



**QUEEN'S
UNIVERSITY
BELFAST**

DSA8002 ASSIGNMENT (RESIT)

OBJECT ORIENTED PROGRAMMING
MEETS DATABASE

JAYAVARDHINI SHANMUGAM

40318915

Jshanmugam01@qub.ac.uk

Table of Contents

1. Introduction.....	2
2. Background Theory	2
2.1 Class.....	2
2.2 Methods.....	2
2.3 SQLite3.....	2
2.4 Pandas.....	3
3. Class Design.....	3
4. Database Design.....	3
4.1 Administrator.....	3
4.2 Library Books.....	5
4.3 Library Members.....	7
4.4 Loan Status.....	8
5. Use Case.....	9
5.1 Login.....	9
5.2 Sign in.....	9
5.3 Add Books.....	9
5.4 Add Members.....	10
5.5 Search Books.....	10
5.6 Loan Books.....	10
5.7 Show Loaned Books.....	11
5.8 Return Books.....	11
5.9 Single Inheritance	11
6. Conclusion.....	12

1.INTRODUCTION

The computer programming model called “Object-oriented programming (OOP)” could be defined as a method of structuring a program by bundling related properties and behaviours into individual objects. The aim of this assignment is to do Object Oriented Programming in python while interfacing a database. In this assignment we created a “Library-Book-Loan-System” which mainly focuses on basic operations in a library like adding new books, updating loan status, searching books by title and return books. In this assignment we can maintain the late fine of members who returns the issued books after the due date. The classes will be designed for establishing a relationship among the predefined classes in order to prepare a suitable relational database. This assignment is developed using Python and used Sqlite3 as database to store the library details.

2.BACKGROUND THEORY

2.1 Class:

A class is a user defined blueprint or prototype from which objects are created. Classes provides a mean of bundling data and functionality together. Creating a new class creates a new type of object, allowing new instances of that type to be made. Each class instance can have attributes attached to it for maintaining its state. Class instances can also have methods (defined by their class) for modifying their state.

2.2 Methods:

A method in python is somewhat similar to a function, except it is associated with object/classes. Methods in python are very similar to functions except for two major differences.

1. The method is implicitly used for an object for which it is called.
2. The method is accessible to data that is contained within the class.

2.3 SQLite3:

SQLite3 is a very easy to use database engine. It is self-contained, serverless, zero-configuration and transactional. It is very fast and lightweight, and the entire database is stored in a single disk file. It is used in a lot of applications as internal data storage. The Python Standard Library includes a module called "sqlite3" intended for working with this database. This module is a SQL interface compliant with the DB-API 2.0 specification.

2.4 Pandas:

Pandas is a package which was created on top of NumPy in python, it is flexible and fast. The data structures of the pandas are designed in a way to give relational or label data easy. It has two different type of data structures which are series and data frame. Series data structure is based on 1 dimension, while the data frame is 2 dimensional. In our project we used 2-dimensional data structure which is data frame.

3.CLASS DESIGN:

The class design corresponding to the object comprises the description relevant to the object bearing similar responsibilities, operations, attributes and semantics. It is carried out via the three means wherein the design corresponding to the class can be ascertained via interface, inheritance and composition. In our code we created a class with a method to access its attributes. The code is given in Appendix.

Our code has five classes, among that first class is named as Library. In this class we added two methods add books and add members. The method add books will add new books to our “LIBRARY BOOKS” table. The method add members will add new members to our “LIBRARY MEMBERS” table.

The second class is named as Administrator. In this class we added three methods administrator, administrator login and administrator sign in. The first method administrator will display a two options login and sign in. The second method administrator login will allow the administrator/operator to login. The third method administrator sign in will help the administrator/operator to create a new account for administrator.

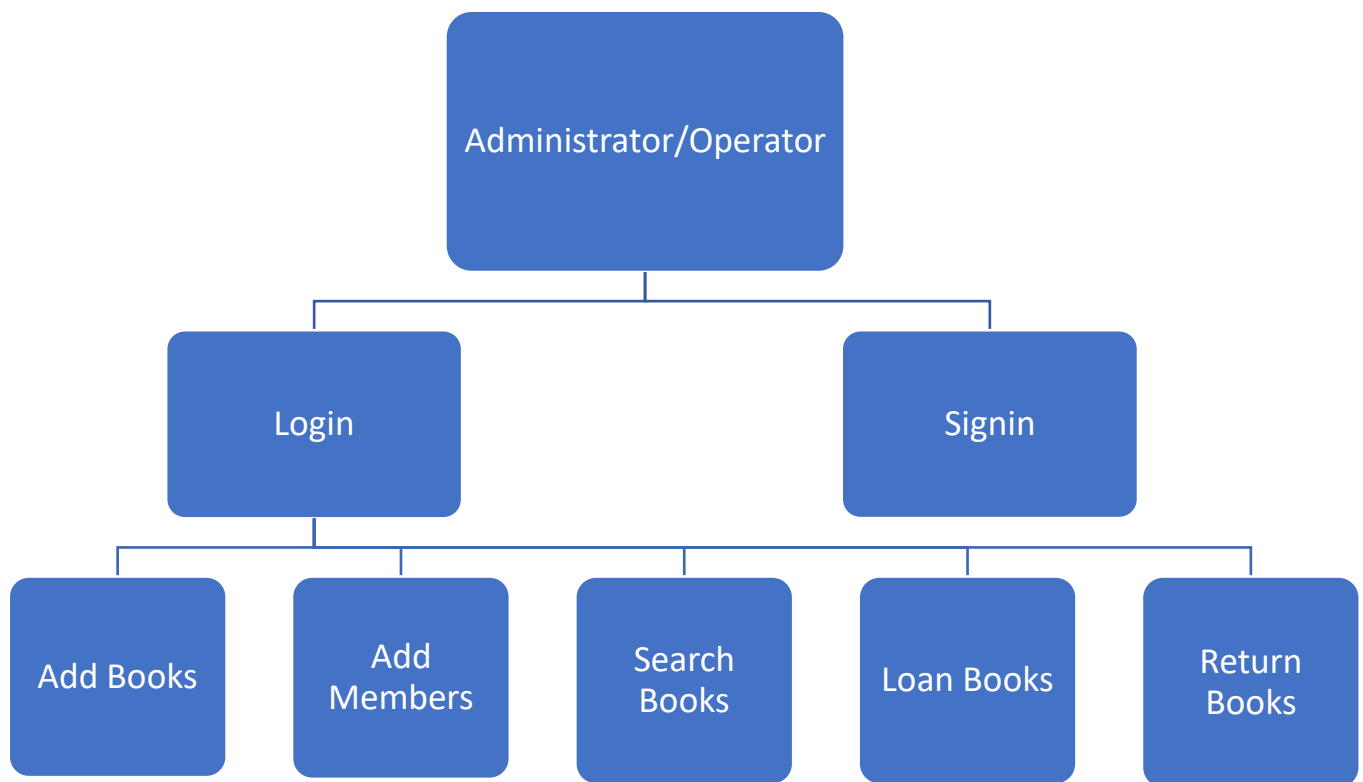
The third class is named as book search. In this class we added two methods search book and search book2. The first method search book will help the administrator/operator to search the books which are available in library. The second method search book2 is used for doing single level inheritance with other class.

The fourth class is named as loan book. The class contains only one method book loan. It is the derived class of book search (Single level inheritance). This method will allow the administrator to loan a book to members.

The fifth and final class is named as loaned books. This class contains two methods show loaned books and return books. The first method show loaned books will allow the

operator/administrator to view the loaned books. The second method return books will allow the operator/administrator to get back the loaned books from the members.

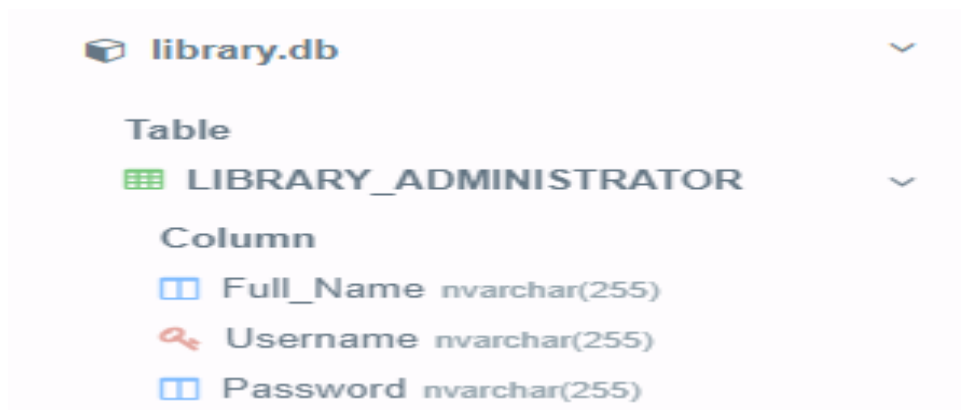
Structure for Library book loan system



4.DATABASE DESIGN:

4.1 Administrator:

1. CREATE TABLE LIBRARY_ADMINISTRATOR (Full_name NVARCHAR (255), Username NVARCHAR (255) PRIMARY KEY, Password NVARCHAR (255) NOT NULL)



2. SELECT * from LIBRARY_ADMINISTRATOR

library.db

Table

LIBRARY_ADMINISTRATOR

Column

- Full_Name nvarchar(255)
- Username nvarchar(255)
- Password nvarchar(255)

LIBRARY_BOOKS

LIBRARY_MEMBERS

LOAN_STATUS

SQLite

MariaDB

library.db

1 SELECT * FROM LIBRARY_ADMINISTRATOR

Full_Name	Username	Password
jayavardhini shanmugam	jayavardhini@gmail.com	1234
Vignesh	vignesh@gmail.com	1234
vijay	vijay@gmail.com	1234
xyz	xyz@gmail.com	1234

Figure: Database output for LIBRARY_ADMINISTRATOR

(Source: Self-created in SQLite browser)

4.2 Library Books:

1. CREATE TABLE LIBRARY_BOOKS (Book_ISBN INT PRIMARY KEY, Title NVARCHAR (255) NOT NULL, Author_Name NVARCHAR (255), Year INT, Edition NVARCHAR (100), Loan_Status NVARCHAR (255))

 library.db

Table

-  LIBRARY_ADMINISTRATOR
-  LIBRARY_BOOKS

Column



-  Book_ISBN int
-  Title nvarchar(255)
-  Author_Name nvarchar(255)
-  Year int
-  Edition nvarchar(100)
-  Loan_Status nvarchar(255)

Figure: Database design for LIBRARY BOOKS

2. SELECT * from LIBRARY_BOOKS

 library.db

1 SELECT * FROM LIBRARY_BOOKS

!	Book_ISBN	Title	Author_Name	Year	Edition	Loan_Status
	1209856720981	Become a digit...	susan j.barnes	2003	1st Edition	Loaned
	1845092348714	Global Librarian...	Martin A.Kessel...	2004	1st Edition	Available
	1784562309124	Library Informat...	Charles Grosch	1994	1st Edition	Available
	1892673540913	An Introduction ...	Roger Flynn	1986	1st Edition	Available
	1082542978143	Scientific and T...	Subramanyam	1981	1st Edition	Available
	1672093568129	Furnishing the ...	Pierce	1980	1st Edition	Available
	9781119562733	Effective Projec...	Robert K. Wyso...	2019	8th Edition	Available
	9781617296062	Zero to AI	Nicolo Valigi	2020	4th Edition	Available
	9781617295980	Classic Comput...	David Kopec	2019	2nd Edition	Loaned
	9781484226049	Blockchain Basics	Daniel Drescher	2017	5th Edition	Loaned
	9781913151171	Python for Begi...	J Foster	2020	5th edition	Loaned
	9781838554767	Expect C++	Vardan Grigoryan	2020	1st edition	Available
	9781839216787	Machine Learni...	Stefan Jansen	2020	2nd Edition	Available

Figure: Database output for LIBRARY BOOKS

4.3 Library Members:

1. CREATE TABLE LIBRARY_MEMBERS (Username NVARCHAR (255) PRIMARY KEY, Password NVARCHAR (255) NOT NULL, Full_Name NVARCHAR (255), Address NVARCHAR (255), Contact_Number INT, Plan NVARCHAR (255))

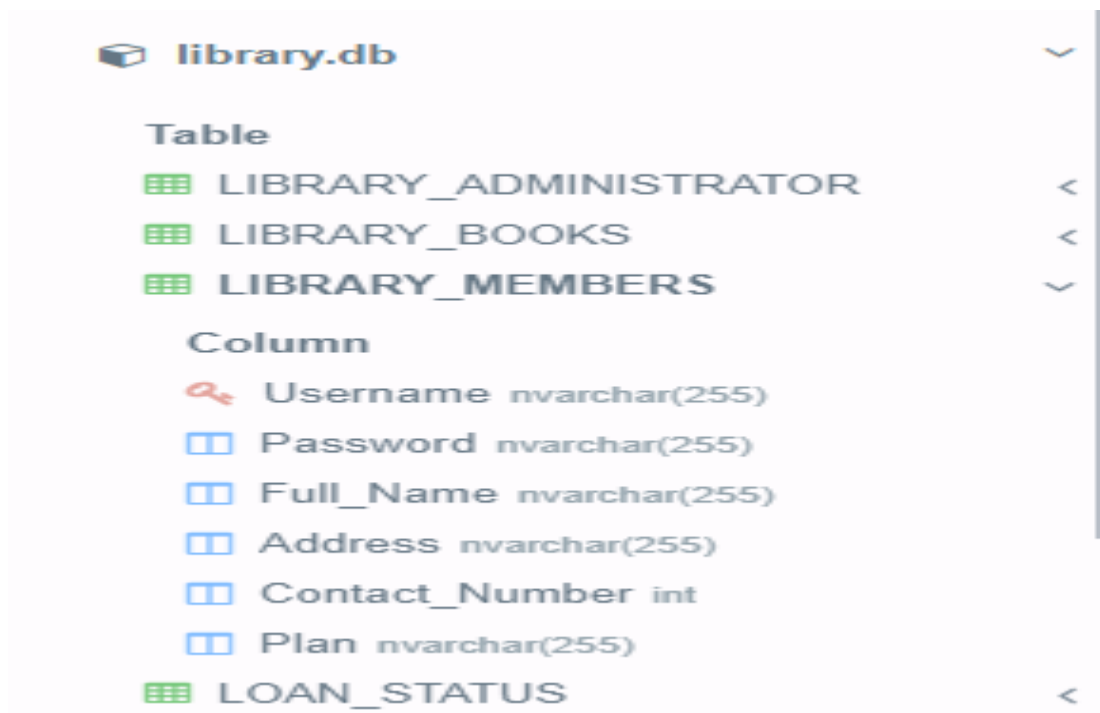
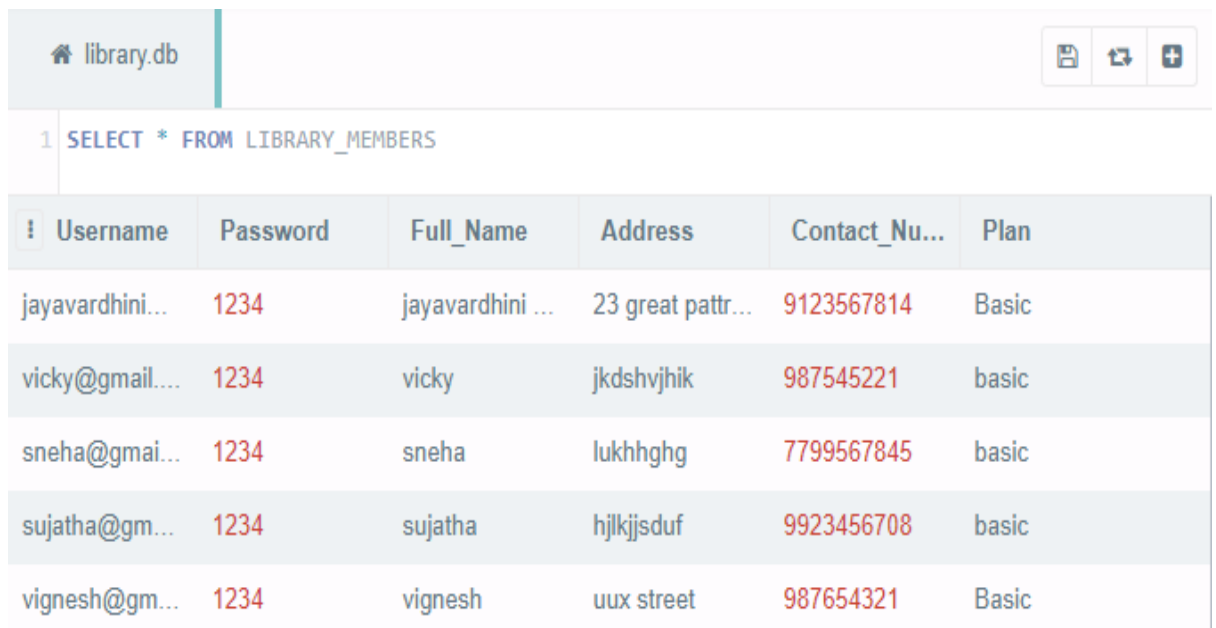


Figure: Database design for LIBRARY_MEMBERS

2. SELECT * from LIBRARY_MEMBERS



library.db

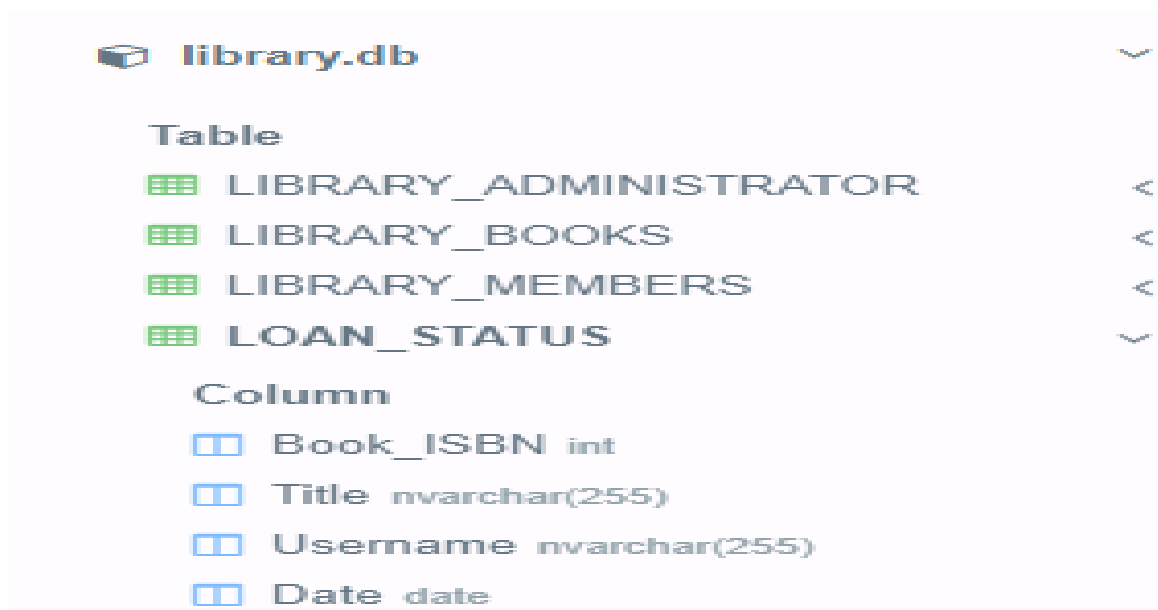
1 SELECT * FROM LIBRARY_MEMBERS

!	Username	Password	Full_Name	Address	Contact_Nu...	Plan
	jayavardhini...	1234	jayavardhini ...	23 great pattr...	9123567814	Basic
	vicky@gmail....	1234	vicky	jkdshvjhik	987545221	basic
	sneha@gmai...	1234	sneha	lukhhghg	7799567845	basic
	sujatha@gm...	1234	sujatha	hjlkjjsdud	9923456708	basic
	vignesh@gm...	1234	vignesh	uux street	987654321	Basic

Figure: Database output for LIBRARY_MEMBERS

4.4 Loan Status:

1. CREATE TABLE LOAN_STATUS (Book_ISBN INT, Title NVARCHAR (255), Username NVARCHAR (255), Date date NOT NULL)



library.db

Table

- LIBRARY_ADMINISTRATOR
- LIBRARY_BOOKS
- LIBRARY_MEMBERS
- LOAN_STATUS

Column

- Book_ISBN int
- Title nvarchar(255)
- Username nvarchar(255)
- Date date

Figure: Database design for LOAN_STATUS

2. SELECT * from LOAN_STATUS

library.db			
1 SELECT * FROM LOAN_STATUS			
Book_ISBN	Title	Username	Date
9781617295980	Classic Computer Scie...	sneha@gmail.com	2021-02-20
9781484226049	Blockchain Basics	jayavardhini@gmail.com	2019-12-12
9781913151171	Python for Beginners	jayavardhini@gmail.com	2020-12-13
1209856720981	Become a digital library	sneha@gmail.com	2021-08-01

Figure: Database output for LOAN_STATUS

5.USE CASES:

5.1 Login:

The login method will allow the administrator or operator to login into the library book loan system. This function gets the input (username and password) from the administrator or operator and checks the values with the corresponding database value. If the values didn't get matched it won't allow the administrator to login.

5.2 Sign in:

The sign in method will create a new account for administrator or operator. This function will get the input from the administrator such as full name, username = email and password and it will store them into a table "LIBRARY ADMINISTARTOR" in library database. The username = email will be unique to identify the administrator here.

5.3 Add Books:

The add book's method will create a new table "LIBRARY BOOKS" in library database. This will get the input from administrator such as book isbn, book title, author name, year and

edition and it will store to the table. It stores book status as available as default. The book isbn will be unique to identify the book here.

5.4 Add Members:

The add members method will create a new table “LIBRARY MEMBERS” in library database. This will get the details about the new member from administrator such as member username = email, password, full name, address, contact number and plan (i.e) ‘Basic’ or ‘Premium’ and stores it to the table. If the plan is basic the member can take up to two books loan as maximum. If the plan is premium the member can take up to four books loan as maximum. The username = email will be unique to identify the member here.

5.5 Search Books:

The search book’s method will help the administrator to search the books from the database using the books title. It will fetch all the books by using title which was provided by the administrator.

SQL QUERY:

```
Select * from LIBRARY_BOOKS where Title like ?
```

5.6 Loan Books:

The loan book’s method will help the administrator or operator to loan a particular book for the member which they needed. This method is derived from the book search class. Here we have done “SINGLE LEVEL INHERITANCE” to access the books which are searched in search books method. This shows a list of books and the member will be loaned a book according to they need. This will check whether the person is already a member in our library, if already a member, then it will check the members plan is basic or premium and whether they have crossed the maximum limit of their plan. If they have crossed thier maximum limit, it won’t allow them to loan a book and asks them to return their loaned book. This method will also display’s whether the book is available or loaned, if the book is already loaned it will denote the administrator that the book is already loaned. If all the conditions are satisfied then it will create a new table “LOAN STATUS” which will store the book isbn, book title, member who is loaning the book and date of loaning the book. These details are given by the administrator or operator to the system.

5.7 Show Loaned Books:

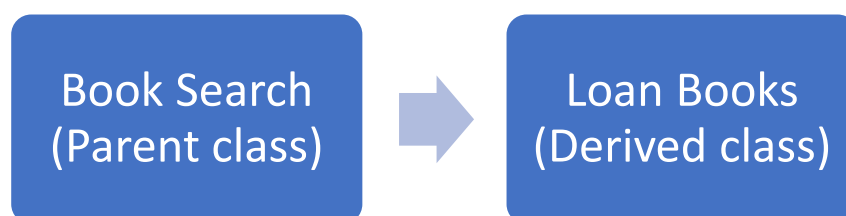
The show loaned books method will allow the administrator to view how many books are loaned from the library. This also shows that which member loans the book from the library. These details are fetched from the LOAN STATUS table in library database.

5.8 Return Books:

This method will allow the administrator to get back the loaned books from the member. This will first check whether the person is already a member in our library, if he/she is already a member then it will check whether they loaned any books. If they loaned any books, it will display how many books they have been loaned. Now the administrator wants to select which book the member want to return. Then it will display the total days which the member had loaned the book. If the days count is more than 60 days then fine will be charged from the member. Again, the book loan status will be updated to available in LIBRARY BOOKS and the member will be deleted from the LOAN STATUS table corresponding to the book isbn.

5.9 Single Inheritance:

We have done a single level inheritance to the class “Book Search” to “Loan Books”. Here the class Book search is the parent class and loan books are the derived class, in other word book search is parent class and loan books are child class. We can access the book search method from loan books class so we can get the book search list in loan books while loaning a book to member.



5.10 Logout:

This method allows the administrator/operator to logout from the library book loan system and directs to the main page of library book loan system.

6.CONCLUSION:

We successfully done the python implementation with more than three classes and methods. Also, we have inherited classes and used in a meaningful relationship. We have created an administrator account, login, add books option, add members option, search books, loan books, return books and logout according to all the conditions given in the assessment.

Appendix

Source Code

```
import sqlite3

import pandas as pd

pd.set_option('display.width', 400)
pd.set_option('display.max_columns', 10)

# Function which displays the main menu of the library
def main_menu():
    try:
        print("                L I B R A R Y   M E N U                \n\n\n\n")
        print("1. Add Books\n\n")
        print("2. Add Members\n\n")
        print("3. Search Book\n\n")
        print("4. Issue Book\n\n")
        print("5. Loaned Books\n\n")
        print("6. Return Book\n\n")
        print("7. Logout\n\n")
        print("8. Exit\n\n")

        choice = int(input("Enter your Choice(Only Numbers).....:"))

        if choice == 1:
            return Library.add_books()

        elif choice == 2:
            return Library.library_members()

        elif choice == 3:
```

```

        return book_search.search_books()
    elif choice == 4:
        return loan_book.search_book2(),loan_book.book_loan()
    elif choice == 5:
        return Loaned_books.show_loaned_books()
    elif choice == 6:
        return Loaned_books.return_books()
    elif choice == 7:
        print("You Have Been Logged Out\n\n")
        return Administrator.administrator()
    else:
        print("Application Closed \n\n")
        pass

```

```

except:
    print("Enter Proper Details \n\n")
    click = input("Enter Any Button to Continue.....:")
    return main_menu()

```

Class which will stores all details abot library

class Library:

Function which will add the new books to our library database

def add_books():

try:

```

    print("                Add Books                \n\n\n")

```

opens a connection to the SQLite database file

```

    con = sqlite3.connect('library.db')

```

creates a cursor which will be used throughout of your database programming with Python

```

cur = con.cursor()

# Checks whether the table is already created
listOfTables = cur.execute(
    """SELECT name FROM sqlite_master WHERE type='table' AND
name='LIBRARY_BOOKS'; """).fetchall()

isbn = int(input("Enter the ISBN :"))
title = input("Enter the Book Title :")
author = input("Enter the Author name :")
year = int(input("Enter the Year :"))
edition = input("Enter the Edition :")
val = (isbn, title, author, year, edition, "Available")

if listOfTables == []:
    cur.execute("""create table LIBRARY_BOOKS (Book_ISBN int PRIMARY KEY,
    Title nvarchar(255) NOT NULL,
    Author_Name nvarchar(255),
    Year int,
    Edition nvarchar(100),
    Loan_Status nvarchar(255));""")

    query = """ Insert into
LIBRARY_BOOKS(Book_ISBN,Title,Author_Name,Year,Edition,Loan_Status)
values(?,?,?,?,?,?)"""

    cur.execute(query, val)
    con.commit()
    cur.close()

# Closes the connection

```



```

        con.close()

        print("Book Added Sucessfully")

        click = input("Enter Any Button to Continue.....:")

        return main_menu()

    else:

        query = """ Insert into
LIBRARY_BOOKS(Book_ISBN,Title,Author_Name,Year,Edition,Loan_Status)
values(?,?,?,?,?,?)"""

        cur.execute(query, val)

        con.commit()

        cur.close()

        con.close()

        print("Book Added Sucessfully")

        click = input("Enter Any Button to Continue.....:")

        return main_menu()

except:

    print("Enter Proper Details \n\n")

    click = input("Enter Any Button to Continue.....:")

    return main_menu()

#Function which will add new members to our database
def library_members():

    try:

        print("                Add Members                \n\n\n")

        con = sqlite3.connect('library.db')

        cur = con.cursor()

        listofTables = cur.execute(

            """SELECT name FROM sqlite_master WHERE type='table' AND
name='LIBRARY_MEMBERS'; """).fetchall()

```

```

username = input("Enter the USERNAME (Email Address) :")
password = input("Enter the Password :")
name = input("Enter the Full name :")
address = input("Enter the Address :")
contact = int(input("Enter the Contact Number :"))
plan = input("Enter the plan Basic or Premium ?")
val = (username,password,name,address,contact,plan)

if listOfTables == []:
    cur.execute("""create table LIBRARY_MEMBERS (Username nvarchar(255)
PRIMARY KEY,
    Password nvarchar(255) NOT NULL,
    Full_Name nvarchar(255),
    Address nvarchar(255) ,
    Contact_Number int,
    Plan nvarchar(255));""")
    # Inserts new members details to our database
    query = """ Insert into
LIBRARY_MEMBERS(Username>Password>Full_Name>Address>Contact_Number>Plan)
values(?,?,?,?,?)"""
    cur.execute(query,val)
    con.commit()
    cur.close()
    con.close()
    print("Member Added\n\n")
    click = input("Enter Any Button to Continue.....:")
    return main_menu()
else:

```

```

        query = """ Insert into
LIBRARY_MEMBERS(Username,Password,Full_Name,Address,Contact_Number,Plan)
values(?,?,?,?,?,?)"""

        cur.execute(query,val)

        con.commit()

        cur.close()

        con.close()

        print("Member Added\n\n")

        click = input("Enter Any Button to Continue.....:")

        return main_menu()

except:

    print("Enter Proper Details \n\n")

    click = input("Enter Any Button to Continue.....:")

    return main_menu()

```

class Administrator:

Function which allows to login administrator/operator

def administrator_login():

try:

```

    print("                Login                \n\n")

    username = input('Enter the USERNAME (Email Address) :')

    password = input("Enter the Password :")

    con = sqlite3.connect('library.db')

    cur = con.cursor()

    query = """select Username from LIBRARY_ADMINISTRATOR"""

    cur.execute(query)

    values = cur.fetchall()

```

```

if (username,) in values:

    query = ("select Password from LIBRARY_ADMINISTRATOR where
Username=?")

    cur.execute(query,(username,))

    pass_val = cur.fetchall()

    if (password,) in pass_val:

        print("Login Succesful \n\n")

        cur.close()

        con.close()

        return main_menu()

    else:

        print("password incorrect \n\n")

        cur.close()

        con.close()

        click = input("Enter Any Button to Continue.....:")

        return Administrator.administrator()

else:

    print("No Username found \n\n")

    cur.close()

    con.close()

    click = input("Enter Any Button to Continue.....:")

    return Administrator.administrator()

except:

    print("Enter Proper Details \n\n")

    click = input("Enter Any Button to Continue.....:")

    return Administrator.administrator()

```

Function which Creates New Administrator/operator Account

```

def administrator_signin():
    try:
        print("                Signin                \n\n")
        con = sqlite3.connect('library.db')
        cur = con.cursor()
        listOfTables = cur.execute(
            """SELECT name FROM sqlite_master WHERE type='table' AND
name='LIBRARY_ADMINISTRATOR'; """).fetchall()

        name = input("Enter the Full name :")
        username = input("Enter the USERNAME (Email Address) :")
        password = input("Enter the Password :")
        val=(name, username, password)

        if listOfTables == []:
            cur.execute("""create table LIBRARY_ADMINISTRATOR (Full_Name
nvarchar(255),Username nvarchar(255) PRIMARY KEY,
                Password nvarchar(255) NOT NULL;""")

            query = """ Insert into LIBRARY_ADMINISTRATOR(Full_Name,Username>Password)
values(?,?,?)"""
            cur.execute(query,val)
            con.commit()
            cur.close()
            con.close()
            print("Account Created \n\n")
            click = input("Enter Any Button to Continue.....:")
            return Administartor.administrator_login()
        else:
            query = """ Insert into LIBRARY_ADMINISTRATOR(Full_Name,Username>Password)
values(?,?,?)"""

```

```

        cur.execute(query, val)
        con.commit()
        cur.close()
        con.close()

        print("Account Created \n\n")
        click = input("Enter Any Button to Continue.....:")
        return Administrator.administrator_login()

except:
    print("Enter Proper Details \n\n")
    click = input("Enter Any Button to Continue.....:")
    return Administrator.administrator()

# Administrator/operator main page function
def administrator():
    try:
        print("
                Adminstrator Login
                \n\n")
        print("1. Login  \n\n")
        print("2. Don't have account? signin  \n\n")

        option = int(input("Enter Your choice 1 or 2 ?"))
        if option == 1:
            return Administrator.administrator_login()
        else:
            return Administrator.administrator_signin()
    except:
        print("Enter Proper Details \n\n")
        click = input("Enter Any Button to Continue.....:")
        return Administrator.administrator()

```

```

class book_search:

    # Function for searching the books using title

    def search_books():

        try:

            print("                Search Books                \n\n\n")

            search = input("Enter the Book Name :")

            key_word = "%" + search + "%"

            con = sqlite3.connect('library.db')

            cur = con.cursor()

            Query = """Select * from LIBRARY_BOOKS where Title like ? """

            query = cur.execute(Query,(key_word,))

            book_val = query.fetchall()

            if len(book_val)>0:

                cols = [column[0] for column in query.description]

                results = pd.DataFrame.from_records(data = book_val, columns = cols)

                print(results,"\n\n")

            else:

                print("No Book Found try different keyword \n\n")

            cur.close()

            con.close()

            click = input("Enter Any Button to Continue.....:")

            return main_menu()

        except:

            print("Enter Proper Details \n\n")

            click = input("Enter Any Button to Continue.....:")

            return main_menu()

```

```

# Function for doing inheritance
def search_book2():
    try:
        print("                Search Books                \n\n\n")
        search = input("Enter the Book Name :")
        key_word = "%" + search + "%"
        con = sqlite3.connect('library.db')
        cur = con.cursor()
        Query = """Select * from LIBRARY_BOOKS where Title like ? """
        query = cur.execute(Query,(key_word,))
        book_val = query.fetchall()
        if len(book_val)>0:
            cols = [column[0] for column in query.description]
            results = pd.DataFrame.from_records(data = book_val, columns = cols)
            print(results,"\n\n")
        else:
            print("No Book Found try different keyword\n\n")
        cur.close()
        con.close()
    except:
        print("Enter Proper Details \n\n")
        click = input("Enter Any Button to Continue.....:")
        return main_menu()

```

Single level inheritance for listing the books search

```

class loan_book(book_search):
    # Function for loaning a book to member and adds it to a table
    def book_loan():
        try:

```



```

print("                Loan Books                \n\n\n")

con = sqlite3.connect('library.db')

cur = con.cursor()

listOfTables = cur.execute(

    """SELECT name FROM sqlite_master WHERE type='table' AND
name='LOAN_STATUS'; """).fetchall()

if listOfTables == []:

    book = int(input("Enter the ISBN of Book :"))
    member = input("Enter the Member Username(Email Address) :")
    date = input("Enter Date(yyyy-mm-dd) :")
    query = """select Title from LIBRARY_BOOKS where Book_ISBN =?"""
    cur.execute(query, (book,))
    value = cur.fetchone()
    book_title = value[0]
    val = (book, book_title, member, date)

    cur.execute("""create table LOAN_STATUS (Book_ISBN int,Title
nvarchar(255),Username nvarchar(255),Date date NOT NULL);""")
    query = """ insert into LOAN_STATUS values(?,?,?,?)"""
    cur.execute(query,(val))
    con.commit()

    # Updates the book status to loaned
    query = """update LIBRARY_BOOKS set Loan_Status=? where Book_ISBN=?"""
    cur.execute(query, ('Loaned',book))
    con.commit()

    cur.close()

    con.close()

    print("Thank You\n\n")

    click = input("Enter Any Button to Continue.....:")

    return main_menu()

```

```

else:

    book = int(input("Enter the ISBN of Book :"))

    cur.execute("select Loan_Status from LIBRARY_BOOKS where Book_ISBN
="+str(book) +"'")

    val = cur.fetchone()

    status = val[0]

    # Checks whether the book is loaned or available

    if status == 'Available':

        member = input("Enter the Member Username (Email Address) :")

        query = """select Plan from LIBRARY_MEMBERS where Username =?"""

        cur.execute(query, (member,))

        plan_val = cur.fetchone()

        plan_value = plan_val[0]

        # Checks whether the member plan is basic or premium

        if plan_value == 'Basic' or 'basic':

            query= """select count(Username) from LOAN_STATUS where Username =?"""

            limit_val = cur.execute(query,(member,))

            for i in limit_val:

                limit = int(i[0])

            # Checks how many books loaned by member

            if limit<2:

                date = input("Enter Date(yyyy-mm-dd) :")

                query = """select Title from LIBRARY_BOOKS where Book_ISBN =?"""

                cur.execute(query,(book,))

                value = cur.fetchone()

                book_title = value[0]

                tot_val = (book,book_title,member,date)

                query = """ insert into LOAN_STATUS values(?,?,?,?)"""

```

```

        cur.execute(query, (tot_val))
        con.commit()

        query = """update LIBRARY_BOOKS set Loan_Status=? where
Book_ISBN=?"""

        cur.execute(query, ("Loaned",book))
        con.commit()
        cur.close()
        con.close()

        print("Thank You\n\n")
        click = input("Enter Any Button to Continue.....:")
        return main_menu()

    else:

        print("Books cannot be loaned, Please return the loaned books\n\n")
        click = input("Enter Any Button to Continue.....:")
        return main_menu()

    else:

        query = """select count(Username) from LOAN_STATUS where Username =?"""
        limit_val=cur.execute(query,(member,))

        for i in limit_val:
            limit = int(i[0])

        if limit<4:

            date = input("Enter Date(yyyy-mm-dd) :")
            query = """select Title from LIBRARY_BOOKS where Book_ISBN =?"""
            cur.execute(query,(book,))
            value = cur.fetchone()
            book_title = value[0]
            tot_val = (book,book_title,member,date)
            query = """ insert into LOAN_STATUS values(?,?,?,?)"""
            cur.execute(query, (tot_val))

```

```

        con.commit()

        query = """update LIBRARY_BOOKS set Loan_Status=? where
Book_ISBN=?"""

        cur.execute(query, ("Loaned",book))

        con.commit()

        cur.close()

        con.close()

        print("Thank You \n\n")

        click = input("Enter Any Button to Continue.....:")

        return main_menu()

    else:

        print("Books cannot be loaned, Please return the loaned books\n\n")

        click = input("Enter Any Button to Continue.....:")

        return main_menu()

    else:

        print("Book Unavailable\n\n")

        click = input("Enter Any Button to Continue.....:")

        return main_menu()

except:

    print("Enter Proper Details \n\n")

    click = input("Enter Any Button to Continue.....:")

    return main_menu()

# Class for displaying the loaned books by member
class Loaned_books:

    def show_loaned_books():

        try:

            print("

                Loaned Books

                \n\n\n")

```

```

con = sqlite3.connect('library.db')
cur=con.cursor()
query=cur.execute("select Title,Username from LOAN_STATUS")
book_val=query.fetchall()
if len(book_val)>0:
    cols = ["Title","Username"]
    results= pd.DataFrame.from_records(data = book_val, columns = cols)
    print(results,"\n\n")
else:
    print("No Books Loaned \n\n")
cur.close()
con.close()
click=input("Enter Any Button to Continue.....:")
return main_menu()
except:
    print("No Books Loaned or Enter Correct Details \n\n")
    click = input("Enter Any Button to Continue.....:")
    return main_menu()

```

Function for returning the books by member to library

```

def return_books():
    try:
        print("                Return Books                \n\n\n")
        username=input("Enter the Username(Email Address) :")
        con = sqlite3.connect('library.db')
        cur=con.cursor()

        Query="""select Book_ISBN,Title,Username from LOAN_STATUS where
Username=?"""

        query=cur.execute(Query,(username,))

```

```

user_val=query.fetchall()
if len(user_val)>0:
    cols = ["Book_ISBN","Title","Username"]
    results= pd.DataFrame.from_records(data = user_val, columns = cols)
    print(results,"\n\n")
    isbn=int(input("Enter the ISBN :"))

    # Checks how many days the book loaned by member

    query="""SELECT julianday('now') - julianday(Date) FROM LOAN_STATUS where
Username=? and Book_ISBN=?"""
    cur.execute(query,(username,isbn))
    fine_days=cur.fetchone()
    days=int(fine_days[0])

    print("Total Days :", days , "days\n\n")
    # Calculates the fine amount
    if days>60:
        print("Fine Amount 10pounds \n\n")
    else:
        print("Thank You \n\n")

    query="""delete from LOAN_STATUS where Book_ISBN=?"""
    cur.execute(query,(isbn,))
    query="""update LIBRARY_BOOKS set Loan_Status=? where Book_ISBN=?"""
    cur.execute(query,('Available',isbn))
    con.commit()

else:
    print("No Books Loaned By User \n\n")

```

```
cur.close()
con.close()
click=input("Enter Any Button to Continue.....:")
return main_menu()
except:
    print("Enter Proper Details \n\n")
    click=input("Enter Any Button to Continue.....:")
    return main_menu()
```

```
Administrator.administrator()
```