

Prepared by- Shubham Mishra

Email – mshraea@amazon.com

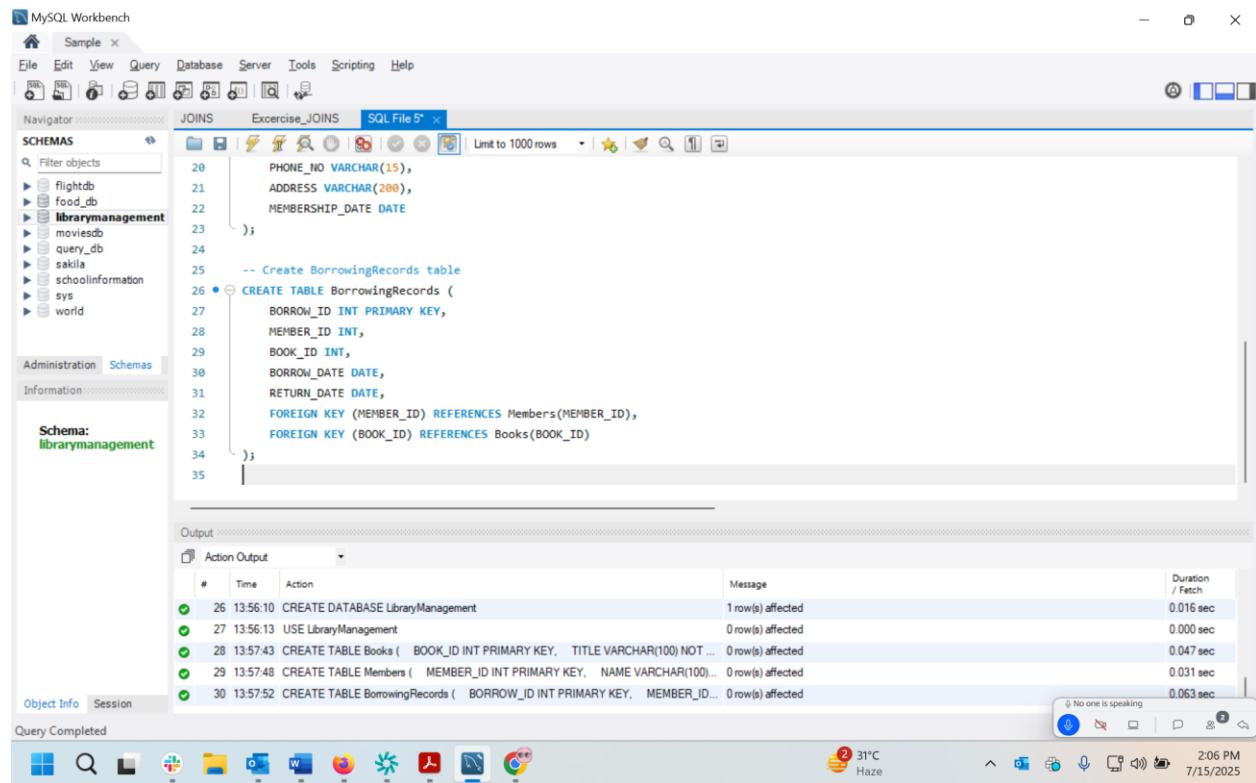
GitHub Account - <https://github.com/mshraea>

I have used SQL workbench for all 4 tasks.

Task 1

Project Title: Library Management System (using SQL)

Database Creation:



The screenshot shows the MySQL Workbench interface with the following details:

- File Menu:** Sample ×, File, Edit, View, Query, Database, Server, Tools, Scripting, Help.
- Navigator:** Schemas (librarymanagement selected), Administration, Information.
- SQL Editor:** JOINS tab, Exercise_JOINS, SQL File 5*, Limit to 1000 rows. The code creates the BorrowingRecords table:

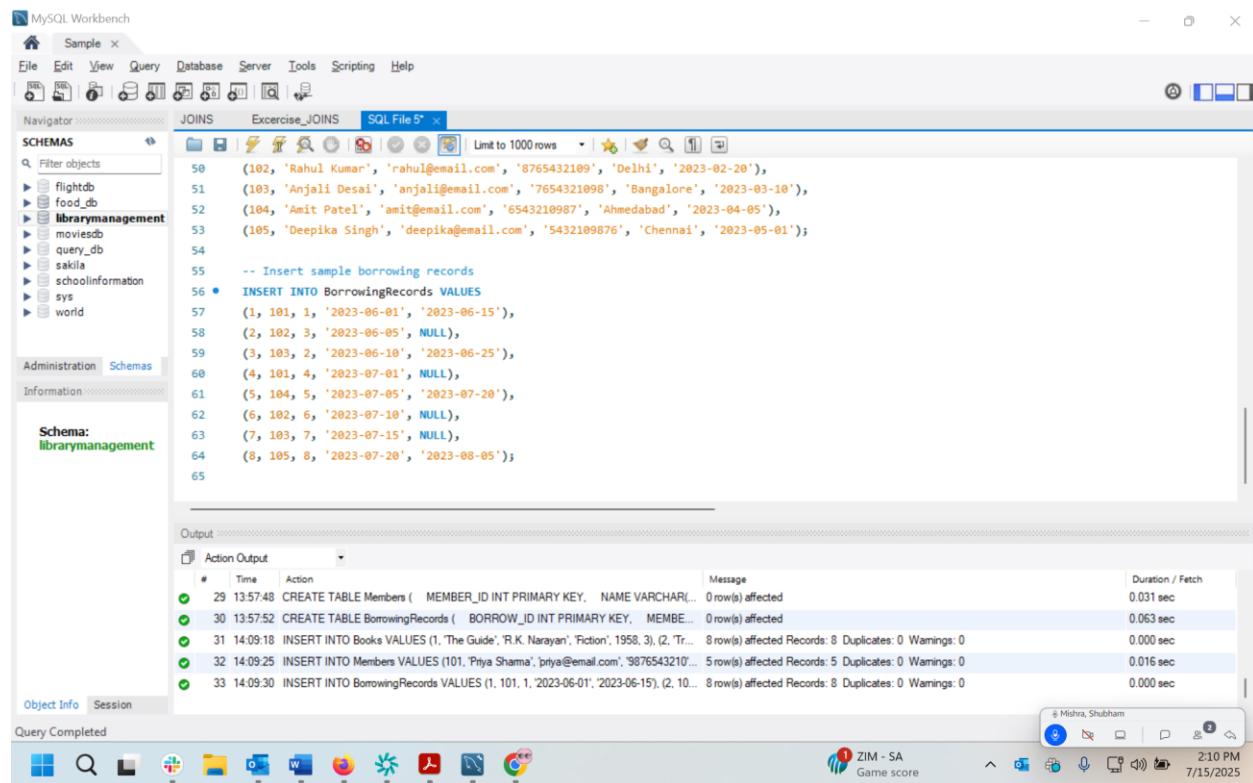
```
20     PHONE_NO VARCHAR(15),
21     ADDRESS VARCHAR(200),
22     MEMBERSHIP_DATE DATE
23 );
24
25 -- Create BorrowingRecords table
26 • CREATE TABLE BorrowingRecords (
27     BORROW_ID INT PRIMARY KEY,
28     MEMBER_ID INT,
29     BOOK_ID INT,
30     BORROW_DATE DATE,
31     RETURN_DATE DATE,
32     FOREIGN KEY (MEMBER_ID) REFERENCES Members(MEMBER_ID),
33     FOREIGN KEY (BOOK_ID) REFERENCES Books(BOOK_ID)
34 );
35 |
```

- Output:** Action Output table showing the execution of SQL statements:

#	Time	Action	Message	Duration / Fetch
26	13:56:10	CREATE DATABASE LibraryManagement	1 row(s) affected	0.016 sec
27	13:56:13	USE LibraryManagement	0 row(s) affected	0.000 sec
28	13:57:43	CREATE TABLE Books (BOOK_ID INT PRIMARY KEY, TITLE VARCHAR(100) NOT ...)	0 row(s) affected	0.047 sec
29	13:57:48	CREATE TABLE Members (MEMBER_ID INT PRIMARY KEY, NAME VARCHAR(100) NOT ...)	0 row(s) affected	0.031 sec
30	13:57:52	CREATE TABLE BorrowingRecords (BORROW_ID INT PRIMARY KEY, MEMBER_ID...)	0 row(s) affected	0.063 sec

- System Status:** 31°C Haze, 2:06 PM, 7/15/2025.

Data Creation:



The screenshot shows the MySQL Workbench interface with a SQL editor tab open. The code in the editor is as follows:

```
MySQL Workbench - Sample
File Edit View Query Database Server Tools Scripting Help
JOINS Excercise_JOINS SQL File 5* ×
schemas
SCHEMAS Filter objects
flightdb food_db librarymanagement moviesdb query_db sakila schoolinformation sys world
Administration Schemas
Information
Schema: librarymanagement
50    (102, 'Rahul Kumar', 'rahul@email.com', '8765432109', 'Delhi', '2023-02-20'),
51    (103, 'Anjali Desai', 'anjali@email.com', '7654321098', 'Bangalore', '2023-03-10'),
52    (104, 'Amit Patel', 'amit@email.com', '6543210987', 'Ahmedabad', '2023-04-05'),
53    (105, 'Deepika Singh', 'deepika@email.com', '5432109876', 'Chennai', '2023-05-01');

54
55    -- Insert sample borrowing records
56    INSERT INTO BorrowingRecords VALUES
57    (1, 101, 1, '2023-06-01', '2023-06-15'),
58    (2, 102, 3, '2023-06-05', NULL),
59    (3, 103, 2, '2023-06-10', '2023-06-25'),
60    (4, 101, 4, '2023-07-01', NULL),
61    (5, 104, 5, '2023-07-05', '2023-07-20'),
62    (6, 102, 6, '2023-07-10', NULL),
63    (7, 103, 7, '2023-07-15', NULL),
64    (8, 105, 8, '2023-07-20', '2023-08-05');
65

Output
Action Output
# Time Action Message Duration / Fetch
29 13:57:48 CREATE TABLE Members ( MEMBER_ID INT PRIMARY KEY, NAME VARCHAR(255) NOT NULL, ADDRESS TEXT, PHONE NUMBER(10,0), EMAIL VARCHAR(255) NOT NULL, DOB DATE, JOIN_DATE DATE, STATUS CHAR(1) ) ENGINE=InnoDB; 0 row(s) affected 0.031 sec
30 13:57:52 CREATE TABLE BorrowingRecords ( BORROW_ID INT PRIMARY KEY, MEMBER_ID INT, BOOK_ID INT, BORROW_DATE DATE, RETURN_DATE DATE, DUE_DATE DATE, FINE DECIMAL(5,2) ); 0 row(s) affected 0.063 sec
31 14:09:18 INSERT INTO Books VALUES (1, 'The Guide', 'R.K. Narayan', 'Fiction', 1958, 3, (2, True)); 8 row(s) affected Records: 8 Duplicates: 0 Warnings: 0 0.000 sec
32 14:09:25 INSERT INTO Members VALUES (101, 'Priya Sharma', 'priya@email.com', '9876543210', 'Mumbai', '2023-06-01', '2023-06-15'); 5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0 0.016 sec
33 14:09:30 INSERT INTO BorrowingRecords VALUES (1, 101, 1, '2023-06-01', '2023-06-15'); 2, 10... 8 row(s) affected Records: 8 Duplicates: 0 Warnings: 0 0.000 sec

Object Info Session
Query Completed
S Mishra, Shubham ZIM - SA Game score 2:10 PM 7/15/2025
```

Information Retrieval:

- a) Retrieve a list of books currently borrowed by a specific member.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 SQL File 5

Navigator

SCHEMAS

- flightdb
- food_db
- librarymanagement
 - Tables
 - Views
 - Stored Procedures
 - Functions
- moviesdb
- payroll_database
- query_db
- sakila
- schoolinformation
- sys
- world

65 -- Information Retrieval Queries
66 -- a) Books borrowed by specific member
67
68 SELECT b.TITLE, b.AUTHOR, br.BORROW_DATE, br.RETURN_DATE
69 FROM Books b
70 JOIN BorrowingRecords br ON b.BOOK_ID = br.BOOK_ID
71 WHERE br.MEMBER_ID = 101;

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

TITLE	AUTHOR	BORROW_DATE	RETURN_DATE
The Guide	R.K. Narayan	2023-06-01	2023-06-15
Wings of Fire	A.P.J. Abdul Kalam	2023-07-01	NULL

Administration Schemas Information

Schema: food_db

Action Output

#	Time	Action	Message	Duration / Fetch
14	15:26:22	SELECT DEPARTMENT, COUNT(*) AS EMPLOYEE_COUNT FROM employees ...	4 row(s) returned	0.000 sec / 0.000 sec
15	15:27:02	SELECT DEPARTMENT, AVG(SALARY) AS AVG_SALARY FROM employees G...	1 row(s) returned	0.000 sec / 0.000 sec
16	15:28:12	SELECT e1.NAME, e1.SALARY FROM employees e1 INNER JOIN employees e2 ON...	0 row(s) returned	0.016 sec / 0.000 sec
17	15:28:38	SELECT e1.NAME, e1.SALARY FROM employees e1 INNER JOIN employees e2 ON...	0 row(s) returned	0.000 sec / 0.000 sec
18	15:30:24	SELECT b.TITLE, b.AUTHOR, br.BORROW_DATE, br.RETURN_DATE FROM Books ...	Error Code: 1146. Table 'payroll_database.books' doesn't exist	0.000 sec
19	15:30:32	USE LibraryManagement	0 row(s) affected	0.000 sec
20	15:30:41	SELECT b.TITLE, b.AUTHOR, br.BORROW_DATE, br.RETURN_DATE FROM Books ...	2 row(s) returned	0.016 sec / 0.000 sec

Object Info Session

b) Find members who have overdue books (borrowed more than 30 days ago, not returned)

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 Task 2

Navigator

SCHEMAS

- flightdb
- food_db
- librarymanagement
 - Tables
 - Views
 - Stored Procedures
 - Functions
- moviesdb
- payroll_database
- query_db
- sakila
- schoolinformation
- sys
- world

74 -- b) Members with overdue books
75 SELECT m.NAME, b.TITLE, br.BORROW_DATE
76 FROM Members m
77 JOIN BorrowingRecords br ON m.MEMBER_ID = br.MEMBER_ID
78 JOIN Books b ON br.BOOK_ID = b.BOOK_ID
79 WHERE br.RETURN_DATE IS NULL
80 AND DATEDIFF(CURRENT_DATE, br.BORROW_DATE) > 30;

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

NAME	TITLE	BORROW_DATE
Rahul Kumar	The God of Small Things	2023-06-05
Priya Sharma	Wings of Fire	2023-07-01
Rahul Kumar	The Immortals of Meluha	2023-07-10
Anjali Desai	Sacred Games	2023-07-15

Administration Schemas Information

Schema: food_db

Action Output

#	Time	Action	Message	Duration / Fetch
15	15:27:02	SELECT DEPARTMENT, AVG(SALARY) AS AVG_SALARY FROM employees G...	1 row(s) returned	0.000 sec / 0.000 sec
16	15:28:12	SELECT e1.NAME, e1.SALARY FROM employees e1 INNER JOIN employees e2 ON...	0 row(s) returned	0.016 sec / 0.000 sec
17	15:28:38	SELECT e1.NAME, e1.SALARY FROM employees e1 INNER JOIN employees e2 ON...	0 row(s) returned	0.000 sec / 0.000 sec
18	15:30:24	SELECT b.TITLE, b.AUTHOR, br.BORROW_DATE, br.RETURN_DATE FROM Books ...	Error Code: 1146. Table 'payroll_database.books' doesn't exist	0.000 sec
19	15:30:32	USE LibraryManagement	0 row(s) affected	0.000 sec
20	15:30:41	SELECT b.TITLE, b.AUTHOR, br.BORROW_DATE, br.RETURN_DATE FROM Books ...	2 row(s) returned	0.016 sec / 0.000 sec
21	15:31:47	SELECT m.NAME, b.TITLE, br.BORROW_DATE FROM Members m JOIN BorrowingRe...	4 row(s) returned	0.015 sec / 0.000 sec

Object Info Session

c) Retrieve books by genre along with the count of available copies.

The screenshot shows the MySQL Workbench interface. The query editor contains the following SQL code:

```
80 AND DATEDIFF(CURRENT_DATE, br.BORROW_DATE) > 30;
81
82 -- c) Books by genre with available copies
83 • SELECT GENRE, COUNT(*) as TOTAL_BOOKS, SUM(AVAILABLE_COPIES) as TOTAL_COPIES
84 FROM Books
85 GROUP BY GENRE;
86
87 -- d) Most borrowed books
88 • SELECT b.TITLE, COUNT(*) as BORROW_COUNT
89 FROM Books b
90 JOIN BorrowingRecords br ON b.BOOK_ID = br.BOOK_ID
```

The results grid displays the following data:

GENRE	TOTAL_BOOKS	TOTAL_COPIES
Fiction	1	3
Historical Fiction	1	2
Literary Fiction	1	4
Autobiography	1	5
Magical Realism	1	2
Mythology	1	3
Crime Fiction	1	2
Romance	1	4

The output pane shows the following information:

#	Time	Action	Message	Duration / Fetch
21	15:31:47	SELECT m.NAME, b.TITLE, br.BORROW_DATE FROM Members m JOIN BorrowingRe...	4 row(s) returned	0.015 sec / 0.000 sec

d) Find the most borrowed book(s) overall.

The screenshot shows the MySQL Workbench interface. In the top-left, the Navigator pane displays various database schemas like flightdb, food_db, librarymanagement, moviesdb, payroll_database, query_db, sakila, schoolinformation, sys, and world. The central area contains a query editor tab titled 'Task_1' with the following SQL code:

```

87 -- d) Most borrowed books
88 •   SELECT b.TITLE, COUNT(*) as BORROW_COUNT
89   FROM Books b
90   JOIN BorrowingRecords br ON b.BOOK_ID = br.BOOK_ID
91   GROUP BY b.TITLE
92   ORDER BY BORROW_COUNT DESC
93   LIMIT 1;
94

```

Below the code is a 'Result Grid' showing one row of data:

TITLE	BORROW_COUNT
The Guide	1

The bottom section, 'Result 4', shows the 'Action Output' log with the following entries:

#	Time	Action	Message	Duration / Fetch
20	15:30:41	SELECT b.TITLE, b.AUTHOR, br.BORROW_DATE, br.RETURN_DATE FROM Books b JOIN BorrowingRecords br ON b.BOOK_ID = br.BOOK_ID WHERE b.TITLE = 'The Guide'	2 row(s) returned	0.016 sec / 0.000 sec
21	15:31:47	SELECT m.NAME, b.TITLE, br.BORROW_DATE FROM Members m JOIN BorrowingRecords br ON m.MEMBER_ID = br.MEMBER_ID JOIN Books b ON b.BOOK_ID = br.BOOK_ID WHERE b.TITLE = 'The Guide'	4 row(s) returned	0.015 sec / 0.000 sec
22	15:32:19	SELECT GENRE, COUNT(*) as TOTAL_BOOKS, SUM(AVAILABLE_COPIES) as TOTAL_COPIES FROM Books b GROUP BY GENRE	8 row(s) returned	0.000 sec / 0.000 sec
23	15:33:05	ELECT b.TITLE, COUNT(*) as BORROW_COUNT FROM Books b JOIN BorrowingRecords br ON b.BOOK_ID = br.BOOK_ID WHERE b.TITLE = 'The Guide'	Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'ELECT b.TITLE, COUNT(*) as BORROW_COUN...' at line 1	0.000 sec
24	15:33:06	ELECT b.TITLE, COUNT(*) as BORROW_COUNT FROM Books b JOIN BorrowingRecords br ON b.BOOK_ID = br.BOOK_ID WHERE b.TITLE = 'The Guide'	Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'ELECT b.TITLE, COUNT(*) as BORROW_COUN...' at line 1	0.000 sec
25	15:33:15	SELECT b.TITLE, COUNT(*) as BORROW_COUNT FROM Books b JOIN BorrowingRecords br ON b.BOOK_ID = br.BOOK_ID WHERE b.TITLE = 'The Guide'	1 row(s) returned	0.000 sec / 0.000 sec

e) Retrieve members who have borrowed books from at least three different genres.

There are no members who have borrowed at least 3+ genres

The screenshot shows the MySQL Workbench interface. In the top-left, the Navigator pane displays various database schemas like flightdb, food_db, and librarymanagement. The main area contains a query editor tab titled 'Task_1' with the following SQL code:

```

96 •  SELECT m.NAME, COUNT(DISTINCT b.GENRE) as GENRES_BORROWED
97   FROM Members m
98  JOIN BorrowingRecords br ON m.MEMBER_ID = br.MEMBER_ID
99  JOIN Books b ON br.BOOK_ID = b.BOOK_ID
100 GROUP BY m.NAME
101 HAVING GENRES_BORROWED >= 3;
102
103 -- Calculate total books borrowed per month:

```

Below the query editor is a 'Result Grid' table with columns 'NAME' and 'GENRES_BORROWED'. The results section shows a single row: NAME | GENRES_BORROWED.

In the bottom-right corner, there is a status bar showing system icons, weather (30°C Haze), and system time (3:34 PM / 7/17/2025).

Reporting and Analytics:

- a) Calculate the total books borrowed per month:

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 Task 2

Limit to 1000 rows

```

102
103    -- Calculate total books borrowed per month:
104 •   SELECT
105        YEAR(BORROW_DATE) as Year,
106        MONTH(BORROW_DATE) as Month,
107        COUNT(*) as Total_Books_Borrowed
108    FROM BorrowingRecords
109    GROUP BY YEAR(BORROW_DATE), MONTH(BORROW_DATE)
110    ORDER BY Year, Month;
111

```

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

Year	Month	Total_Books_Borrowed
2023	6	3
2023	7	5

Administration Schemas

Information

Schema: food_db

Action Output

#	Time	Action	Message	Duration / Fetch
25	15:33:15	SELECT b.TITLE, COUNT(*) as BORROW_COUNT FROM Books b JOIN BorrowingRe...	1 row(s) returned	0.000 sec / 0.000 sec
26	15:33:57	SELECT m.NAME, COUNT(DISTINCT b.GENRE) as GENRES_BORROWED FROM M...	0 row(s) returned	0.000 sec / 0.000 sec
27	15:35:34	SELECT YEAR(BORROW_DATE) as Year, MONTH(BORROW_DATE) as Month, COUNT(*) as Total_Books_Borrowed	2 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

b) Top three most active members:

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 Task 2

Limit to 1000 rows

```

112    -- Top three most active members:
113 •   SELECT
114        m.NAME,
115        COUNT(br.BORROW_ID) as Books_Borrowed
116    FROM Members m
117    JOIN BorrowingRecords br ON m.MEMBER_ID = br.MEMBER_ID
118    GROUP BY m.MEMBER_ID, m.NAME
119    ORDER BY Books_Borrowed DESC
120    LIMIT 3;
121

```

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

NAME	Books_Borrowed
Priya Sharma	2
Rahul Kumar	2
Anjali Desai	2

Administration Schemas

Information

Schema: food_db

Action Output

#	Time	Action	Message	Duration / Fetch
26	15:33:57	SELECT m.NAME, COUNT(DISTINCT b.GENRE) as GENRES_BORROWED FROM M...	0 row(s) returned	0.000 sec / 0.000 sec
27	15:35:34	SELECT YEAR(BORROW_DATE) as Year, MONTH(BORROW_DATE) as Month, COUNT(*) as Total_Books_Borrowed	2 row(s) returned	0.000 sec / 0.000 sec
28	15:36:14	SELECT m.NAME, COUNT(br.BORROW_ID) as Books_Borrowed	3 row(s) returned	0.015 sec / 0.000 sec

Object Info Session

c) Authors whose books were borrowed at least 10 times: **No Records**

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 Task 2

SCHEMAS

- flighthdb
- food_db
- librarymanagement**
 - Tables
 - Views
 - Stored Procedures
 - Functions
- moviesdb
- payroll_database
- query_db
- sakila
- schoolinformation
- sys
- world

Result Grid Filter Rows: Export: Wrap Cell Content: Result Grid Form Editor Read Only

```

123 • SELECT
124     b.AUTHOR,
125     COUNT(br.BORROW_ID) as Times_Borrowed
126 FROM Books b
127 JOIN BorrowingRecords br ON b.BOOK_ID = br.BOOK_ID
128 GROUP BY b.AUTHOR
129 HAVING COUNT(br.BORROW_ID) >= 10
130 ORDER BY Times_Borrowed DESC;
131
132 -- Members who have never borrowed a book:

```

AUTHOR	Times_Borrowed

Administration Schemas

Information

Schema: food_db

Action Output

#	Time	Action	Message	Duration / Fetch
27	15:35:34	SELECT	YEAR(BORROW_DATE) as Year, MONTH(BORROW_DATE) as Month, COUNT(BORROW_ID) as Books_Borrowed FROM Member...	0.000 sec / 0.000 sec
28	15:36:14	SELECT	m.NAME, COUNT(br.BORROW_ID) as Books_Borrowed FROM Member... 3 row(s) returned	0.015 sec / 0.000 sec
29	15:36:47	SELECT	b.AUTHOR, COUNT(br.BORROW_ID) as Times_Borrowed FROM Book... 0 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

d) Members who have never borrowed a book:

The screenshot shows the MySQL Workbench interface. In the top navigation bar, the 'File' and 'Edit' tabs are selected. Below the menu, there are several icons for database management. The 'Navigator' panel on the left lists various schemas, including 'librarymanagement' which is expanded to show 'Tables', 'Views', 'Stored Procedures', and 'Functions'. The main workspace contains a query editor tab titled 'Task_1' with the following SQL code:

```
131 -- Members who have never borrowed a book:
132
133 • SELECT
134     m.NAME,
135     m.EMAIL
136 FROM Members m
137 LEFT JOIN BorrowingRecords br ON m.MEMBER_ID = br.MEMBER_ID
138 WHERE br.BORROW_ID IS NULL;
139
```

Below the code, a 'Result Grid' is displayed with columns 'NAME' and 'EMAIL'. The results section shows three rows of output:

Action	Time	Message	Duration / Fetch
28	15:36:14	SELECT m.NAME, COUNT(br.BORROW_ID) as Books_Borrowed FROM Members m LEFT JOIN BorrowingRecords br ON m.MEMBER_ID = br.MEMBER_ID WHERE br.BORROW_ID IS NULL; 3 row(s) returned	0.015 sec / 0.000 sec
29	15:36:47	SELECT b.AUTHOR, COUNT(br.BORROW_ID) as Times_Borrowed FROM Books b LEFT JOIN BorrowingRecords br ON b.BOOK_ID = br.BOOK_ID WHERE br.BORROW_ID IS NULL; 0 row(s) returned	0.000 sec / 0.000 sec
30	15:37:11	SELECT m.NAME, m.EMAIL FROM Members m LEFT JOIN BorrowingRecords br ON m.MEMBER_ID = br.MEMBER_ID WHERE br.BORROW_ID IS NULL; 0 row(s) returned	0.000 sec / 0.000 sec

The bottom status bar shows system icons and the date/time: 3:37 PM 7/17/2025.

TASK 2

Project: Employee Payroll Management System

Database Setup:

Create a database named payroll database.

The screenshot shows the MySQL Workbench interface. In the top navigation bar, the 'Query' tab is selected. Below it, the 'JOINS', 'Exercise_JOINS', and 'Task_1' tabs are visible. The main workspace contains a SQL editor with the following script:

```
1 -- Database Setup
2 • CREATE DATABASE payroll_database;
3 • USE payroll_database;
4
5
```

In the left sidebar, under the 'SCHEMAS' section, the 'food_db' schema is selected. The 'Output' pane at the bottom shows the results of the executed queries:

#	Time	Action	Message	Duration / Fetch
1	14:58:11	CREATE DATABASE payroll_database	1 row(s) affected	0.016 sec
2	14:58:16	USE payroll_database	0 row(s) affected	0.000 sec

Create a table employee with columns: EMPLOYEE_ID (integer), NAME (text), DEPARTMENT (text), EMAIL (text), PHONE_NO (numeric), JOINING_DATE (date), SALARY (numeric), BONUS (numeric), TAX_PERCENTAGE (numeric).

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 SQL File 5° X

Navigator

SCHEMAS Filter objects

- flightdb
- food_db
- librarymanagement
- moviesdb
- payroll_database
- query_db
- sakila
- schoolinformation
- sys
- world

SQL Editor

```

1 -- Database Setup
2 • CREATE DATABASE payroll_database;
3 • USE payroll_database;
4
5 -- Create employees table
6 • CREATE TABLE employees (
7     EMPLOYEE_ID INT PRIMARY KEY,
8     NAME VARCHAR(100) NOT NULL,
9     DEPARTMENT VARCHAR(50),
10    EMAIL VARCHAR(100),
11    PHONE_NO BIGINT,
12    JOINING_DATE DATE,
13    SALARY DECIMAL(10,2),
14    BONUS DECIMAL(10,2),
15    TAX_PERCENTAGE DECIMAL(5,2)
16 );

```

Administration Schemas

Information

Schema: food_db

Action Output

#	Time	Action	Message	Duration / Fetch
1	14:58:11	CREATE DATABASE payroll_database	1 row(s) affected	0.016 sec
2	14:58:16	USE payroll_database	0 row(s) affected	0.000 sec
3	15:03:12	CREATE TABLE employees (EMPLOYEE_ID INT PRIMARY KEY, NAME VARCHAR... 0 row(s) affected		0.047 sec

Object Info Session

Data Entry:

Insert 10 sample employee records.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 SQL File 5° X

Navigator

SCHEMAS Filter objects

- flightdb
- food_db
- librarymanagement
- moviesdb
- payroll_database
- query_db
- sakila
- schoolinformation
- sys
- world

SQL Editor

```

13
14
15
16
17
18 -- Data Entry (10 sample records )
19 • INSERT INTO employees VALUES
20 (1, 'Satendra Mishra', 'Sales', 'satendra.m@company.com', 9876543210, '2023-01-15', 85000, 15000, 20),
21 (2, 'Rajendra Singh', 'IT', 'rajendra.s@company.com', 9876543211, '2022-08-20', 95000, 12000, 25),
22 (3, 'Shubham Sharma', 'HR', 'shubham.s@company.com', 9876543212, '2023-05-10', 75000, 8000, 18),
23 (4, 'Himanshu Mishra', 'Sales', 'himanshu.m@company.com', 9876543213, '2023-03-01', 82000, 14000, 20),
24 (5, 'Rajan Khanna', 'IT', 'rajan.k@company.com', 9876543214, '2022-11-15', 98000, 13000, 25),
25 (6, 'Amrit Tripathi', 'Finance', 'amrit.t@company.com', 9876543215, '2023-02-01', 88000, 10000, 22),
26 (7, 'Abhishek Pandey', 'Sales', 'abhishek.p@company.com', 9876543216, '2022-12-10', 80000, 12000, 20),
27 (8, 'Aniket Pandey', 'IT', 'aniket.p@company.com', 9876543217, '2023-04-15', 92000, 11000, 25),
28 (9, 'Ashu Kumar', 'Finance', 'ashu.k@company.com', 9876543218, '2022-09-01', 86000, 9000, 22),
29 (10, 'JP Chawla', 'HR', 'jp.c@company.com', 9876543219, '2023-06-01', 78000, 7000, 18);

```

Administration Schemas

Information

Schema: food_db

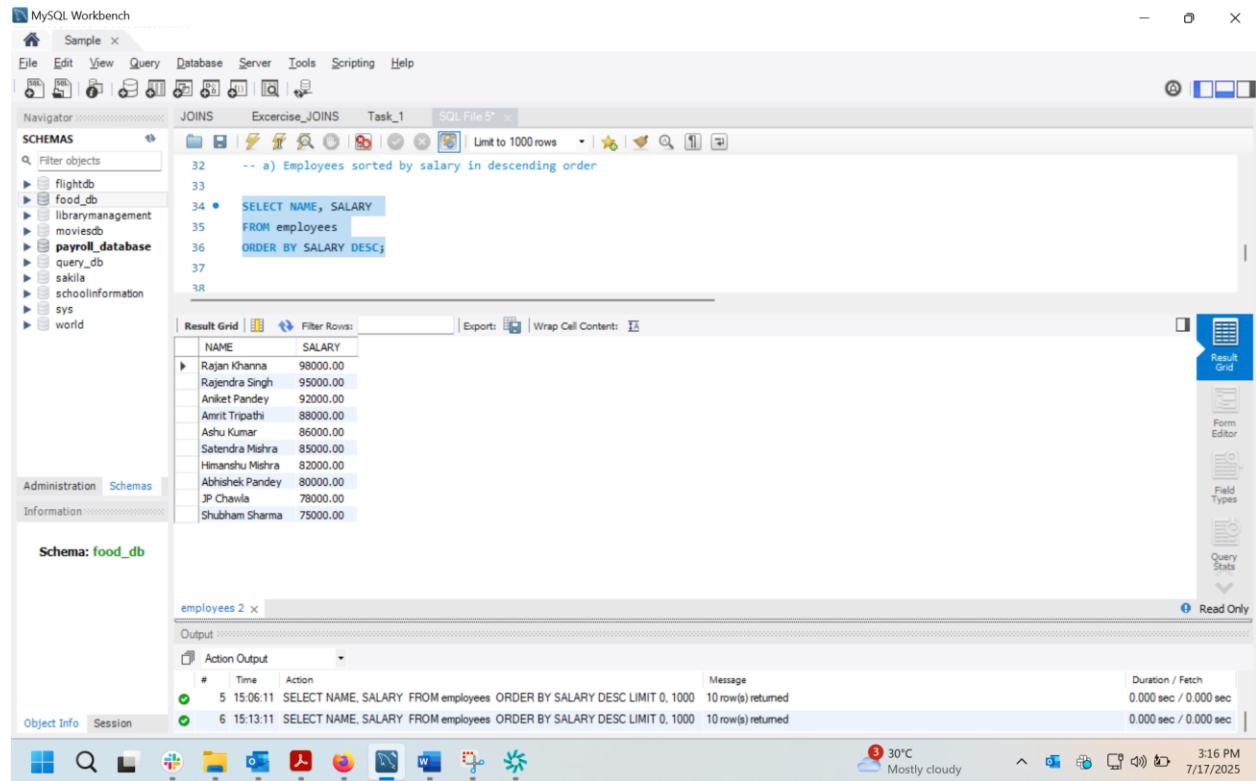
Action Output

#	Time	Action	Message	Duration / Fetch
1	14:58:11	CREATE DATABASE payroll_database	1 row(s) affected	0.016 sec
2	14:58:16	USE payroll_database	0 row(s) affected	0.000 sec
3	15:03:12	CREATE TABLE employees (EMPLOYEE_ID INT PRIMARY KEY, NAME VARCHAR... 0 row(s) affected		0.047 sec
4	15:04:18	INSERT INTO employees VALUES (1, 'Satendra Mishra', 'Sales', 'satendra.m@company.co... 10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0		0.016 sec

Object Info Session

Payroll Queries:

a) Retrieve the list of employees sorted by salary in descending order



The screenshot shows the MySQL Workbench interface. In the top navigation bar, the 'Query' tab is selected. Below it, the 'JOINS', 'Excercise_JOINS', and 'Task_1' tabs are visible. The main area contains a SQL editor window with the following query:

```
32 -- a) Employees sorted by salary in descending order
33
34 • SELECT NAME, SALARY
  FROM employees
  ORDER BY SALARY DESC;
35
36
37
38
```

Below the query, the 'Result Grid' shows the results:

NAME	SALARY
Rajan Khanna	98000.00
Rajendra Singh	95000.00
Aniket Pandey	92000.00
Amrit Tripathi	88000.00
Ashu Kumar	86000.00
Satendra Mishra	85000.00
Himanshu Mishra	82000.00
Abhishek Pandey	80000.00
JP Chawla	78000.00
Shubham Sharma	75000.00

At the bottom of the interface, there is an 'Output' section showing two log entries:

- 5 15:06:11 SELECT NAME, SALARY FROM employees ORDER BY SALARY DESC LIMIT 0, 1000 10 row(s) returned
- 6 15:13:11 SELECT NAME, SALARY FROM employees ORDER BY SALARY DESC LIMIT 0, 1000 10 row(s) returned

b) Find employees with total compensation (SALARY + BONUS) greater than \$100,000.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 SQL File 5

Limit to 1000 rows

```

32 -- a) Employees sorted by salary in descending order
33
34 • SELECT NAME, SALARY
  FROM employees
  ORDER BY SALARY DESC;
35
36
37
38
39 -- b) Employees with total compensation > $100,000
40
41 • SELECT NAME, (SALARY + BONUS) as TOTAL_COMPENSATION
  FROM employees
  WHERE (SALARY + BONUS) > 100000;

```

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

NAME TOTAL_COMPENSATION

Rajendra Singh	107000.00
Rajan Khanna	111000.00
Aniket Pandey	103000.00

Schema: food_db

Result 3 x Read Only

Action Output

#	Time	Action	Message	Duration / Fetch
6	15.13.11	SELECT NAME, SALARY FROM employees ORDER BY SALARY DESC LIMIT 0, 1000	10 row(s) returned	0.000 sec / 0.000 sec

Object Info Session 3: 30°C Mostly cloudy 3:16 PM 7/17/2025

c) Update the bonus for employees in the ‘Sales’ department by 10%.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 SQL File 5

Limit to 1000 rows

```

28 (9, 'Ashu Kumar', 'Finance', 'ashu.k@company.com', 9876543218, '2022-09-01', 86000, 9000, 22),
29 (10, 'JP Chawla', 'HR', 'jp.c@company.com', 9876543219, '2023-06-01', 78000, 7000, 18);

30 -- Payroll Queries
31 -- a) Employees sorted by salary in descending order
32
33
34 • SELECT NAME, SALARY
  FROM employees
  ORDER BY SALARY DESC;
35
36
37
38
39 -- b) Employees with total compensation > $100,000
40
41 • SELECT NAME, (SALARY + BONUS) as TOTAL_COMPENSATION
  FROM employees
  WHERE (SALARY + BONUS) > 100000;
42
43
44
45
46 -- c) Update bonus for Sales department
47 • UPDATE employees
  SET BONUS = BONUS * 1.1
  WHERE DEPARTMENT = 'Sales';
48
49

```

Output

Action Output

#	Time	Action	Message	Duration / Fetch
7	15.16.48	SELECT NAME, (SALARY + BONUS) as TOTAL_COMPENSATION FROM employees ...	3 row(s) returned	0.000 sec / 0.000 sec

Object Info Session 3: 30°C Mostly cloudy 3:17 PM 7/17/2025

d) Calculate the net salary after deducting tax for all employees.

The screenshot shows the MySQL Workbench interface. The SQL editor tab contains the following query:

```
-- d) Calculate net salary after tax
SELECT
    NAME,
    SALARY,
    BONUS,
    (SALARY + BONUS) * (1 - TAX_PERCENTAGE/100) as NET_SALARY
FROM employees;
```

The Result Grid shows the output of the query:

NAME	SALARY	BONUS	NET_SALARY
Satendra Mishra	85000.00	15000.00	80000.0000000000
Rajendra Singh	95000.00	12000.00	80250.0000000000
Shubham Sharma	75000.00	8000.00	68060.0000000000
Himanshu Mishra	82000.00	14000.00	76800.0000000000
Rajan Khanna	98000.00	13000.00	83250.0000000000
Amrit Tripathi	88000.00	10000.00	76440.0000000000
Abhishek Pandey	80000.00	12000.00	73600.0000000000
Aniket Pandey	92000.00	11000.00	77250.0000000000
Ashu Kumar	86000.00	9000.00	74100.0000000000
JP Chawla	78000.00	7000.00	69700.0000000000

The Result Grid panel also displays the following information:

- Result Grid
- Filter Rows:
- Export:
- Wrap Cell Content:
- Result 5
- Output
- Action Output
- # Time Action Message Duration / Fetch
- 9 15:19:01 SELECT NAME, SALARY, BONUS, (SALARY + BONUS) * (1 - TAX_PERCE... 10 row(s) returned 0.000 sec / 0.000 sec
- 10 15:19:46 SELECT NAME, SALARY, BONUS, (SALARY + BONUS) * (1 - TAX_PERCE... 10 row(s) returned 0.000 sec / 0.000 sec

The status bar at the bottom right shows: Hot days ahead 30°C 3:21 PM 7/17/2025

e) Retrieve the average, minimum, and maximum salary per department.

The screenshot shows the MySQL Workbench interface. In the top navigation bar, the 'File' tab is selected. Below the menu, there's a toolbar with various icons. The main area contains a query editor window titled 'JOINS' with the sub-tab 'Exercise_JOINs'. The SQL code in the editor is:

```

60 -- e) Average, minimum, and maximum salary per department
61 • SELECT
62   DEPARTMENT,
63   AVG(SALARY) as AVG_SALARY,
64   MIN(SALARY) as MIN_SALARY,
65   MAX(SALARY) as MAX_SALARY
66 FROM employees
67 GROUP BY DEPARTMENT;

```

Below the code is a 'Result Grid' table with the following data:

DEPARTMENT	AVG_SALARY	MIN_SALARY	MAX_SALARY
Sales	82333.333333	80000.00	85000.00
IT	95000.000000	92000.00	98000.00
HR	76500.000000	75000.00	78000.00
Finance	87000.000000	86000.00	88000.00

On the right side of the interface, there's a vertical toolbar with icons for 'Result Grid', 'Form Editor', and 'Field Types'. At the bottom left, there are tabs for 'Object Info' and 'Session'. The status bar at the bottom right indicates 'Read Only'.

Advanced Queries:

- a) Retrieve employees who have joined in the last 6 months.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 SQL File 5* x

Limit to 1000 rows

```

67 GROUP BY DEPARTMENT;
68
69 -- Advanced Queries
70
71 -- a) Employees who joined in last 6 months
72 • SELECT NAME, JOINING_DATE
73 FROM employees
74 WHERE JOINING_DATE >= DATE_SUB(CURRENT_DATE, INTERVAL 6 MONTH);

```

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

NAME	JOINING_DATE

Administration Schemas

Information

Schema: food_db

employees 7 x Read Only

Action Output

#	Time	Action	Message	Duration / Fetch
12	15:24:55	ELECT NAME,JOINING_DATE FROM employees WHERE JOINING_DATE >= DATE...	Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'ELECT NAME,JOINING_DATE FROM employees WHERE JOINING_DATE >= DAT...' at line 1	0.000 sec
13	15:25:01	SELECT NAME,JOINING_DATE FROM employees WHERE JOINING_DATE >= DAT...	0 row(s) returned	0.015 sec / 0.000 sec

Object Info Session 3: Very humid Now 3:25 PM 7/17/2025

b) Group employees by department and count how many employees each has.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 SQL File 5* x

Limit to 1000 rows

```

75
76 -- b) Employee count by department
77 • SELECT
78   DEPARTMENT,
79   COUNT(*) AS EMPLOYEE_COUNT
80 FROM employees
81 GROUP BY DEPARTMENT;
82

```

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

DEPARTMENT	EMPLOYEE_COUNT
Sales	3
IT	3
HR	2
Finance	2

Administration Schemas

Information

Schema: food_db

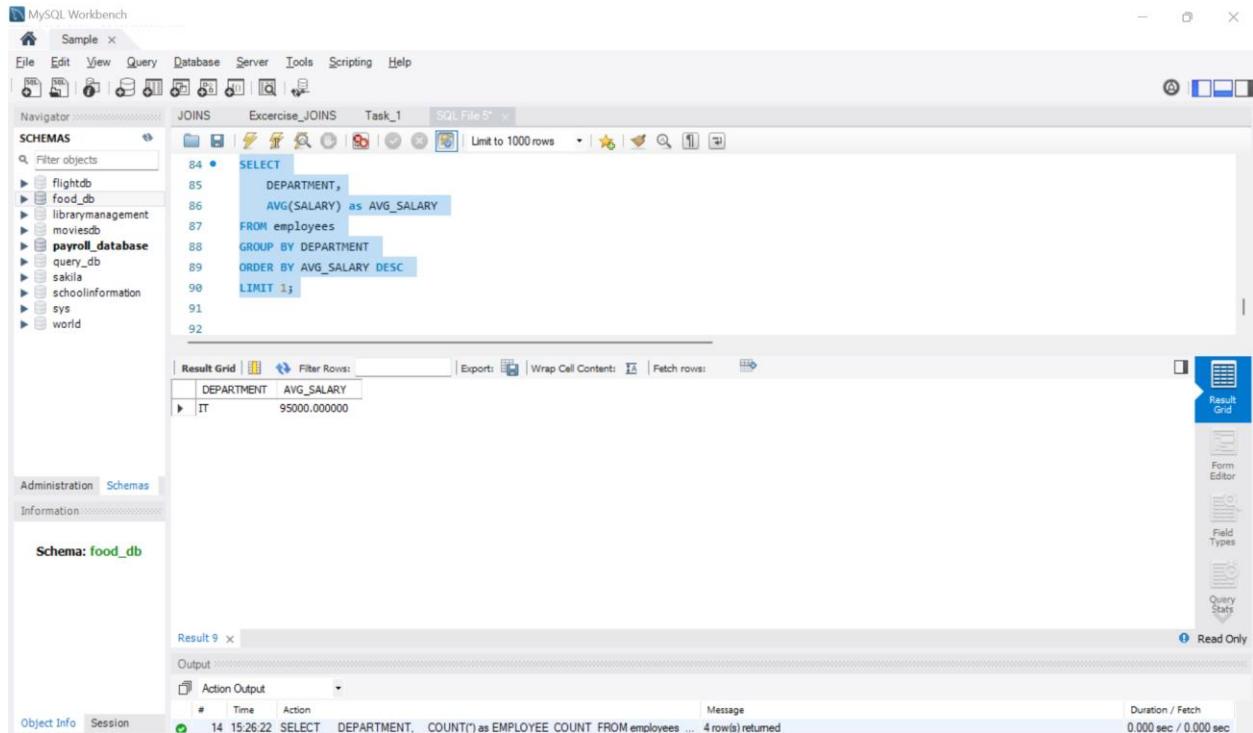
Result 8 x Read Only

Action Output

#	Time	Action	Message	Duration / Fetch
13	15:25:01	SELECT NAME,JOINING_DATE FROM employees WHERE JOINING_DATE >= DAT...	0 row(s) returned	0.015 sec / 0.000 sec
14	15:26:22	SELECT DEPARTMENT, COUNT(*) AS EMPLOYEE_COUNT FROM employees ...	4 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

c) Find the department with the highest average salary.



The screenshot shows the MySQL Workbench interface. In the center, there is a SQL editor tab titled "Task_1" containing the following query:

```
84 •    SELECT
85      DEPARTMENT,
86      AVG(SALARY) as AVG_SALARY
87  FROM employees
88  GROUP BY DEPARTMENT
89  ORDER BY AVG_SALARY DESC
90  LIMIT 1;
91
92
```

Below the SQL editor is a "Result Grid" table with two columns: "DEPARTMENT" and "AVG_SALARY". The table contains one row with the value "IT" in the DEPARTMENT column and "95000.000000" in the AVG_SALARY column.

At the bottom of the interface, there is a "Result 9" window showing the execution details of the query. The message area indicates "4 rows(s) returned" and the duration is "0.000 sec / 0.000 sec".

d) Identify employees who have the same salary as at least one other employee.

The screenshot shows the MySQL Workbench interface. In the top-left, the Navigator pane displays various databases: flightdb, food_db, librarymanagement, moviesdb, payroll_database, query_db, sakila, schoolinformation, sys, and world. The central area contains a SQL editor window titled 'SQL File 5' with the following code:

```

94 FROM employees e1
95 INNER JOIN employees e2
96 ON e1.SALARY = e2.SALARY
97 AND e1.EMPLOYEE_ID != e2.EMPLOYEE_ID
98 ORDER BY e1.SALARY
qq

```

Below the SQL editor is a 'Result Grid' table with columns 'NAME' and 'SALARY'. The bottom section of the interface shows the 'Result 11' tab, which displays the execution history of the session. The log entries are as follows:

#	Time	Action	Message	Duration / Fetch
11	15:21:56	SELECT DEPARTMENT, AVG(SALARY) as AVG_SALARY, MIN(SALARY) as ...	4 row(s) returned	0.000 sec / 0.000 sec
12	15:24:55	ELECT NAME, JOINING_DATE FROM employees WHERE JOINING_DATE >= DATE... Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'ELECT NAME, JOINING_DATE FROM employees WHERE JOINING_DATE >= DATE...' at line 1		0.000 sec
13	15:25:01	SELECT NAME, JOINING_DATE FROM employees WHERE JOINING_DATE >= DAT...	0 row(s) returned	0.015 sec / 0.000 sec
14	15:26:22	SELECT DEPARTMENT, COUNT(*) as EMPLOYEE_COUNT FROM employees ...	4 row(s) returned	0.000 sec / 0.000 sec
15	15:27:02	SELECT DEPARTMENT, AVG(SALARY) as AVG_SALARY FROM employees G...	1 row(s) returned	0.000 sec / 0.000 sec
16	15:28:12	SELECT e1.NAME, e1.SALARY FROM employees e1 INNER JOIN employees e2 ON...	0 row(s) returned	0.016 sec / 0.000 sec
17	15:28:38	SELECT e1.NAME, e1.SALARY FROM employees e1 INNER JOIN employees e2 ON...	0 row(s) returned	0.000 sec / 0.000 sec

Task 3

Project: Online Store Order Management System

Database Creation:

Create a database named Online Store.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 Task 2 SQL File 6*

Navigator Schemas

Filter objects

SCHEMAS

- flightdb
- food_db
- librarymanagement
- moviesdb
- payroll_database
- query_db
- sakila
- schoolinformation
- sys
- world

Administration Schemas

Information

No object selected

Output

Action Output

#	Time	Action	Message	Duration / Fetch
30	15:37:11	SELECT m.NAME, m.EMAIL FROM Members m LEFT JOIN BorrowingRecords br ...	0 row(s) returned	0.000 sec / 0.000 sec
31	15:49:27	CREATE DATABASE OnlineStore	1 row(s) affected	0.000 sec
32	15:49:40	USE OnlineStore	0 row(s) affected	0.000 sec

Object Info Session

30°C Haze 3:49 PM 7/17/2025

The screenshot shows the MySQL Workbench interface. In the top navigation bar, 'File', 'Edit', 'View', 'Query', 'Database', 'Server', 'Tools', 'Scripting', and 'Help' are visible. Below the menu is a toolbar with various icons. The main area has tabs for 'JOINS', 'Exercise_JOINS', 'Task_1', 'Task 2', and 'SQL File 6*'. The 'SQL File 6*' tab is active, displaying the following SQL code:

```
1 -- Create the database
2
3 • CREATE DATABASE OnlineStore;
4
5 -- Use the database
6 • USE OnlineStore;
```

On the left, the 'Navigator' and 'Schemas' panes are open, showing a list of databases including 'flightdb', 'food_db', 'librarymanagement', 'moviesdb', 'payroll_database', 'query_db', 'sakila', 'schoolinformation', 'sys', and 'world'. The 'Information' pane below shows 'No object selected'. The 'Output' pane at the bottom displays the execution log with three entries:

#	Time	Action	Message	Duration / Fetch
30	15:37:11	SELECT m.NAME, m.EMAIL FROM Members m LEFT JOIN BorrowingRecords br ...	0 row(s) returned	0.000 sec / 0.000 sec
31	15:49:27	CREATE DATABASE OnlineStore	1 row(s) affected	0.000 sec
32	15:49:40	USE OnlineStore	0 row(s) affected	0.000 sec

The status bar at the bottom right shows the temperature (30°C), weather (Haze), time (3:49 PM), and date (7/17/2025).

Create tables:

Customers (CUSTOMER_ID, NAME, EMAIL, PHONE, ADDRESS)

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Excercise_JOINS Task_1 Task 2 SQL File 6*

SCHEMAS

- flighthdb
- food_db
- librarymanagement
- moviesdb
- payroll_database
- query_db
- sakila
- schoolinformation
- sys
- world

1 -- Create the database
2
3 • CREATE DATABASE OnlineStore;
4
5 -- Use the database
6 • USE OnlineStore;
7
8 -- Create Customers table
9 • CREATE TABLE Customers (
10 CUSTOMER_ID INT AUTO_INCREMENT PRIMARY KEY,
11 NAME VARCHAR(100),
12 EMAIL VARCHAR(100),
13 PHONE VARCHAR(20),
14 ADDRESS TEXT
15);

Administration Schemas

No object selected

Output

#	Time	Action	Message	Duration / Fetch
31	15:49:27	CREATE DATABASE OnlineStore	1 row(s) affected	0.000 sec
32	15:49:40	USE OnlineStore	0 row(s) affected	0.000 sec
33	15:51:02	CREATE TABLE Customers (CUSTOMER_ID INT AUTO_INCREMENT PRIMARY K...	0 row(s) affected	0.016 sec

Object Info Session

Products (PRODUCT_ID, PRODUCT_NAME, CATEGORY, PRICE, STOCK)

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Excercise_JOINS Task_1 Task 2 SQL File 6*

SCHEMAS

- flighthdb
- food_db
- librarymanagement
- moviesdb
- payroll_database
- query_db
- sakila
- schoolinformation
- sys
- world

1 -- Use the database
2 • USE OnlineStore;
3
4 -- Create Customers table
5 • CREATE TABLE Customers (
6 CUSTOMER_ID INT AUTO_INCREMENT PRIMARY KEY,
7 NAME VARCHAR(100),
8 EMAIL VARCHAR(100),
9 PHONE VARCHAR(20),
10 ADDRESS TEXT
11);

12
13 -- Create Products table
14 • CREATE TABLE Products (
15 PRODUCT_ID INT AUTO_INCREMENT PRIMARY KEY,
16 PRODUCT_NAME VARCHAR(100),
17 CATEGORY VARCHAR(50),
18 PRICE DECIMAL(10, 2),
19 STOCK INT
20);

21
22
23
24);

Administration Schemas

No object selected

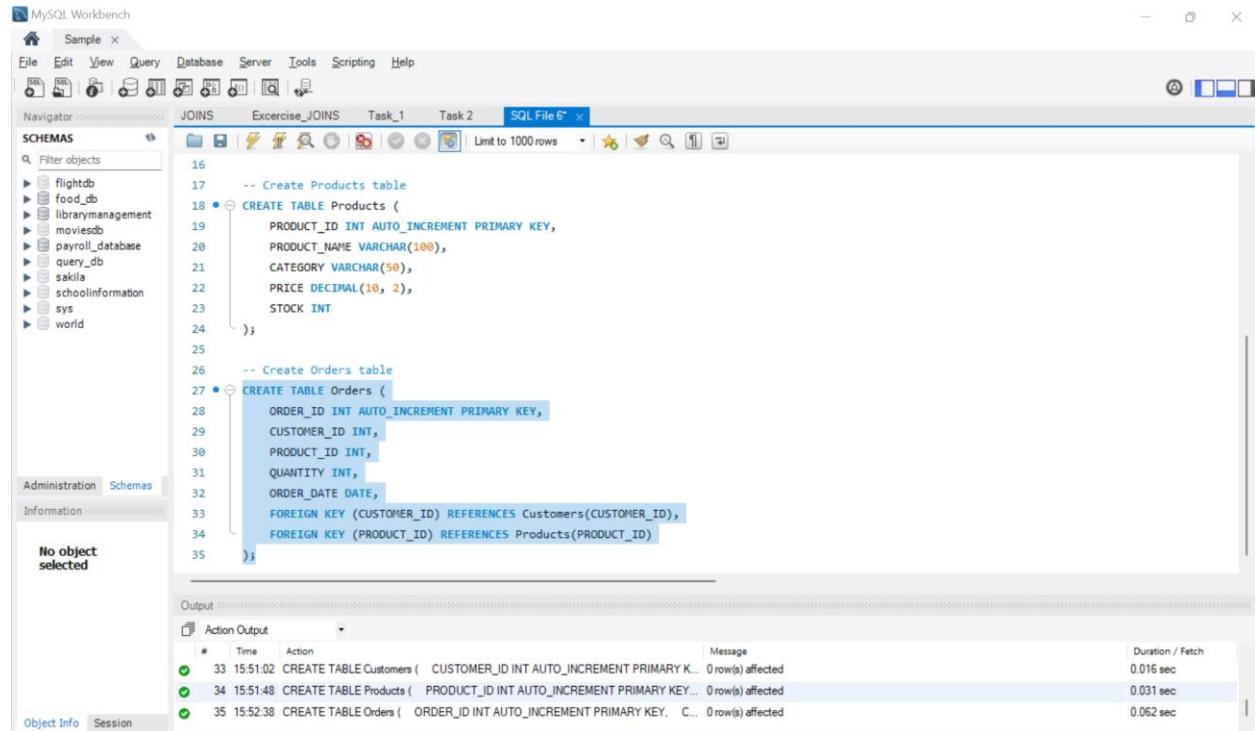
Output

#	Time	Action	Message	Duration / Fetch
32	15:49:40	USE OnlineStore	0 row(s) affected	0.000 sec
33	15:51:02	CREATE TABLE Customers (CUSTOMER_ID INT AUTO_INCREMENT PRIMARY K...	0 row(s) affected	0.016 sec
34	15:51:48	CREATE TABLE Products (PRODUCT_ID INT AUTO_INCREMENT PRIMARY KEY...	0 row(s) affected	0.031 sec

Object Info Session

Orders (ORDER_ID, CUSTOMER_ID, PRODUCT_ID, QUANTITY, ORDER_DATE)

Set up foreign keys linking Orders to Customers and Products



The screenshot shows the MySQL Workbench interface with the following details:

- Navigator:** Shows various databases: flightdb, food_db, librarymanagement, moviesdb, payroll_database, query_db, sakila, schoolinformation, sys, and world.
- SQL Editor:** Contains the following SQL code:

```
16 -- Create Products table
17 CREATE TABLE Products (
18     PRODUCT_ID INT AUTO_INCREMENT PRIMARY KEY,
19     PRODUCT_NAME VARCHAR(100),
20     CATEGORY VARCHAR(50),
21     PRICE DECIMAL(10, 2),
22     STOCK INT
23 );
24
25
26 -- Create Orders table
27 CREATE TABLE Orders (
28     ORDER_ID INT AUTO_INCREMENT PRIMARY KEY,
29     CUSTOMER_ID INT,
30     PRODUCT_ID INT,
31     QUANTITY INT,
32     ORDER_DATE DATE,
33     FOREIGN KEY (CUSTOMER_ID) REFERENCES Customers(CUSTOMER_ID),
34     FOREIGN KEY (PRODUCT_ID) REFERENCES Products(PRODUCT_ID)
35 );
```

The code creates two tables: `Products` and `Orders`. The `Products` table has columns for `PRODUCT_ID` (primary key), `PRODUCT_NAME`, `CATEGORY`, `PRICE`, and `STOCK`. The `Orders` table has columns for `ORDER_ID` (primary key), `CUSTOMER_ID`, `PRODUCT_ID`, `QUANTITY`, and `ORDER_DATE`. It includes foreign key constraints linking `Customer_ID` in `Orders` to `CUSTOMER_ID` in `Customers` and `Product_ID` in `Orders` to `PRODUCT_ID` in `Products`.

Output: Shows the execution results:

#	Time	Action	Message	Duration / Fetch
33	15:51:02	CREATE TABLE Customers (CUSTOMER_ID INT AUTO_INCREMENT PRIMARY K...	0 row(s) affected	0.016 sec
34	15:51:48	CREATE TABLE Products (PRODUCT_ID INT AUTO_INCREMENT PRIMARY KEY...	0 row(s) affected	0.031 sec
35	15:52:38	CREATE TABLE Orders (ORDER_ID INT AUTO_INCREMENT PRIMARY KEY, C...	0 row(s) affected	0.062 sec

Data Creation:

Insert sample data for customers, products, and orders.

Customers-

The screenshot shows the MySQL Workbench interface with the following details:

- Schemas:** A list of databases including flightdb, food_db, librarymanagement, moviesdb, payroll_database, query_db, sakila, schoolinformation, sys, and world.
- SQL Editor:** Contains the following SQL code:

```
34      FOREIGN KEY (PRODUCT_ID) REFERENCES Products(PRODUCT_ID)
35  );
36  -- Insert sample customers
37  •  INSERT INTO Customers (NAME, EMAIL, PHONE, ADDRESS) VALUES
38  ('Aarav Sharma', 'aarav@email.com', '9876540001', 'Mumbai, Maharashtra'),
39  ('Zara Patel', 'zara@email.com', '9876540002', 'Delhi, NCR'),
40  ('Vihaan Singh', 'vihaan@email.com', '9876540003', 'Bangalore, Karnataka'),
41  ('Anaya Reddy', 'anaya@email.com', '9876540004', 'Hyderabad, Telangana'),
42  ('Adwait Kapoor', 'adwait@email.com', '9876540005', 'Chennai, Tamil Nadu'),
43  ('Ishaan Mehta', 'ishaan@email.com', '9876540006', 'Pune, Maharashtra'),
44  ('Myra Joshi', 'myra@email.com', '9876540007', 'Kolkata, West Bengal'),
45  ('Arjun Kumar', 'arjun@email.com', '9876540008', 'Ahmedabad, Gujarat'),
46  ('Aanya Malhotra', 'aanya@email.com', '9876540009', 'Jaipur, Rajasthan'),
47  ('Reyansh Verma', 'reyansh@email.com', '9876540010', 'Lucknow, UP')
48
49
50
51
52
53
```

Output: Shows the execution results:

#	Time	Action	Message	Duration / Fetch
34	15:51:48	CREATE TABLE Products (PRODUCT_ID INT AUTO_INCREMENT PRIMARY KEY...	0 row(s) affected	0.031 sec
35	15:52:38	CREATE TABLE Orders (ORDER_ID INT AUTO_INCREMENT PRIMARY KEY...	0 row(s) affected	0.062 sec
36	15:54:06	INSERT INTO Customers (NAME, EMAIL, PHONE, ADDRESS) VALUES (Aarav Sham...	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.000 sec

Products-

The screenshot shows the MySQL Workbench interface with the following details:

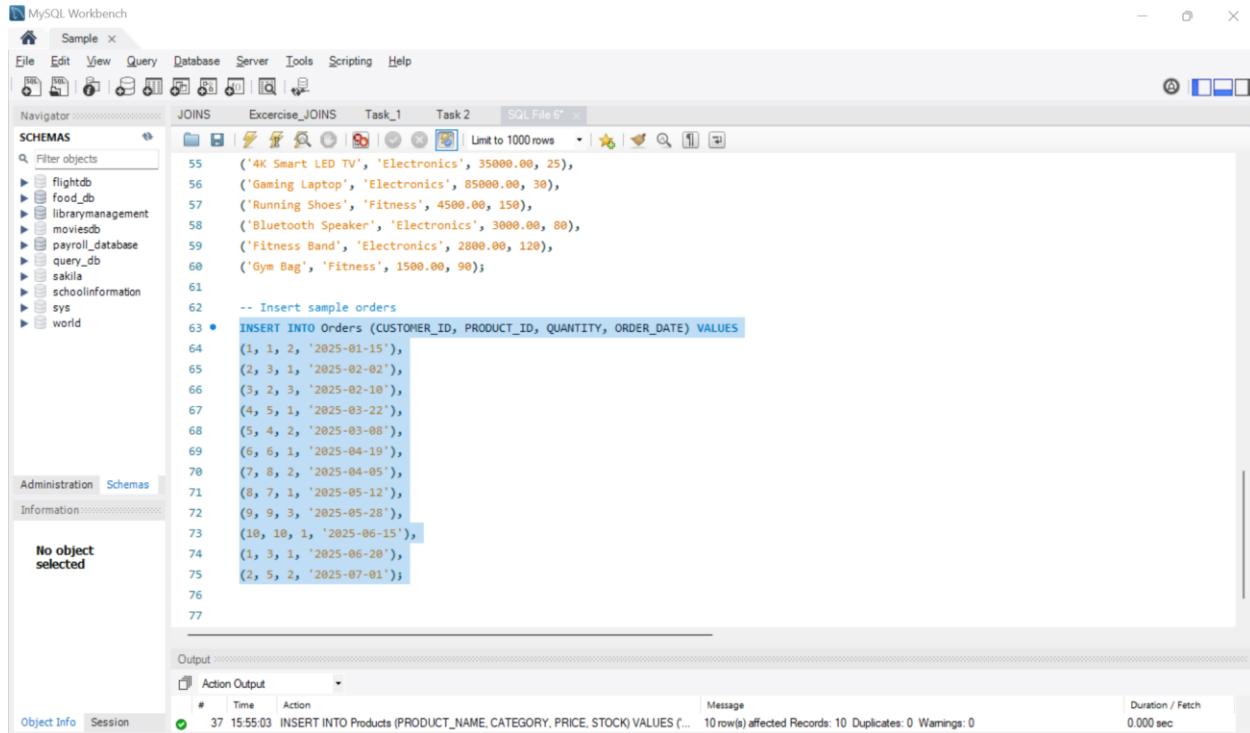
- Schemas:** A list of databases including flightdb, food_db, librarymanagement, moviesdb, payroll_database, query_db, sakila, schoolinformation, sys, and world.
- SQL Editor:** Contains the following SQL code:

```
44  ('Myra Joshi', 'myra@email.com', '9876540007', 'Kolkata, West Bengal'),
45  ('Arjun Kumar', 'arjun@email.com', '9876540008', 'Ahmedabad, Gujarat'),
46  ('Aanya Malhotra', 'aanya@email.com', '9876540009', 'Jaipur, Rajasthan'),
47  ('Reyansh Verma', 'reyansh@email.com', '9876540010', 'Lucknow, UP')
48
49  -- Insert sample products
50  •  INSERT INTO Products (PRODUCT_NAME, CATEGORY, PRICE, STOCK) VALUES
51  ('Smart Watch Pro', 'Electronics', 15000.00, 50),
52  ('Premium Yoga Mat', 'Fitness', 1200.00, 100),
53  ('Wireless Earbuds Plus', 'Electronics', 8000.00, 75),
54  ('Whey Protein 1kg', 'Fitness', 2500.00, 200),
55  ('4K Smart LED TV', 'Electronics', 35000.00, 25),
56  ('Gaming Laptop', 'Electronics', 85000.00, 30),
57  ('Running Shoes', 'Fitness', 4500.00, 150),
58  ('Bluetooth Speaker', 'Electronics', 3000.00, 80),
59  ('Fitness Band', 'Electronics', 2800.00, 120),
60  ('Gym Bag', 'Fitness', 1500.00, 90);
61
62
63
64
65
```

Output: Shows the execution results:

#	Time	Action	Message	Duration / Fetch
36	15:54:06	INSERT INTO Customers (NAME, EMAIL, PHONE, ADDRESS) VALUES (Aarav Sham...	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.000 sec

Orders



The screenshot shows the MySQL Workbench interface with a query editor containing the following SQL code:

```
55     ('4K Smart LED TV', 'Electronics', 35000.00, 25),
56     ('Gaming Laptop', 'Electronics', 85000.00, 30),
57     ('Running Shoes', 'Fitness', 4500.00, 150),
58     ('Bluetooth Speaker', 'Electronics', 3000.00, 80),
59     ('Fitness Band', 'Electronics', 2800.00, 120),
60     ('Gym Bag', 'Fitness', 1500.00, 90);
61
62 -- Insert sample orders
63 • INSERT INTO Orders (CUSTOMER_ID, PRODUCT_ID, QUANTITY, ORDER_DATE) VALUES
64     (1, 1, 2, '2025-01-15'),
65     (2, 3, 1, '2025-02-02'),
66     (3, 2, 3, '2025-02-10'),
67     (4, 5, 1, '2025-03-22'),
68     (5, 4, 2, '2025-03-08'),
69     (6, 6, 1, '2025-04-19'),
70     (7, 8, 2, '2025-04-05'),
71     (8, 7, 1, '2025-05-12'),
72     (9, 9, 3, '2025-05-28'),
73     (10, 10, 1, '2025-06-15'),
74     (1, 3, 1, '2025-06-20'),
75     (2, 5, 2, '2025-07-01');
76
77
```

The output pane shows the result of the insertion:

#	Time	Action	Message	Duration / Fetch
37	15:55:03	INSERT INTO Products (PRODUCT_NAME, CATEGORY, PRICE, STOCK) VALUES (...)	10 row(s) affected Records: 10 Duplicates: 0 Warnings: 0	0.000 sec

Order Management:

- a) Retrieve all orders placed by a specific customer.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 Task 2 Task 3*

Limit to 1000 rows

```

79
80 • SELECT
81   c.NAME AS Customer_Name,
82   p.PRODUCT_NAME,
83   o.QUANTITY,
84   p.PRICE,
85   (o.QUANTITY * p.PRICE) AS Total_Amount,
86   o.ORDER_DATE
87 FROM Orders o
88 JOIN Customers c ON o.CUSTOMER_ID = c.CUSTOMER_ID
89 JOIN Products p ON o.PRODUCT_ID = p.PRODUCT_ID
90 WHERE c.CUSTOMER_ID = 2 -- Change this number to search for different customers
91 ORDER BY o.ORDER_DATE;
92

```

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

Customer_Name	PRODUCT_NAME	QUANTITY	PRICE	Total_Amount	ORDER_DATE
Zara Patel	Wireless Earbuds Plus	1	8000.00	8000.00	2025-02-02
Zara Patel	4K Smart LED TV	2	35000.00	70000.00	2025-07-01

No object selected

Result 2 | Output

Action Output

#	Time	Action	Message	Duration / Fetch
38	15:56:05	INSERT INTO Orders (CUSTOMER_ID, PRODUCT_ID, QUANTITY, ORDER_DATE) V...	12 row(s) affected Records: 12 Duplicates: 0 Warnings: 0	0.016 sec
39	16:12:03	SELECT c.NAME AS Customer_Name, p.PRODUCT_NAME, o.QUANTITY, ...	2 row(s) returned	0.000 sec / 0.000 sec
40	16:12:39	SELECT c.NAME AS Customer_Name, p.PRODUCT_NAME, o.QUANTITY, ...	2 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

b) Find products that are out of stock.

There are 0 products which are currently OOS.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 Task 2 Task 3*

Limit to 1000 rows

```

90 WHERE c.CUSTOMER_ID = 2 -- Change this number to search for different customers
91 ORDER BY o.ORDER_DATE;
92
93 -- Products that are OOS
94 • SELECT
95   PRODUCT_ID,
96   PRODUCT_NAME,
97   CATEGORY,
98   PRICE,
99   STOCK
100 FROM Products
101 WHERE STOCK = 0
102 ORDER BY PRODUCT_NAME;
103

```

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

PRODUCT_ID	PRODUCT_NAME	CATEGORY	PRICE	STOCK
HULL	HULL	HULL	HULL	HULL

No object selected

Products 3 | Output

Action Output

#	Time	Action	Message	Duration / Fetch
39	16:12:03	SELECT c.NAME AS Customer_Name, p.PRODUCT_NAME, o.QUANTITY, ...	2 row(s) returned	0.000 sec / 0.000 sec
40	16:12:39	SELECT c.NAME AS Customer_Name, p.PRODUCT_NAME, o.QUANTITY, ...	2 row(s) returned	0.000 sec / 0.000 sec
41	16:13:34	SELECT PRODUCT_ID, PRODUCT_NAME, CATEGORY, PRICE, STOCK	0 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

c) Calculate the total revenue generated per product.

The screenshot shows the MySQL Workbench interface. The top menu bar includes File, Edit, View, Query, Database, Server, Tools, Scripting, and Help. The main window has tabs for JOINS, Exercise_JOINS, Task_1, Task 2, and Task 3*. The JOINS tab is active. A query editor window displays the following SQL code:

```
105 •    SELECT
106      p.PRODUCT_NAME,
107      p.CATEGORY,
108      SUM(o.QUANTITY) AS Total_Units_Sold,
109      p.PRICE AS Unit_Price,
110      SUM(o.QUANTITY * p.PRICE) AS Total_Revenue
111  FROM Products p
112  LEFT JOIN Orders o ON p.PRODUCT_ID = o.PRODUCT_ID
113  GROUP BY p.PRODUCT_ID, p.PRODUCT_NAME, p.CATEGORY, p.PRICE
114  ORDER BY Total_Revenue DESC;
115
```

The Result Grid shows the following data:

PRODUCT_NAME	CATEGORY	Total_Units_Sold	Unit_Price	Total_Revenue
4K Smart LED TV	Electronics	3	35000.00	105000.00
Gaming Laptop	Electronics	1	85000.00	85000.00
Smart Watch Pro	Electronics	2	15000.00	30000.00
Wireless Earbuds Plus	Electronics	2	8000.00	16000.00
Fitness Band	Electronics	3	2800.00	8400.00
Bluetooth Speaker	Electronics	2	3000.00	6000.00
Whey Protein 1kg	Fitness	2	2500.00	5000.00
Running Shoes	Fitness	1	4500.00	4500.00
Premium Yoga Mat	Fitness	3	1200.00	3600.00
Gym Bag	Fitness	1	1500.00	1500.00

The bottom pane shows the Action Output log with one entry:

#	Time	Action	Message	Duration / Fetch
41	16:13:34	SELECT	PRODUCT_ID, PRODUCT_NAME, CATEGORY, PRICE, STOC...	0 row(s) returned 0.000 sec / 0.000 sec

d) Retrieve the top 5 customers by total purchase amount.

The screenshot shows the MySQL Workbench interface with a query editor and results grid.

Query Editor:

```
118 • SELECT
119     c.NAME AS Customer_Name,
120     COUNT(o.ORDER_ID) AS Total_Orders,
121     SUM(o.QUANTITY) AS Total_Items_Purchased,
122     ROUND(SUM(o.QUANTITY * p.PRICE), 2) AS Total_Purchase_Amount
123
124     FROM Customers c
125     JOIN Orders o ON c.CUSTOMER_ID = o.CUSTOMER_ID
126     JOIN Products p ON o.PRODUCT_ID = p.PRODUCT_ID
127
128     GROUP BY c.CUSTOMER_ID, c.NAME
129
130     ORDER BY Total_Purchase_Amount DESC
```

Result Grid:

Customer_Name	Total_Orders	Total_Items_Purchased	Total_Purchase_Amount
Ishaan Mehra	1	1	85000.00
Zara Patel	2	3	78000.00
Aarav Sharma	2	3	38000.00
Anaya Reddy	1	1	35000.00
Aanya Malhotra	1	3	8400.00

Action Output:

#	Time	Action	Message	Duration / Fetch
41	16:13:34	SELECT PRODUCT_ID, PRODUCT_NAME, CATEGORY, PRICE, STOCK_QTY FROM Products	0 row(s) returned	0.000 sec / 0.000 sec
42	16:14:48	SELECT p.PRODUCT_NAME, p.CATEGORY, SUM(o.QUANTITY) AS Total_Quantity FROM Orders o JOIN Products p ON o.PRODUCT_ID = p.PRODUCT_ID GROUP BY p.PRODUCT_ID	10 row(s) returned	0.000 sec / 0.000 sec
43	16:16:13	SELECT c.NAME AS Customer_Name, COUNT(o.ORDER_ID) AS Total_Orders, SUM(o.QUANTITY) AS Total_Items_Purchased, ROUND(SUM(o.QUANTITY * p.PRICE), 2) AS Total_Purchase_Amount FROM Customers c JOIN Orders o ON c.CUSTOMER_ID = o.CUSTOMER_ID JOIN Products p ON o.PRODUCT_ID = p.PRODUCT_ID GROUP BY c.CUSTOMER_ID, c.NAME ORDER BY Total_Purchase_Amount DESC	5 row(s) returned	0.016 sec / 0.000 sec

e) Find customers who place orders in at least two different product categories.

There are no customers with different categories ordered.

The screenshot shows the MySQL Workbench interface. In the top-left corner, the title bar says "MySQL Workbench" and "Sample x". Below it is a menu bar with File, Edit, View, Query, Database, Server, Tools, Scripting, Help. The main area has tabs for JOINs, Exercise_JOINS, Task_1, Task 2, and Task 3*. The Task 3* tab is active, displaying the following SQL query:

```
130  -- Find customers who placed orders in at least two different product categories:
131  • SELECT
132      c.NAME AS Customer_Name,
133      COUNT(DISTINCT p.CATEGORY) AS Different_Categories_Ordered
134  FROM Customers c
135  JOIN Orders o ON c.CUSTOMER_ID = o.CUSTOMER_ID
136  JOIN Products p ON o.PRODUCT_ID = p.PRODUCT_ID
137  GROUP BY c.CUSTOMER_ID, c.NAME
138  HAVING COUNT(DISTINCT p.CATEGORY) >= 2
139  ORDER BY Different_Categories_Ordered DESC;
```

Below the query is a "Result Grid" table with columns "Customer_Name" and "Different_Categories_Ordered". The table is currently empty. To the right of the result grid is a vertical toolbar with icons for Result Grid, Form Editor, and Field Types. At the bottom of the interface, there's a "Result 6" tab, an "Output" section showing action logs, and a system status bar indicating "Finance headline US Trade Probe..." and the date/time "4:17 PM 7/17/2025".

Analytics:

a) Find the month with the highest total sales.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 Task 2 Task 3*

```

146     YEAR(ORDER_DATE) AS Year,
147     COUNT(ORDER_ID) AS Total_Orders,
148     SUM(o.QUANTITY) AS Total_Items_Sold,
149     ROUND(SUM(o.QUANTITY * p.PRICE), 2) AS Total_Sales
150   FROM Orders o
151   JOIN Products p ON o.PRODUCT_ID = p.PRODUCT_ID
152   GROUP BY YEAR(ORDER_DATE), MONTH(ORDER_DATE), MONTHNAME(ORDER_DATE)
153   ORDER BY Total_Sales DESC
154
155   LIMIT 1;
156

```

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

Month	Year	Total_Orders	Total_Items_Sold	Total_Sales
April	2025	2	3	91000.00

No object selected

Administration Schemas Information

Result 7 ×

Action Output

#	Time	Action	Message	Duration / Fetch
44	16:17:07	SELECT c.NAME AS Customer_Name, COUNT(DISTINCT p.CATEGORY) AS Diff...	0 row(s) returned	0.000 sec / 0.000 sec
45	16:19:35	SELECT MONTHNAME(ORDER_DATE) AS Month, YEAR(ORDER_DATE) AS Y...	1 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

b) Identify products with no orders in the last 6 months.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 Task 2 Task 3*

```

155
156 -- Identify products with no orders in the last 6 months:
157 • SELECT
158   p.PRODUCT_ID,
159   p.PRODUCT_NAME,
160   p.CATEGORY,
161   p STOCK,
162   p.PRICE,
163   MAX(o.ORDER_DATE) AS Last_Order_Date
164   FROM Products p
165   LEFT JOIN Orders o ON p.PRODUCT_ID = o.PRODUCT_ID
166   WHERE o.ORDER_DATE IS NULL
167   OR o.ORDER_DATE < DATE_SUB(CURRENT_DATE, INTERVAL 6 MONTH)
168   GROUP BY p.PRODUCT_ID, p.PRODUCT_NAME, p.CATEGORY, p STOCK, p.PRICE
169   ORDER BY p.CATEGORY, p.PRODUCT_NAME;
170

```

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

PRODUCT_ID	PRODUCT_NAME	CATEGORY	STOCK	PRICE	Last_Order_Date
1	Smart Watch Pro	Electronics	50	15000.00	2025-01-15

No object selected

Administration Schemas Information

Result 9 ×

Action Output

#	Time	Action	Message	Duration / Fetch
46	16:21:15	SELECT p.PRODUCT_ID, p.PRODUCT_NAME, p.CATEGORY, p STOCK, ...	1 row(s) returned	0.000 sec / 0.000 sec
47	16:21:57	SELECT p.PRODUCT_ID, p.PRODUCT_NAME, p.CATEGORY, p STOCK, ...	1 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

c) Retrieve customers who have never placed an order.

MySQL Workbench

Sample

File Edit View Query Database Server Tools Scripting Help

JOINS Excercise_JOINS Task_1 Task 2 Task 3*

Limit to 1000 rows

172 g) Retrieve customers who have never placed an order:

```
173
174
175 -- Retrieve customers who have never placed an order:
176 SELECT
177     c.CUSTOMER_ID,
178     c.NAME,
179     c.EMAIL,
180     c.PHONE,
181     c.ADDRESS
182 FROM Customers c
183 LEFT JOIN Orders o ON c.CUSTOMER_ID = o.CUSTOMER_ID
184 WHERE o.ORDER_ID IS NULL;
```

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

CUSTOMER_ID	NAME	EMAIL	PHONE	ADDRESS

Result 10 x Read Only

Action Output

#	Time	Action	Message	Duration / Fetch
47	16:21:57	SELECT	p.PRODUCT_ID, p.PRODUCT_NAME, p.CATEGORY, p STOCK, ... 1 row(s) returned	0.000 sec / 0.000 sec
48	16:23:11	SELECT	c.CUSTOMER_ID, c.NAME, c.EMAIL, c.PHONE, c.ADDRESS F... 0 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

30°C Haze 4:23 PM 7/17/2025

d) Calculate the average order value across all orders.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 Task 2 Task 3*

SCHEMAS

- flightdb
- food_db
- librarymanagement
- moviesdb
- payroll_database
- query_db
- sakila
- schoolinformation
- sys
- world

```

188 • SELECT
189     ROUND(AVG(order_total), 2) AS Average_Order_Value
190     FROM (
191         SELECT
192             o.ORDER_ID,
193             SUM(o.QUANTITY * p.PRICE) as order_total
194             FROM Orders o
195             JOIN Products p ON o.PRODUCT_ID = p.PRODUCT_ID
196             GROUP BY o.ORDER_ID
197     ) AS order_totals;
198

```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Result Grid | Form Editor | Read Only

Average_Order_Value
22083.33

Administration Schemas

No object selected

Information

Result 15 x

Action Output

#	Time	Action	Message	Duration / Fetch
51	16:23:45	SELECT CUSTOMER_ID, NAME, EMAIL, PHONE, ADDRESS FROM Cus...	0 row(s) returned	0.000 sec / 0.000 sec
52	16:24:02	SELECT c.CUSTOMER_ID, c.NAME, c.EMAIL, c.PHONE, c.ADDRESS F...	0 row(s) returned	0.000 sec / 0.000 sec
53	16:24:58	SELECT ROUND(AVG(order_total), 2) AS Average_Order_Value FROM (1 row(s) returned	0.000 sec / 0.000 sec
54	16:25:18	SELECT ROUND(MIN(order_total), 2) AS Min_Order_Value, ROUND(AVG(order_t...	1 row(s) returned	0.000 sec / 0.000 sec
55	16:25:41	SELECT ROUND(AVG(order_total), 2) AS Average_Order_Value FROM (1 row(s) returned	0.000 sec / 0.000 sec

Object Info Session

Task 4

Project: Movie Rental Analysis System

Database Creation:

The screenshot shows the MySQL Workbench interface. In the top navigation bar, 'File', 'Edit', 'View', 'Query', 'Database', 'Server', 'Tools', 'Scripting', and 'Help' are visible. Below the menu is a toolbar with various icons. The 'Navigator' pane on the left lists several databases, including 'movierental' which is currently selected. The main SQL editor tab is titled 'SQL File 7' and contains the following code:

```
1 CREATE DATABASE MovieRental;
2
3 USE MovieRental;
```

The 'Output' pane at the bottom shows the execution results:

#	Time	Action	Message	Duration / Fetch
1	18:56:24	CREATE DATABASE MovieRental	1 row(s) affected	0.016 sec
2	18:56:31	USE MovieRental	0 row(s) affected	0.000 sec

The status bar at the bottom right indicates the system temperature is 30°C, the weather is Haze, the time is 6:56 PM, and the date is 7/17/2025.

Create table rental data with columns:

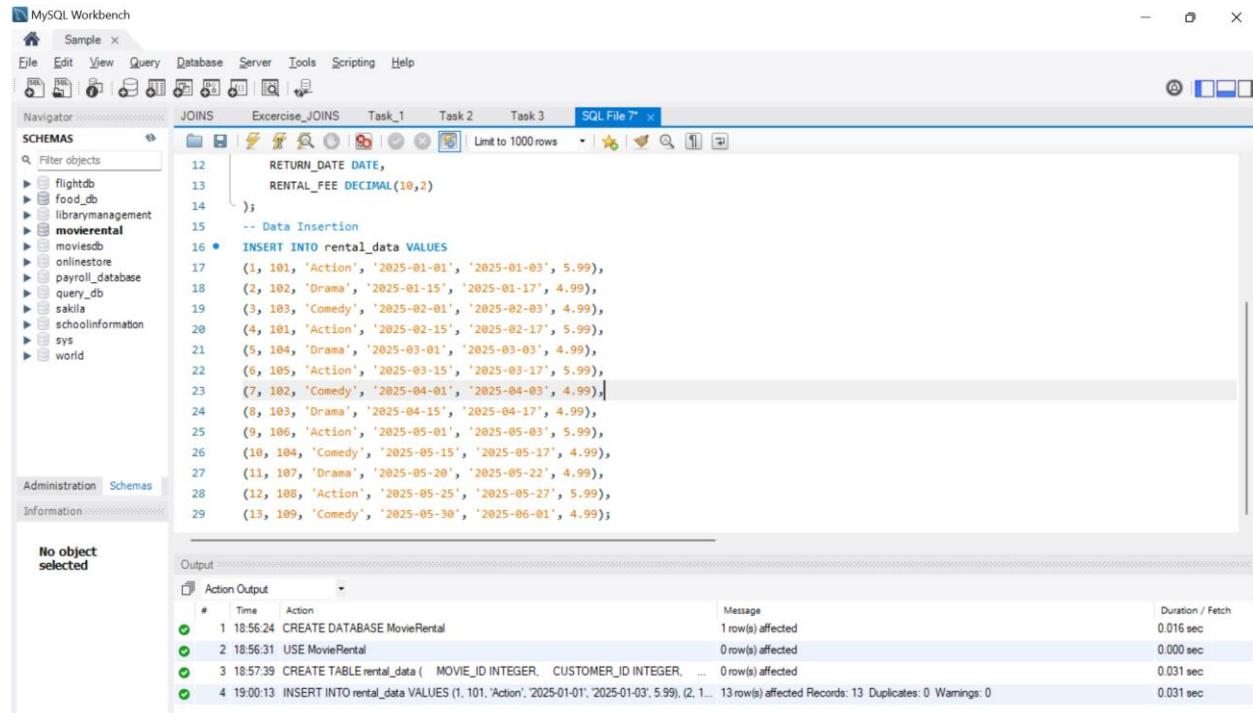
The screenshot shows the MySQL Workbench interface. The 'Navigator' pane on the left shows the 'movierental' database is selected. The main SQL editor tab is titled 'SQL File 7' and contains the following code:

```
1 CREATE DATABASE MovieRental;
2
3 USE MovieRental;
4
5 CREATE TABLE rental_data (
6     MOVIE_ID INTEGER,
7     CUSTOMER_ID INTEGER,
8     GENRE VARCHAR(50),
9     RENTAL_DATE DATE,
10    RETURN_DATE DATE,
11    RENTAL_FEE DECIMAL(10,2)
12);
```

The 'Output' pane at the bottom shows the execution results:

#	Time	Action	Message	Duration / Fetch
1	18:56:24	CREATE DATABASE MovieRental	1 row(s) affected	0.016 sec
2	18:56:31	USE MovieRental	0 row(s) affected	0.000 sec
3	18:57:39	CREATE TABLE rental_data (MOVIE_ID INTEGER, CUSTOMER_ID INTEGER, ...)	0 row(s) affected	0.031 sec

Data Creation: Added 13 sample records:



The screenshot shows the MySQL Workbench interface. The main window displays a SQL editor with the following code:

```
12     RETURN_DATE DATE,
13     RENTAL_FEE DECIMAL(10,2)
14 );
15 -- Data Insertion
16 • INSERT INTO rental_data VALUES
17 (1, 101, 'Action', '2025-01-01', '2025-01-03', 5.99),
18 (2, 102, 'Drama', '2025-01-15', '2025-01-17', 4.99),
19 (3, 103, 'Comedy', '2025-02-01', '2025-02-03', 4.99),
20 (4, 101, 'Action', '2025-02-15', '2025-02-17', 5.99),
21 (5, 104, 'Drama', '2025-03-01', '2025-03-03', 4.99),
22 (6, 105, 'Action', '2025-03-15', '2025-03-17', 5.99),
23 (7, 102, 'Comedy', '2025-04-01', '2025-04-03', 4.99),
24 (8, 103, 'Drama', '2025-04-15', '2025-04-17', 4.99),
25 (9, 106, 'Action', '2025-05-01', '2025-05-03', 5.99),
26 (10, 104, 'Comedy', '2025-05-15', '2025-05-17', 4.99),
27 (11, 107, 'Drama', '2025-05-20', '2025-05-22', 4.99),
28 (12, 108, 'Action', '2025-05-25', '2025-05-27', 5.99),
29 (13, 109, 'Comedy', '2025-05-30', '2025-06-01', 4.99);
```

The Output pane shows the execution results:

#	Time	Action	Message	Duration / Fetch
1	18:56:24	CREATE DATABASE MovieRental	1 row(s) affected	0.016 sec
2	18:56:31	USE MovieRental	0 row(s) affected	0.000 sec
3	18:57:39	CREATE TABLE rental_data (MOVIE_ID INTEGER, CUSTOMER_ID INTEGER, ...) ENGINE=InnoDB	0 row(s) affected	0.031 sec
4	19:00:13	INSERT INTO rental_data VALUES (1, 101, 'Action', '2025-01-01', '2025-01-03', 5.99), (2, 102, 'Drama', '2025-01-15', '2025-01-17', 4.99), (3, 103, 'Comedy', '2025-02-01', '2025-02-03', 4.99), (4, 101, 'Action', '2025-02-15', '2025-02-17', 5.99), (5, 104, 'Drama', '2025-03-01', '2025-03-03', 4.99), (6, 105, 'Action', '2025-03-15', '2025-03-17', 5.99), (7, 102, 'Comedy', '2025-04-01', '2025-04-03', 4.99), (8, 103, 'Drama', '2025-04-15', '2025-04-17', 4.99), (9, 106, 'Action', '2025-05-01', '2025-05-03', 5.99), (10, 104, 'Comedy', '2025-05-15', '2025-05-17', 4.99), (11, 107, 'Drama', '2025-05-20', '2025-05-22', 4.99), (12, 108, 'Action', '2025-05-25', '2025-05-27', 5.99), (13, 109, 'Comedy', '2025-05-30', '2025-06-01', 4.99);	13 row(s) affected Records: 13 Duplicates: 0 Warnings: 0	0.031 sec

OLAP Operations:

a) Drill Down: Analyze rentals from genre to individual movie level.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 Task 2 Task 3 SQL File 7

Filter objects

SCHEMAS

- flightdb
- food_db
- librarymanagement
- movierental**
- moviesdb
- onlinestore
- payroll_database
- query_db
- sakila
- schoolinformation
- sys
- world

Administration Schemas Information

No object selected

```

32 -- Drill Down (from genre to individual movie level):
33
34 -- Individual movies within genre
35 • SELECT GENRE, MOVIE_ID, COUNT(*) as rental_count, SUM(RENTAL_FEE) as total_fee
36 FROM rental_data
37 GROUP BY GENRE, MOVIE_ID
38 ORDER BY GENRE, MOVIE_ID;

```

Result Grid | Filter Rows: Export: Wrap Cell Content: Result Grid Form Editor Field Types Query Stats

GENRE	MOVIE_ID	rental_count	total_fee
Action	1	1	5.99
Action	4	1	5.99
Action	6	1	5.99
Action	9	1	5.99
Action	12	1	5.99
Comedy	3	1	4.99
Comedy	7	1	4.99
Comedy	10	1	4.99
Comedy	13	1	4.99
Drama	2	1	4.99
Drama	5	1	4.99
Drama	8	1	4.99
Drama	11	1	4.99

Result 2 x Read Only

Action Output

#	Time	Action
5	19:02:48	SELECT GENRE,COUNT(*) as rental_count, SUM(RENTAL_FEE) as total_fee FROM r... 3 rows returned

Object Info Session Duration / Fetch 0.000 sec / 0.000 sec

b) Rollup: Summarize total rental fees by genre and then overall.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 Task 2 Task 3 SQL File 7*

Navigator

SCHEMAS

- flighthdb
- food_db
- librarymanagement
- moviereental**
- moviesdb
- onlinestore
- payroll_database
- query_db
- sakila
- schoolinformation
- sys
- world

Result Grid | Filter Rows: Export: Wrap Cell Content: Result Grid Form Editor Field Types

```

39
40    -- Rollup
41 •     SELECT GENRE, SUM(RENTAL_FEE) as total_fee
42     FROM rental_data
43     GROUP BY GENRE WITH ROLLUP;

```

GENRE	total_fee
Action	29.95
Comedy	19.96
Drama	19.96
HULL	69.87

Administration Schemas Information

No object selected

Result 3 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
3	18:57:39	CREATE TABLE rental_data (MOVIE_ID INTEGER, CUSTOMER_ID INTEGER, ...) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci	0 row(s) affected	0.031 sec
4	19:00:13	INSERT INTO rental_data VALUES (1, 101, 'Action', '2025-01-01', '2025-01-03', 5.99), (2, 102, 'Action', '2025-01-01', '2025-01-03', 5.99), (3, 103, 'Comedy', '2025-01-01', '2025-01-03', 4.99), (4, 104, 'Drama', '2025-01-01', '2025-01-03', 5.99), (5, 105, 'Action', '2025-01-01', '2025-01-03', 5.99), (6, 106, 'Action', '2025-01-01', '2025-01-03', 5.99), (7, 107, 'Action', '2025-01-01', '2025-01-03', 5.99), (8, 108, 'Action', '2025-01-01', '2025-01-03', 5.99), (9, 109, 'Action', '2025-01-01', '2025-01-03', 11.98), (10, 110, 'Action', '2025-01-01', '2025-01-03', 29.95), (11, 111, 'Action', '2025-01-01', '2025-01-03', 29.95), (12, 112, 'Action', '2025-01-01', '2025-01-03', 29.95)	13 row(s) affected Records: 13 Duplicates: 0 Warnings: 0	0.031 sec
5	19:02:48	SELECT GENRE, COUNT(*) as rental_count, SUM(RENTAL_FEE) as total_fee FROM rental_data GROUP BY GENRE	3 row(s) returned	0.000 sec / 0.000 sec
6	19:03:54	SELECT GENRE, MOVIE_ID, COUNT(*) as rental_count, SUM(RENTAL_FEE) as total_fee FROM rental_data GROUP BY GENRE, MOVIE_ID	13 row(s) returned	0.000 sec / 0.000 sec
7	19:05:07	SELECT GENRE, SUM(RENTAL_FEE) as total_fee FROM rental_data GROUP BY GENRE	4 row(s) returned	0.000 sec / 0.000 sec

Object Info Session Read Only

c) Cube: Analyze total rental fees across combinations of genre, rental date, and customer.

Query-

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 Task 2 Task 3 SQL File 7*

Navigator

SCHEMAS

- flighthdb
- food_db
- librarymanagement
- moviereental**
- moviesdb
- onlinestore
- payroll_database
- query_db
- sakila
- schoolinformation
- sys
- world

Result Grid | Filter Rows: Export: Wrap Cell Content: Result Grid Form Editor Field Types

```

45    -- cube
46 •     SELECT
47         GENRE,
48         YEAR(RENTAL_DATE) as year,
49         MONTH(RENTAL_DATE) as month,
50         CUSTOMER_ID,
51         SUM(RENTAL_FEE) as total_fee
52     FROM rental_data
53     GROUP BY GENRE, YEAR(RENTAL_DATE), MONTH(RENTAL_DATE), CUSTOMER_ID WITH ROLLUP;

```

GENRE	year	month	CUSTOMER_ID	total_fee
Action	2025	1	101	5.99
Action	2025	1	HULL	5.99
Action	2025	2	101	5.99
Action	2025	2	HULL	5.99
Action	2025	3	105	5.99
Action	2025	3	HULL	5.99
Action	2025	5	106	5.99
Action	2025	5	108	5.99
Action	2025	5	HULL	11.98
Action	2025	HULL	HULL	29.95
Action	HULL	HULL	HULL	29.95
Comedy	2025	2	103	4.99
Comedy	2025	2	HULL	4.99

Administration Schemas Information

No object selected

Result 4 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
2	18:56:31	USE MovieRental	0 row(s) affected	0.000 sec

Object Info Session Read Only

Output-

The screenshot shows the MySQL Workbench interface with a query editor window titled "SQL File 7". The query is:

```
48     YEAR(RENTAL_DATE) as year,
49     MONTH(RENTAL_DATE) as month,
50     CUSTOMER_ID,
51     SUM(RENTAL_FEE) as total_fee
52 FROM rental_data
```

The results grid displays the following data:

GENRE	year	month	CUSTOMER_ID	total_fee
Action	2025	1	101	5.99
Action	2025	1	NULL	5.99
Action	2025	2	101	5.99
Action	2025	2	NULL	5.99
Action	2025	3	105	5.99
Action	2025	3	NULL	5.99
Action	2025	5	106	5.99
Action	2025	5	108	5.99
Action	2025	5	NULL	11.98
Action	2025	NULL	NULL	29.95
Action	2025	NULL	NULL	29.95
Comedy	2025	2	103	4.99
Comedy	2025	2	NULL	4.99
Comedy	2025	4	102	4.99
Comedy	2025	4	NULL	4.99
Comedy	2025	5	104	4.99
Comedy	2025	5	109	4.99
Comedy	2025	5	NULL	9.98
Comedy	2025	NULL	NULL	19.96
Comedy	NULL	NULL	NULL	19.96
Drama	2025	1	102	4.99

The results grid has 4 columns: GENRE, year, month, and CUSTOMER_ID. The total_fee column is explicitly labeled in the query. The results show various genres (Action, Comedy, Drama) across different years (2025) and months (1, 2, 3, 5). Some rows have NULL values for certain fields.

The screenshot shows the MySQL Workbench interface with a query editor window titled "SQL File 7". The query is identical to the one in the previous screenshot:

```
48     YEAR(RENTAL_DATE) as year,
49     MONTH(RENTAL_DATE) as month,
50     CUSTOMER_ID,
51     SUM(RENTAL_FEE) as total_fee
52 FROM rental_data
```

The results grid displays the following data:

GENRE	year	month	CUSTOMER_ID	total_fee
Action	2025	2	103	4.99
Action	2025	2	NULL	4.99
Action	2025	4	102	4.99
Action	2025	4	NULL	4.99
Action	2025	5	104	4.99
Action	2025	5	109	4.99
Action	2025	5	NULL	9.98
Action	2025	NULL	NULL	19.96
Action	NULL	NULL	NULL	19.96
Comedy	2025	1	102	4.99
Comedy	2025	1	NULL	4.99
Comedy	2025	3	104	4.99
Comedy	2025	3	NULL	4.99
Comedy	2025	4	103	4.99
Comedy	2025	4	NULL	4.99
Comedy	2025	5	107	4.99
Comedy	2025	5	NULL	4.99
Comedy	2025	NULL	NULL	19.96
Comedy	NULL	NULL	NULL	19.96
Drama	2025	1	102	4.99
Drama	2025	1	NULL	4.99
Drama	2025	3	104	4.99
Drama	2025	3	NULL	4.99
Drama	2025	4	103	4.99
Drama	2025	4	NULL	4.99
Drama	2025	5	107	4.99
Drama	2025	5	NULL	4.99
Drama	2025	NULL	NULL	19.96
Drama	NULL	NULL	NULL	19.96
NULL	NULL	NULL	NULL	69.87

This result set is identical to the one in the first screenshot, showing the same genres, years, months, and total fees. The last row is a new addition, showing a total fee of 69.87 for entries where all four primary key fields (GENRE, year, month, CUSTOMER_ID) are NULL.

d) Slice: Extract rentals only from the 'Action' genre.

The screenshot shows the MySQL Workbench interface. In the SQL Editor tab, a query is written to extract rentals from the 'Action' genre:

```
43 GROUP BY GENRE WITH ROLLUP;
44
45 -- cube
46 • SELECT
47   GENRE,
48   YEAR(RENTAL_DATE) AS year,
49   MONTH(RENTAL_DATE) AS month,
50   CUSTOMER_ID,
51   SUM(RENTAL_FEE) AS total_fee
52 FROM rental_data
53 GROUP BY GENRE, YEAR(RENTAL_DATE), MONTH(RENTAL_DATE), CUSTOMER_ID WITH ROLLUP;
54
55 -- SLICE
```

The Result Grid displays the following data:

MOVIE_ID	CUSTOMER_ID	GENRE	RENTAL_DATE	RETURN_DATE	RENTAL_FEE
1	101	Action	2025-01-01	2025-01-03	5.99
4	101	Action	2025-02-15	2025-02-17	5.99
6	105	Action	2025-03-15	2025-03-17	5.99
9	106	Action	2025-05-01	2025-05-03	5.99
12	108	Action	2025-05-25	2025-05-27	5.99

In the Output pane, two actions are listed:

- Action 8 at 19:06:15: SELECT GENRE, YEAR(RENTAL_DATE) as year, MONTH(RENTAL_DATE) a... 31 row(s) returned Duration / Fetch: 0.000 sec / 0.000 sec
- Action 9 at 19:10:50: SELECT * FROM rental_data WHERE GENRE = 'Action' LIMIT 0, 1000 5 row(s) returned Duration / Fetch: 0.000 sec / 0.000 sec

e) Dice: Extract rentals where GENRE = 'Action' or 'Drama' and RENTAL_DATE is in the last 3 months.

MySQL Workbench

File Edit View Query Database Server Tools Scripting Help

JOINS Exercise_JOINS Task_1 Task 2 Task 3 SQL File 7*

SCHEMAS

- flighthdb
- food_db
- librarymanagement
- moviereental**
- moviesdb
- onlinestore
- payroll_database
- query_db
- sakila
- schoolinformation
- sys
- world

Filter objects

```
56 • SELECT *
57   FROM rental_data
58   WHERE GENRE = 'Action';
59
60   -- Dice
61
62   -- Assuming current date is 2025-05-15
63 • SELECT *
64   FROM rental_data
65   WHERE GENRE IN ('Action', 'Drama')
66   AND RENTAL_DATE >= '2025-02-15'
67   AND RENTAL_DATE <= '2025-05-15'
68   ORDER BY RENTAL_DATE;
```

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

MOVIE_ID	CUSTOMER_ID	GENRE	RENTAL_DATE	RETURN_DATE	RENTAL_FEE
4	101	Action	2025-02-15	2025-02-17	5.99
5	104	Drama	2025-03-01	2025-03-03	4.99
6	105	Action	2025-03-15	2025-03-17	5.99
8	103	Drama	2025-04-15	2025-04-17	4.99
9	106	Action	2025-05-01	2025-05-03	5.99

No object selected

rental_data 7 ×

Output

Action Output

#	Time	Action	Message	Duration / Fetch
10	19:11:59	SELECT * FROM rental_data WHERE GENRE IN ('Action', 'Drama')	5 row(s) returned	0.000 sec / 0.000 sec
11	19:12:03	SELECT * FROM rental_data WHERE GENRE IN ('Action', 'Drama')	5 row(s) returned	0.000 sec / 0.000 sec

Object Info Session