

**GROUP ASSIGNMENT (PART2).**

**TECHNOLOGY PARK MALAYSIA.**

**CT042-3-1-IDB.**

**INTRODUCTION TO DATABASE.**

**APD1F2209CS(CYB).**

**HAND-OUT DATE: WEEK 3.**

**HAND-IN DATE: WEEK 13.**

**WEIGHTAGE: 60%.**

|  |  |
| --- | --- |
| ***TP\_NUMBER:*** | ***STUDENT NAME:*** |
| *TP065857* | *TERENCE LING CHEE YEW* |
| *TP065775* | *NAIM NASRUDDIN BIN NAZRI* |
| *TP071534* | *OOI HONG PING* |
| *TP071012* | *ABDULRAHMAN GAMIL MOHAMMED AHMED* |

Contents

[**1** **Database Schema.** 3](#_Toc137332770)

[**1.1** **Entity Relationship Diagram.** 3](#_Toc137332771)

[**1.2** **Generated Database Diagram from DMBS.** 3](#_Toc137332772)

[**2** **SQL- Data Definition Language (DDL).** 4](#_Toc137332773)

[**2.1** **Creating Database & Tables and Insert the Data.** 4](#_Toc137332774)

[**3** **SQL-Data Manipulation Language (DML).** 12](#_Toc137332775)

[**3.1** **Q1: Retrieve the names of members who have more than 2 books on loan.** 12](#_Toc137332776)

[**3.2** **Q2: Count the total number of books in each category and sort them in descending order based on the count.** 12](#_Toc137332777)

[**3.3** **Q3: Find the genre(s) that have the highest number of books.** 12](#_Toc137332778)

[**3.4** **Q4: Get the book ID and name for books that either have no reservation or have a reservation status of 'Available'.** 13](#_Toc137332779)

[**3.5** **Q5: Retrieve the member ID and name for members who have borrowed more than 2 books.** 13](#_Toc137332780)

[**3.6** **Q6: Get the book name and author for books whose author's name contains a comma.** 14](#_Toc137332781)

[**3.7** **Q7: Find the member ID and name for members who have incurred fines on their loans.** 14](#_Toc137332782)

[**3.8** **Q8: Get the copies ID and the total count of loans for copies that have the highest loan count.** 15](#_Toc137332783)

[**3.9** **Q9: Retrieve the Publisher name and the total count of books published by each publisher.** 15](#_Toc137332784)

[**3.10** **Q10: Get the book names for books that have a reservation status of 'Pending' and sort them in ascending order.** 16](#_Toc137332785)

[**4** **Work Matrix.** 17](#_Toc137332786)

# **Database Schema.**

## **Entity Relationship Diagram.**

A picture containing diagram, plan, technical drawing, line

Description automatically generated

## **Generated Database Diagram from DMBS.**

**A picture containing text, diagram, plan, parallel

Description automatically generated**

# **SQL- Data Definition Language (DDL).**

## **Creating Database & Tables and Insert the Data.**

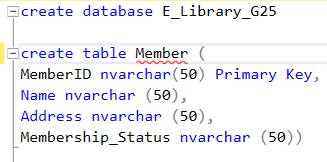


Figure (1).

In the first line of the query, we built the database so that we could create the tables, input the values, and then get the data.‎ After creating the database, we built tables. and the first table we have in our database is the Member table. It consists of four columns (MemberID, Name, Address, and Membership\_status) and each column has been identified with the data type in the query. In this table, all four columns hold the same data type which is nvarchar(50). A primary key for the MemberID is in order to know the identification of the member.

A picture containing text, font, screenshot, line

Description automatically generated

Figure (2).

In the second table, we have the ISBN table, which holds the name of the columns and the data type of it, (ISBN, Author, and Edition) the data type is nvarchar(50) for each column in the table. The Primary key for this table is the ISBN. ‎‎

A picture containing text, font, screenshot, line

Description automatically generated

Figure (3).

The Publisher table, the third table in our database, has one Primary Key that we implemented for the first column (PublisherID) so that we can identify the publisher's name and address. All the data types in this table are nvarchar(50). ‎

A picture containing text, screenshot, font

Description automatically generated

Figure (4).

The Book table contains the basic information about each book in our library. In this table, BookID serves as the Primary Key, whereas PublisherID and ISBN are stored in separate columns as Foreign Keys. The second table is referred to by ISBN, while the third table is referred to by PublisherID. and the table's entire data type is nvarchar(50). ‎ ‎

A picture containing text, font, screenshot, line

Description automatically generated

Figure (5).

This table for Reservations contains one Primary Key and two Foreign Keys for the respective values of ReservationID, MemberID, and BookID. BookID refers to the fourth table, while MemberID refers to the first table.‎‎ Only one column in this table has the date type Date (Reservation\_Date), while all other data types are referred to as nvarchar(50).‎‎‎ ‎‎

A picture containing text, font, screenshot, line

Description automatically generated

Figure (6).

CopiesID, BookID, and Available are the three columns in the Copies table. CopiesID is a Primary Key, BookID is a Foreign Key pointing to the Book table, and nvarchar(50) is the only data type used in the table.‎‎‎‎ ‎

A picture containing text, screenshot, font, line

Description automatically generated

Figure (7).

The final table in this database is for loans. It contains six columns, one of which is a Primary Key (LoanID), and two Foreign Keys, one for each MemberID and CopiesID, which refers to the Member table and the Copies table, respectively.

A picture containing text, font, screenshot, line

Description automatically generated

Figure (8).

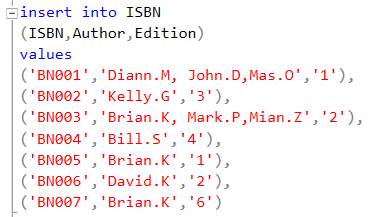


Figure (9).

A screenshot of a computer code

Description automatically generated with low confidence

Figure (10).

A screen shot of a computer

Description automatically generated with low confidence

Figure (11).

A picture containing text, font, screenshot

Description automatically generated

Figure (12).

A screenshot of a computer code

Description automatically generated with low confidence

Figure (13).

A picture containing text, font, screenshot, number

Description automatically generated

Figure (14).

The previous seven figures (8–14) demonstrate that a value was placed into each of the seven identified tables in the database. Each table has five to ten rows of data from various data types. The newly added data is derived from tables that were previously supplied from the part of the Third Normalized Form.‎

A picture containing text, screenshot, font, electric blue

Description automatically generated

Figure (15).

Figure (15) shows how to write (select \* from *Table\_Name*) to display all the database's tables. You must use the same command as it is shown in the image above for all the tables.‎

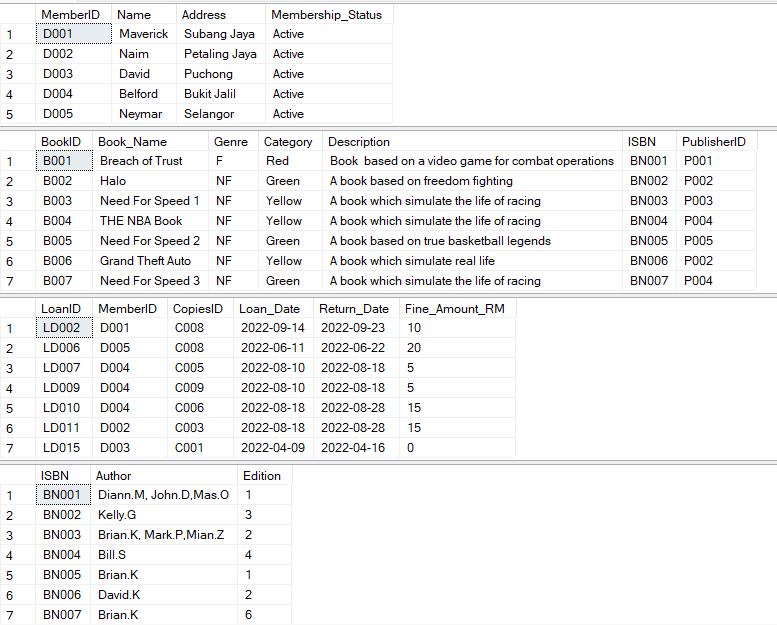


Figure (16).

A screenshot of a computer

Description automatically generated with medium confidence

Figure (17).

Additionally, using a database enables efficient documentation and management of the outcomes from the additional tables, containing their respective associated values. The previous two figures demonstrate this in a very straightforward manner. It's crucial to remember that the tables in the picture correspond to those in Figure (15). As a result, the database acts as a central location for organizing and maintaining the data obtained from various tables, making it easier to retrieve and manipulate data.

# **SQL-Data Manipulation Language (DML).**

## **Q1: Retrieve the names of members who have more than 2 books on loan.**

**A picture containing text, screenshot, font

Description automatically generated**

**A picture containing text, screenshot, font

Description automatically generated**

This query retrieves the names of members and the number of books they have on loan. It joins the **‘Member’** and **‘Loan’** tables on the **‘MemberID’** column and groups the results by member name. The **HAVING** clause filters out members who have more than 2 books on loan. The results are sorted in ascending order by member name.

## **Q2: Count the total number of books in each category and sort them in descending order based on the count.**

**A picture containing text, font, screenshot

Description automatically generated**

**A screenshot of a computer

Description automatically generated with medium confidence**

This query calculates the total number of books in each category. It selects the **‘Category’** column from the **‘Book’** table and uses the **‘COUNT (\*)’** function to count the number of books in each category. The results are grouped by category and sorted in descending order based on the total number of books.

## **Q3: Find the genre(s) that have the highest number of books.**

**A picture containing text, screenshot, font, line

Description automatically generated**

**A screenshot of a computer

Description automatically generated with medium confidence**

This query determines the genre(s) that have the highest number of books. It calculates the count of books in each genre using the **‘COUNT (\*)’** function and groups the results by genre. The **‘HAVING’** clause filters the results to include only the genre(s) with the maximum count. The subquery **‘(SELECT MAX(BookCount) FROM (SELECT COUNT (\*) AS BookCount FROM Book GROUP BY Genre) AS BookCounts)’** finds the maximum count of books across genres.

## **Q4: Get the book ID and name for books that either have no reservation or have a reservation status of 'Available'.**

**A picture containing text, screenshot, font

Description automatically generated**

**A screenshot of a computer

Description automatically generated with medium confidence**

This query retrieves the book ID and book name from the **‘Book’** table for books that either have no reservation or have a reservation status of 'Available'. It performs a left join between the **‘Book’** and **‘Reservation’** tables on the **‘BookID’** column. The **‘WHERE’** clause filters out books where either the reservation ID is null, or the reservation status is 'Available'.

## **Q5: Retrieve the member ID and name for members who have borrowed more than 2 books.**

**A picture containing text, screenshot, font

Description automatically generated**

**A screenshot of a computer

Description automatically generated with medium confidence**

This query selects the member ID and name from the **‘Member’** table for members who have borrowed more than 2 books. It joins the **‘Member’** and **‘Loan’** tables based on the **‘MemberID’** column. The results are grouped by member ID and name, and the **‘HAVING’** clause filters out members who have borrowed less than or equal to 2 books.

## **Q6: Get the book name and author for books whose author's name contains a comma.**

**A close-up of a logo

Description automatically generated with low confidence**

**A screenshot of a computer

Description automatically generated with low confidence**

This query retrieves the book name and author from the **‘Book’** table for books where the author's name contains a comma. It joins the **‘Book’** and **‘ISBN’** tables based on the ISBN column. The **‘WHERE’** clause filters out books whose author's name includes a comma.

## **Q7: Find the member ID and name for members who have incurred fines on their loans.**

**A screenshot of a website

Description automatically generated with low confidence**

**A screenshot of a computer

Description automatically generated with medium confidence**

This query selects the distinct member ID and name from the **‘Member’** table for members who have incurred fines (fine amount is not equal to 0) based on their loans. It performs an inner join between the **‘Member’** and **‘Loan’** tables using the **‘MemberID’** column. The **‘WHERE’** clause filters out members who have not incurred any fines.

## **Q8: Get the copies ID and the total count of loans for copies that have the highest loan count.**

**A picture containing text, screenshot, font, line

Description automatically generated**

**A screenshot of a computer

Description automatically generated with medium confidence**

This query retrieves the copies ID and the total count of loans for copies that have the highest loan count. It selects the **‘CopiesID’** column and calculates the loan count for each copy using the **‘COUNT (\*)’** function. The results are grouped by **‘CopiesID’**, and the HAVING clause filters out copies that do not have the maximum loan count. The subquery **‘(SELECT MAX(Amount) FROM (SELECT CopiesID, COUNT (\*) AS Amount FROM Loan GROUP BY CopiesID) AS subquery) ‘finds** the maximum loan count across copies.

## **Q9: Retrieve the Publisher name and the total count of books published by each publisher.**

**A picture containing text, screenshot, font

Description automatically generated**

**A screenshot of a computer

Description automatically generated with medium confidence**

This query retrieves the publisher’s name and the total count of books published by each publisher. It performs a left join between the **‘Publisher’** and **‘Book’** tables on the **‘PublisherID’** column. The **‘GROUP BY’** clause groups the results by publisher name, and the **‘COUNT (Book.BookID)’** function calculates the total count of books published by each publisher.

## **Q10: Get the book names for books that have a reservation status of 'Pending' and sort them in ascending order.**

**A picture containing text, font, screenshot

Description automatically generated**

**A screenshot of a computer

Description automatically generated with medium confidence**

This query retrieves the book names from the **‘Book’** table for books that have a reservation status of 'Pending'. It joins the **‘Book’** and **‘Reservation’** tables based on the **‘BookID’** column. The **‘WHERE’** clause filters out books with a reservation status other than 'Pending'. The results are sorted in ascending order by book name.

# **Work Matrix.**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Part** | **Component** | **Naim Nasruddin bin Nazri** | **Terence Ling Chee Yew** | **Ooi Hong Ping** | **ABDULRAHMAN GAMIL MOHAMMED AHMED** | **Total** |
| **2** | d) Database Schema | **25%** | **25%** | **25%** | **25%** | **100%** |
| **2** | * e) SQL-Data Definition Language (DDL) | **25%** | **25%** | **25%** | **25%** | **100%** |
| **2** | f) SQL-Data Manipulation Language (DML) | **25%** | **25%** | **25%** | **25%** | **100%** |