**ASSIGNMENT DBI202**

**Topic: Database of library management system**

**STUDENT NAME: TRAN HOANG MINH | SE161584**

**Table contents**

1. **Introduction of the problem**
2. **Define the entity in the system**
3. **ERD diagram**
4. **Convert ERD diagram to relation diagram**
5. **Define input entity from data type**
6. **List of constraint between entity**
7. **Set up SQL database**
8. **Conclusion of the management system**
9. **Introduction of the problem**

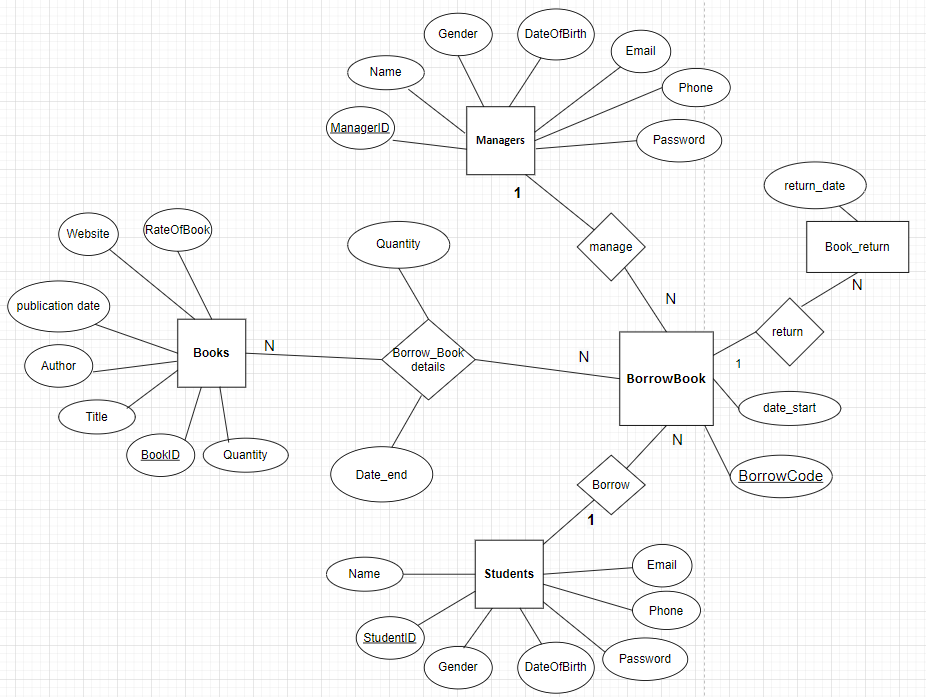
In recent years, with the rapid development of information and student in universities want to collect that information in their library. However, almost libraries use physical managements (for example the librarians have to rearrange the new book by themselves when the new one comes to their libraries), trouble in identify the student who borrowed the books or lack of storage space and sometimes lose the location of some academic materials. So I think we have to create a website library management to solve this problem.

Our website can retrieve information on the computer instead of manual management. This helps the librarian keep track of each person who borrows materials, and needs to register from the library account with their student information such as their student Id, name, address, so on…. All academic materials are arranged by the author and the academic fields which are easy for readers to borrow.

1. **Define the entity in the system**

* The entity Student represents the student who wants to borrow the book in the library on the website. Each Student will have the StudentID, Name, DateOfBirth, Gender, Email, Phone, and Password( to login to the website)
* In the library, the university will need a manager to manage and maintain the book in and out. Also, the Manager will manage the number of students who borrow the book. Each manager includes ManagerID, Name, DateOfBirth, Gender, Email, Phone, and Password( to login to the website)
* For each student who wants to borrow a book on a website, the information of that student and the manager to keep track of the student’s info will represent in the entity BorrowBook. Every BorrowBook will have its code for specific students and managers. The entity include BorrowCode, StudentID, ManagerID, date\_start( the date student start to borrow the book)
* The Entity Book will provide the all book that the library has on the website for a student to borrow. That Entity provides BookID, Title, Author, publication\_date, Quantity (The number of specific books will provide on the website), RateOfBook, Award
* The entity Borrow\_Book\_details stores the BorrowCode of each student who wants to borrow the number of books in the library and the deadline of return the book. It includes BorrowCode, BookID, Quantity, Date\_end
* When people finish the book and want to return the book to the library. We will use the entity Book\_return to show the date of returning the book. The entity Book\_return is BorrowCode, return\_date

1. **ERD diagram**

****

1. **Convert ERD diagram to relation diagram**

Our database library management objectives can be represented with some entities and attributes of the entity:

**Managers**: ManagerID, Name, DateOfBirth, Gender, Email, Phone, Password

**Students**: StudentID, Name, DateOfBirth, Gender, Email, Phone, Password

**Diagram

Description automatically generated**

**BorrowBook**: BorrowCode, StudentID, ManagerID, date\_start

**A picture containing shoji

Description automatically generated**

**Book\_return:** BorrowCode, return\_date

**A picture containing shoji, whiteboard, clock

Description automatically generated**

**Books:** BookID, Title, Author, publication\_date, Quantity, Website, RateOfBook

**Borrow\_Book details:** BorrowCode, BookID, Quantity, Date\_end

**Diagram

Description automatically generated**

1. **Define input entity from data type**

**a/ Tables of Managers**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column Name** | **Description** | **Data Type** | **Length** | **Value** |
| ManagerID | Each manager has their own id to distinguish from another manager | Char | 50 | Number, Text, Space permitted |
| Name | Name of the library manager | Nvarchar | 100 | Text, Space permitted |
| DateOfBirth | Date of Birth of manager | Date |  | Local date;  dd = 01-31;  mm = 01-12;  yyyy = 1970-2022 |
| Gender | Type of gender of each manager | Bit | 1 | Text, Space permitted |
| Email | The email address to contact another or to login to manage the website | Char | 100 | Number ,Text, Space permitted |
| Phone | Phone number to contact | Char | 50 | number |
| Password | Each manager have their own password to login in the website | Char | 50 | Number ,Text, Space permitted |

**b/ Table of Students**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column Name** | **Description** | **Data Type** | **Length** | **Value** |
| StudentID | Each Student have their id to distinguish from another student | char | 50 | Number, Text, Space permitted |
| Name | Name of student account | nvarchar | 100 | Text, Space permitted |
| DateOfBirth | Date of Birth of student | date |  | Local date;  dd = 01-31;  mm = 01-12;  yyyy = 1970-2022 |
| Gender | Type of gender of each student | Bit |  | Text, Space permitted |
| Email | The email address to contact another or to login to the website | Char | 100 | Number ,Text, Space permitted |
| Phone | Phone number to contact | Char | 50 | number |
| Password | Each student has their own password to login in to the website | Char | 50 | Number ,Text, Space permitted |

**c/ Table of book**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column**  **Name** | **Description** | **Data Type** | **Length** | **Value** |
| BookID | Each Book have its id to keep track of them | char | 50 | Number ,Text, Space permitted |
| Title | The name of the book | nvarchar | 100 | Text, Space permitted |
| Author | The name of the writer who wrote this book | nvarchar | 100 | Text, Space permitted |
| publication\_date | To know the date the book comes to public people | Date |  | Local date;  dd = 01-31;  mm = 01-12;  yyyy = 1970-2022 |
| Quantity | To know how many books in the website | Int | 3 | number |
| Website | The book appears of different platform | char | 100 | Number ,Text, Space permitted |
| RateOfBook | The comment of readers about this book. Is it worth reading? | Double | 1 | number |
| Award | Does this book have another trophy or certificate?? | nvarchar | 100 | Number ,Text, Space permitted |

**D/ Tables of Borrow\_Book**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column**  **Name** | **Description** | **Data Type** | **Length** | **Value** |
| BorrowCode | Each Student want to borrow book will have the Borrowcode it also manage by the managers | char | 50 | Number, Text, Space permitted |
| StudentID | Each Student have their own id to distinguish with another student | char | 50 | Number, Text, Space permitted |
| ManagerID | Each manager have their own id to distinguish with another manager | char | 50 | Number, Text, Space permitted |
| date\_start | The date start to store the time student borrow the book | Date |  | Local date;  dd = 01-31;  mm = 01-12;  yyyy = 1970-2022 |

**E/Table of Borrow\_Book\_details**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column**  **Name** | **Description** | **Data Type** | **Length** | **Value** |
| BorrowCode | Each Student who want to borrow a book will have the Borrowcode it also managed by the managers | char | 50 | Number, Text, Space permitted |
| BookID | Each Book have its id to keep track of them | char | 50 | Number, Text, Space permitted |
| Quantity | To know how many books you want to borrow | Int | 3 | number |
| Date\_end | The date ended we you have to return this book | Date |  | Local date;  dd = 01-31;  mm = 01-12;  yyyy = 1970-2022 |

**F/ Table of Book\_return**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Column**  **Name** | **Description** | **Data Type** | **Length** | **Value** |
| BorrowCode | Each Student who wants to borrow a book will have the Borrowcode it is also managed by the managers | char | 50 | Number, Text, Space permitted |
| return\_date | Each Book has its date for the student to return the book | date |  | Local date;  dd = 01-31;  mm = 01-12;  yyyy = 1970-2022 |

1. **List of constraint between entity**

|  |  |  |  |
| --- | --- | --- | --- |
| **Table** | **Primary Key** | **Foreign key** | **Check Validate** |
| Managers | ManagerID |  | NOT NULL, MA[6] |
| Students | StudentID |  | NOT NULL, FU[6] |
| Book\_return | BorrowCode | BorrowCode | NOT NULL, BC[6] |
| Borrow\_Book | BorrowCode |  | NOT NULL, BC[6] |
|  | ManagerID | NOT NULL, MA[6] |
|  | StudentID | NOT NULL, FU[6] |
| Borrow\_Book\_details | BorrowCode | BorrowCode | NOT NULL, BC[6] |
| BookID | BookID | NOT NULL |
| Book | BookID |  | NOT NULL |

1. **Set up SQL database**

**Create Table Managers:**

create table [Managers](

[ManagerID] char(50) NOT NULL PRIMARY KEY CHECK([ManagerID] like 'MA[0-9][0-9][0-9][0-9][0-9][0-9]'),

[Name] nvarchar(100) NOT NULL,

[DateOfBirth] date NOT NULL check([DateOfBirth] < getDate()),

[Gender] bit NOT NULL check([Gender] like 1 or [Gender] like 0),

[Email] char(100) NOT NULL,

[Phone] char(50),

[Password] char (50) NOT NULL

)

--Insert input to Managers

INSERT INTO dbo.Managers(ManagerID, Name, DateOfBirth, Gender, Email, Phone, Password)

VALUES

('MA102819', N'Nguyễn Thanh Tùng', '1990-02-10', 1, 'thanhtung@fpt.edu.com', 1526612771, 'thanhtung123'),

('MA291672', N'Trần Thị Yến ', '1992-12-12', 0, 'yenthi@fpt.edu.com', 18729266621, 'thiyen321'),

('MA145261', N'Hoàng Đức', '1987-05-9', 1, 'hoangduc@fpt.edu.com', 9876251311, 'duc@gmail'),

('MA776622', N'Nguyễn Linh Chi', '1994-4-30', 0, 'linhchi@fpt.edu.com', 88771572991, 'linhchi@hcm123'),

('MA442661', N'Nguyễn Võ Tòng', '1997-07-16', 1, 'tongvonguyen@fpt.edu.com', 17829072612, 'votong9876'),

('MA101982', N'Hoàng Thị Lan', '1998-12-02', 0, 'htlan@fpt.edu.com', 7224612771, 'htlan1998')

--The displayment of table manager

**Table

Description automatically generated**

**Create Table Students:**

create table [Students](

[StudentID] char(50) NOT NULL PRIMARY KEY CHECK([StudentID] like 'FU[0-9][0-9][0-9][0-9][0-9][0-9]'),

[Name] nvarchar(100) NOT NULL,

[DateOfBirth] date NOT NULL check([DateOfBirth] < getDate()),

[Gender] bit NOT NULL check([Gender] like 1 or [Gender] like 0),

[Email] char(100) NOT NULL,

[Phone] char(50),

[Password] char (50) NOT NULL

)

--Insert input to Students

INSERT INTO dbo.Students(StudentID, Name, DateOfBirth, Gender, Email, Phone, Password)

VALUES

('FU161334', N'Lê Văn Luyện', '2002-04-10', 1, 'lvluyen@fpt.edu.com', 1698212771, '00998877'),

('FU161442', N'Hoàng Văn Bách', '2000-02-22', 1, 'hvbach@fpt.edu.com', 54312632621, 'bachvh111'),

('FU161586', N'Lâm Đức', '1999-05-09', 1, 'duc@fpt.edu.com', 8877651891, 'duc\_12300'),

('FU151678', N'Nguyễn Mỹ Anh', '2001-05-20', 0, 'myanh@fpt.edu.com', 4453781211, 'MyAnh321123'),

('FU152588', N'Trần Minh Anh', '2000-03-26', 0, 'minhanhtran@fpt.edu.com', 8800212277, 'minhanh999'),

('FU146882', N'Hoàng Lê Linh', '1999-11-22', 0, 'htlinh@fpt.edu.com', 4784612371, '11221999'),

('FU173852', N'Phát Huy', '2003-01-22', 1, 'huyphat@fpt.edu.com', 665182371, 'huy01222003'),

('FU171835', N'Nhật Hà', '2003-01-22', 0, 'ha@fpt.edu.com', 009911223, 'nhatha2003')

--The displayment of table Students

**Table

Description automatically generated**

**Create Table Book:**

CREATE TABLE [Book]

(

[BookID] CHAR(50) PRIMARY KEY NOT NULL,

[Title] NVARCHAR(100) NOT NULL,

[Author] NVARCHAR(100) NOT NULL,

[publication\_date] date NOT NULL,

[Quantity] INT CHECK([Quantity] > 0 AND [Quantity] < 200),

[Website] CHAR(100),

[RateOfBook] INT CHECK([RateOfBook] >= 0 AND [RateOfBook] <= 5)

)

--insert input to Book

INSERT INTO dbo.Book(BookID, Title, Author, publication\_date, Quantity, Website, RateOfBook)

VALUES

('BS2008', N'Batman vs Superman', N'Peter', '2008-12-25', 130, 'amazon.com', 4),

('EH1122', N'The Economics of the World', N'Robert Sirico', '2005-12-11', 30, 'amazon.com', 3),

('HA0907', N'Harry Potter', N'J.K.Rowling', '2007-07-14', 22, 'harrypotter.com', 5),

('HM1961', N'Sapiens: A brief History of Humankind', N'Noah Harari', '2015-02-10', 21, 'history.com', 4),

('JM0722', N'Journey into Mystery', N'Patsy Walker', '1961-07-22', 5, 'marvelcomic.com', 4),

('HY2012', N'A Little Life', N'Hanya Yanagihara', '2016-01-26', 15, 'alibaba.com', 4),

('PA7762', N'To Paradise', N'Hanya Yanagihara', '2022-01-11', 10, 'shopee.com', 3),

('IA1616', N'Cambridge Ielts 16 Academic', N'Cambrigde', '2021-09-07', 180, 'ieltsgroup.com', 5)

--The displayment of table Book

**Graphical user interface, table

Description automatically generated**

**Create Tables Borrow\_Book**

CREATE TABLE [Borrow\_Book]

(

[BorrowCode] CHAR(50) NOT NULL PRIMARY KEY CHECK([BorrowCode] like 'BC[0-9][0-9][0-9][0-9][0-9][0-9]'),

[ManagerID] char(50) NOT NULL CHECK([ManagerID] like 'MA[0-9][0-9][0-9][0-9][0-9][0-9]'),

CONSTRAINT FK\_Manager\_ID FOREIGN KEY([ManagerID]) REFERENCES dbo.Managers([ManagerID]),

[StudentID] char(50) NOT NULL CHECK([StudentID] like 'FU[0-9][0-9][0-9][0-9][0-9][0-9]'),

CONSTRAINT FK\_Student\_ID FOREIGN KEY([StudentID]) REFERENCES dbo.Students([StudentID]),

date\_start DATE NOT NULL

)

--insert input to Borrow\_Book

INSERT INTO dbo.Borrow\_Book(BorrowCode, ManagerID, StudentID, date\_start)

VALUES

('BC000001', 'MA101982', 'FU146882', '2021-05-29'),

('BC000002', 'MA101982', 'FU151678', '2021-12-20'),

('BC000003', 'MA102819', 'FU152588', '2022-01-21'),

('BC000004', 'MA102819', 'FU161334', '2022-01-01'),

('BC000005', 'MA102819', 'FU161442', '2022-02-26'),

('BC000006', 'MA291672', 'FU161334', '2020-12-30'),

('BC000007', 'MA442661', 'FU161442', '2021-04-29'),

('BC000008', 'MA442661', 'FU171835', '2021-11-30'),

('BC000009', 'MA291672', 'FU151678', '2021-11-30'),

('BC000010', 'MA776622', 'FU173852', '2021-11-30')

--the displayment of table Borrow\_Book

Table

Description automatically generated

**Create Tables Borrow\_Book\_details**

CREATE TABLE [Borrow\_Book\_details]

(

[BorrowCode] CHAR(50) NOT NULL CHECK([BorrowCode] like 'BC[0-9][0-9][0-9][0-9][0-9][0-9]'),

CONSTRAINT FK\_BorrowCode FOREIGN KEY([BorrowCode]) REFERENCES dbo.[Borrow\_Book]([BorrowCode]),

[BookID] CHAR(50) NOT NULL,

CONSTRAINT FK\_Book\_ID FOREIGN KEY([BookID]) REFERENCES dbo.Book([BookID]),

[Quantity] INT NOT NULL,

[Date\_end] DATE NOT NULL,

PRIMARY KEY([BorrowCode], [BookID])

)

--insert input to Borrow\_Book\_details

INSERT INTO dbo.Borrow\_Book\_details(BorrowCode, BookID, Quantity, Date\_end)

VALUES

('BC000001', 'BS2008', 30, '2021-12-30'),

('BC000001', 'EH1122', 3, '2021-12-30'),

('BC000001', 'HA0907', 1, '2021-12-30'),

('BC000002', 'HM1961', 4, '2022-03-20'),

('BC000003', 'HA0907', 12, '2022-05-11'),

('BC000004', 'PA7762', 9, '2022-04-12'),

('BC000005', 'HM1961', 7, '2022-05-11'),

('BC000005', 'JM0722', 5, '2022-05-11'),

('BC000006', 'IA1616', 12, '2022-06-22'),

('BC000007', 'HY2012', 12, '2022-07-01'),

('BC000007', 'HA0907', 2, '2022-07-01'),

('BC000008', 'EH1122', 3, '2022-08-01'),

('BC000008', 'HM1961', 7, '2022-06-11'),

('BC000010', 'BS2008', 11, '2022-04-24')

--the displayment of table Borrow\_Book\_details

Table

Description automatically generated

**Create Tables Book\_return**

CREATE TABLE [Book\_return](

[BorrowCode] CHAR(50) NOT NULL CHECK([BorrowCode] like 'BC[0-9][0-9][0-9][0-9][0-9][0-9]')

CONSTRAINT FK\_BorrowCode\_BookReturn FOREIGN KEY([BorrowCode]) REFERENCES dbo.[Borrow\_Book]([BorrowCode]),

[return\_date] DATE NOT NULL,

PRIMARY KEY([BorrowCode])

)

--insert input to Book\_return

INSERT INTO dbo.Book\_return(BorrowCode, return\_date)

VALUES

('BC000001', '2022-02-17'),

('BC000002', '2022-02-27'),

('BC000003', '2022-04-22'),

('BC000004', '2022-04-17'),

('BC000005', '2022-05-13'),

('BC000006', '2022-06-06'),

('BC000007', '2022-02-17'),

('BC000008', '2022-07-30'),

('BC000009', '2022-03-17'),

('BC000010', '2022-03-28')

--the displayment of Book\_return

**Table

Description automatically generated with medium confidence**

**Overview of database\_library\_management**

**Diagram

Description automatically generated**

SQL COMMAND

# A/ SQL ORDER BY Keyword

**Code:**

SELECT \* FROM dbo.Students

ORDER BY [StudentID] ASC

**Result:**

**Table

Description automatically generated**

We use SQL query ORDER BY to sort the list ascending or descending by the values which we want it to display. Select \* from **Students** to give users all record in the **Students** table and cooperate with the Order By query so the table displays all records ascending by StudentID.

# B/ SQL LEFT JOIN Keyword

# **Code:**

SELECT BB.BorrowCode, BB.ManagerID, BB.StudentID, BB.date\_start, BBd.Date\_end, BBd.Quantity

FROM

# dbo.Borrow\_Book BB LEFT JOIN dbo.Borrow\_Book\_details BBd ON BBd.BorrowCode = BB.BorrowCode

**Result:**

Table, Excel

Description automatically generated

Below this picture, We left Join **Borrow\_Book** with **Borrow\_Book\_details**. The null of the return date means that the record can not connect to any column in **Borrow\_Book\_details** so the SQL will set this to the null value.

# C/ QUERY USING AGGREGATE FUNCTIONS

# **Code:**

SELECT Br.BorrowCode, count(Br.return\_date) AS NumberOfBorrowBookIn2002

FROM dbo.Book\_return Br

WHERE YEAR(Br.return\_date) = 2022 AND MONTH(Br.return\_date) = 5

GROUP BY Br.BorrowCode

**Result**:

# **Graphical user interface, application Description automatically generated**

We use count() function to count the number of **BorrowCode** which students return the Book. In this picture, only 1 people have a return date of ‘May 2022’

# D/ QUERY USING THE GROUP BY AND HAVING CLAUSES

# **Code:**

SELECT b.BookID, b.Title, COUNT(\*) AS NumberOfBorrowBook

FROM dbo.Book b JOIN dbo.Borrow\_Book\_details BBd ON BBd.BookID = b.BookID

GROUP BY b.BookID, b.Title

HAVING COUNT(\*) >= 3

ORDER BY b.BookID ASC

**Result:**

# Graphical user interface, text, application Description automatically generated

# In this picture, we want to find the number of books which borrow greater or equal than 3 according to the ascending by the **BookID**. Then having have to use

# Having() and count()

# E/ QUERY THAT USES A SUB-QUERY AS A RELATION

# **Code:**

SELECT bb.StudentID, s.Name AS StudentName, COUNT(bb.BorrowCode) AS StudentBorrowBook

FROM dbo.Borrow\_Book bb JOIN dbo.Students s ON s.StudentID = bb.StudentID

GROUP BY bb.StudentID, s.Name

HAVING COUNT(bb.BorrowCode) = (

SELECT MAX(B.StudentBorrowBook) FROM(SELECT COUNT(bb.BorrowCode) AS StudentBorrowBook

FROM dbo.Borrow\_Book bb JOIN dbo.Students s ON s.StudentID = bb.StudentID

GROUP BY bb.StudentID, s.Name) AS B)

**Result:**

Graphical user interface, text, table

Description automatically generated

We use Group By() and Having() clauses to find the student who have the total **BorrowBook** most

F/ QUERY THAT FIND APPROXIMATE MATCHING IN THE WHERE CLAUSE

# **Code:**

SELECT \*

FROM dbo.Students S

WHERE S.Name LIKE 'T%' OR S.Name LIKE 'N%'

**Result:**

Table

Description automatically generated

If we want to find the student who has name beginning with ‘N’ or ‘T’. We can use Like ‘%’ in **Students** table

G/ STORE PROCEDURE

# **Code:**

CREATE PROC count\_Book @BorrowCode CHAR(50), @NumberOfBook int OUTPUT

AS

BEGIN

SET @NumberOfBook = (

SELECT COUNT(\*)

FROM dbo.Borrow\_Book\_details

WHERE BorrowCode = @BorrowCode

GROUP BY BorrowCode

)

END

DECLARE @x INT

EXEC count\_Book 'BC000001', @x OUTPUT

SELECT @x AS totalOfBook

**Result:**



We use the procedure to count the total number of Book student want to borrow which uses the **BorrowCode**

# H/ TRIGGER PROCEDURE

# **Code:**

CREATE TRIGGER tr\_insert\_quantity ON Borrow\_Book\_details after insert

AS

BEGIN

SELECT i.BorrowCode, i.BookID, b.Title, b.Author, i.Quantity, i.Date\_end

FROM inserted i JOIN dbo.Book b ON i.BookID = b.BookID

END

INSERT INTO dbo.Borrow\_Book\_details(BorrowCode, BookID, Quantity, Date\_end)

VALUES ('BC000009','IA1616', 2, '2020-08-01')

**Result:**

**Graphical user interface, text, application

Description automatically generated**

We use trigger procedure to insert the new **BorrowCode** and display it with Title and Author of that Book