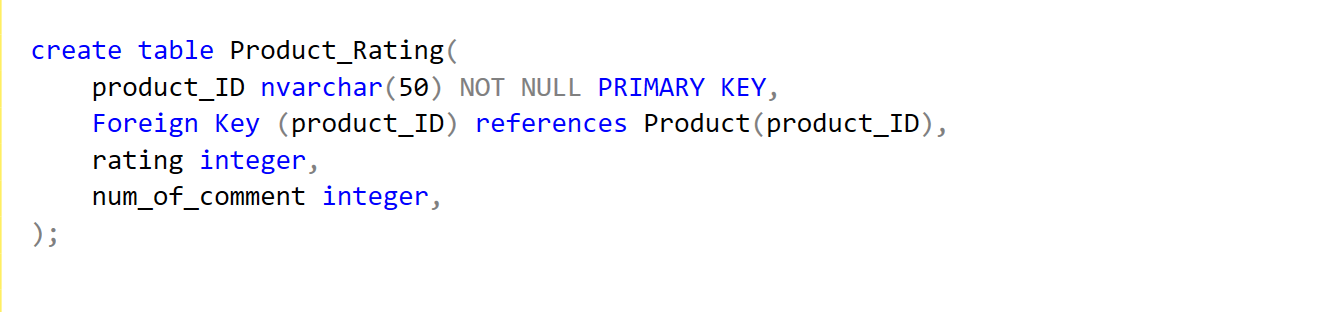
Starting insert data into table stock count after the name of table and column inserted. Based on the sequence of insert query of column name, key in the data accordingly, and make sure all of the data follow the data type and format.

Table

Description automatically generated

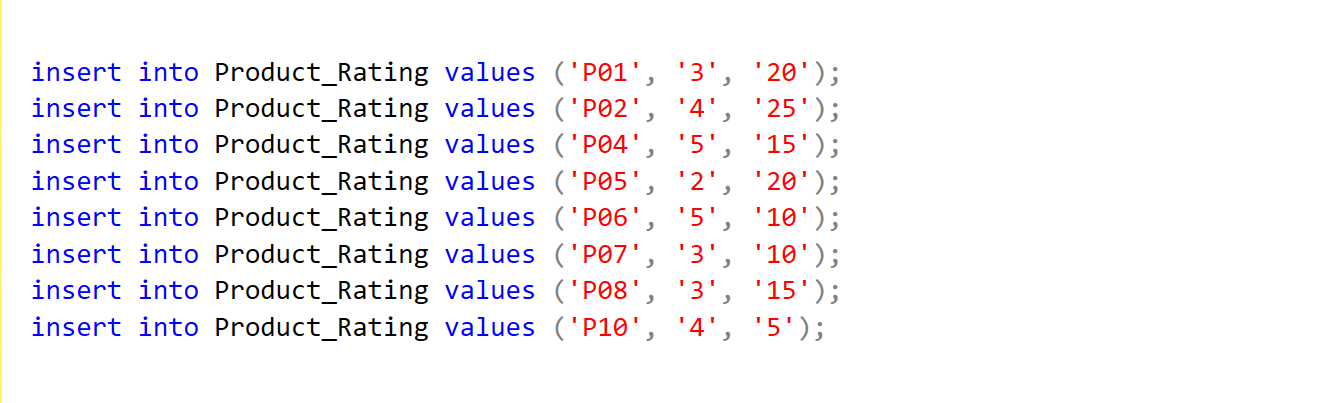
***Figure 4.6.3:*** *Screenshot of output*

# 4.7 Product Rating



***Figure 4.7.1:*** *Query of Table Product Rating*

After creating stock count table, we create another table name product rating. Based on figure 4.7.1 shown the query for creating stock count table which includes product ID, rating, and number of comment. Product ID is a foreign key in product rating table. Customer can provide rating for a product by following the score, 1 is terrible and meanwhile 5 is excellence and the customer can giving some comment regarding to the product that they has been received and their comment will convert into integer to collect the number of comment for specific product.



***Figure 4.7.2:*** *Query of Insert Value into Table Product Rating*

These are the data that inserted into product rating table

*Table

Description automatically generated*

***Figure 4.7.3:*** *Screenshot of output*

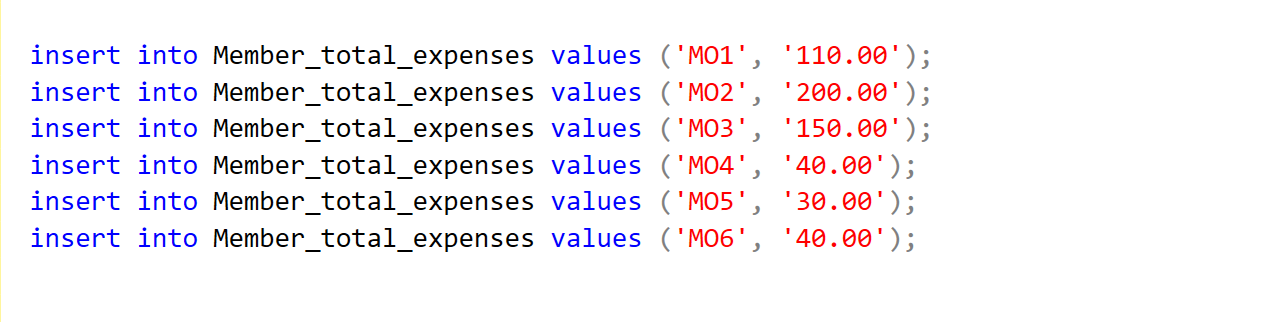
# 4.8 Member total expenses

Graphical user interface, application

Description automatically generated

***Figure 4.8.1:*** *Query of Table Member total expenses*

After creating table of product rating, we create a new table name member total expenses with queries of 2 column which are member ID and total expenses. Member ID is a primary key and is a foreign key too at a same time. This will be tracking different member total expenses according by unique member ID.



***Figure 4.8.2:*** *Query of Insert Value into Table Member total expenses*

We key in those data into the table member total expenses that has been created

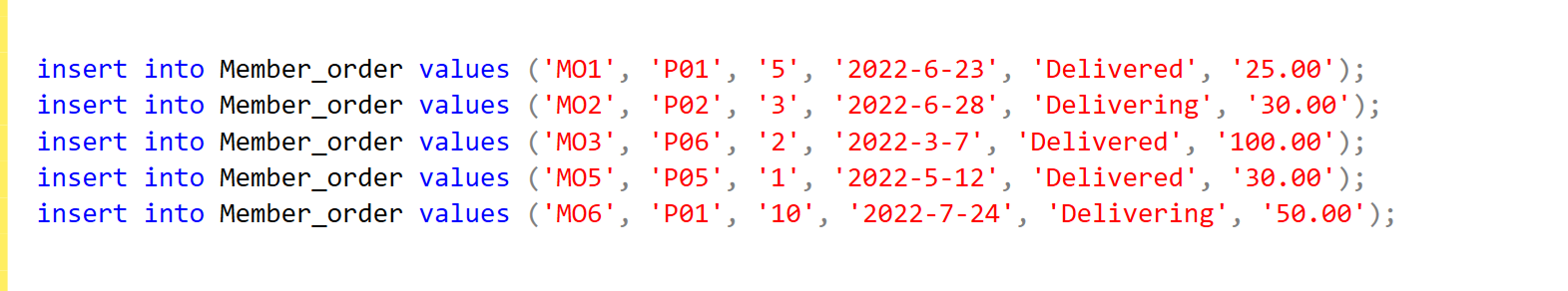
Table

Description automatically generated

***Figure 4.8.3:*** *Screenshot of output*

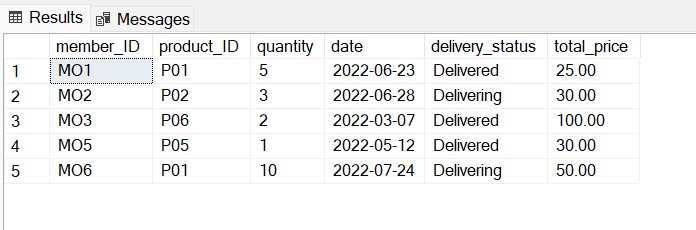
# 4.9 Member order

***Figure 4.9.1:*** *Query of Table Member order*

 After the member total expenses is created, we create a new table name member order. The figure shows the query for creating the table which includes member ID, product ID, quantity, date, delivery status and total price. Member ID is the primary key and there are two foreign key in this table which are member ID and product ID so we can check the member’ product order. This table will be assist member to track their quantity of product, delivery status such as delivered or delivering, and will list out the total price of their order to let the customer keep it as prove and further confirmation.

***Figure 4.9.2:*** *Query of Insert Value into Table Member order*

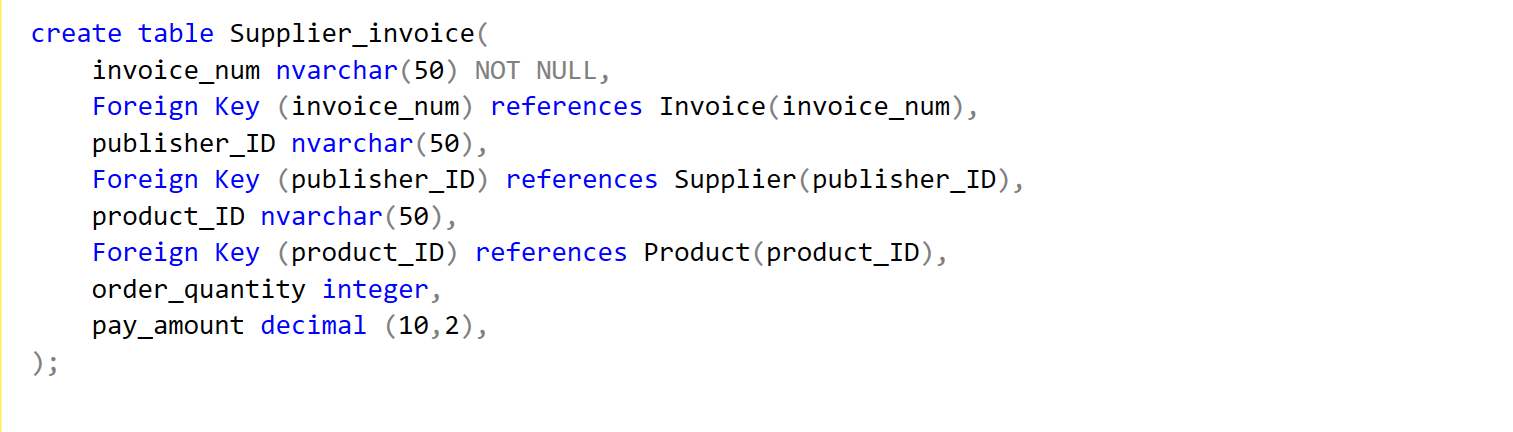
These are the data that has been inserted by us into the table

**

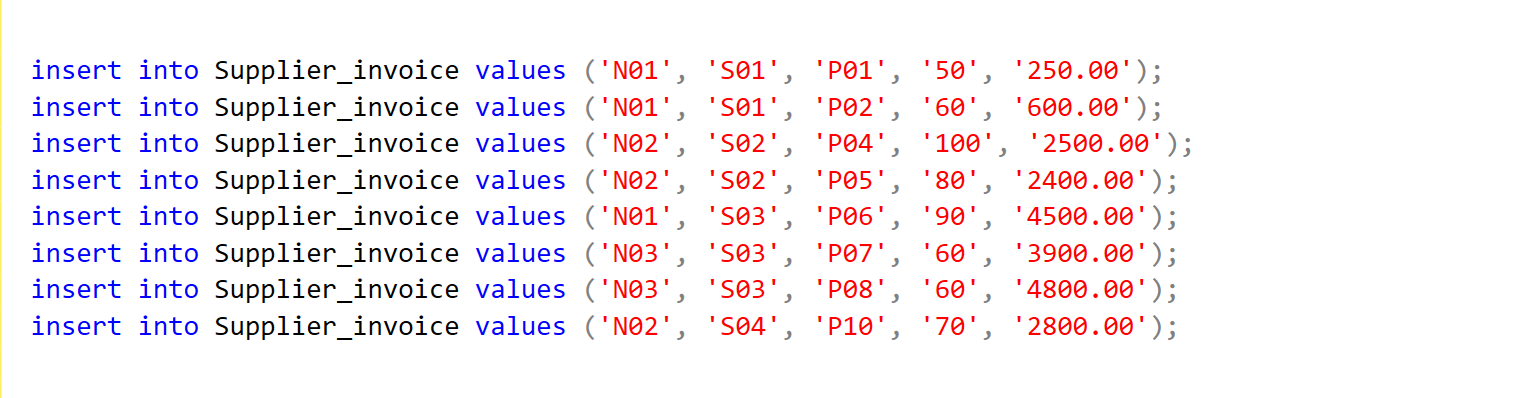
***Figure 4.9.3:*** *Screenshot of output*

# 

# 4.10 Supplier Invoice



***Figure 4.10.1:*** *Query of Table Supplier Invoice*

** After the member order table is created, we created a new table name supplier invoice. The query for generating table member, which includes invoice number, publisher ID, product ID, order quantity and pay amount. Based on the figure 4.10.1 that shown above, invoice number is the primary key in this query. In this table, there are 3 foreign key which are invoice number, publisher ID and product ID. This will help supplier to have a more detail on invoice by order quantity of product and pay amount to know how much that has been given out by according the value of product.

***Figure 4.10.2:*** *Query of Insert Value into Table Supplier Invoice*

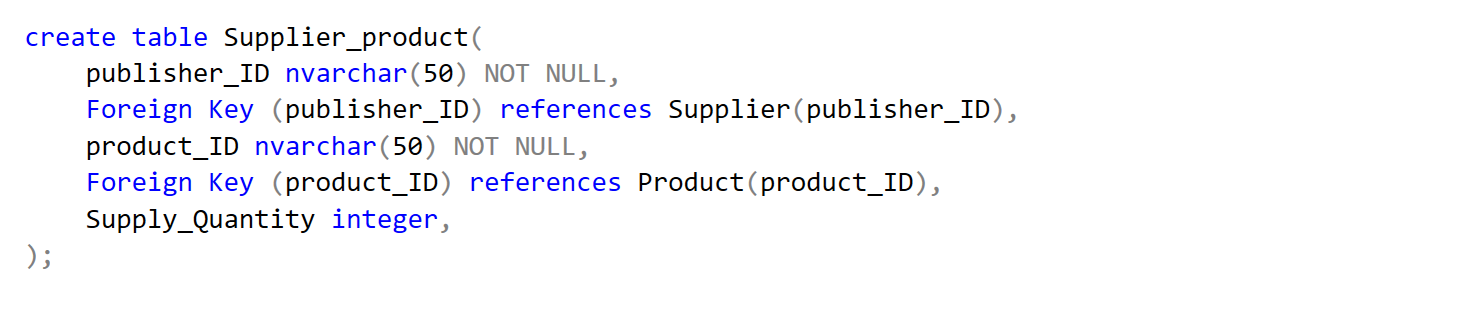
We key in the data that’s relavant to supplier invoice details into it

*Table

Description automatically generated*

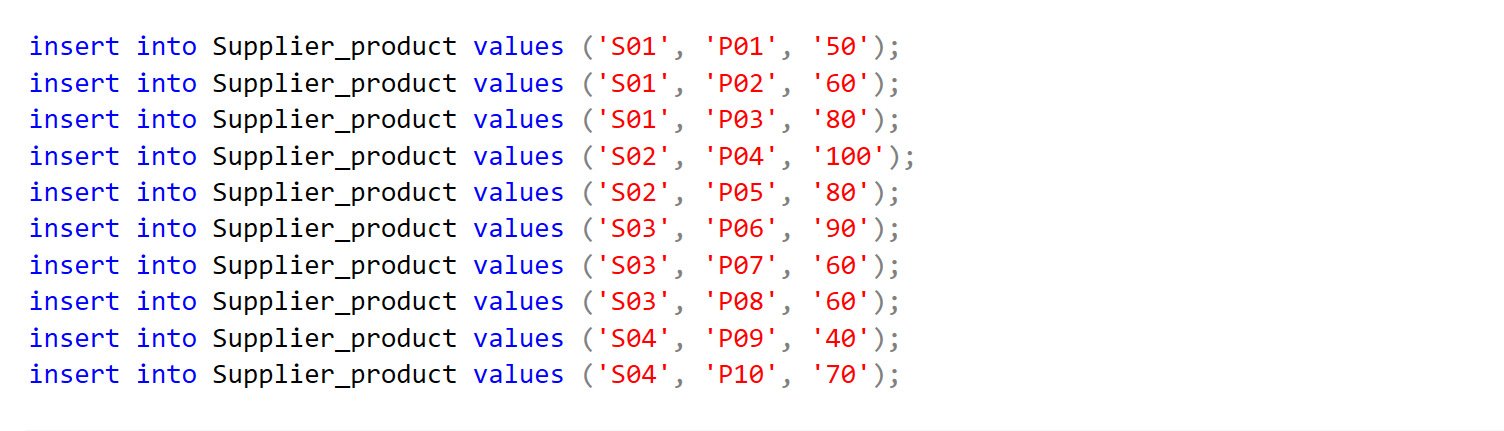
***Figure 4.10.3:*** *Screenshot of output*

# 4.11 Supplier Product

**

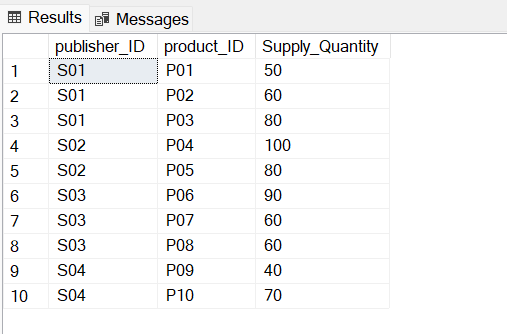
***Figure 4.11.1:*** *Query of Table Supplier product*

Finally, we create a new table name supplier product after supplier invoice table. In table supplier product contains 3 column which are publisher ID, product and supply quantity. The primary key will be the “publisher ID”. Meanwhile, publisher ID and product ID are also foreign key to help supplier track product ID and supply ID. Also, the existing of supply quantity is to help supplier well known how many they have the product.

**

***Figure 4.11.2:*** *Query of Insert Value into Table Supplier product*

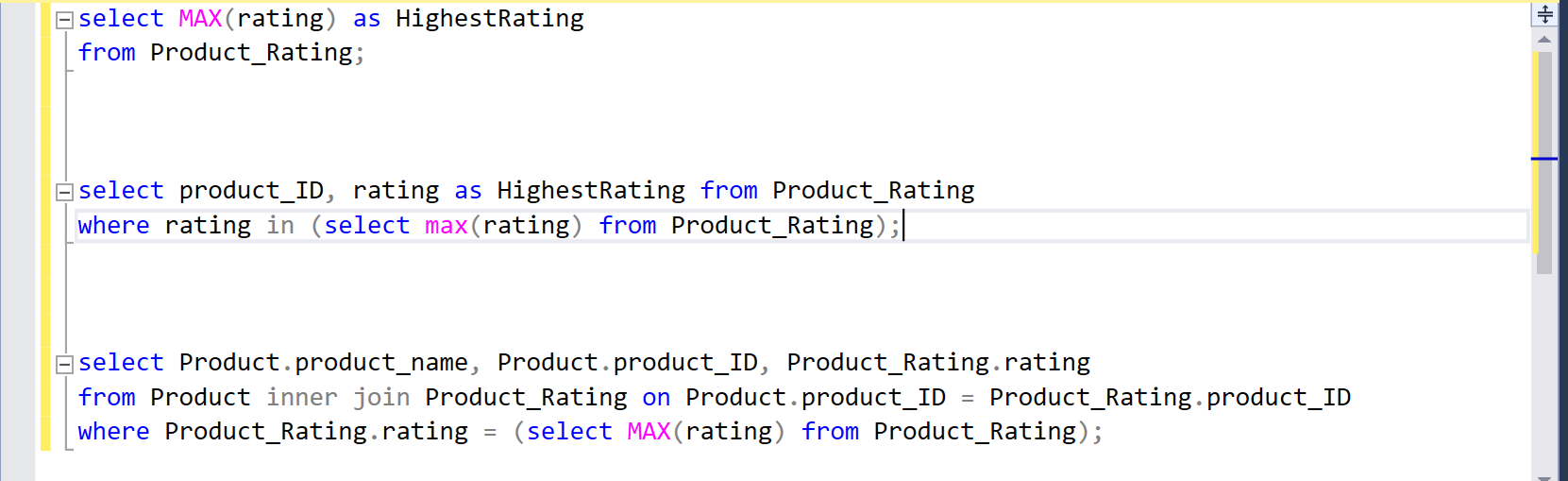
Starting insert data into table supplier product after the name of table and column inserted. Based on the sequence of insert query of column name, key in the data accordingly, and make sure all of the data follow the data type and format.



***Figure 4.11.3:*** *Screenshot of output*

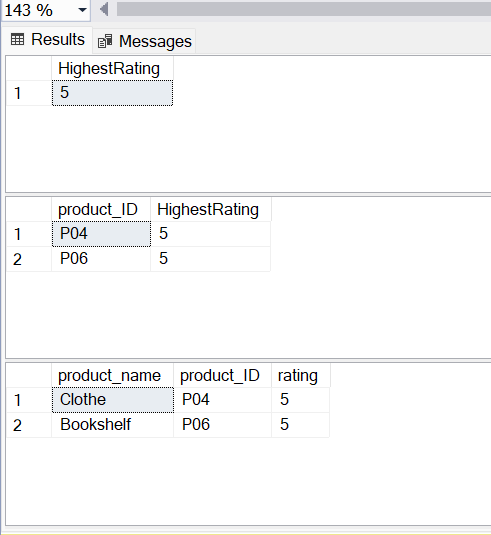
# 5.0 SQL-Data Manipulation Language (DML)

Kong Jun Hinn TP064398

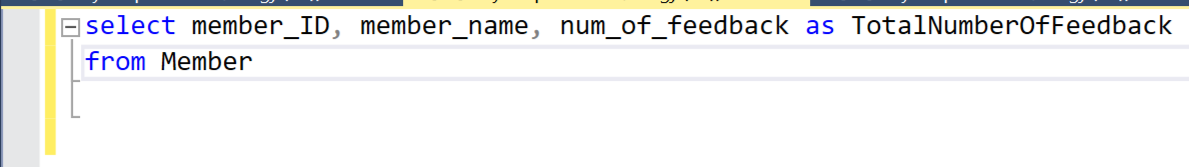


*Figure 5.1.1: Query of Question 1*

Question 1 requested that we display the highest rating in our product system, and grouping by product id and product name. We used the max function to find the highest rating. Then, we utilized the inner join function to combine records from product and rating. The group by clause is then used to group the product id and product name that question 1 requested. Finally, we used the having clause to allow the group by search criteria to be applied. The below diagram shows the results of the Question 1

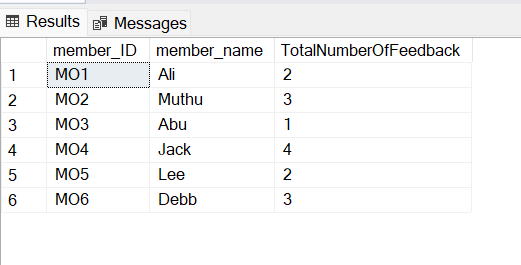


*Figure 5.1.2: Screenshot of Output (Product with Highest Rating)*

**

*Figure 5.2.1: Query of Question 2*

Question 2 asked us to calculate the total feedback per member and to display both the member id and the member’s name. As a result, we must choose the title that corresponds to the query. Then we needed to write a from clause that specified which table contained these data. The member id, member name, and feedback were all created in the same table. As a result, we only need to utilise the group by clause to group the generated rows into sets to make it easier. The picture below shows that the results of the Question 2.



*Figure 5.2.2: Screenshot of Output (Total number of feedbacks per member)*

Koh Hong Kai (TP064049)

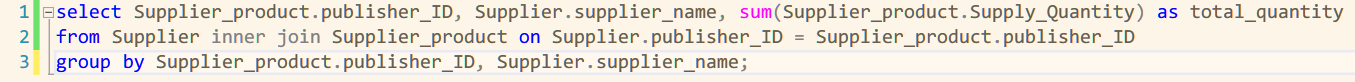


Figure 5.3.1 Query of Question 3

For Question 3, we will need to obtain an output showing the total number of products supplied by each supplier. The table of output should consist of publisher\_ID, supplier\_name, and total\_quantity. Hence, sum function is used to calculate the number of products supplied by each supplier. Then, inner join syntax is utilised to combine both Supplier table and Supplier\_product table, in order to have the output of supplier\_name matching the publisher\_ID. Lastly, the total\_quantity of each supplier is shown by using the group by syntax. The output is as below.

Table

Description automatically generated

Figure 5.3.2 Output for total number of products supplied by each supplier

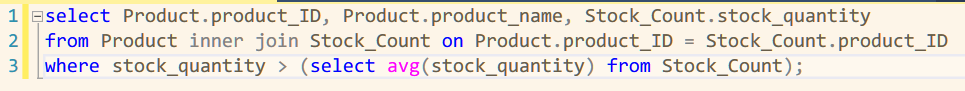


Figure 5.4.1 Query of Question 4

Question 4 requested for a list of in store products with quantity higher than the average quantity of all products. The output will consist of product\_ID, product\_name, and stock\_quantity. Hence, the Product table and Stock\_Count table are combined with inner join syntax to look for the matching results in both tables. The condition of query is set to stock\_quantity of a product must be more than the average of all stock\_quantity, which is obtained through the avg function. The resulted table is shown below.

Table

Description automatically generated

Figure 5.4.2 Output for product with quantity more than average

Lee Jia Heng (TP063938)

Graphical user interface, application, Word

Description automatically generated

*Figure 5.5.1 Query of Question 5*

Question 5 need to find out total number of products ordered by e-store manager from various suppliers. So, we need to sum up the total number of products ordered which name order\_quantity from table Supplier\_invoice.

Graphical user interface, application

Description automatically generated

*Figure 5.5.2 Query of Question 5*

Graphical user interface, text, application

Description automatically generated

*Figure 5.6.1 Query of Question 6*

Question 6 need to join three tables, so we used SELECT command, JOIN function with three tables, JOIN function with three tables. It joined three tables which are Supplier\_invoice, Invoice and Supplier.

Table

Description automatically generated

*Figure 5.6.2 Query of Question 6*

Text

Description automatically generated

*Figure 5.7.1 Query of Question 7*

Question 7 need to show the total member based on gender who are registered as member and total number of registered members. To create question 7 using DML, we use SELECT command and COUNT function. As created before, the table member only include member who registered in Unique Dot Com, so only need to select the gender from the table member and use COUNT function . Besides, the total number of registered members we just use COUNT function for the member\_ID from member table.

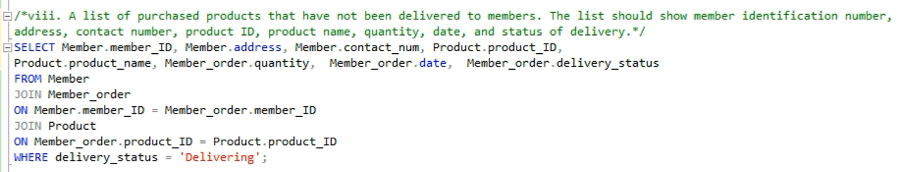
Graphical user interface, text, application, email

Description automatically generated

*Figure 5.7.2 Query of Question 7*

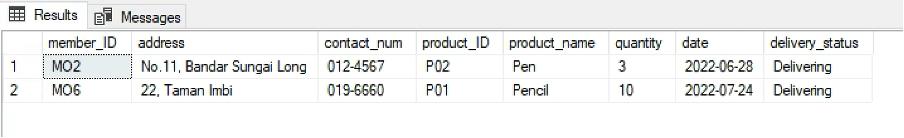
Louis Soin Keng Yong (TP064103)

Question 8:

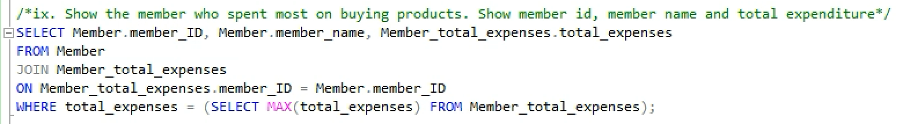
*Figure 5.8.1 Query of Question 8*

Answer:

The question asked us list out the purchased products that **have not been delivered** to members. So the condition should be delivering. Then we pick up these data “ *member\_ID*, *address*, *contact\_num, product\_ID, product\_name, quantity, date* and *delivery\_status”*. After that, we applied **JOIN** function to combine table *Member*, table *Member\_order* and table *Product* together.

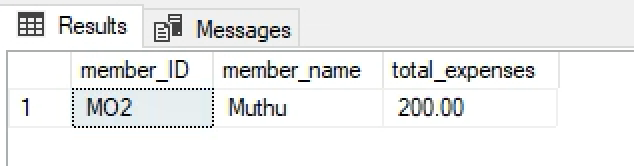
*Figure 5.8.2 Output of Question 8*

Question 9:

*Figure 5.9.1 Query of Question 9*

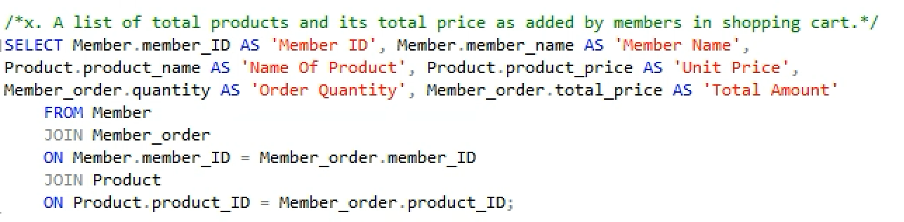
Answer:

In order to answer this question, we had to identify the person who had spent the most money on product purchases, hence the requirement is a maximum of total expenses.We pick data: *member\_ID, member\_name & total\_expenses*. **JOIN** these two tables, *Member* and *Member\_total\_expenses* together. Next, we utilise **WHERE** to determine the maximum value of the "*total expenses*”.



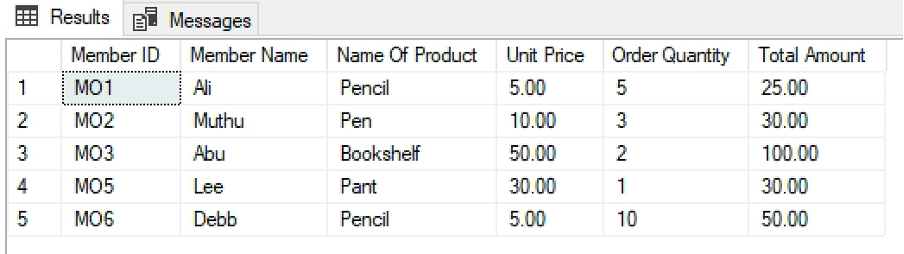
*Figure 5.9.2 Output of Question 9*

Question 10:

*Figure 5.10.1 Query of Question 10*

Answer:

This question required us to find the total products and total price that members added in shopping cart. So we need to show all of the information relevant to these requirements. Firstly, we choose *member\_ID & member\_name* from table *Member, product\_name* & *product\_price* from table *Product and quantity & total\_price* from table *Member\_order*. After that we labelled all of these names and use **JOIN** function to combine the table *Member* to the table *Member\_order* by relationship *member\_ID*. Finally we combine the result to another table name *Product* by relationship *product\_ID*, we shall have the final result as the following below:

*Figure 5.10.2 Output of Question 10*

# 6.0 Workload Matrix

|  |  |
| --- | --- |
| **Name** | **Task** |
| Kong Jun Hinn (TP064398) | Data Schema |
| Data Dictionary |
| SQL – Data Definition Language (DDL) |
| SQL-Data Manipulation Language (DML)  Question 1 - 2 |
| Koh Hong Kai (TP064049) | Data Diagram |
| SQL-Data Manipulation Language (DML)  Question 3 - 4 |
| Lee Jia Heng (TP063938) | SQL-Data Manipulation Language (DML)  Question 5 - 7 |
| Louis Soin Keng Yong (TP064103) | SQL-Data Manipulation Language (DML)  Question 8 - 10 |