

## LIBRARY MANAGEMENT SYSTEM

Topic : Library Management System

You are going to build a project based on Library Management System. It keeps track of all information about books in the library, their cost, status and total number of books available in the library.

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

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

Attributes for the tables:

1. Branch
  - Branch\_no
  - Set as PRIMARY KEY
    - Manager\_Id
    - Branch\_address
    - Contact\_no

The screenshot shows a database management tool interface. The top pane displays SQL queries for creating a table and inserting data. The bottom pane shows the 'Result Grid' with 10 rows of data for the 'branch' table.

**SQL Queries:**

```
18 (5,105,'trissure','9987876543'),
19 (6,106,'goa','7878909009'),
20 (7,107,'hydrabad','6778899009'),
21 (8,108,'shirur','7665545456'),
22 (9,109,'palakad','8989877667'),
23 (10,110,'mysore','9067566778');
```

**Result Grid:**

branch_no	manager_id	branch_address	contact_no
1	101	kottayam	9876543210
2	102	malappuram	9786543211
3	103	kannur	9678543212
4	104	kozhikod	9778765434
5	105	trissure	9987876543
6	106	goa	7878909009
7	107	hydrabad	6778899009
8	108	shirur	7665545456
9	109	palakad	8989877667
10	110	mysore	9067566778

**Output Log:**

#	Time	Action	Message	Duration
15	03:49:45	alter table branch modify column contact_no varchar(15)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0	0.282 s
16	03:49:54	insert into branch (branch_no,manager_id,branch_address,contact_no) ...	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.015 s
17	03:50:00	desc branch	4 row(s) returned	0.000 s
18	03:54:32	select*from branch LIMIT 0, 1000	10 row(s) returned	0.000 s

## 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 shows a SQL IDE interface with a query editor and a result grid. The query editor contains the following SQL code:

```
43      (26,'jebi','office staff',30000,7),
44      (27,'fathima','hr',58000,8),
45      (28,'noora','manager',68000,9),
46      (29,'jaleel','salesman',70000,10);
47 •   select * from employee;
48
```

The result grid displays the following data:

	emp_id	emp_name	position	salary	branch_no
▶	20	alice	manager	68000.00	1
	21	joby	salesman	30000.00	2
	22	rahul	hr	55000.00	3
	23	babu	staff	34000.00	4
	24	daisy	manager	67000.00	5
	25	chinju	hr	38000.00	6
	26	jebi	office staff	30000.00	7
	27	fathima	hr	58000.00	8
	28	noora	manager	68000.00	9
	29	jaleel	salesman	70000.00	10

The interface also shows a toolbar with various icons and a status bar at the bottom.

## 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

STORE PROCEDURE\*    project    project 1\*    SQL File 5\* x

Limit to 1000 rows

```

67      (26,'the great gatsby','fiction',100,'yes','scot fitsgiralld','charles scribners'),
68      (27,'the road','fiction',90,'no','cormac mccarthy','alfred k knolf'),
69      (28,'the alchemist','fantasy',110,'yes','paulo coelho','harperone'),
70      (29,'educated','memoir',60,'no','tara westover','random house');
71 •   select * from books;
72

```

Result Grid    Filter Rows:    Edit:    Export/Import:    Wrap Cell Content: [IA](#)

	isbn	book_title	category	rental_price	status	author	publisher
▶	20	history of rome	history	100.00	yes	livy	penguin
	21	advanced mathematics	maths	150.00	yes	isaac newton	cambridge
	22	hery potter	fantasy	120.00	no	j.k.rowling	bloomsberry
	23	charlottes web	children fiction	200.00	no	e.b.white	harper and brothers
	24	to kill a mockingbird	fiction	180.00	no	harper lee	j.b.lippincot and co
	25	becoming	fiction	50.00	yes	michelle obama	crown
	26	the great gatsby	fiction	100.00	yes	scot fitsgiralld	charles scribners
	27	the road	fiction	90.00	no	cormac mccarthy	alfred k knolf
	28	the alchemist	fantasy	110.00	yes	paulo coelho	harperone
	29	educated	memoir	60.00	no	tara westover	random house

books 8 x

Output

Action Output

#### 4. Customer

- Customer\_Id

- Set as PRIMARY KEY
  - Customer\_name
  - Customer\_address
  - Reg\_date

The screenshot shows a SQL IDE interface. At the top, there are tabs for 'STORE PROCEDURE\*', 'project', 'project 1\*', and 'SQL File 5\*'. Below the tabs is a toolbar with various icons, including a 'Limit to 1000 rows' dropdown. The main area displays a SQL query with line numbers 87 to 92. The query inserts four records into a table and then selects all records from the 'customers' table. Below the query, there is a 'Result Grid' section with a toolbar for filtering, editing, and exporting. The result grid shows a table with five columns: 'customer\_id', 'customer\_name', 'customer\_address', and 'reg\_date'. The table contains 13 rows of data, with the last row highlighted. Below the result grid, there is a tab labeled 'customers 13' and an 'Output' section.

```

87  (937,'chinju','uttarpradesh','2018-03-23'),
88  (938,'aeiza','shrilankha','2024-12-03'),
89  (939,'ayshu','mannali','2020-04-14'),
90  (940,'nusaiba','uttarpradesh','2021-10-23');
91  • select * from customers;
92

```

	customer_id	customer_name	customer_address	reg_date
▶	931	laila	jammu	2023-09-04
	932	mery	palakad	2024-01-02
	933	hyra	kannur	2019-03-22
	934	raseena	wayanad	2020-09-23
	935	habeeb	kashmir	2021-12-23
	936	jojo	tamilnadu	2019-02-15
	937	chinju	uttarpradesh	2018-03-23
	938	aeiza	shrilankha	2024-12-03
	939	ayshu	mannali	2020-04-14
	940	nusaiba	uttarpradesh	2021-10-23

customers 13 x

Output

## 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 list of SQL insert statements for a table named 'Issued\_book\_name' is shown, with rows numbered 124 to 130. Below this, a 'Result Grid' shows a table with 5 columns: Issue\_Id, Issued\_cust, Issued\_book\_name, Issue\_date, and Isbn\_book. The table contains 10 rows of data. To the right of the grid are buttons for 'Apply', 'Revert', and 'Context'. Below the grid, an 'Action Output' log shows two entries: an INSERT statement affecting 10 rows and a SELECT statement returning 10 rows.

Issue_Id	Issued_cust	Issued_book_name	Issue_date	Isbn_book
300	931	pathumayude aad	2023-02-12	20
301	932	aotubiograpghy	2019-12-23	21
302	933	baalasakhi	2020-11-23	22
303	934	tom and jerry	2022-03-15	23
304	935	train to pakisthan	2020-03-12	24
305	936	circe	2024-01-14	25
306	937	normal people	2022-12-23	26
307	938	the night circus	2022-09-18	27
308	939	herrypotter	2023-07-05	28
309	940	becoming	2021-06-17	29

#	Time	Action	Message
46	06:22:45	INSERT INTO IssueStatus (Issue_Id, Issued_cust, Issued_book_name, Issue_date, Isbn_book...	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0
47	06:22:51	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



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

The screenshot shows a database management interface with a SQL editor at the top and a results pane below. The SQL editor contains two queries: one to retrieve book details and another to list employees by salary. The results pane displays the output of the first query as a table of book titles, categories, and rental prices. Below the table, the 'Action Output' section shows the execution status and row counts for both queries.

```
154
155 -- retrieve the book title,category,and rental price of all available book
156 • select book_title,category,rental_price from books;
157
158 -- list the employee names and their respective salaries in descending order of salary
159 • select emp_name,salary from employee order by salary desc;
160
```

book_title	category	rental_price
history of rome	history	100.00
advanced mathematics	maths	150.00
hery potter	fantasy	120.00
charlottes web	children fiction	200.00
to kill a mockingbird	fiction	180.00
becoming	fiction	50.00
the great gatsby	fiction	100.00
the road	fiction	90.00
the alchemist	fantasy	110.00
educated	memoir	60.00

books 19 x Read Only

Output

Action Output

#	Time	Action	Message
50	06:24:27	select * from return_status LIMIT 0, 1000	10 row(s) returned
51	06:52:16	select book_title,category,rental_price from books LIMIT 0, 1000	15 row(s) returned

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

The screenshot shows a SQL IDE interface with a query editor at the top and a results pane at the bottom. The query editor contains two SQL statements: a comment about retrieving book information and a SELECT statement for employee salaries. The results pane shows a table with employee names and salaries, and an action log at the bottom.

```
154
155 -- retrieve the book title,category,and rentel price of all available book
156 • select book_title,category,rentel_price from books;
157
158 -- list the employee names and their respective salaries in descending order of salary
159 • select emp_name,salary from employee order by salary desc;
160
161
```

emp_name	salary
jaleel	70000.00
alice	68000.00
noora	68000.00
daisy	67000.00
fathima	58000.00
rahul	55000.00
chinju	38000.00
babu	34000.00
joby	30000.00
jebi	30000.00

#	Time	Action	Message
✓ 51	06:52:16	select book_title,category,rentel_price from books LIMIT 0, 1000	15 row(s) returned
✓ 52	06:55:41	select emp_name,salary from employee order by salary desc LIMIT 0, 1000	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 with a query editor and a results grid. The query editor contains a SQL query to retrieve book titles and customer names from the 'issuestatus' table, joined with 'books' and 'customer' tables. The results grid displays 10 rows of data.

```
166
167
168 -- 3. Retrieve the book titles and the corresponding customers who have issued those books.
169 • SELECT books.Book_title, issuestatus.Issued_cust, customer.Customer_name
170 FROM issuestatus
171 INNER JOIN books ON issuestatus.Isbn_book = books.ISBN
172 INNER JOIN customer ON issuestatus.Issued_cust = customer.Customer_Id; -- Assuming Issued_cust matches Customer_Id
173
```

Result Grid

	Book_title	Issued_cust	Customer_name
▶	history of rome	931	laila
	advanced mathematics	932	mery
	hery potter	933	hyra
	charlottes web	934	raseena
	to kill a mockingbird	935	habeeb
	becoming	936	jojo
	the great gatsby	937	chinju
	the road	938	aeiza
	the alchemist	939	ayshu
	educated	940	nusaiba

Result 21 x Read Only

Output

Action Output

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

The screenshot shows a SQL IDE interface with a toolbar at the top containing icons for file operations, execution, and settings. The main editor area displays a SQL query with line numbers 174 through 181. The query is a SQL statement to select the category and the total count of books from a table named 'books', grouped by category. Below the editor, there is a 'Result Grid' section with a table showing the results of the query. The table has two columns: 'category' and 'total\_books'. The results are as follows:

category	total_books
history	2
maths	2
fantasy	3
children fiction	2
fiction	5
memoir	1

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

The screenshot shows a database IDE with a SQL editor and a results pane. The SQL editor contains the following queries:

```
177
178 -- Display the total count of books in each category.
179 • select category,count(*) as total_books from books group by category;
180
181 -- Retrieve the employee names and their positions for the employees whose salaries are above Rs.50,000
182 • select emp_name,position,salary from employee where salary>50000;
183
184 -- List the customer names who registered before 2022-01-01 and have not found any books yet
```

The results pane shows the output of the second query, displaying a table with 6 rows and 3 columns: emp\_name, position, and salary.

emp_name	position	salary
alice	manager	68000.00
rahul	hr	55000.00
daisy	manager	67000.00
fathima	hr	58000.00
noora	manager	68000.00
jaleel	salesman	70000.00

The output pane shows the execution of the queries:

```
employee 23 x
Output :
Action Output
# Time Action Message
61 07:12:51 select category,count(*) as total_books from books group by category LIMIT 0, 1000 6 row(s) returned
62 07:14:40 select emp_name,position,salary from employee where salary>50000 LIMIT 0, 1000 6 row(s) returned
```

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

The screenshot shows a SQL IDE interface with a query editor and a results pane. The query editor contains the following SQL code:

```
190
191 -- 6. List the customer names who registered before 2023-07-01 and have not issued any books yet
192 • SELECT customer.Customer_name, customer.Reg_date
193 FROM customer
194 LEFT JOIN issuestatus ON customer.Customer_Id = issuestatus.Issued_cust
195 WHERE customer.Reg_date < '2023-07-01'
196 AND issuestatus.issue_id IS NULL;
```

The results pane displays a table with two columns: Customer\_name and Reg\_date. The table contains 19 rows of data. The results are as follows:

Customer_name	Reg_date
sabu	2021-05-10
fathima	2019-03-22
rosy	2020-09-23
sabu	2021-05-10
fathima	2019-03-22
rosy	2020-09-23
meri	2021-12-23
jojo	2019-02-15
chinju	2018-03-23
ayshu	2020-04-14

The results pane also shows a message indicating that 19 row(s) were returned.

However, books has been issued to all the customers.

STORE PROCEDURE\* project project 1\* project\*

Limit to 1000 rows

```

185 • select customers.customer_name,customers.reg_date
186 from customers
187 left join issuestatus on customers.customer_id = issuestatus.issued_cust
188 where customers.reg_date < '2019-12-12'
189 and issuestatus.issue_id is null;
190
191 -- 6. List the customer names who registered before 2023-07-01 and have not issued any books yet
192 • SELECT customers.Customer_name,customers.Reg_date

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [IA](#)

customer_name	reg_date
---------------	----------

Result 31 x

Output

Action Output

#	Time	Action	Message
✓ 70	07:23:24	SELECT customer.Customer_name,customer.Reg_date FROM customer LEFT JOIN issuestatu...	26 row(s) returned
✓ 71	07:23:54	select customers.customer_name,customers.reg_date from customers left join issuestatus on cu...	0 row(s) returned

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

STORE PROCEDURE\* project project 1\* project x

Limit to 1000 rows

```
196 AND issuestatus.issue_id IS NULL;
197
198 -- Display the branch numbers and the total count of employees in each branch.
199 • select branch_no,count(*) as total_employees
200 from employee
201 group by branch_no;
202
203
```

Result Grid

	branch_no	total_employees
▶	1	1
	2	1
	3	1
	4	1
	5	1
	6	1
	7	1
	8	1
	9	1
	10	1

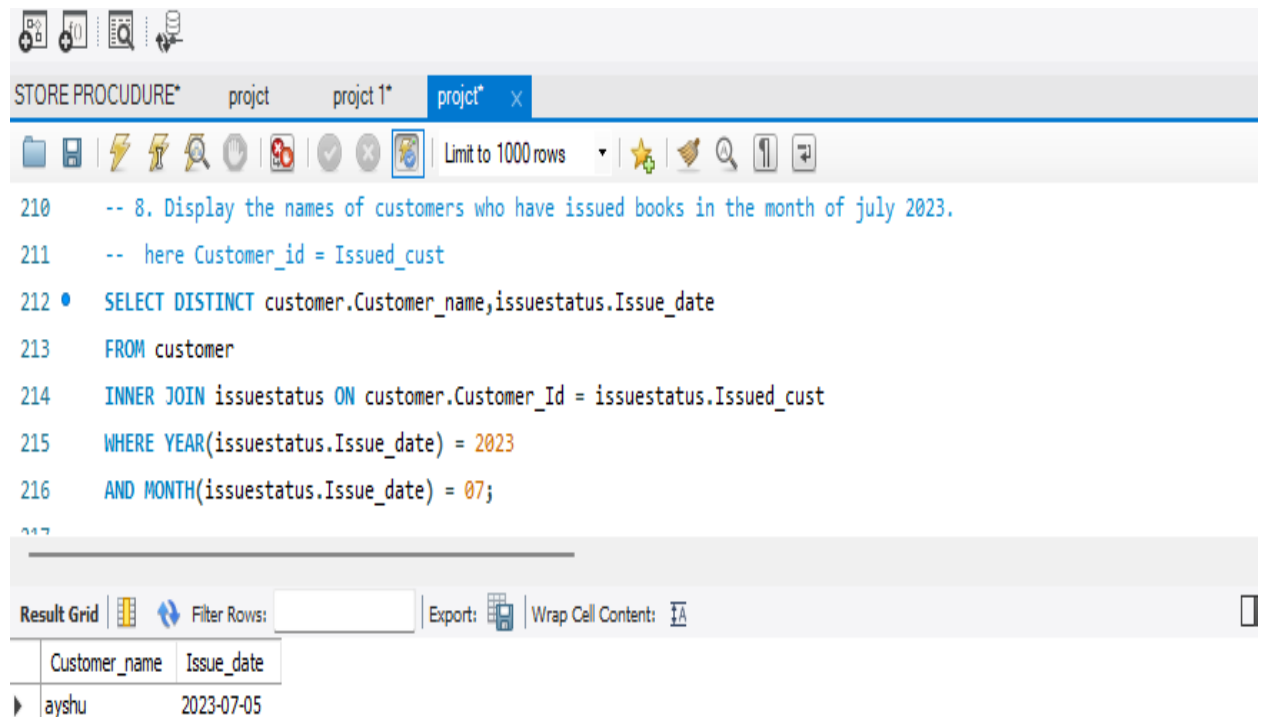
Result 32 x

Output

Action Output

#	Time	Action	Message
✓ 71	07:23:54	select customers.customer_name,customers.reg_date from customers left join issuestatus on cu...	0 row(s) returned
✓ 72	07:25:25	select branch_no,count(*) as total_employees from employee group by branch_no LIMIT 0, 1000	10 row(s) returned

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

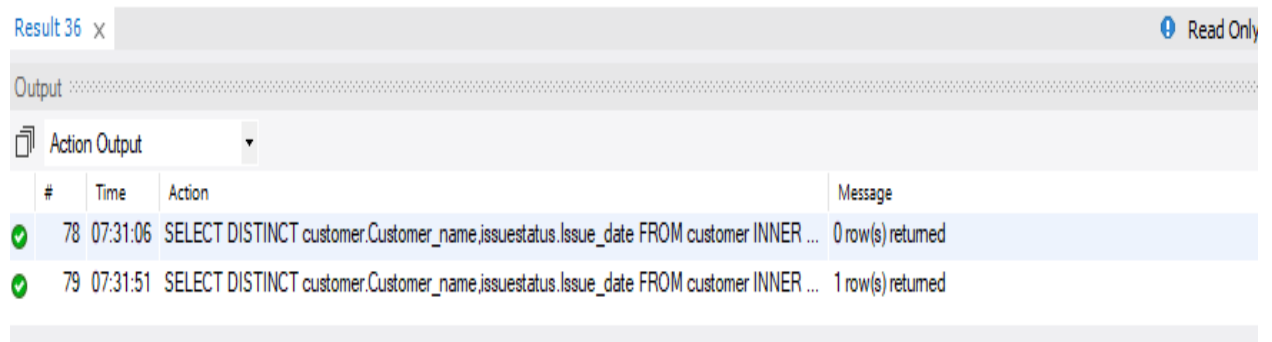


The screenshot shows a SQL IDE interface with a query editor and a results pane. The query editor contains the following SQL code:

```
210 -- 8. Display the names of customers who have issued books in the month of july 2023.
211 -- here Customer_id = Issued_cust
212 • SELECT DISTINCT customer.Customer_name,issuestatus.Issue_date
213 FROM customer
214 INNER JOIN issuestatus ON customer.Customer_Id = issuestatus.Issued_cust
215 WHERE YEAR(issuestatus.Issue_date) = 2023
216 AND MONTH(issuestatus.Issue_date) = 07;
```

The results pane shows a table with two columns: Customer\_name and Issue\_date. The table contains one row with the values 'ayshu' and '2023-07-05'.

Customer_name	Issue_date
ayshu	2023-07-05



The screenshot shows the execution log of the SQL query. The log contains two entries, both marked with a green checkmark, indicating successful execution.

#	Time	Action	Message
78	07:31:06	SELECT DISTINCT customer.Customer_name,issuestatus.Issue_date FROM customer INNER ...	0 row(s) returned
79	07:31:51	SELECT DISTINCT customer.Customer_name,issuestatus.Issue_date FROM customer INNER ...	1 row(s) returned

9. Retrieve book\_title from book table containing fiction.

The screenshot shows a database IDE interface with a SQL editor and a results pane. The SQL editor contains the following code:

```
214 INNER JOIN issuestatus ON customer.Customer_Id = issuestatus.Issued_cust
215 WHERE YEAR(issuestatus.Issue_date) = 2023
216 AND MONTH(issuestatus.Issue_date) = 07;
217
218
219 -- retrieve book_title from book table containing fiction.
220 • select book_title,category from books where category = 'fiction';
221
```

Below the editor, the "Result Grid" shows the results of the query:

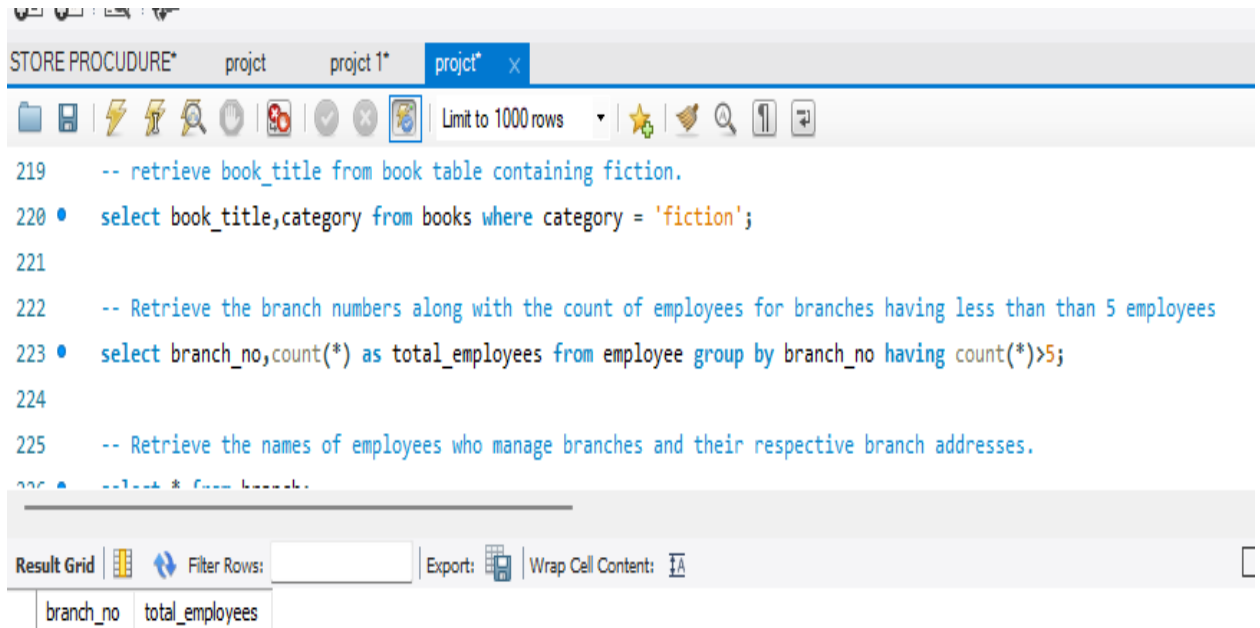
book_title	category
to kill a mockingbird	fiction
becoming	fiction
the great gatsby	fiction
the road	fiction
to kill a mockingbird	fiction

At the bottom, the "Output" pane shows the "Action Output" for the query:

#	Time	Action	Message
82	07:37:42	select book_title,category from books where book_title = 'history of rome' LIMIT 0, 1000	2 row(s) returned
83	07:40:03	select book_title,category from books where category = 'fiction' LIMIT 0, 1000	5 row(s) returned



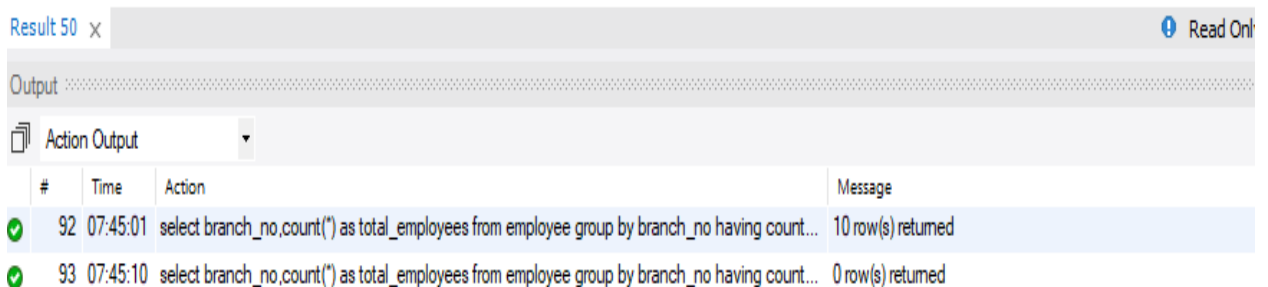
10. Retrieve the branch numbers along with the count of employees for branches having more than 5 employees.



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

```
219 -- retrieve book_title from book table containing fiction.
220 • select book_title,category from books where category = 'fiction';
221
222 -- Retrieve the branch numbers along with the count of employees for branches having less than than 5 employees
223 • select branch_no,count(*) as total_employees from employee group by branch_no having count(*)>5;
224
225 -- Retrieve the names of employees who manage branches and their respective branch addresses.
226 • select * from branch;
```

Below the query editor is a 'Result Grid' section. It includes a 'Filter Rows' input field, an 'Export' button, and a 'Wrap Cell Content' checkbox. The result grid displays two columns: 'branch\_no' and 'total\_employees'.



The screenshot shows the 'Action Output' window in the SQL IDE. It displays a table with execution results:

#	Time	Action	Message
✓ 92	07:45:01	select branch_no,count(*) as total_employees from employee group by branch_no having count...	10 row(s) returned
✓ 93	07:45:10	select branch_no,count(*) as total_employees from employee group by branch_no having count...	0 row(s) returned

11. Retrieve the names of employees who manage branches and their respective branch addresses.

The screenshot shows a SQL IDE interface with a query editor and a results pane. The query editor contains the following SQL code:

```
234 • SELECT*FROM employee;  
235 -- Here Manager_Id= Emp_Id  
236 • SELECT employee.Emp_name AS Manager_Name, branch.Branch_Address  
237 FROM employee  
238 INNER JOIN branch ON employee.Emp_Id = branch.Manager_Id;  
239  
240  
241 -- display the names of employees who have found back with a profit ratio higher than 95
```

Below the query editor, the 'Result Grid' tab is active, showing a table with two columns: 'Manager\_Name' and 'Branch\_Address'. The table is currently empty.

The 'Output' pane shows the execution log for 'Result 56':

#	Time	Action	Message
99	07:50:56	SELECT*FROM employee LIMIT 0, 1000	10 row(s) returned
100	07:51:05	SELECT employee.Emp_name AS Manager_Name, branch.Branch_Address FROM employee I...	0 row(s) returned

12. Display the names of customers who have issued books with a rental price higher than Rs. 25.

The screenshot shows a SQL IDE interface with a query editor at the top and a results pane at the bottom. The query editor contains a SQL query to select distinct customer names and their issued book names, filtered by a rental price greater than 25. The results pane displays a table with two columns: 'issuestatus' and 'issued\_book\_name'. The table contains 10 rows of data. The output pane at the bottom shows two messages: an error message (Error Code: 1054) and a success message (10 row(s) returned).

```
242 • select * from issue_status;
243 • select*from customers;
244 • select distinct customers.customer_name issuestatus,books.rentel_price issued_book_name
245   from customers
246  inner join issuestatus on customers.customer_id = issuestatus.issued_cust
247  inner join books on issuestatus.isbn_book = books.isbn
248  where books.rentel_price >25;
```

issuestatus	issued_book_name
laila	100.00
mery	150.00
hyra	120.00
raseena	200.00
habeeb	180.00
jojo	50.00
chinju	100.00
aeiza	90.00
ayshu	110.00
nusaiba	60.00

Result 57 x Read Only

Output

Action Output

#	Time	Action	Message
101	07:53:14	select distinct customers.customer_name issuestatus,books.rentel_priceissued_book_name fro...	Error Code: 1054. Unknown column 'books.rentel_priceissued_book_name' in field li
102	07:53:37	select distinct customers.customer_name issuestatus,books.rentel_price issued_book_name fro...	10 row(s) returned