Author: Ngu Hui En

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Overview of the Topic

The impact of COVID-19 has impacted many businesses including the physical bookstore. Since 2018, the Malaysian Book Publishers Association stated that 95 bookstores have been closed down which includes 32 outlets being closed down in 2020 (Daim and Krishnan, 2022). With the evolution of technology and e-commerce, e-bookstore played an important role for profitability, business and marketing strategies. The president of Malaysian Book Publishers Association expected that it will be a positive growth trend for book sales through ecommerce (Malek, 2020). The popular bookstore chains in Malaysia - MPH started to adopt online sales strategies in order to gain more customers (Daim and Krishnan, 2022).

Based on the case study, a bookstore which is located in Kuala Lumpur Malaysia expands their operation by launching an online store. This is to expand their business and to facilitate the book order to more customers. The operation of the bookstore business involves buying various genres of books from publishers and then selling them to their customers. The aim of operating a-bookstore is to provide 24/7 shopping experience to customers while improving the effectiveness and efficiency of book orders. In order to achieve the objective, an e-bookstore database system is needed to be designed and implemented for achieving the massive data including books, members, orders, payments and shipment.

1.2 Advantages of Database and Database Management System

A database is defined as an interrelated data set where insertion, deletion, extraction can be performed (Sharma et al., 2022). A database management system (DBSM) is known as an application which functions in management and storage of the data (Sharma et al., 2022). A well-designed database system allows data integration. Various data sources can be integrated into a centralised system. The features of being unique and data validation ensure the accuracy and consistency of data throughout the system. This improves the management of book inventory which avoids understocking or overstocking. DBMS maintains the data integrity as insertion, deletion, updating, extraction of data can be performed. Hence, data searching can be easily performed to search particular information. Besides, DBMS provides data security such as backup, safeguard of data from unauthorised access, encryption and data corruption. Other than that, data scalability is achieved through DBMS. It can be scaled up and down to accommodate varying workloads and data volumes. It helps to accommodate the growing business needs of the bookstore. In summary, databases and DBMS provide a reliable, secure and efficient way in managing large data volumes, which is beneficial for e-bookstore database systems.

1.3 Objectives

The objective of this project is to use Microsoft SQL Server to construct a database system for the e-bookstore. The next objective is to determine the advantages of database and database management system for e-bookstore.

CHAPTER 2

DESIGN OF THE DATABASE

2.1 List of Business Rules

Below are the lists of business rules relevant to the database:

1. A customer must register as a member to purchase books from the online store.
2. A member can make one, zero or multiple orders to the bookstore.
3. A member can buy one or multiple books in one order.
4. Bookstores can buy one or multiple books in one order.
5. An order must be associated with one and only one member.
6. A publisher can publish zero, one or multiple books.
7. Bookstores can make one, zero or multiple orders to publishers.
8. A payment must be associated with one and only one order.
9. Maximum one review per member per book is allowed.
10. A book can have zero, one or multiple reviews from different members.
11. Each book can only belong to only one category.
12. A delivery is associated with just one order.
13. A shopping cart can have many books.
14. One category ID can have many books.
15. One genre can have many books.

2.2 Entity relationship Diagram using Crow's foot notation

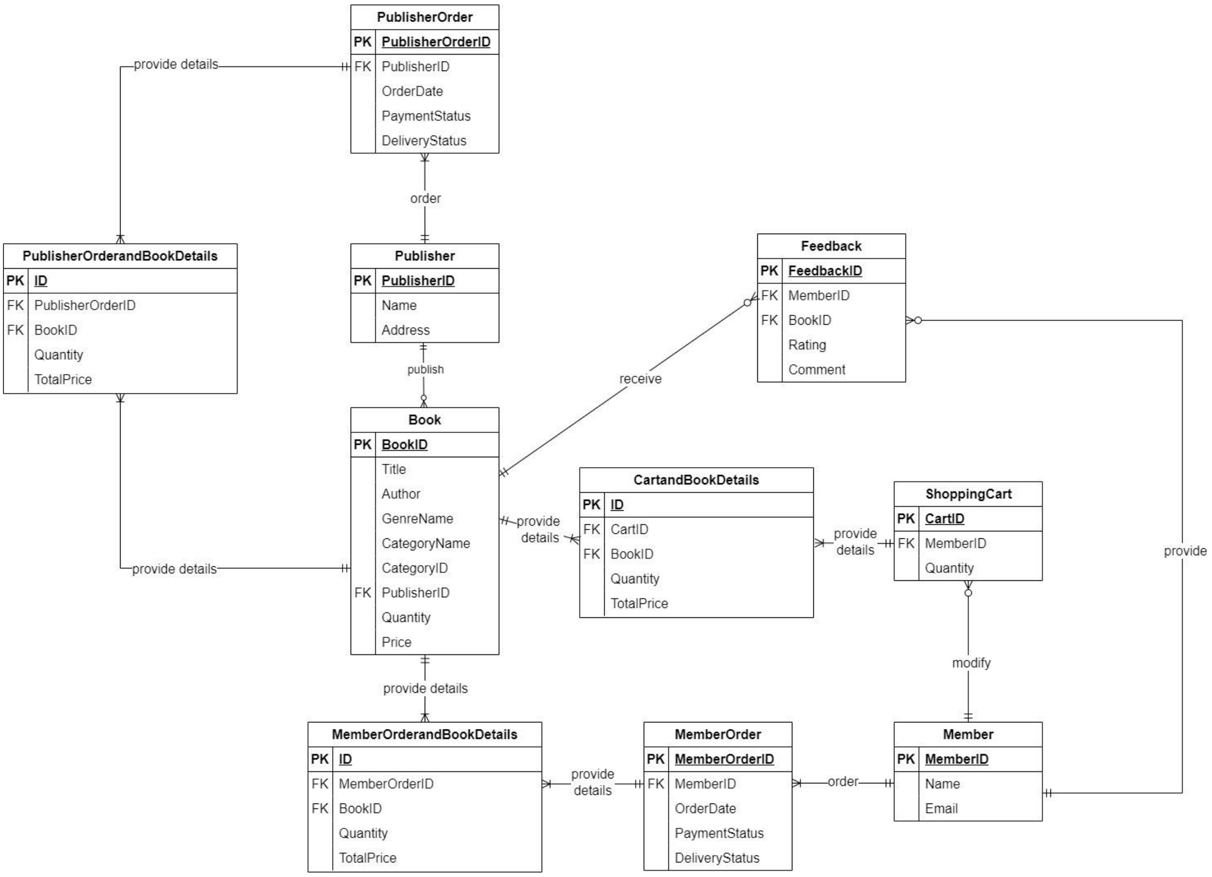


Figure 2.2.1. Entity relationship diagram of E-book database system using Crow's foot notation.

2.3 Database Diagram

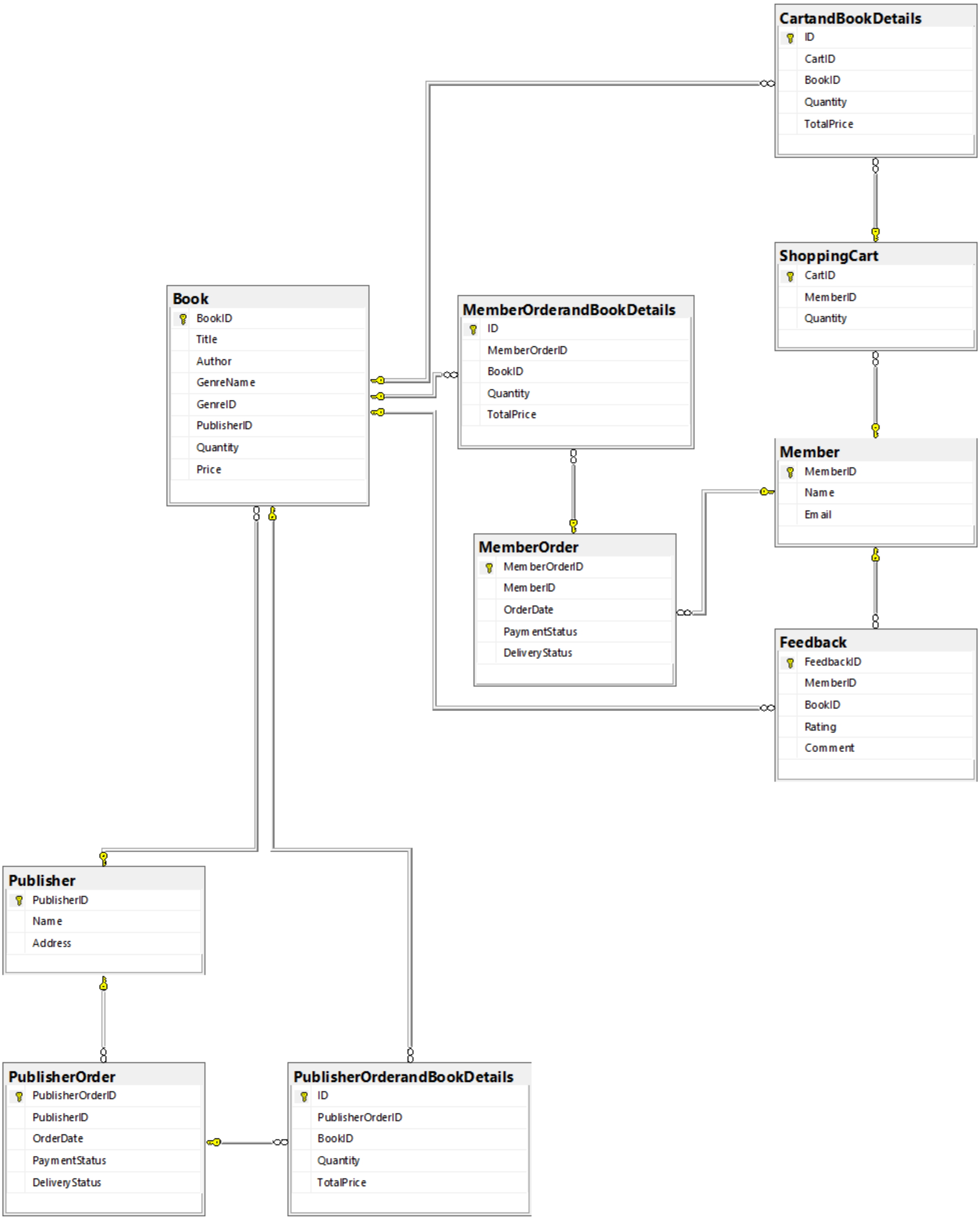


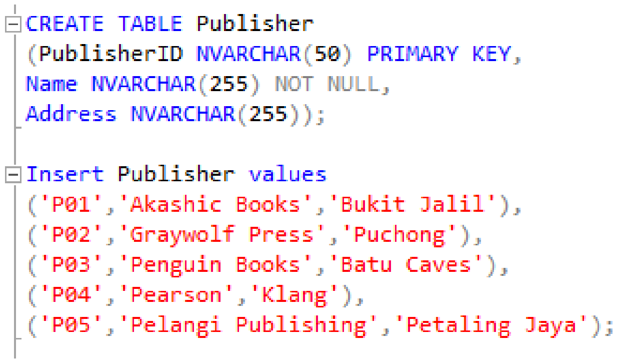
Figure 2.3.1. The database diagram of E-book database system generated by Microsoft SQL Server.

CHAPTER 3

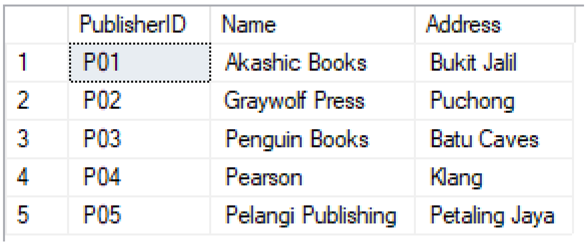
SQL STATMENTS

3.1 Data Definition Language

3.1.1 Publisher

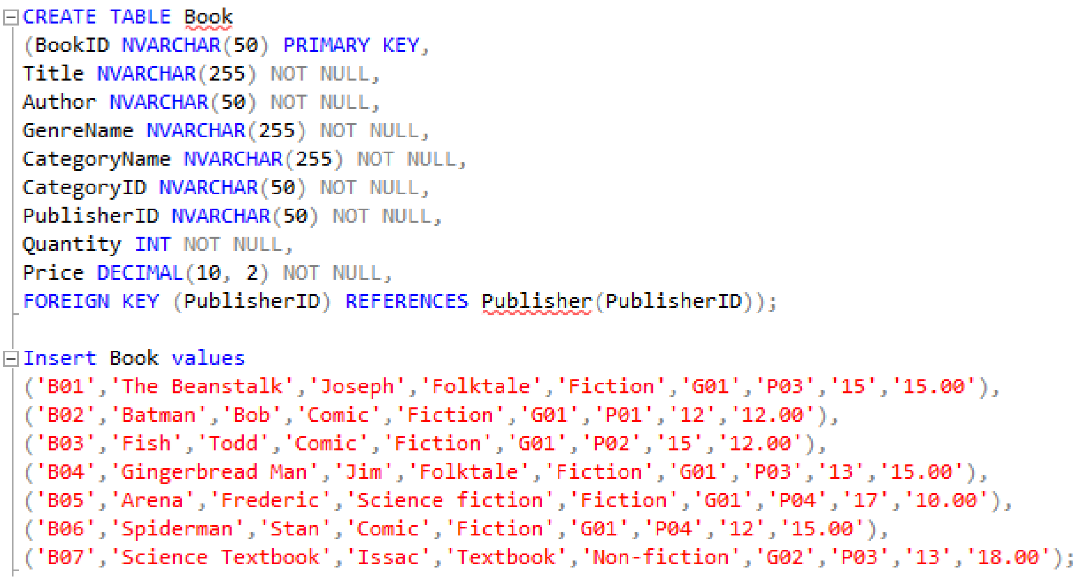


```
CREATE TABLE Publisher
(PublisherID NVARCHAR(50) PRIMARY KEY,
Name NVARCHAR(255) NOT NULL,
Address NVARCHAR(255));

Insert Publisher values
('P01','Akashic Books','Bukit Jalil'),
('P02','Graywolf Press','Puchong'),
('P03','Penguin Books','Batu Caves'),
('P04','Pearson','Klang'),
('P05','Pelangi Publishing','Petaling Jaya');
```

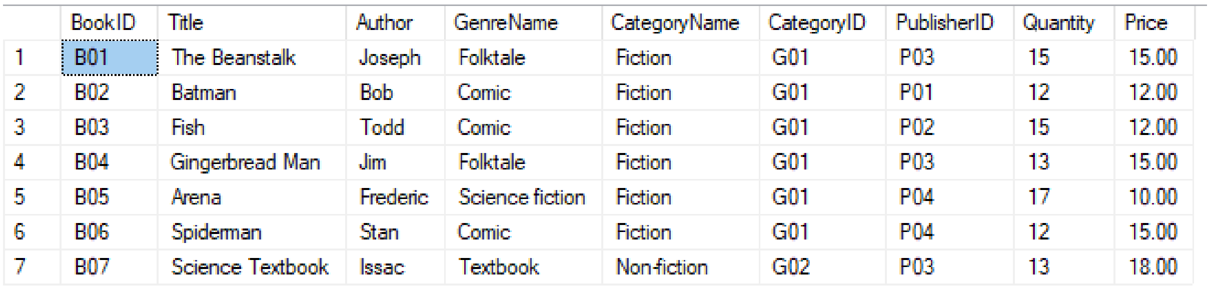


	PublisherID	Name	Address
1	P01	Akashic Books	Bukit Jalil
2	P02	Graywolf Press	Puchong
3	P03	Penguin Books	Batu Caves
4	P04	Pearson	Klang
5	P05	Pelangi Publishing	Petaling Jaya

3.1.2 Book

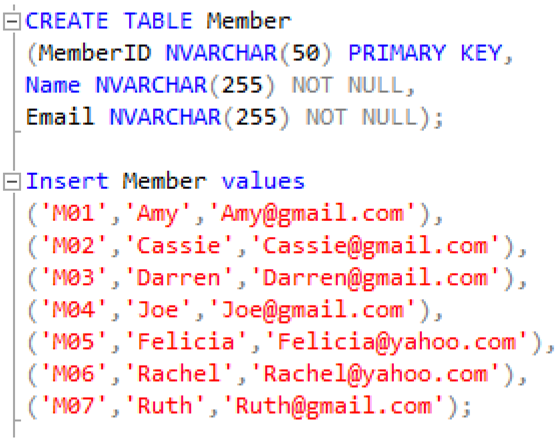


```
CREATE TABLE Book
(BookID NVARCHAR(50) PRIMARY KEY,
Title NVARCHAR(255) NOT NULL,
Author NVARCHAR(50) NOT NULL,
GenreName NVARCHAR(255) NOT NULL,
CategoryName NVARCHAR(255) NOT NULL,
CategoryID NVARCHAR(50) NOT NULL,
PublisherID NVARCHAR(50) NOT NULL,
Quantity INT NOT NULL,
Price DECIMAL(10, 2) NOT NULL,
FOREIGN KEY (PublisherID) REFERENCES Publisher(PublisherID));

Insert Book values
('B01','The Beanstalk','Joseph','Folktale','Fiction','G01','P03','15','15.00'),
('B02','Batman','Bob','Comic','Fiction','G01','P01','12','12.00'),
('B03','Fish','Todd','Comic','Fiction','G01','P02','15','12.00'),
('B04','Gingerbread Man','Jim','Folktale','Fiction','G01','P03','13','15.00'),
('B05','Arena','Frederic','Science fiction','Fiction','G01','P04','17','10.00'),
('B06','Spiderman','Stan','Comic','Fiction','G01','P04','12','15.00'),
('B07','Science Textbook','Issac','Textbook','Non-fiction','G02','P03','13','18.00');
```

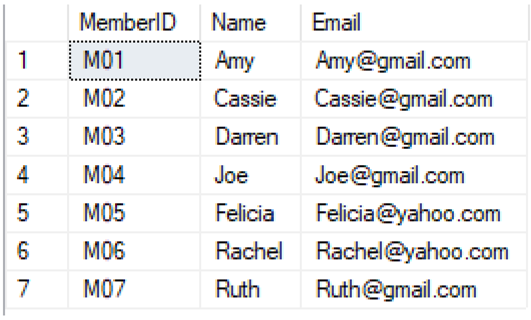


	BookID	Title	Author	GenreName	CategoryName	CategoryID	PublisherID	Quantity	Price
1	B01	The Beanstalk	Joseph	Folktale	Fiction	G01	P03	15	15.00
2	B02	Batman	Bob	Comic	Fiction	G01	P01	12	12.00
3	B03	Fish	Todd	Comic	Fiction	G01	P02	15	12.00
4	B04	Gingerbread Man	Jim	Folktale	Fiction	G01	P03	13	15.00
5	B05	Arena	Frederic	Science fiction	Fiction	G01	P04	17	10.00
6	B06	Spiderman	Stan	Comic	Fiction	G01	P04	12	15.00
7	B07	Science Textbook	Issac	Textbook	Non-fiction	G02	P03	13	18.00

3.1.3 Member

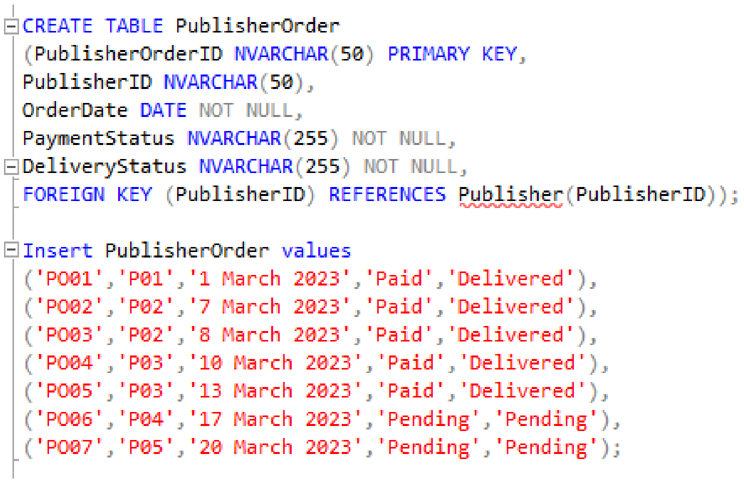


```
CREATE TABLE Member
(MemberID NVARCHAR(50) PRIMARY KEY,
Name NVARCHAR(255) NOT NULL,
Email NVARCHAR(255) NOT NULL);

Insert Member values
('M01','Amy','Amy@gmail.com'),
('M02','Cassie','Cassie@gmail.com'),
('M03','Darren','Darren@gmail.com'),
('M04','Joe','Joe@gmail.com'),
('M05','Felicia','Felicia@yahoo.com'),
('M06','Rachel','Rachel@yahoo.com'),
('M07','Ruth','Ruth@gmail.com');
```

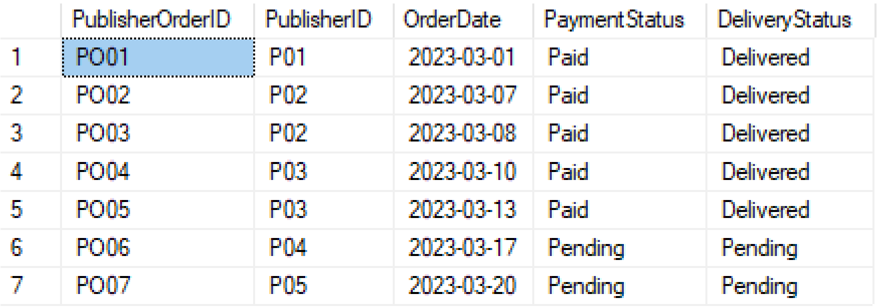


	MemberID	Name	Email
1	M01	Amy	Amy@gmail.com
2	M02	Cassie	Cassie@gmail.com
3	M03	Darren	Darren@gmail.com
4	M04	Joe	Joe@gmail.com
5	M05	Felicia	Felicia@yahoo.com
6	M06	Rachel	Rachel@yahoo.com
7	M07	Ruth	Ruth@gmail.com

3.1.4 Publisher Order

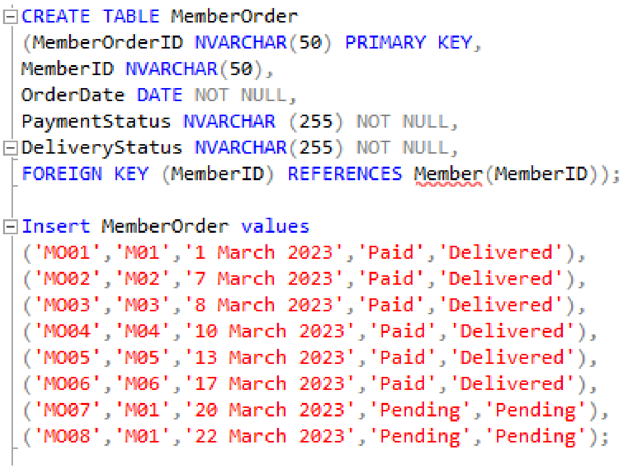


```
CREATE TABLE PublisherOrder
(PublisherOrderID NVARCHAR(50) PRIMARY KEY,
PublisherID NVARCHAR(50),
OrderDate DATE NOT NULL,
PaymentStatus NVARCHAR(255) NOT NULL,
DeliveryStatus NVARCHAR(255) NOT NULL,
FOREIGN KEY (PublisherID) REFERENCES Publisher(PublisherID));

Insert PublisherOrder values
('PO01','P01','1 March 2023','Paid','Delivered'),
('PO02','P02','7 March 2023','Paid','Delivered'),
('PO03','P02','8 March 2023','Paid','Delivered'),
('PO04','P03','10 March 2023','Paid','Delivered'),
('PO05','P03','13 March 2023','Paid','Delivered'),
('PO06','P04','17 March 2023','Pending','Pending'),
('PO07','P05','20 March 2023','Pending','Pending');
```

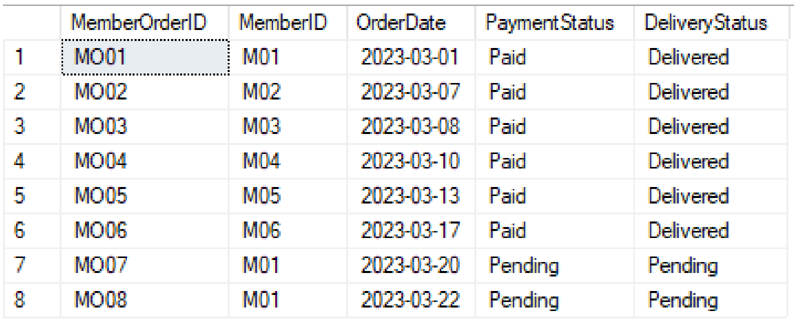


	PublisherOrderID	PublisherID	OrderDate	PaymentStatus	DeliveryStatus
1	PO01	P01	2023-03-01	Paid	Delivered
2	PO02	P02	2023-03-07	Paid	Delivered
3	PO03	P02	2023-03-08	Paid	Delivered
4	PO04	P03	2023-03-10	Paid	Delivered
5	PO05	P03	2023-03-13	Paid	Delivered
6	PO06	P04	2023-03-17	Pending	Pending
7	PO07	P05	2023-03-20	Pending	Pending

3.1.5 Member Order

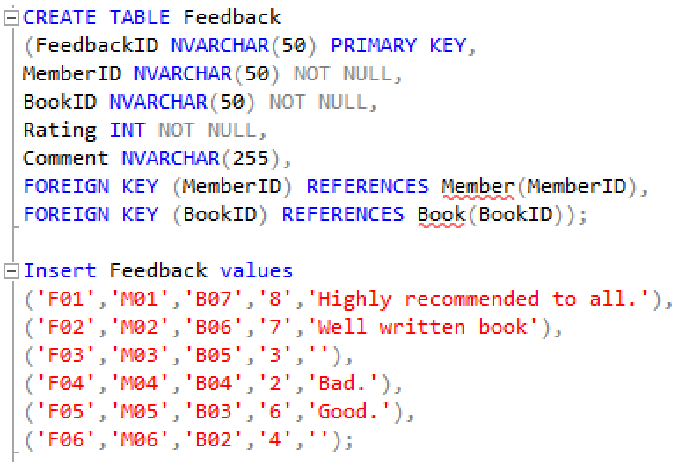


```
CREATE TABLE MemberOrder
(MemberOrderID NVARCHAR(50) PRIMARY KEY,
MemberID NVARCHAR(50),
OrderDate DATE NOT NULL,
PaymentStatus NVARCHAR (255) NOT NULL,
DeliveryStatus NVARCHAR(255) NOT NULL,
FOREIGN KEY (MemberID) REFERENCES Member(MemberID));

Insert MemberOrder values
('MO01','M01','1 March 2023','Paid','Delivered'),
('MO02','M02','7 March 2023','Paid','Delivered'),
('MO03','M03','8 March 2023','Paid','Delivered'),
('MO04','M04','10 March 2023','Paid','Delivered'),
('MO05','M05','13 March 2023','Paid','Delivered'),
('MO06','M06','17 March 2023','Paid','Delivered'),
('MO07','M01','20 March 2023','Pending','Pending'),
('MO08','M01','22 March 2023','Pending','Pending');
```

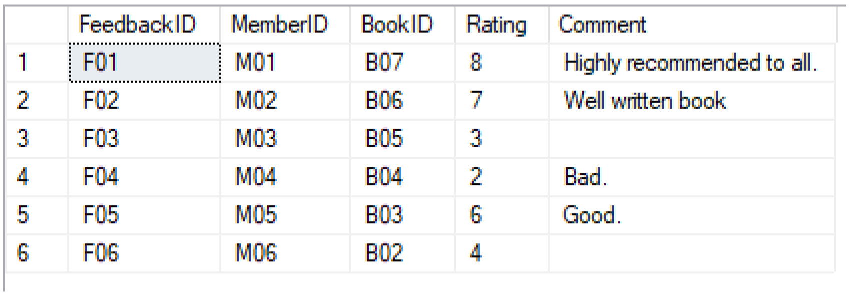


	MemberOrderID	MemberID	OrderDate	PaymentStatus	DeliveryStatus
1	MO01	M01	2023-03-01	Paid	Delivered
2	MO02	M02	2023-03-07	Paid	Delivered
3	MO03	M03	2023-03-08	Paid	Delivered
4	MO04	M04	2023-03-10	Paid	Delivered
5	MO05	M05	2023-03-13	Paid	Delivered
6	MO06	M06	2023-03-17	Paid	Delivered
7	MO07	M01	2023-03-20	Pending	Pending
8	MO08	M01	2023-03-22	Pending	Pending

3.1.6 Feedback

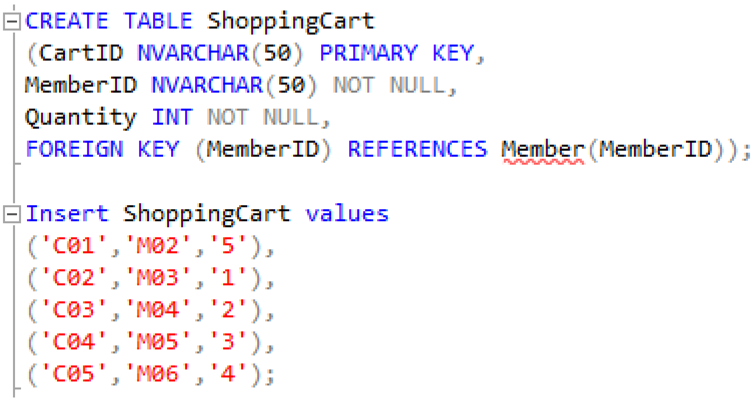


```
CREATE TABLE Feedback
(FeedbackID NVARCHAR(50) PRIMARY KEY,
MemberID NVARCHAR(50) NOT NULL,
BookID NVARCHAR(50) NOT NULL,
Rating INT NOT NULL,
Comment NVARCHAR(255),
FOREIGN KEY (MemberID) REFERENCES Member(MemberID),
FOREIGN KEY (BookID) REFERENCES Book(BookID));

Insert Feedback values
('F01','M01','B07','8','Highly recommended to all.'),
('F02','M02','B06','7','Well written book'),
('F03','M03','B05','3',''),
('F04','M04','B04','2','Bad.'),
('F05','M05','B03','6','Good.'),
('F06','M06','B02','4','');
```

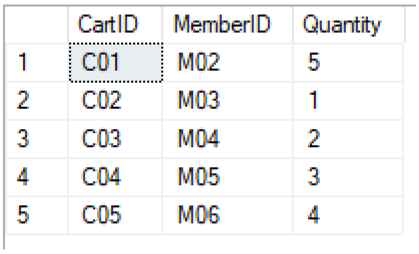


	FeedbackID	MemberID	BookID	Rating	Comment
1	F01	M01	B07	8	Highly recommended to all.
2	F02	M02	B06	7	Well written book
3	F03	M03	B05	3	
4	F04	M04	B04	2	Bad.
5	F05	M05	B03	6	Good.
6	F06	M06	B02	4	

3.1.7 Shopping Cart

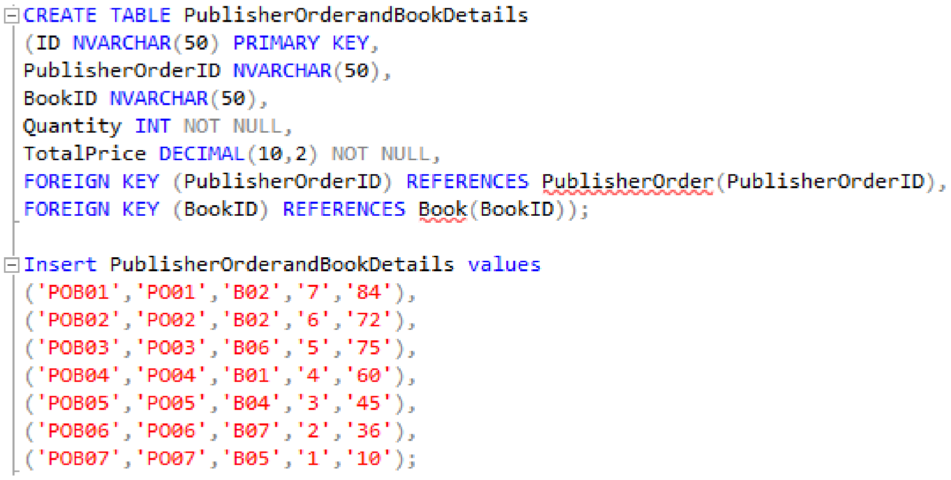


```
CREATE TABLE ShoppingCart
(CartID NVARCHAR(50) PRIMARY KEY,
MemberID NVARCHAR(50) NOT NULL,
Quantity INT NOT NULL,
FOREIGN KEY (MemberID) REFERENCES Member(MemberID));

Insert ShoppingCart values
('C01','M02','5'),
('C02','M03','1'),
('C03','M04','2'),
('C04','M05','3'),
('C05','M06','4');
```

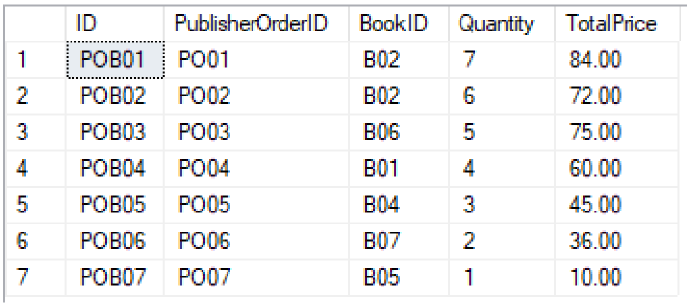


	CartID	MemberID	Quantity
1	C01	M02	5
2	C02	M03	1
3	C03	M04	2
4	C04	M05	3
5	C05	M06	4

3.1.8 Publisher Order and Book Details

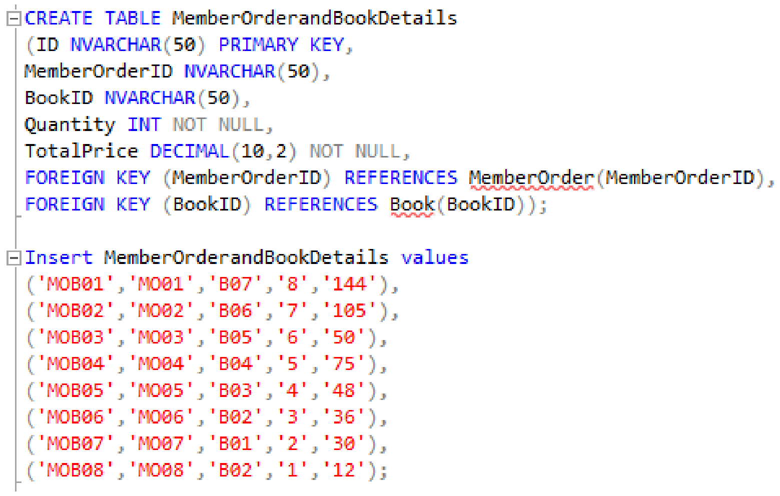


```
CREATE TABLE PublisherOrderandBookDetails
(ID NVARCHAR(50) PRIMARY KEY,
PublisherOrderID NVARCHAR(50),
BookID NVARCHAR(50),
Quantity INT NOT NULL,
TotalPrice DECIMAL(10,2) NOT NULL,
FOREIGN KEY (PublisherOrderID) REFERENCES PublisherOrder(PublisherOrderID),
FOREIGN KEY (BookID) REFERENCES Book(BookID));

Insert PublisherOrderandBookDetails values
('POB01','PO01','B02','7','84'),
('POB02','PO02','B02','6','72'),
('POB03','PO03','B06','5','75'),
('POB04','PO04','B01','4','60'),
('POB05','PO05','B04','3','45'),
('POB06','PO06','B07','2','36'),
('POB07','PO07','B05','1','10');
```

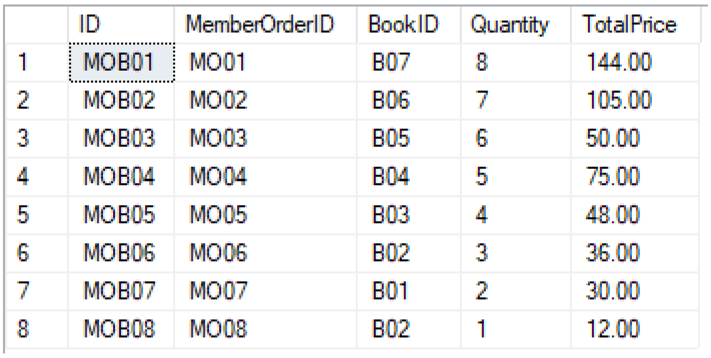


	ID	PublisherOrderID	BookID	Quantity	TotalPrice
1	POB01	PO01	B02	7	84.00
2	POB02	PO02	B02	6	72.00
3	POB03	PO03	B06	5	75.00
4	POB04	PO04	B01	4	60.00
5	POB05	PO05	B04	3	45.00
6	POB06	PO06	B07	2	36.00
7	POB07	PO07	B05	1	10.00

3.1.9 Member Order and Book Details

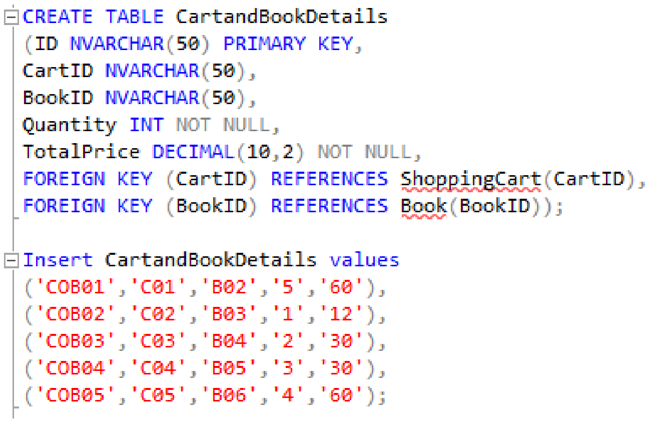


```
CREATE TABLE MemberOrderandBookDetails
(ID NVARCHAR(50) PRIMARY KEY,
MemberOrderID NVARCHAR(50),
BookID NVARCHAR(50),
Quantity INT NOT NULL,
TotalPrice DECIMAL(10,2) NOT NULL,
FOREIGN KEY (MemberOrderID) REFERENCES MemberOrder(MemberOrderID),
FOREIGN KEY (BookID) REFERENCES Book(BookID));

Insert MemberOrderandBookDetails values
('MOB01','MO01','B07','8','144'),
('MOB02','MO02','B06','7','105'),
('MOB03','MO03','B05','6','50'),
('MOB04','MO04','B04','5','75'),
('MOB05','MO05','B03','4','48'),
('MOB06','MO06','B02','3','36'),
('MOB07','MO07','B01','2','30'),
('MOB08','MO08','B02','1','12');
```

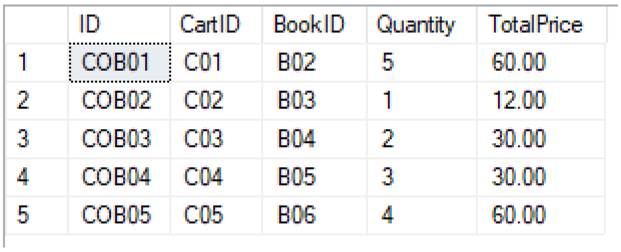


	ID	MemberOrderID	BookID	Quantity	TotalPrice
1	MOB01	MO01	B07	8	144.00
2	MOB02	MO02	B06	7	105.00
3	MOB03	MO03	B05	6	50.00
4	MOB04	MO04	B04	5	75.00
5	MOB05	MO05	B03	4	48.00
6	MOB06	MO06	B02	3	36.00
7	MOB07	MO07	B01	2	30.00
8	MOB08	MO08	B02	1	12.00

3.1.10 Shopping Cart and Book Details



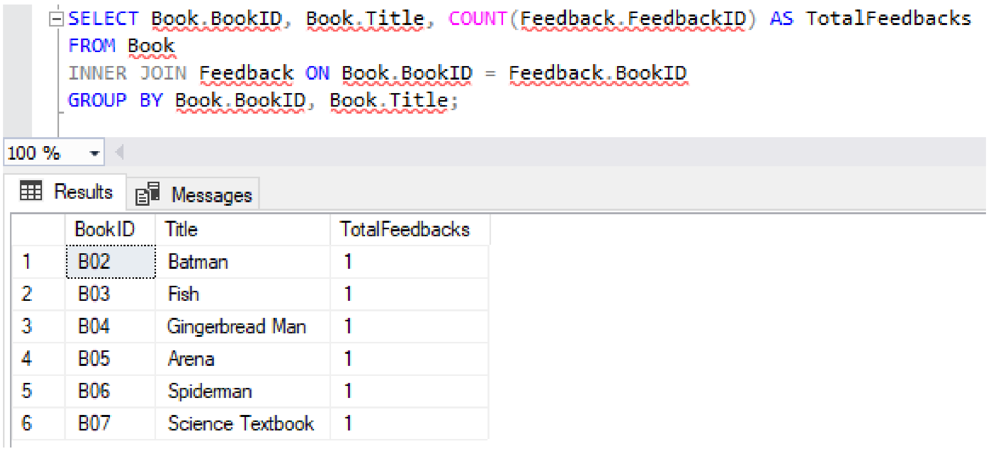
```
CREATE TABLE CartandBookDetails
(ID NVARCHAR(50) PRIMARY KEY,
CartID NVARCHAR(50),
BookID NVARCHAR(50),
Quantity INT NOT NULL,
TotalPrice DECIMAL(10,2) NOT NULL,
FOREIGN KEY (CartID) REFERENCES ShoppingCart(CartID),
FOREIGN KEY (BookID) REFERENCES Book(BookID));

Insert CartandBookDetails values
('COB01','C01','B02','5','60'),
('COB02','C02','B03','1','12'),
('COB03','C03','B04','2','30'),
('COB04','C04','B05','3','30'),
('COB05','C05','B06','4','60');
```



	ID	CartID	BookID	Quantity	TotalPrice
1	COB01	C01	B02	5	60.00
2	COB02	C02	B03	1	12.00
3	COB03	C03	B04	2	30.00
4	COB04	C04	B05	3	30.00
5	COB05	C05	B06	4	60.00

3.2 Data Manipulation Language

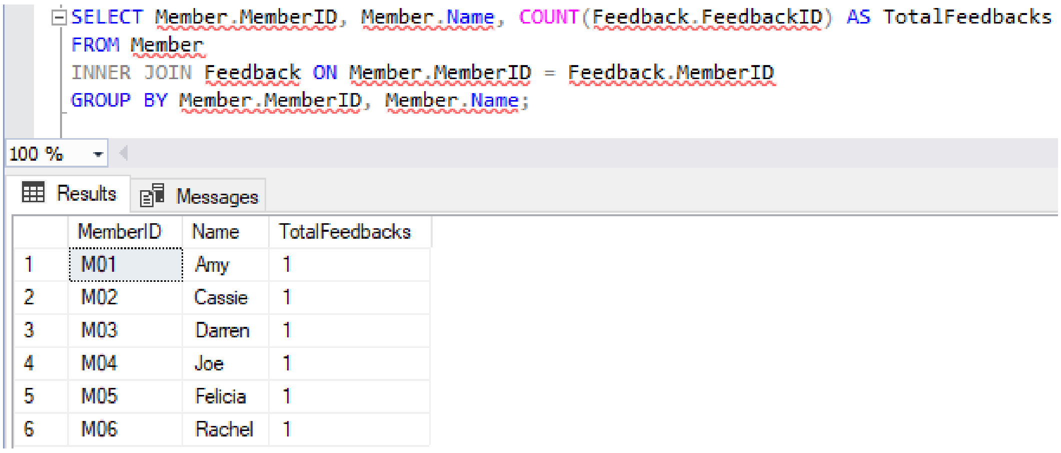
3.2.1 Find the total number of feedbacks per book. Show book id, book title, and total number of feedbacks per book



```
SELECT Book.BookID, Book.Title, COUNT(Feedback.FeedbackID) AS TotalFeedbacks
FROM Book
INNER JOIN Feedback ON Book.BookID = Feedback.BookID
GROUP BY Book.BookID, Book.Title;
```

	BookID	Title	TotalFeedbacks
1	B02	Batman	1
2	B03	Fish	1
3	B04	Gingerbread Man	1
4	B05	Arena	1
5	B06	Spiderman	1
6	B07	Science Textbook	1

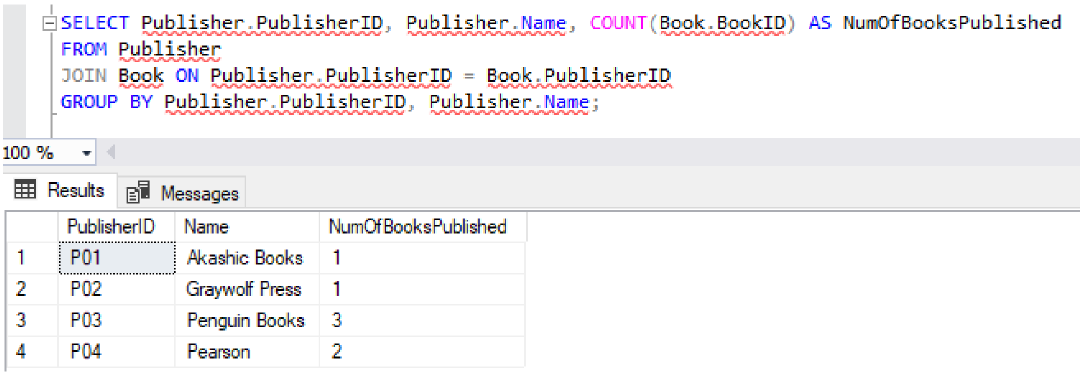
3.2.2 Find the total number of feedbacks per member. Show member id, member name, and total number of feedbacks per member



```
SELECT Member.MemberID, Member.Name, COUNT(Feedback.FeedbackID) AS TotalFeedbacks
FROM Member
INNER JOIN Feedback ON Member.MemberID = Feedback.MemberID
GROUP BY Member.MemberID, Member.Name;
```

	MemberID	Name	TotalFeedbacks
1	M01	Amy	1
2	M02	Cassie	1
3	M03	Darren	1
4	M04	Joe	1
5	M05	Felicia	1
6	M06	Rachel	1

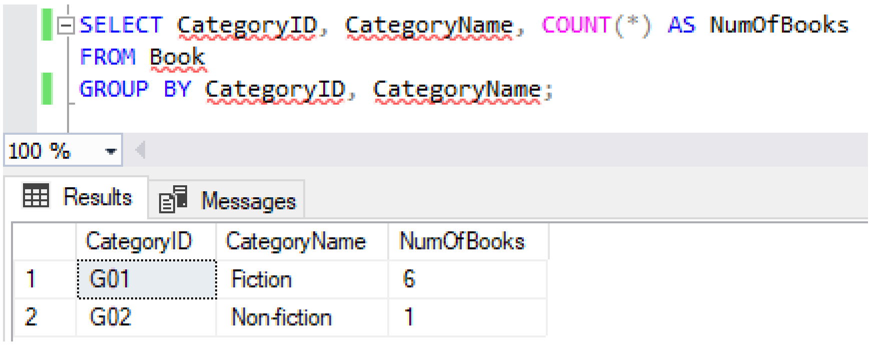
3.2.3 Find the total number of books published by each publisher. Show publisher id, publisher name, and number of books published



```
SELECT Publisher.PublisherID, Publisher.Name, COUNT(Book.BookID) AS NumOfBooksPublished
FROM Publisher
JOIN Book ON Publisher.PublisherID = Book.PublisherID
GROUP BY Publisher.PublisherID, Publisher.Name;
```

	PublisherID	Name	NumOfBooksPublished
1	P01	Akashic Books	1
2	P02	Graywolf Press	1
3	P03	Penguin Books	3
4	P04	Pearson	2

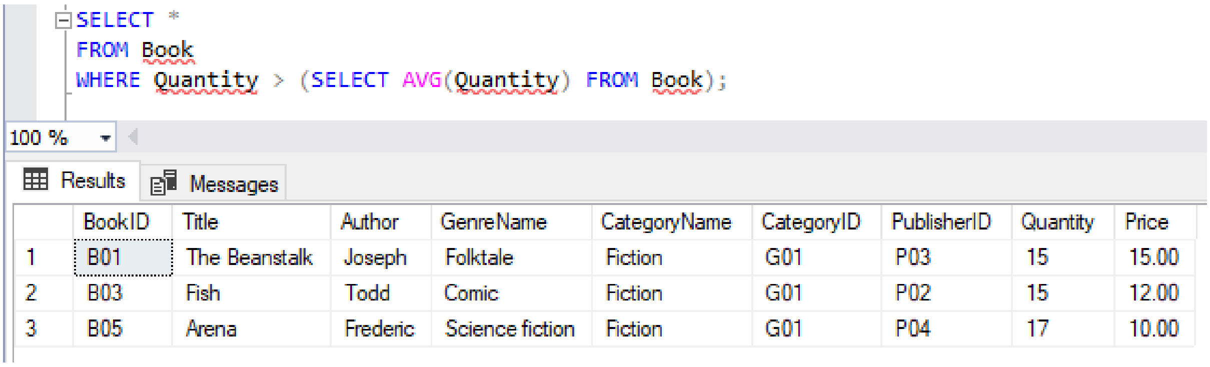
3.2.4 Find the total number of books for each category. Show category id, category name, and number of books for each category



```
SELECT CategoryID, CategoryName, COUNT(*) AS NumOfBooks
FROM Book
GROUP BY CategoryID, CategoryName;
```

	CategoryID	CategoryName	NumOfBooks
1	G01	Fiction	6
2	G02	Non-fiction	1

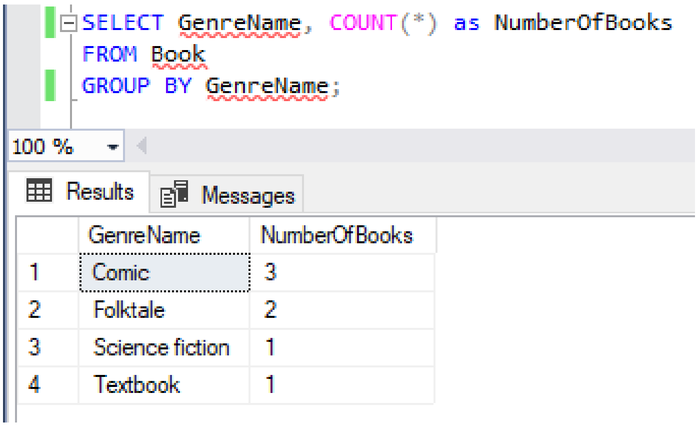
3.2.5 From the book table, list the books where quantity is more than the average quantity of all books



```
SELECT *
FROM Book
WHERE Quantity > (SELECT AVG(Quantity) FROM Book);
```

	BookID	Title	Author	GenreName	CategoryName	CategoryID	PublisherID	Quantity	Price
1	B01	The Beanstalk	Joseph	Folktale	Fiction	G01	P03	15	15.00
2	B03	Fish	Todd	Comic	Fiction	G01	P02	15	12.00
3	B05	Arena	Frederic	Science fiction	Fiction	G01	P04	17	10.00

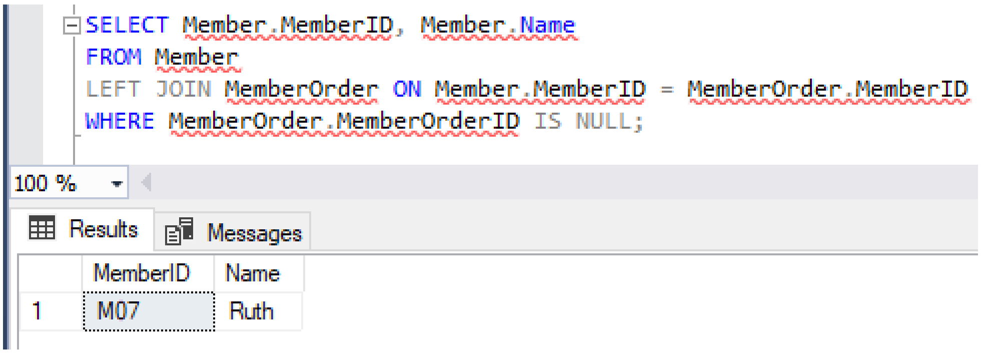
3.2.6 Show how many books are there for each genre



```
SELECT GenreName, COUNT(*) as NumberOfBooks
FROM Book
GROUP BY GenreName;
```

	GenreName	NumberOfBooks
1	Comic	3
2	Folktale	2
3	Science fiction	1
4	Textbook	1

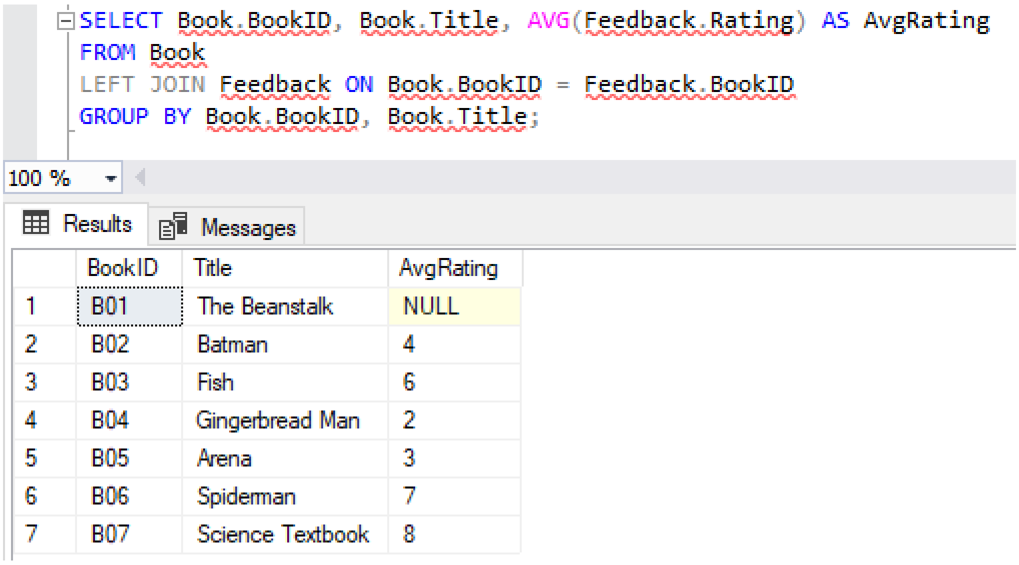
3.2.7 Show the members who did not make any order



```
SELECT Member.MemberID, Member.Name
FROM Member
LEFT JOIN MemberOrder ON Member.MemberID = MemberOrder.MemberID
WHERE MemberOrder.MemberOrderID IS NULL;
```

	MemberID	Name
1	M07	Ruth

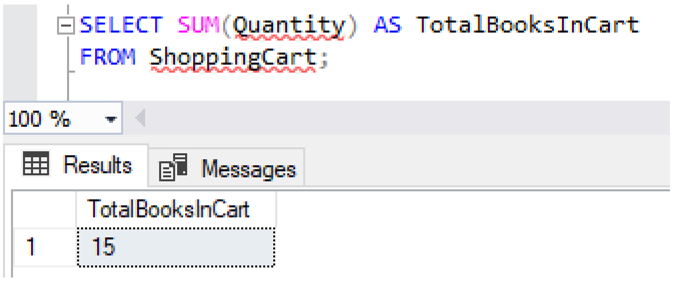
3.2.8 Find the average rating for each book



```
SELECT Book.BookID, Book.Title, AVG(Feedback.Rating) AS AvgRating
FROM Book
LEFT JOIN Feedback ON Book.BookID = Feedback.BookID
GROUP BY Book.BookID, Book.Title;
```

	BookID	Title	AvgRating
1	B01	The Beanstalk	NULL
2	B02	Batman	4
3	B03	Fish	6
4	B04	Gingerbread Man	2
5	B05	Arena	3
6	B06	Spiderman	7
7	B07	Science Textbook	8

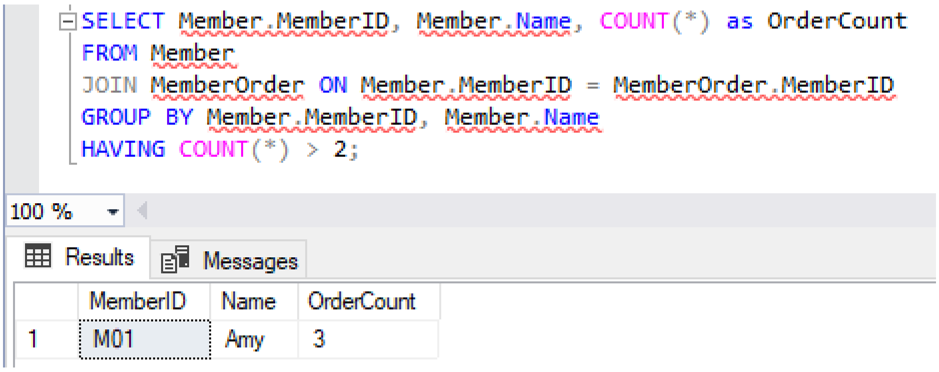
3.2.9 Show the total number of books added to the shopping cart



```
SELECT SUM(Quantity) AS TotalBooksInCart
FROM ShoppingCart;
```

	TotalBooksInCart
1	15

3.2.10 Show the members who made more than 2 orders



```
SELECT Member.MemberID, Member.Name, COUNT(*) as OrderCount
FROM Member
JOIN MemberOrder ON Member.MemberID = MemberOrder.MemberID
GROUP BY Member.MemberID, Member.Name
HAVING COUNT(*) > 2;
```

	MemberID	Name	OrderCount
1	M01	Amy	3

CHAPTER 4

CONCLUSION

The objective of this project has been achieved. A database system for the e-bookstore has been constructed using Microsoft SQL Server. The advantages of database and DBMS are data redundancy, data integrity, data searching, data security, data scalability.

REFERENCES

Daim, N. and Krishnan, D.B. (2022, May 21). Book industry must embrace change to stay relevant. *New Straits Times.* https://www.nst.com.my/news/nation/2022/05/798048/book-industry-must-embrace-change-stay-relevant

Malek, N.H.A. (2020, June 9). Book industry takes a hit as bookstores close amid MCO. *The Malaysian Reserve.* https://themalaysianreserve.com/2020/06/09/book-industry-takes-a-hit-as-bookstores-close-amid-mco/

Sharma, A., Karamchandani, A., Dave, D., Patel, A., & Doshi, N. (2022). Database Management Systems—An Efficient, Effective, and Augmented Approach for Organizations. In *ICT with Intelligent Applications: Proceedings of ICTIS 2021, Volume 1* (pp. 465-478). Springer Singapore.