

MySQL PROJECT

Topic : Library Management System

Create a database named library and following TABLES in the database:

1. Branch
2. Employee
3. Books
4. Customer
5. IssueStatus
5. ReturnStatus

Attributes for the tables:

1. Branch

Branch_no - Set as PRIMARY KEY
Manager_Id
Branch_address
Contact_no

```
9  -- ;
10
11  -- Insert 10 rows into the branch table
12  • INSERT INTO Branch (Branch_no, Manager_Id, Branch_address, Contact_no)
13  VALUES
14  (1, 101, '123_Main_St', '555-1234'),
15  (2, 102, '456 Elm St', '555-5678'),
16  (3, 103, '789 Oak St', '555-9012'),
17  (4, 104, '321 Pine St', '555-3456'),
18  (5, 105, '654 Maple St', '555-7890'),
19  (6, 106, '987 Cedar St', '555-2345'),
20  (7, 107, '654 Birch St', '555-6789'),
21  (8, 108, '321 Walnut St', '555-1234'),
22  (9, 109, '789 Cherry St', '555-5678'),
```

Result Grid	Filter Rows:	Edit:	Export/Import:	Wrap Cell Content:
Branch_no	Manager_Id	Branch_address	Contact_no	
1	101	123_Main_St	555-1234	
2	102	456 Elm St	555-5678	
3	103	789 Oak St	555-9012	
4	104	321 Pine St	555-3456	
5	105	654 Maple St	555-7890	
6	106	987 Cedar St	555-2345	

Branch 1 x

2. Employee

Emp_Id – Set as PRIMARY KEY

Emp_name

Position

Salary

Branch_no - Set as FOREIGN KEY and it refer Branch_no in Branch table

The screenshot displays a database management interface. At the top, a toolbar includes icons for file operations, search, and execution, along with a 'Limit to 1000 rows' dropdown. The SQL editor shows an `INSERT INTO` statement for the `library.Employee` table, followed by a `VALUES` clause with 10 rows of data. Below the editor, the 'Result Grid' tab is active, showing the first five rows of the inserted data. The 'Output' tab is also visible, showing the execution log with two entries: a successful insert and a successful select query.

```
37 • INSERT INTO library.Employee (Emp_Id, Emp_name, Position, Salary, Branch_no)
38 VALUES
39     (1, 'John_Davis', 'Librarian', 50000.00, 1),
40     (2, 'Willow_Smith', 'Assistant_Librarian', 40000.00, 2),
41     (3, 'Alice_Johnson', 'Cataloger', 35000.00, 3),
42     (4, 'Bob_Williams', 'Reference_Librarian', 45000.00, 1),
43     (5, 'Emily_Brown', 'Library_Assistant', 30000.00, 2),
44     (6, 'Michael_Davis', 'Archivist', 40000.00, 3),
45     (7, 'Jessica_Wilson', 'Library_Director', 60000.00, 1),
46     (8, 'David_Martinez', 'Children's_Librarian', 45000.00, 2),
47     (9, 'Sarah_Anderson', 'Acquisitions_Librarian', 45000.00, 3),
48     (10, 'Christopher_Taylor', 'Systems_Librarian', 50000.00, 2);
49 • SELECT * FROM Employee;
```

Emp_Id	Emp_name	Position	Salary	Branch_no
1	John_Davis	Librarian	50000.00	1
2	Willow_Smith	Assistant_Librarian	40000.00	2
3	Alice_Johnson	Cataloger	35000.00	3
4	Bob_Williams	Reference_Librarian	45000.00	1
5	Emily_Brown	Library_Assistant	30000.00	2

Employee 2 x

Output

Action Output

#	Time	Action	Message
✓ 17	16:21:05	INSERT INTO library.Employee (Emp_Id, Emp_name, Position, Salary, Branch_no) VALU...	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0
✓ 18	16:21:11	SELECT * FROM Employee LIMIT 0, 1000	10 row(s) returned

3. Books

ISBN - Set as PRIMARY KEY

Book_title

Category

Rental_Price

Status [Give yes if book available and no if book not available]

Author

Publisher

mysql task 4* mysql task 5* mysql task 6* mysql task 7 stored procedures* Triggers* mysql task 8* mysql task 9* mysql task 10* mysql pr*

VALUES

```
87
88 ('9780679601085', 'The Prophet', 'Fiction', 10.99, 'yes', 'Kahlil Gibran', 'Alfred A. Knopf'),
89 ('9780141439662', 'Sense and Sensibility', 'Classic', 8.99, 'yes', 'Jane Austen', 'Penguin Classics'),
90 ('9780062316110', 'Sapiens: A Brief History of Humankind', 'Non-fiction', 12.99, 'yes', 'Yuval Noah Harari', 'Harper'),
91 ('9788172235652', 'God of Small Things', 'Fiction', 11.99, 'yes', 'Arundhati Roy', 'IndiaInk'),
92 ('9788170289186', 'Gitanjali', 'Poetry', 9.99, 'yes', 'Rabindranath Tagore', 'Rupa & Co.'),
93 ('9780307947055', '1984', 'Classic', 9.99, 'yes', 'George Orwell', 'Vintage'),
94 ('9780061120084', 'To Kill a Mockingbird', 'Classic', 10.99, 'yes', 'Harper Lee', 'Harper Perennial Modern Classics'),
95 ('9780307277671', 'The Catcher in the Rye', 'Fiction', 8.99, 'yes', 'J.D. Salinger', 'Little, Brown and Company'),
96 ('9780307477729', 'The Girl with the Dragon Tattoo', 'Mystery', 11.99, 'yes', 'Stieg Larsson', 'Vintage Crime/Black Lizard'),
97 ('9781400031702', 'The Da Vinci Code', 'Mystery', 10.99, 'yes', 'Dan Brown', 'Anchor');
98
99 • select * from books;
```

Result Grid

ISBN	Book_title	Category	Rental_Price	Status	Author	Publisher
9780061120084	To Kill a Mockingbird	Classic	10.99	yes	Harper Lee	Harper Perennial Modern Classics
9780062316110	Sapiens: A Brief History of Humankind	Non-fiction	12.99	yes	Yuval Noah Harari	Harper
9780141439662	Sense and Sensibility	Classic	8.99	yes	Jane Austen	Penguin Classics
9780307277671	The Catcher in the Rye	Fiction	8.99	yes	J.D. Salinger	Little, Brown and Company
9780307477729	The Girl with the Dragon Tattoo	Mystery	11.99	yes	Stieg Larsson	Vintage Crime/Black Lizard

books 4 x Apply Revert

Output

Action Output

#	Time	Action	Message	Duration / Fetch
23	16:24:20	INSERT INTO library.Books (ISBN, Book_title, Category, Rental_Price, Status, Author, Pu...	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.015 sec
24	16:24:29	select * from books LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec

4. Customer

Customer_Id - Set as PRIMARY KEY

Customer_name

Customer_address

Reg_date

The screenshot shows a MySQL IDE interface. The top toolbar includes icons for file operations, execution, and search. Below the toolbar, a SQL editor displays a series of INSERT statements for a 'customer' table, followed by a SELECT statement. The SQL editor is line-numbered from 61 to 73. Below the editor, a 'Result Grid' shows the first five rows of data. At the bottom, an 'Action Output' pane displays the execution results of the SQL statements.

```
61 VALUES
62     (1, 'Homer Simpson', '742 Evergreen Terrace, Springfield', '2023-01-15'),
63     (2, 'Phoebe Buffay', 'Apartment 20, 495 Grove Street, New York', '2023-02-28'),
64     (3, 'Barney Stinson', 'Apartment 4C, 79th Street, New York', '2023-03-10'),
65     (4, 'Chandler Bing', 'Apartment 19, 15 Yemen Road, Yemen', '2023-04-05'),
66     (5, 'Michael Scott', 'Dunder Mifflin Paper Company, Scranton', '2023-05-20'),
67     (6, 'Leslie Knope', 'City Hall, Pawnee', '2023-06-12'),
68     (7, 'Sheldon Cooper', 'Apartment 4A, 2311 Los Robles Avenue, Pasadena', '2023-07-18'),
69     (8, 'Phyllis Vance', 'Dunder Mifflin Paper Company, Scranton', '2023-08-02'),
70     (9, 'Ron Swanson', 'City Hall, Pawnee', '2023-09-09'),
71     (10, 'Monica Geller', 'Apartment 20, 495 Grove Street, New York', '2023-10-30');
72
73 • select * from customer;
```

Customer_Id	Customer_name	Customer_address	Reg_date
1	Homer Simpson	742 Evergreen Terrace, Springfield	2023-01-15
2	Phoebe Buffay	Apartment 20, 495 Grove Street, New York	2023-02-28
3	Barney Stinson	Apartment 4C, 79th Street, New York	2023-03-10
4	Chandler Bing	Apartment 19, 15 Yemen Road, Yemen	2023-04-05
5	Michael Scott	Dunder Mifflin Paper Company, Scranton	2023-05-20

customer 3 x

Output :

Action Output

#	Time	Action	Message
20	16:22:14	INSERT INTO library.Customer (Customer_Id, Customer_name, Customer_address, Reg_...	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0
21	16:23:15	select * from customer LIMIT 0, 1000	10 row(s) returned

5. IssueStatus

Issue_Id - Set as PRIMARY KEY

Issued_cust – Set as FOREIGN KEY and it refer customer_id in CUSTOMER table

Issued_book_name

Issue_date

Isbn_book – Set as FOREIGN KEY and it should refer isbn in BOOKS table

The screenshot displays a database management interface. At the top, a toolbar includes icons for file operations and a 'Limit to 1000 rows' dropdown. Below the toolbar, a SQL editor shows a query: `select * from issuestatus;`. The results are displayed in a 'Result Grid' with columns: Issue_Id, Issued_cust, Issued_book_name, Issue_date, and Isbn_book. The grid contains 10 rows of data. Below the grid, an 'Output' section shows the execution log with two entries: an INSERT statement at 16:25:36 and a SELECT statement at 16:26:01.

VALUES

```
(1, 1, 'The Prophet', '2024-02-01', '9780679601085'),
(2, 2, 'Sense and Sensibility', '2024-02-02', '9780141439662'),
(3, 3, 'Sapiens: A Brief History of Humankind', '2024-02-03', '9780062316110'),
(4, 4, 'God of Small Things', '2024-02-04', '9788172235652'),
(5, 5, 'Gitanjali', '2024-02-05', '9788170289186'),
(6, 6, '1984', '2024-02-06', '9780307947055'),
(7, 7, 'To Kill a Mockingbird', '2024-02-07', '9780061120084'),
(8, 8, 'The Catcher in the Rye', '2024-02-08', '9780307277671'),
(9, 9, 'The Girl with the Dragon Tattoo', '2024-02-09', '9780307477729'),
(10, 10, 'The Da Vinci Code', '2024-02-10', '9781400031702');
```

select * from issuestatus;

Result Grid

	Issue_Id	Issued_cust	Issued_book_name	Issue_date	Isbn_book
1	1	1	The Prophet	2024-02-01	9780679601085
2	2	2	Sense and Sensibility	2024-02-02	9780141439662
3	3	3	Sapiens: A Brief History of Humankind	2024-02-03	9780062316110
4	4	4	God of Small Things	2024-02-04	9788172235652
5	5	5	Gitanjali	2024-02-05	9788170289186

ssuestatus 5 x

Output

Action Output

#	Time	Action	Message
26	16:25:36	INSERT INTO library.IssueStatus (Issue_Id, Issued_cust, Issued_book_name, Issue_date...	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0
27	16:26:01	select * from issuestatus LIMIT 0, 1000	10 row(s) returned

6. ReturnStatus

Return_Id - Set as PRIMARY KEY

Return_cust

Return_book_name

Return_date

Isbn_book2 - Set as FOREIGN KEY and it should refer isbn in BOOKS table

The screenshot shows a MySQL IDE interface. The SQL editor contains the following code:

```
136 );
137
138 -- Insert 6 rows into the ReturnStatus table
139 • INSERT INTO library.ReturnStatus (Return_Id, Return_cust, Return_book_name, Return_date, Isbn_book2)
140   VALUES
141     (1, 1, 'The Prophet', '2024-02-11', '9780679601085'),
142     (2, 2, 'Sense and Sensibility', '2024-02-12', '9780141439662'),
143     (3, 3, 'Sapiens: A Brief History of Humankind', '2024-02-13', '9780062316110'),
144     (4, 4, 'God of Small Things', '2024-02-14', '9788172235652'),
145     (5, 5, 'Gitanjali', '2024-02-15', '9788170289186'),
146     (6, 6, '1984', '2024-02-16', '9780307947055');
147
148 • select * from returnstatus;
```

Below the editor, the 'Result Grid' shows the data inserted into the 'returnstatus' table:

Return_Id	Return_cust	Return_book_name	Return_date	Isbn_book2
1	1	The Prophet	2024-02-11	9780679601085
2	2	Sense and Sensibility	2024-02-12	9780141439662
3	3	Sapiens: A Brief History of Humankind	2024-02-13	9780062316110
4	4	God of Small Things	2024-02-14	9788172235652
5	5	Gitanjali	2024-02-15	9788170289186

The 'Output' pane shows the execution results:

#	Time	Action	Message
29	16:26:57	INSERT INTO library.ReturnStatus (Return_Id, Return_cust, Return_book_name, Return...	6 row(s) affected Records: 6 Duplicates: 0 Warnings: 0
30	16:27:18	select * from returnstatus LIMIT 0, 1000	6 row(s) returned

Display all the tables and Write the queries for the following :

1. Retrieve the book title, category, and rental price of all available books.

The screenshot shows a database management tool interface. The top toolbar includes icons for file operations, execution, and settings, with a dropdown menu set to "Limit to 1000 rows". The SQL editor contains the following queries:

```
149
150 -- Execute the statement under the keyboard cursor and rental price of all available books.
151 • SELECT Book_title, Category, Rental_Price FROM library.Books WHERE Status = 'yes';
152
153 -- 2. List the employee names and their respective salaries in descending order of salary.
154 • SELECT Emp_name, Salary FROM library.Employee ORDER BY Salary DESC;
155
156 -- 3. Retrieve the book titles and the corresponding customers who have issued those books.
157 • SELECT Books.Book_title, Customer.Customer_name
158 FROM library.BOOKS
159 JOIN library.Issuestatus ON Books.ISBN = Issuestatus.Isbn_book
160 JOIN library.Customer ON Customer.Customer_id = Issuestatus.Issued_cust;
161
```

Below the editor, the "Result Grid" shows the output of the first query:

Book_title	Category	Rental_Price
To Kill a Mockingbird	Classic	10.99
Sapiens: A Brief History of Humankind	Non-fiction	12.99
Sense and Sensibility	Classic	8.99
The Catcher in the Rye	Fiction	8.99
The Girl with the Dragon Tattoo	Mystery	11.99

The "Output" section shows the execution of the first query, returning 6 rows. The "Action Output" section shows the execution of the second query, returning 10 rows.

2. List the employee names and their respective salaries in descending order of salary.

```
.9
0      -- 1.Retrieve the book title, category, and rental price of all available books.
1 •   SELECT Book_title, Category, Rental_Price FROM library.Books WHERE Status = 'yes';
2
3      -- 2. List the employee names and their respective salaries in descending order of salary.
4 •   SELECT Emp_name, Salary FROM library.Employee ORDER BY Salary DESC;
5
6      -- 3. Retrieve the book titles and the corresponding customers who have issued those books.
7 •   SELECT Books.Book_title, Customer.Customer_name
8      FROM library.Books
9      JOIN library.Issuestatus ON Books.ISBN = Issuestatus.Isbn_book
10     JOIN library.Customer ON Customer.Customer_id = Issuestatus.Issued_cust;
11
12     -- 4. Retrieve the book titles and the corresponding customers who have issued those books.
```

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

Emp_name	Salary
Jessica_Wilson	60000.00
John_Davis	50000.00
Christopher_Taylor	50000.00
Bob_Williams	45000.00
David_Martinez	45000.00

Employee 8 x

Output

Action Output

#	Time	Action	Message
31	16:28:04	SELECT Book_title, Category, Rental_Price FROM library.Books WHERE Status = 'yes' L...	10 row(s) returned
32	16:28:50	SELECT Emp_name, Salary FROM library.Employee ORDER BY Salary DESC LIMIT 0, 1...	10 row(s) returned

3. Retrieve the book titles and the corresponding customers who have issued those books

The screenshot shows a SQL IDE interface. The top toolbar includes icons for file operations, execution, and navigation, along with a 'Limit to 1000 rows' dropdown. The SQL editor contains the following code:

```
149
150 -- Execute the statement under the keyboard cursor and rental price of all available books.
151 • SELECT Book_title, Category, Rental_Price FROM library.Books WHERE Status = 'yes';
152
153 -- 2. List the employee names and their respective salaries in descending order of salary.
154 • SELECT Emp_name, Salary FROM library.Employee ORDER BY Salary DESC;
155
156 -- 3. Retrieve the book titles and the corresponding customers who have issued those books.
157 • SELECT Books.Book_title, Customer.Customer_name
158 FROM library.BOOKS
159 JOIN library.Issuestatus ON Books.ISBN = Issuestatus.Isbn_book
160 JOIN library.Customer ON Customer.Customer_id = Issuestatus.Issued_cust;
161
```

Below the editor is the 'Result Grid' section, which displays the results of the third query. It includes a 'Filter Rows' input, an 'Export' button, and a 'Wrap Cell Content' checkbox. The results are shown in a table:

Book_title	Customer_name
To Kill a Mockingbird	Sheldon Cooper
Sapiens: A Brief History of Humankind	Barney Stinson
Sense and Sensibility	Phoebe Buffay
The Catcher in the Rye	Phyllis Vance
The Girl with the Dragon Tattoo	Ron Swanson

Below the result grid is the 'Output' section, which shows the execution log. It includes a dropdown for 'Action Output' and a table of execution details:

#	Time	Action	Message
32	16:28:50	SELECT Emp_name, Salary FROM library.Employee ORDER BY Salary DESC LIMIT 0, 1...	10 row(s) returned
33	16:30:03	SELECT Books.Book_title, Customer.Customer_name FROM library.BOOKS JOIN library.I...	10 row(s) returned

4. Display the total count of books in each category.

```
2  -- 4.Display the total count of books in each category.
3  • SELECT Category, COUNT(*) AS Total_Count FROM library.Books GROUP BY Category;
4
5  -- 5. Retrieve the employee names and their positions for the employees whose salaries are above
6  • SELECT Emp_name, Position FROM library.Employee WHERE Salary > 50000;
7
```

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

Category	Total_Count
Classic	3
Non-fiction	1
Fiction	3
Mystery	2
Poetry	1

ult 10 x




put

Action Output

#	Time	Action	Message
33	16:30:03	SELECT Books.Book_title, Customer.Customer_name FROM library.BOOKS JOIN library.I...	10 row(s) returned

5. Retrieve the employee names and their positions for the employees whose salaries are above Rs.50,000.

```
165 -- 5. Retrieve the employee names and their positions for the employees whose salaries are above Rs.50,0
166 • SELECT Emp_name, Position FROM library.Employee WHERE Salary > 50000;
167
168 -- 6. List the customer names who registered before 2022-01-01 and have not issued any books yet.
169 • SELECT Customer_name FROM library.Customer WHERE Reg_date < '2022-01-01' AND Customer_Id NOT IN
170 (SELECT Issued_cust FROM library.IssueStatus);
171
172 -- 7. Display the branch numbers and the total count of employees in each branch.
173 • SELECT Branch_no, COUNT(*) AS Total_Employees
174 FROM library.Employee GROUP BY Branch_no;
```





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

Emp_name	Position
Jessica_Wilson	Library_Director

Employee 11 x

6. List the customer names who registered before 2022-01-01 and have not issued any books yet.

```
168 -- 6. List the customer names who registered before 2022-01-01 and have not issued any books yet.
169 • SELECT Customer_name FROM library.Customer WHERE Reg_date < '2022-01-01' AND Customer_Id NOT IN
170 (SELECT Issued_cust FROM library.IssueStatus);
171
172 -- 7. Display the branch numbers and the total count of employees in each branch.
173 • SELECT Branch_no, COUNT(*) AS Total_Employees
174 FROM library.Employee GROUP BY Branch_no;
---
```

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

Customer_name

7. Display the branch numbers and the total count of employees in each branch.

```
172 -- 7. Display the branch numbers and the total count of employees in each branch.
173 • SELECT Branch_no, COUNT(*) AS Total_Employees
174 FROM library.Employee GROUP BY Branch_no;
175
176 -- 8.Display the names of customers who have issued books in the month of June 2023.
177 • SELECT DISTINCT Customer_name
178 FROM library.Customer WHERE Customer_Id IN (
179 SELECT Issued_cust
180 FROM library.Issued_Books)
```

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

	Branch_no	Total_Employees
1	1	3
2	2	4
3	3	3

Result 13 x

Output

Action Output

#	Time	Action	Message
36	16:32:30	SELECT Customer_name FROM library.Customer WHERE Reg_date < '2022-01-01' AND...	0 row(s) returned

8. Display the names of customers who have issued books in the month of June 2023.

```
176 -- 8.Display the names of customers who have issued books in the month of June 2023.
177 • SELECT DISTINCT Customer_name
178 FROM library.Customer WHERE Customer_Id IN (
179     SELECT Issued_cust
180     FROM library.IssueStatus
181     WHERE MONTH(Issue_date) = 6 AND YEAR(Issue_date) = 2023
182 );
183
184 -- 9. Retrieve book_title from book table containing history.
185 • SELECT Book_title
186 FROM library.Book
```

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

Customer_name

Customer 14 x




Output

Action Output

#	Time	Action	Message
✓ 37	16:33:28	SELECT Branch_no, COUNT(*) AS Total_Employees FROM library.Employee GROUP BY...	3 row(s) returned
38	16:34:13	SELECT DISTINCT Customer_name FROM library.Customer WHERE Customer_Id IN (...)	0 row(s) returned

9. Retrieve book_title from book table containing history.

```
183
184 -- 9. Retrieve book_title from book table containing history.
185 ❌ SELECT Book_title from where category="History";
186 • select book_title from books
187 WHERE book_title LIKE '%history%' or book_title like "Histoty";
188
```

Result Grid   Filter Rows: Export:  Wrap Cell Content: 

	book_title
▶	Sapiens: A Brief History of Humankind

10.Retrieve the branch numbers along with the count of employees for branches having more than 5 employees


```
189 -- 10.Retrieve the branch numbers along with the count of employees for branches having more than 5 employees.  
190 • SELECT Branch_no, COUNT(Emp_ID) AS Employee_Count  
191 FROM library.Employee  
192 GROUP BY Branch_no  
193 HAVING COUNT(Emp_ID)>5;
```

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

	Branch_no	Employee_Count
--	-----------	----------------

11.Retrieve the names of customers who have issued books authored by George Orwell.





```
194
195  -- 11.Retrieve the names of customers who have issued books authored by George Orwell.
196
197 •  SELECT DISTINCT c.Customer_name
198  FROM library.Customer c
199  JOIN library.IssueStatus i ON c.Customer_Id = i.Issued_cust
200  JOIN library.Books b ON i.Isbn_book = b.ISBN
201  WHERE b.Author = 'George Orwell';
```

Result Grid   Filter Rows: Export:  Wrap Cell Content: 

	Customer_name
▶	Leslie Knope

12.Retrieve the names of customers who have issued books published by Penguin Classics





```
197 • SELECT DISTINCT c.Customer_name
198 FROM library.Customer c
199 JOIN library.IssueStatus i ON c.Customer_Id = i.Issued_cust
200 JOIN library.Books b ON i.Isbn_book = b.ISBN
201 WHERE b.Author = 'George Orwell';
202
203 -- 12.Retrieve the names of customers who have issued books published by Penguin Classics
204
205 • SELECT DISTINCT c.Customer_name
206 FROM library.Customer c
```

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

	Customer_name
▶	Leslie Knope

13.Display the total count of books issued in each month of 2024




```
L1 -- 13.Display the total count of books issued in each month of 2024
L2
L3 • SELECT MONTH(Issue_date) AS Issue_Month, COUNT(*) AS Total_Books_Issued
L4 FROM library.IssueStatus
L5 WHERE YEAR(Issue_date) = 2024
L6 GROUP BY MONTH(Issue_date);
L7
```

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

Issue_Month	Total_Books_Issued
2	10

14.Retrieve the names of customers who have issued books authored by Jane Austen and have not returned them yet





```
218 -- 14.Retrieve the names of customers who have issued books authored by Jane Austen and have not returned them yet
219
220 • SELECT DISTINCT c.Customer_name
221 FROM library.Customer c
222 JOIN library.IssueStatus i ON c.Customer_Id = i.Issued_cust
223 JOIN library.Books b ON i.Isbn_book = b.ISBN
224 WHERE b.Author = 'Jane Austen'
225 AND c.Customer_Id NOT IN (SELECT Return_cust FROM library.ReturnStatus);
226
```

Result Grid   Filter Rows: Export:  Wrap Cell Content: 

Customer_name

15.Retrieve the names of employees who are not managers.

```
228 -- 15.Retrieve the names of employees who are not managers.
229
230 • SELECT Emp_name
231 FROM library.Employee
232 WHERE Emp_Id NOT IN (SELECT Manager_Id FROM library.Branch);
233
234 -- 16.Display the total count of books issued by each customer.
235
236 • SELECT c.Customer_name, COUNT(i.Issue_Id) AS Total_Books_Issued
237 FROM library.Customer c
238 JOIN library.Issue i ON c.Customer_Id = i.Customer_Id
239 GROUP BY c.Customer_name;
```


Result Grid   Filter Rows: Export:  Wrap Cell Content: 

	Emp_name
▶	John_Davis
	Willow_Smith
	Alice_Johnson
	Bob_Williams
	Emily_Brown

Employee 38 x

16.Display the total count of books issued by each customer.

```
234 -- 16.Display the total count of books issued by each customer.
235
236 • SELECT c.Customer_name, COUNT(i.Issue_Id) AS Total_Books_Issued
237 FROM library.Customer c
238 LEFT JOIN library.IssueStatus i ON c.Customer_Id = i.Issued_cust
239 GROUP BY c.Customer_name;
240
241 -- 17. List the authors along with the total number of books they have written in the Mystery category.
242
```

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

	Customer_name	Total_Books_Issued
▶	Homer Simpson	1
	Phoebe Buffay	1
	Barney Stinson	1
	Chandler Bing	1
	Michael Scott	1

Result 39 x

17. List the authors along with the total number of books they have written in the Mystery category.

```
241  -- 17. List the authors along with the total number of books they have written in the Mystery category.
242
243  • SELECT Author, COUNT(*) AS Total_Books
244  FROM library.Books
245  WHERE Category = 'Mystery'
246  GROUP BY Author;
247
248  -- 18. Identify the branches where the average salary of employees is below Rs. 45,000.
249
250  • SELECT Branch, AVG(Salary) AS Avg_Salary
251  FROM library.Employees
252  WHERE Avg_Salary < 45000
253  GROUP BY Branch;
```

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

	Author	Total_Books
▶	Stieg Larsson	1
	Dan Brown	1

18. Identify the branches where the average salary of employees is below Rs. 45,000.

```
248 -- 18. Identify the branches where the average salary of employees is below Rs. 45,000.
249
250 • select branch_no, avg(salary) as avg_sal from employee group by branch_no having avg(salary) < 45000;
251
252 -- 19. #Retrieve the details of customers who have issued books separately in the years 2021, 2022, 2023, and 2024.
253 • create view issue_details as
```

Result Grid   Filter Rows: Export:  Wrap Cell Content: 




	branch_no	avg_sal
▶	2	41250.000000
	3	40000.000000

-- 19. #Retrieve the details of customers who have issued books separately in the years 2021, 2022, 2023, and 2024.

```
-- 19. #Retrieve the details of customers who have issued books separat
• create view issue_details as
  select c.customer_name,i.issued_cust,i.issue_date,i.isbn_book
  from customer c join issuestatus i
  on c.customer_id = i.issued_cust;

• select * from issue_details;

delimiter $
• create procedure issued_year(issue_year int)
begin
    select customer_name,issue_date,isbn_book from issue_details
    where year(issue_date) = issue_year
```

ult Grid |  Filter Rows: | Export:  | Wrap Cell Content: 

customer_name	issue_date	isbn_book
Barney Stinson	2024-02-03	9780062316110
Chandler Bing	2024-02-04	9788172235652
Homer Simpson	2024-02-01	9780679601085
Leslie Knope	2024-02-06	9780307947055
Michael Scott	2024-02-05	9788170289186

lt 41 x

20. display name and position of employee who have the highest salary.

```
271
272  -- 20. #display name and position of employee who have the highest salary.
273 • select emp_name, position,salary from employee where salary= (select max(salary) from employee);
274
```

Result Grid   Filter Rows: Export:  Wrap Cell Content: 

	emp_name	position	salary
▶	John_Davis	Librarian	50000.00
	Willow_Smith	Assistant_Librarian	40000.00
	Alice_Johnson	Cataloger	35000.00
	Bob_Williams	Reference_Librarian	45000.00
	Emily_Brown	Library_Assistant	30000.00

employee 42 x

Suchithra.p
D18