

# **PROJECT ON**

## **LIBRARY MANAGEMENT SYSTEM**

Submitted By:

Brahmaiah Rachapudi

Abhishek Nandagawali

Sudhamsa Nagala

## INDEX

Sr.no	Particulars
1.	Introduction
2.	Project Analysis
3.	Functions And Modules
4.	System Design
5.	Source code
6.	Outputs and Tables

# INTRODUCTION

The aims and objectives are as follows:

- The main aim in library management is adding, issuing and returning of books, and also generating reports on it.

Adding books:

Here we add new books into our database.

Issuing books:

Here we issue books to the candidates.

Returning books:

We get books return from candidates.

Delete books:

We delete books here.

Displaying books:

Here we track all the book details.

Reports:

Here we get information about issued and returned books.

## PROJECT ANALYSIS

### OPERATION ENVIRONMENT :

FRONT END	PYTHON
DATA BASE	MYSQL

Here we have created a online data base:

To connect to database we have to give,

Host	<a href="http://sql12.freessqldatabase.com">sql12.freessqldatabase.com</a>
Database name	sql12621331
Database user	sql12621331
Database password	1ssdLqTHym
Port number	3306

## FUNCTIONS AND MODULES:

Modules:

➤ **Import mysql.connector:**

By importing this module, we are able to get the connection between SQL and python.

Functions:

Connect():

It establishes the connection between Python and MYSQL.

Cursor():

It facilitates the row-by-row processing of records in the result set.

Syntax: <cursor object>=<connection object>.cursor()

Execute():

It is used to execute the sql query and get records using python.

Syntax:<cursor object>.execute<sql query string>

Def():

Is a block of code which only runs when it is called.

Fetchall():

Here it will return all the rows from the result set in the form of a tuple containing the records.

Fetchone():

It will return one row from the result set in the form of a tuple containing the records.

Commit():

It provides changes in the database physically.

## SYSTEM DESIGN

Table Design:

-Our project has 3 MYSQL tables:-

Book Table for keeping track of book

Field	Data Type	Default	key
bname	Varchar(100)	Not null	
author	Varchar(100)		
bcode	Varchar(10)	Not null	Primary key
total	Int		
Subject	Varchar(50)		

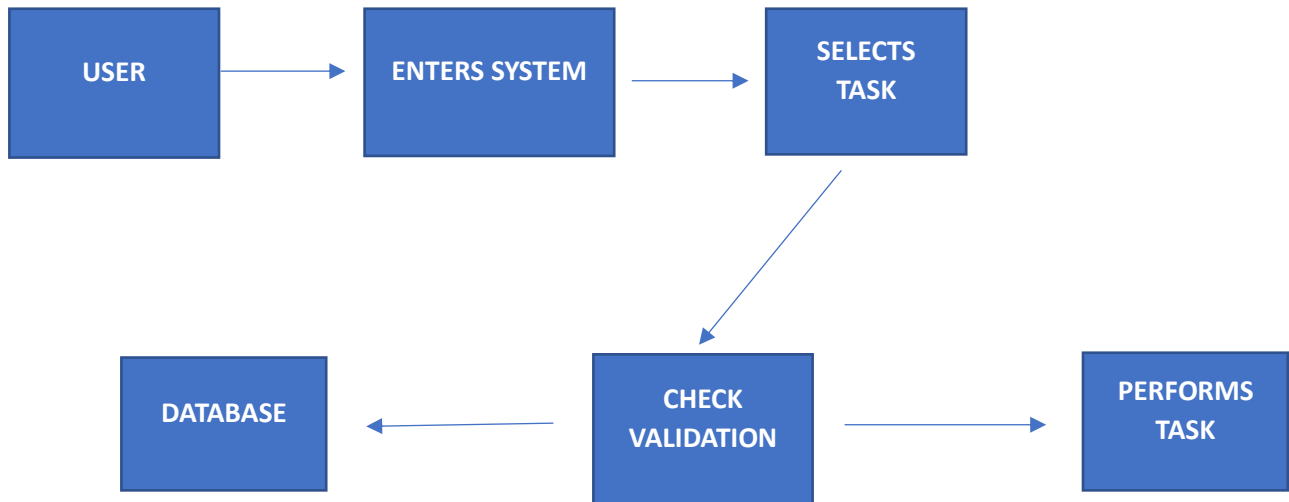
Issue table to issue a book

Field	Data type	Default	Key
name	Varchar(100)		
regno	Int	Not null	Primary key
bcode	Varchar(10)	Not null	
issue_date	Datetime		

Return\_books table for returning a book

field	Data type	Default	Key
name	Varchar(100)		
regno	Int	Not null	Primary key
bcode	Varchar(10)	Not null	
Return_date	Datetime		

## Data Flow Diagram:





## IMPLEMENTATION

### Source code:

```
import mysql.connector as a

conn =
a.connect(host='sql12.freemdb.com',user='sql12621331',password='1ssdLqTHym',database='sql12621331')

my_cursor=conn.cursor()

conn.commit()

print('connection successfully created')


def addbook():
    book_name = input('enter book name: ')
    book_author = input('enter author name: ')
    book_code = input('enter book code: ')
    total_books = int(input('total books: '))
```

```
subject = input('enter subject: ')
data = (book_name, book_author, book_code, total_books, subject)
sql = 'insert into books values (%s, %s, %s, %s, %s);'
my_cursor = conn.cursor()
my_cursor.execute(sql, data)
conn.commit()
print('book added successfully')
wait = input('press enter to continue')

main()
```

```
def issuebook():
    sname = input('enter student name: ')
    reg_no = int(input('enter reg no: '))
    book_code = (input('enter book code: '))
    issue_date = input('enter date: ')
    data = (sname, reg_no, book_code, issue_date)
    sql = 'insert into issue values (%s, %s, %s, %s);'
    my_cursor = conn.cursor()
    my_cursor.execute(sql, data)
    conn.commit()
    print('book issued successfully to:', sname)
```

```
wait = input('press enter to continue')
```

```
bookupdate(book_code, -1)
```

```
main()
```

```
def returnbook():
```

```
    sname = input('Enter student name: ')
```

```
    reg_no = int(input('Enter reg no: '))
```

```
    book_code = (input('Enter book code: '))
```

```
    return_date = input('Enter date: ')
```

```
    data = (sname, reg_no, book_code, return_date)
```

```
    sql = 'insert into return_books values (%s, %s, %s, %s);'
```

```
    my_cursor = conn.cursor()
```

```
    my_cursor.execute(sql, data)
```

```
    conn.commit()
```

```
    print('book returned by:', sname)
```

```
    wait = input('press enter to continue')
```

```
    bookupdate(book_code, 1)
```

```
def bookupdate(book_code, update):
```

```
    sql_select = 'select total from books WHERE bcode = %s;'
```

```
    data = (book_code,)
```

```
    my_cursor = conn.cursor()
```

```
my_cursor.execute(sql_select, data)
myresult = my_cursor.fetchone()
t = myresult[0] + update
sql_update = 'update books SET total = %s WHERE bcode = %s;'
data=(t,book_code)
my_cursor.execute(sql_update, data)
conn.commit()
wait = input('press enter to continue')
main()
```

```
def deletebook():
    book_code = int(input('enter book code: '))
    sql = 'delete from books WHERE bcode = %s;'
    data = (book_code,)
    my_cursor = conn.cursor()
    my_cursor.execute(sql, data)
    conn.commit()
    print('book deleted successfully')
    wait = input('press enter to continue')
    main()
```

```
def displaybook():
```

```
sql = 'select * from books;'
my_cursor = conn.cursor()
my_cursor.execute(sql)
myresult = my_cursor.fetchall()
for i in myresult:
    print('book_name:', i[0])
    print('book_author:', i[1])
    print('book_code:', i[2])
    print('total_books:', i[3])
    print('subject:', i[4])
wait = input('press enter to continue')
main()
```

```
def report_issued_books():
    sql = 'select * from issue;'
    my_cursor = conn.cursor()
    my_cursor.execute(sql)
    myresult = my_cursor.fetchall()
    for i in myresult:
        print(myresult)
    wait = input('press enter to continue')
    main()
```

```
def report_return_books():  
    sql = 'select * from return_books;'  
    my_cursor = conn.cursor()  
    my_cursor.execute(sql)  
    myresult = my_cursor.fetchall()  
    for i in myresult:  
        print(myresult)  
    wait = input('press enter to continue')  
    main()
```

```
def main():  
    print("""  
    LIBRARY MANAGEMENT SYSTEM APPLICATION  
    1. ADD BOOK  
    2. ISSUE BOOK  
    3. RETURN BOOK  
    4. DELETE BOOK  
    5. DISPLAY BOOKS  
    6. REPORT MENU  
    7. EXIT PROGRAM  
    """)
```

```
choice = input('enter task no: ')

if choice == '1':
    addbook()
elif choice == '2':
    issuebook()
elif choice == '3':
    returnbook()
elif choice == '4':
    deletebook()
elif choice == '5':
    displaybook()
elif choice == '6':
    print("""REPORT MENU
1. ISSUED BOOKS
2. RETURNED BOOKS
3. GO BACK TO MAIN MENU
""")
    choice = input('enter task no: ')
    if choice == '1':
        report_issued_books()
```

```
elif choice == '2':  
    report_return_books()  
elif choice == '3':  
    main()  
else:  
    print('please try again')  
    main()  
elif choice == '7':  
    print('Thank you and have a great day ahead')  
else:  
    print('please try again')  
    main()  
  
main()
```



# System Testing

Here we perform dry testing,

The aim of the system testing process is to determine all defects in our project.

Unit Testing:-

Here we check complete environment i.e. by importing module whether the sql connection has established or not.

Integration Testing:-

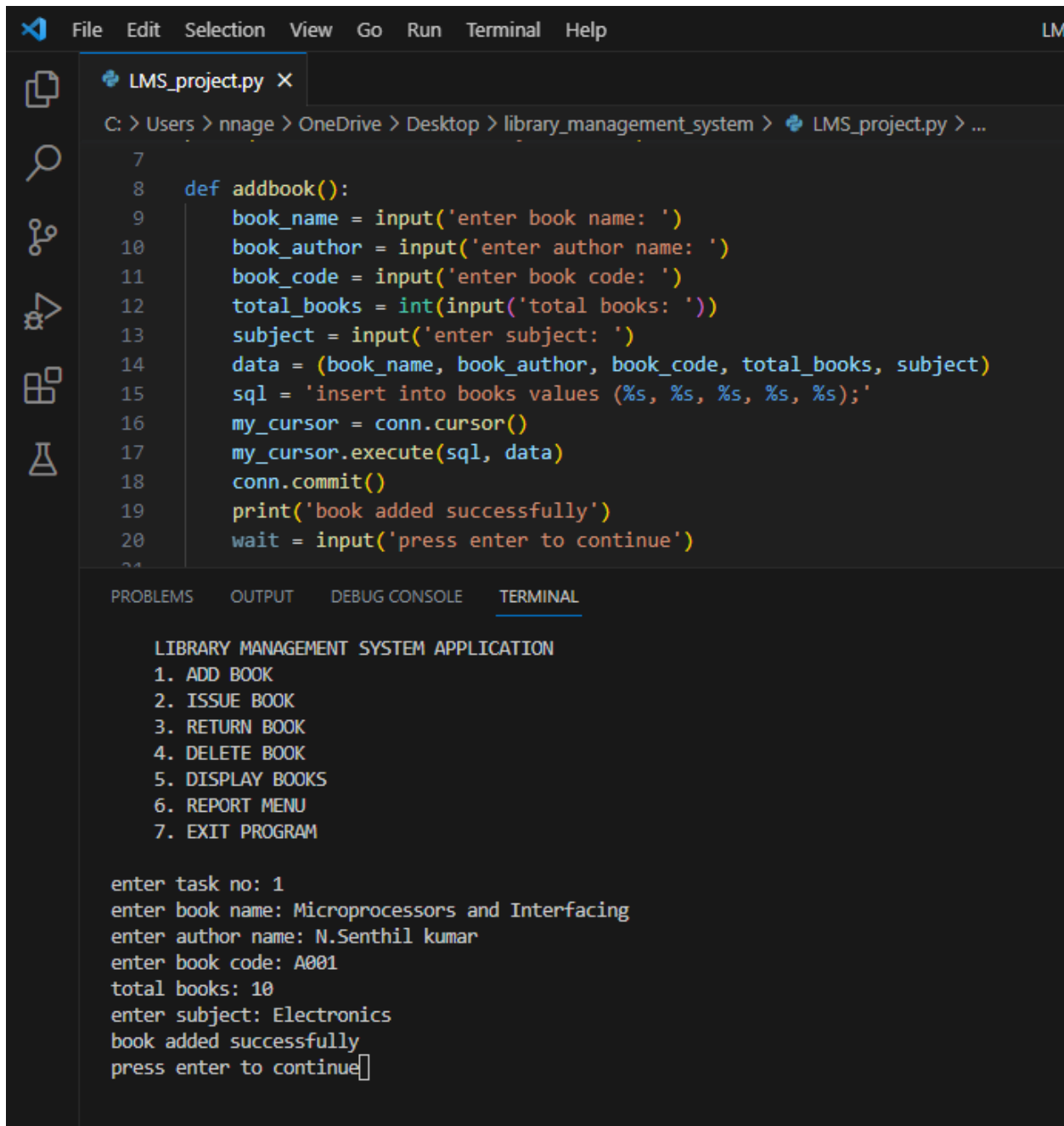
Here we check whether we are getting expected outputs are not.

- ✓ To add book, it will check the book code should be unique value.
- ✓ To issue book, before it will validate the student details like student registration number should be unique value.
- ✓ Return book, here validates the book code and student registration number
- ✓ Report on books, validates whether the system updating about book issued and book returned details.

We user selects other than the mention tasks it will display as “please try again”.

## OUTPUTS AND TABLES

### Add Book:



The image shows a Visual Studio Code editor window with a Python file named `LMS_project.py`. The file path is `C:\Users\> nname > OneDrive > Desktop > library_management_system > LMS_project.py > ...`. The code defines an `addbook()` function that prompts the user for book details and inserts them into a database. The terminal output shows the program's menu and the successful execution of the 'Add Book' task.

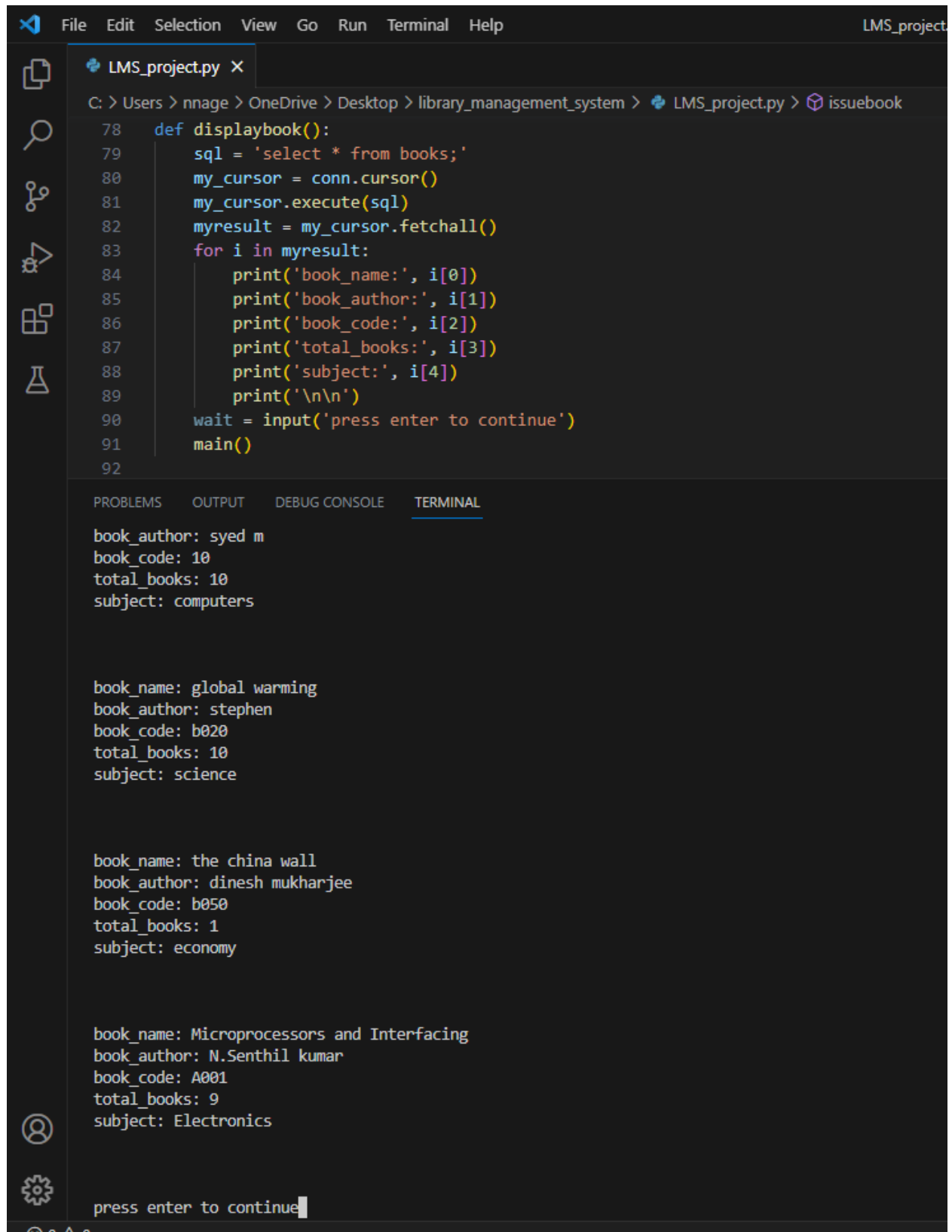
```
7
8 def addbook():
9     book_name = input('enter book name: ')
10    book_author = input('enter author name: ')
11    book_code = input('enter book code: ')
12    total_books = int(input('total books: '))
13    subject = input('enter subject: ')
14    data = (book_name, book_author, book_code, total_books, subject)
15    sql = 'insert into books values (%s, %s, %s, %s, %s);'
16    my_cursor = conn.cursor()
17    my_cursor.execute(sql, data)
18    conn.commit()
19    print('book added successfully')
20    wait = input('press enter to continue')
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
LIBRARY MANAGEMENT SYSTEM APPLICATION
1. ADD BOOK
2. ISSUE BOOK
3. RETURN BOOK
4. DELETE BOOK
5. DISPLAY BOOKS
6. REPORT MENU
7. EXIT PROGRAM

enter task no: 1
enter book name: Microprocessors and Interfacing
enter author name: N.Senthil kumar
enter book code: A001
total books: 10
enter subject: Electronics
book added successfully
press enter to continue
```

## Display Books:



The image shows a Visual Studio Code editor window with a Python file named `LMS_project.py` open. The file path is `C:\Users\> nnage > OneDrive > Desktop > library_management_system > LMS_project.py > issuebook`. The code defines a `displaybook()` function that queries a database for book information and prints it. The terminal output shows the results of the function being called, displaying details for four books.

```
78 def displaybook():
79     sql = 'select * from books;'
80     my_cursor = conn.cursor()
81     my_cursor.execute(sql)
82     myresult = my_cursor.fetchall()
83     for i in myresult:
84         print('book_name:', i[0])
85         print('book_author:', i[1])
86         print('book_code:', i[2])
87         print('total_books:', i[3])
88         print('subject:', i[4])
89         print('\n\n')
90     wait = input('press enter to continue')
91     main()
92
```

book\_author: syed m  
book\_code: 10  
total\_books: 10  
subject: computers

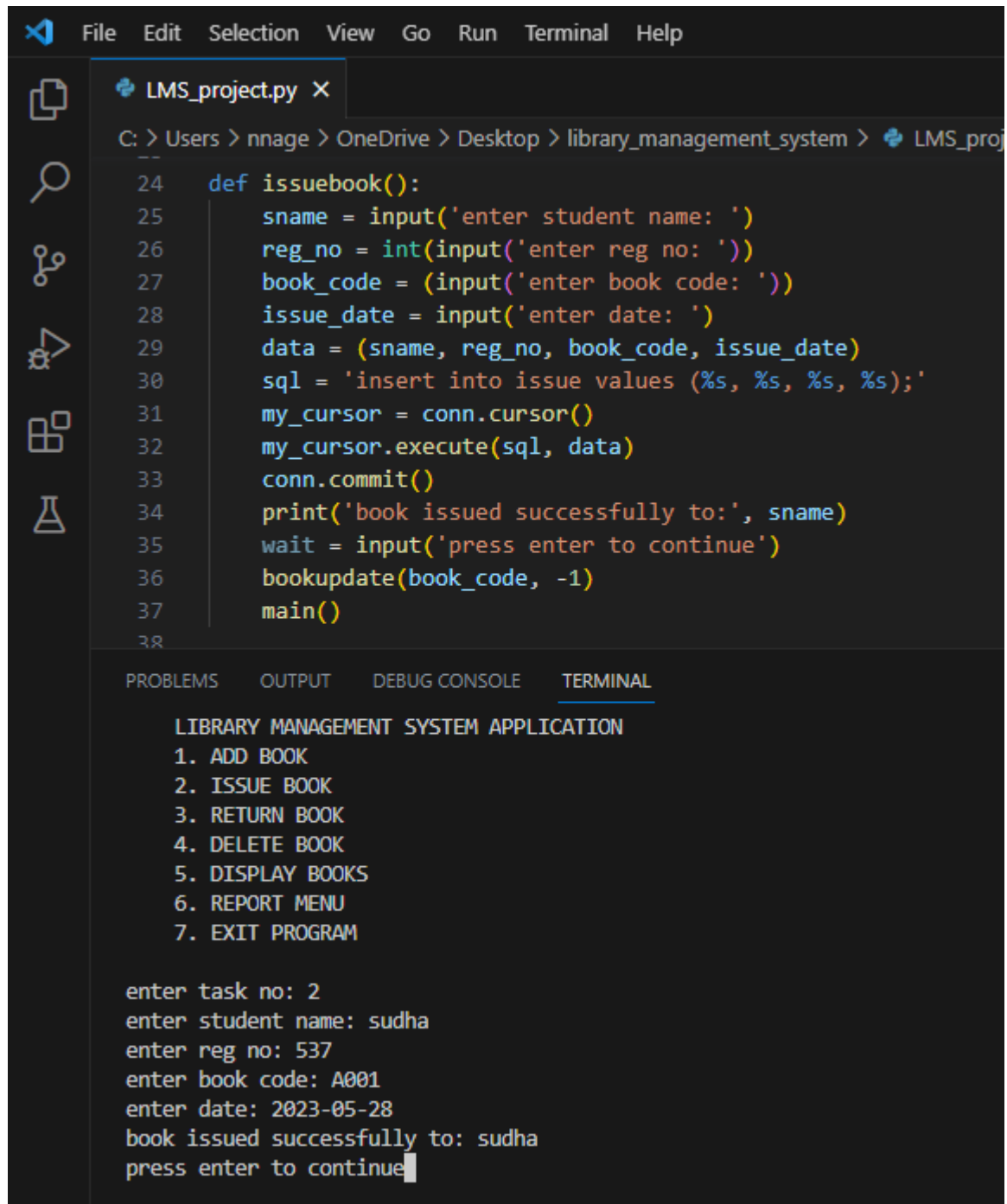
book\_name: global warming  
book\_author: stephen  
book\_code: b020  
total\_books: 10  
subject: science

book\_name: the china wall  
book\_author: dinesh mukharjee  
book\_code: b050  
total\_books: 1  
subject: economy

book\_name: Microprocessors and Interfacing  
book\_author: N.Senthil kumar  
book\_code: A001  
total\_books: 9  
subject: Electronics

press enter to continue

## Issue Book:



The image shows a Visual Studio Code editor window with a Python file named `LMS_project.py` open. The file path is `C:\> Users > nnage > OneDrive > Desktop > library_management_system > LMS_proj`. The code defines a function `issuebook()` that takes user input for student name, registration number, book code, and issue date, then inserts this data into a database. The terminal output shows the program's menu and the successful execution of the 'Issue Book' task.

```
24 def issuebook():
25     sname = input('enter student name: ')
26     reg_no = int(input('enter reg no: '))
27     book_code = (input('enter book code: '))
28     issue_date = input('enter date: ')
29     data = (sname, reg_no, book_code, issue_date)
30     sql = 'insert into issue values (%s, %s, %s, %s);'
31     my_cursor = conn.cursor()
32     my_cursor.execute(sql, data)
33     conn.commit()
34     print('book issued successfully to:', sname)
35     wait = input('press enter to continue')
36     bookupdate(book_code, -1)
37     main()
38
```

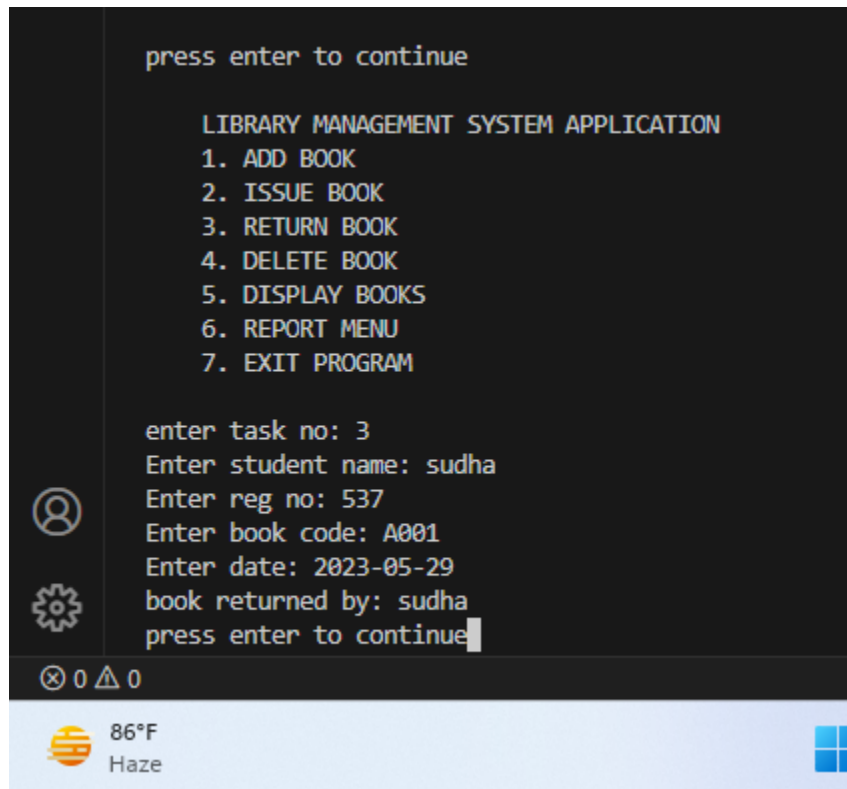
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

LIBRARY MANAGEMENT SYSTEM APPLICATION

1. ADD BOOK
2. ISSUE BOOK
3. RETURN BOOK
4. DELETE BOOK
5. DISPLAY BOOKS
6. REPORT MENU
7. EXIT PROGRAM

enter task no: 2  
enter student name: sudha  
enter reg no: 537  
enter book code: A001  
enter date: 2023-05-28  
book issued successfully to: sudha  
press enter to continue

## Return Book:



# Returned Book\_Report:

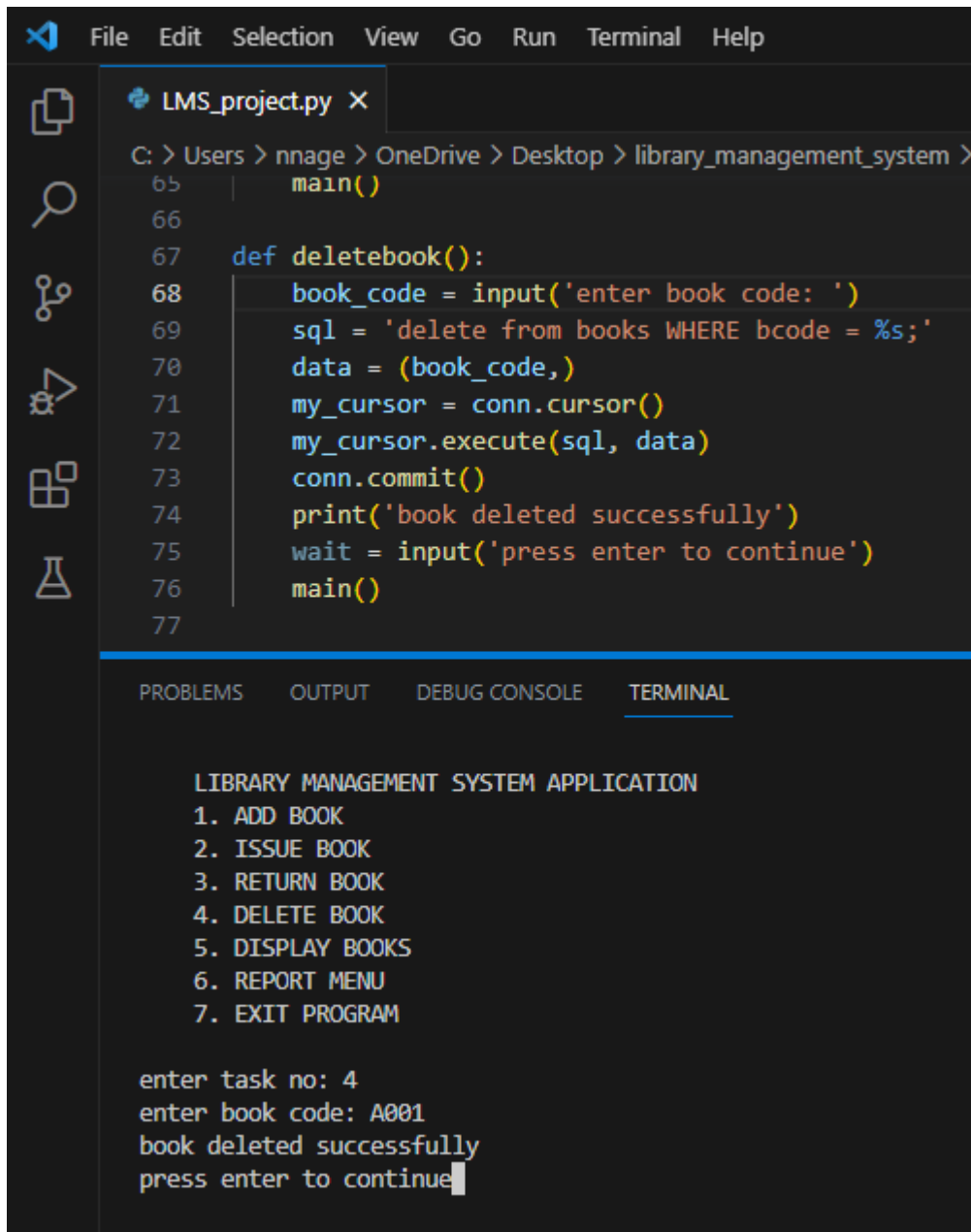
```
LMS_project.py X
C:\Users> nmage> OneDrive> Desktop> library_management_system> LMS_project.py> displaybook> displaybook.py
72 my_cursor.execute(sql, data)
73 conn.commit()
74 print('book deleted successfully')
75 wait = input('press enter to continue')
76 main()
77
78 def displaybook():
79
80     sql = 'select * from books;'
81     my_cursor = conn.cursor()
82     my_cursor.execute(sql)
83     myresult = my_cursor.fetchall()
84     for i in myresult:
85         print('book name:', i[0])
86         print('book author:', i[1])

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
), ('ROJA', '540', 'B015', datetime.datetime(2023, 5, 10, 0, 0)), ('SEETHA', '541', 'B012', datetime.datetime(2023, 3, 5, 0, 0)), ('VYSHU', '542', 'B001', datetime.datetime(2023, 5, 31, 0, 0)), ('ABHI', '543', 'B008', datetime.datetime(2023, 4, 14, 0, 0)), ('RAM', '544', 'B011', datetime.datetime(2023, 2, 20, 0, 0)), ('TARUN', '545', 'B003', datetime.datetime(2023, 2, 5, 0, 0)), ('ABHISHEK', '546', 'B004', datetime.datetime(2023, 5, 30, 0, 0)), ('POOJA', '547', 'B005', datetime.datetime(2023, 4, 16, 0, 0)), ('PRIYA', '548', 'B006', datetime.datetime(2023, 5, 11, 0, 0)), ('sudha', '537', '1', None), ('vasu', '538', 'B001', None), ('sudha', '537', '100', None), ('devi', '561', '200', datetime.datetime(2023, 5, 25, 0, 0)), ('hamsa', '500', '1', datetime.datetime(2023, 5, 29, 0, 0)), ('sudha', '537', '10', datetime.datetime(2023, 5, 28, 0, 0)), ('vasu', '538', '20', datetime.datetime(2023, 5, 20, 0, 0)), ('sudha', '537', '20', datetime.datetime(2023, 5, 30, 0, 0)), ('vasu', '538', '10', datetime.datetime(2023, 2, 1, 0, 0)), ('sudha', '537', 'b020', datetime.datetime(2023, 5, 1, 0, 0)), ('sudha', '537', 'b020', datetime.datetime(2023, 5, 20, 0, 0)), ('sudha', '537', 'A001', datetime.datetime(2023, 5, 29, 0, 0))]
[('RAJU', '539', 'B010', datetime.datetime(2023, 2, 12, 0, 0)), ('SUDHA', '537', 'B001', datetime.datetime(2023, 3, 28, 0, 0)), ('VASU', '538', 'B006', datetime.datetime(2023, 2, 25, 0, 0)), ('ROJA', '540', 'B015', datetime.datetime(2023, 5, 10, 0, 0)), ('SEETHA', '541', 'B012', datetime.datetime(2023, 3, 5, 0, 0)), ('VYSHU', '542', 'B001', datetime.datetime(2023, 5, 31, 0, 0)), ('ABHI', '543', 'B008', datetime.datetime(2023, 4, 14, 0, 0)), ('RAM', '544', 'B011', datetime.datetime(2023, 2, 20, 0, 0)), ('TARUN', '545', 'B003', datetime.datetime(2023, 2, 5, 0, 0)), ('ABHISHEK', '546', 'B004', datetime.datetime(2023, 5, 30, 0, 0)), ('POOJA', '547', 'B005', datetime.datetime(2023, 4, 16, 0, 0)), ('PRIYA', '548', 'B006', datetime.datetime(2023, 5, 11, 0, 0)), ('sudha', '537', '1', None), ('vasu', '538', 'B001', None), ('sudha', '537', '100', None), ('devi', '561', '200', datetime.datetime(2023, 5, 25, 0, 0)), ('hamsa', '500', '1', datetime.datetime(2023, 5, 29, 0, 0)), ('sudha', '537', '10', datetime.datetime(2023, 5, 28, 0, 0)), ('vasu', '538', '20', datetime.datetime(2023, 5, 20, 0, 0)), ('sudha', '537', '20', datetime.datetime(2023, 5, 30, 0, 0)), ('vasu', '538', '10', datetime.datetime(2023, 2, 1, 0, 0)), ('sudha', '537', 'b020', datetime.datetime(2023, 5, 1, 0, 0)), ('sudha', '537', 'b020', datetime.datetime(2023, 5, 20, 0, 0)), ('sudha', '537', 'A001', datetime.datetime(2023, 5, 29, 0, 0))]
[('RAJU', '539', 'B010', datetime.datetime(2023, 2, 12, 0, 0)), ('SUDHA', '537', 'B001', datetime.datetime(2023, 3, 28, 0, 0)), ('VASU', '538', 'B006', datetime.datetime(2023, 2, 25, 0, 0)), ('ROJA', '540', 'B015', datetime.datetime(2023, 5, 10, 0, 0)), ('SEETHA', '541', 'B012', datetime.datetime(2023, 3, 5, 0, 0)), ('VYSHU', '542', 'B001', datetime.datetime(2023, 5, 31, 0, 0)), ('ABHI', '543', 'B008', datetime.datetime(2023, 4, 14, 0, 0)), ('RAM', '544', 'B011', datetime.datetime(2023, 2, 20, 0, 0)), ('TARUN', '545', 'B003', datetime.datetime(2023, 2, 5, 0, 0)), ('ABHISHEK', '546', 'B004', datetime.datetime(2023, 5, 30, 0, 0)), ('POOJA', '547', 'B005', datetime.datetime(2023, 4, 16, 0, 0)), ('PRIYA', '548', 'B006', datetime.datetime(2023, 5, 11, 0, 0)), ('sudha', '537', '1', None), ('vasu', '538', 'B001', None), ('sudha', '537', '100', None), ('devi', '561', '200', datetime.datetime(2023, 5, 25, 0, 0)), ('hamsa', '500', '1', datetime.datetime(2023, 5, 29, 0, 0)), ('sudha', '537', '10', datetime.datetime(2023, 5, 28, 0, 0)), ('vasu', '538', '20', datetime.datetime(2023, 5, 20, 0, 0)), ('sudha', '537', '20', datetime.datetime(2023, 5, 30, 0, 0)), ('vasu', '538', '10', datetime.datetime(2023, 2, 1, 0, 0)), ('sudha', '537', 'b020', datetime.datetime(2023, 5, 1, 0, 0)), ('sudha', '537', 'b020', datetime.datetime(2023, 5, 20, 0, 0)), ('sudha', '537', 'A001', datetime.datetime(2023, 5, 29, 0, 0))]
[('RAJU', '539', 'B010', datetime.datetime(2023, 2, 12, 0, 0)), ('SUDHA', '537', 'B001', datetime.datetime(2023, 3, 28, 0, 0)), ('VASU', '538', 'B006', datetime.datetime(2023, 2, 25, 0, 0)), ('ROJA', '540', 'B015', datetime.datetime(2023, 5, 10, 0, 0)), ('SEETHA', '541', 'B012', datetime.datetime(2023, 3, 5, 0, 0)), ('VYSHU', '542', 'B001', datetime.datetime(2023, 5, 31, 0, 0)), ('ABHI', '543', 'B008', datetime.datetime(2023, 4, 14, 0, 0)), ('RAM', '544', 'B011', datetime.datetime(2023, 2, 20, 0, 0)), ('TARUN', '545', 'B003', datetime.datetime(2023, 2, 5, 0, 0)), ('ABHISHEK', '546', 'B004', datetime.datetime(2023, 5, 30, 0, 0)), ('POOJA', '547', 'B005', datetime.datetime(2023, 4, 16, 0, 0)), ('PRIYA', '548', 'B006', datetime.datetime(2023, 5, 11, 0, 0)), ('sudha', '537', '1', None), ('vasu', '538', 'B001', None), ('sudha', '537', '100', None), ('devi', '561', '200', datetime.datetime(2023, 5, 25, 0, 0)), ('hamsa', '500', '1', datetime.datetime(2023, 5, 29, 0, 0)), ('sudha', '537', '10', datetime.datetime(2023, 5, 28, 0, 0)), ('vasu', '538', '20', datetime.datetime(2023, 5, 20, 0, 0)), ('sudha', '537', '20', datetime.datetime(2023, 5, 30, 0, 0)), ('vasu', '538', '10', datetime.datetime(2023, 2, 1, 0, 0)), ('sudha', '537', 'b020', datetime.datetime(2023, 5, 1, 0, 0)), ('sudha', '537', 'b020', datetime.datetime(2023, 5, 20, 0, 0)), ('sudha', '537', 'A001', datetime.datetime(2023, 5, 29, 0, 0))]
press enter to continue
```

## Delete Book:



```
File Edit Selection View Go Run Terminal Help
LMS_project.py X
C: > Users > nnage > OneDrive > Desktop > library_management_system >
65     main()
66
67     def deletebook():
68         book_code = input('enter book code: ')
69         sql = 'delete from books WHERE bcode = %s;'
70         data = (book_code,)
71         my_cursor = conn.cursor()
72         my_cursor.execute(sql, data)
73         conn.commit()
74         print('book deleted successfully')
75         wait = input('press enter to continue')
76         main()
77
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

LIBRARY MANAGEMENT SYSTEM APPLICATION

1. ADD BOOK
2. ISSUE BOOK
3. RETURN BOOK
4. DELETE BOOK
5. DISPLAY BOOKS
6. REPORT MENU
7. EXIT PROGRAM

enter task no: 4  
enter book code: A001  
book deleted successfully  
press enter to continue

## SQL Tables:

## Book Table:

The screenshot shows a SQL IDE interface with a menu bar (File, Edit, View, Query, Database, Server, Tools, Scripting, Help) and a toolbar. The left sidebar displays the 'SCHEMAS' tree for 'sql12621331', listing tables: books, issue, return\_books, Views, Stored Procedures, and Functions. The main query editor shows a single query: `SELECT * FROM sql12621331.books;`. The 'Result Grid' at the bottom displays the query results in a table format.

bname	author	bcode	total	subject
Happy Dreams	Jia Pingwa	B011	9	literature
The Only zero	Julian Barnes	B012	1	mathematics
harry porter	NULL	4023	10	2
introduction to electrical...	jhonsmyth	b001	2	electrical engin...
money matters	rangarajan	100	5	statistics
my experiments with truth	MK Gandhi	1	9	philosophy
who will cry when you die	Robinsharma	b025	5	Philosophy
sudhamurthy	sudhamurthy	789	5	English
100 questions program...	veronica	10	1	computers
career in testing	arjun rathod	30	15	computers
career in testing	arjun rathod	30	10	computers
learning mysql	syed m	10	10	computers
global warming	stephen	b020	10	science
the china wall	dinesh mukharjee	b050	1	economy

The interface also includes a 'Filter Rows' field, an 'Export' button, and a 'Wrap Cell Content' option. The bottom status bar shows 'books 1 x' and 'Read Only'.



## Issue Table:

MySQL Workbench

myonline\_sql - Warning - not... x

File Edit View Query Database Server Tools Scripting Help

Navigator: books issue return\_books

**SCHEMAS**

Filter objects

sql12621331

- Tables
  - books
  - issue
  - return\_books
- Views
- Stored Procedures
- Functions

Limit to 1000 rows

1 • `SELECT * FROM sql12621331.issue;`

**Result Grid** | Filter Rows: | Export: | Wrap Cell Content: |

	name	regno	bcode	issue_date
	sudha	537	10	2023-05-27 00:00:00
	sudha	537	10	2023-05-27 00:00:00
	sudha	537	10	2023-05-27 00:00:00
	vasu	538	20	2023-01-01 00:00:00
	sudha	537	20	2023-05-27 00:00:00
	vasu	538	10	2023-01-01 00:00:00
	sudha	537	30	2023-01-01 00:00:00
	sudha	537	10	2023-05-05 00:00:00
	sudha	537	b001	2023-01-10 00:00:00
	vasu	538	30	2023-01-01 00:00:00
	sudha	537	b020	2023-04-10 00:00:00
	sudha	537	b020	2023-05-01 00:00:00
	sudha	537	b050	2023-05-05 00:00:00
	sudha	537	A001	2023-05-28 00:00:00

Administration Schemas

Information

Result Grid

Form Editor

Field Types

Query Stats

## Return Table:

MySQL Workbench

myonline\_sql - Warning - not... x

File Edit View Query Database Server Tools Scripting Help

Navigator: myonline\_sql - Warning - not... x

SCHEMAS

Filter objects

sql12621331

- Tables
  - books
  - issue
  - return\_books
- Views
- Stored Procedures
- Functions

books issue return\_books x

Limit to 1000 rows

1 • SELECT \* FROM sql12621331.return\_books;

Result Grid

name	regno	bcode	return_date
POOJA	547	B005	2023-04-16 00:00:00
PRIYA	548	B006	2023-05-11 00:00:00
sudha	537	1	0000-00-00 00:00:00
vasu	538	b001	0000-00-00 00:00:00
sudha	537	100	0000-00-00 00:00:00
devi	561	200	2023-05-25 00:00:00
hamsa	500	1	2023-05-29 00:00:00
sudha	537	10	2023-05-28 00:00:00
vasu	538	20	2023-05-20 00:00:00
sudha	537	20	2023-05-30 00:00:00
vasu	538	10	2023-02-01 00:00:00
sudha	537	b020	2023-05-01 00:00:00
sudha	537	b020	2023-05-20 00:00:00
sudha	537	A001	2023-05-29 00:00:00

etum\_books 1 x

Read Only

Output

Action Output

#	Time	Action	Message
1	02:52:39	SELECT * FROM sql12621331.books LIMIT 0, 1000	24 row(s) re
2	02:52:42	SELECT * FROM sql12621331.issue LIMIT 0, 1000	39 row(s) re
3	02:52:43	SELECT * FROM sql12621331.return_books LIMIT 0, 1000	24 row(s) re

No object selected

