

## Assignment No. 7

**Aim:** Implement nested sub queries. Perform a test for set membership (in, not in), set comparison (<some, >=some, <all etc.) and set cardinality (unique, not unique).

**Objective:**

- To learn different types of Joins.
- To implement different sub queries.

**Theory :**

MySQL JOINS are used with SELECT statement. It is used to retrieve data from multiple tables. It is performed whenever you need to fetch records from two or more tables.

There are three types of MySQL joins:

- MySQL INNER JOIN (or sometimes called simple join)
- MySQL LEFT OUTER JOIN (or sometimes called LEFT JOIN)
- MySQL RIGHT OUTER JOIN (or sometimes called RIGHT JOIN)

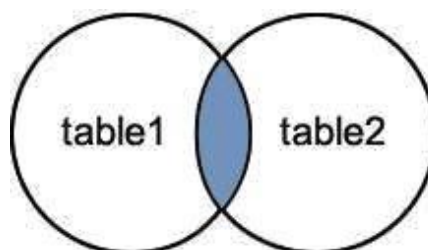
### MySQL Inner JOIN (Simple Join)

The MySQL INNER JOIN is used to return all rows from multiple tables where the join condition is satisfied. It is the most common type of join.

**Syntax:**

```
SELECT columns  
FROM table1  
  
INNER JOIN table2  
ON table1.column = table2.column;
```

**Image representation:**



### Let's take an example:

Consider two tables "officers" and "students", having the following data.

#### Execute the following query:

```
SELECT officers.officer_name, officers.address, students.course_name
FROM officers
INNER JOIN students
ON officers.officer_id = students.student_id;
```

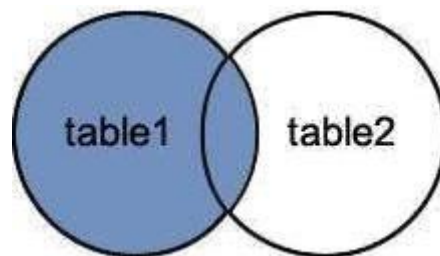
### MySQL Left Outer Join

The LEFT OUTER JOIN returns all rows from the left hand table specified in the ON condition and only those rows from the other table where the join condition is fulfilled.

#### Syntax:

```
SELECT columns
FROM table1
LEFT [OUTER] JOIN table2
ON table1.column = table2.column;
```

#### Image representation:



### Let's take an example:

Consider two tables "officers" and "students", having the following data.

#### Execute the following query:

```
SELECT officers.officer_name, officers.address, students.course_name
FROM officers
LEFT JOIN students
ON officers.officer_id = students.student_id;
```

### MySQL Right Outer Join

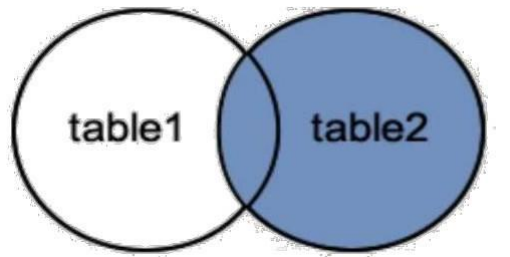
The MySQL Right Outer Join returns all rows from the RIGHT-hand table specified in the ON condition and only those rows from the other table where the join condition is fulfilled.

#### Syntax:

```
SELECT columns
```

```
FROM table1  
RIGHT [OUTER] JOIN table2  
ON table1.column = table2.column;
```

**Image representation:**



**Let's take an example:**

Consider two tables "officers" and "students", having the following data.

**Execute the following query:**

```
SELECT      officers.officer_name, officers.address,  
            students.course_name,  
            students.student_name  
FROM officers  
RIGHT JOIN students  
ON officers.officer_id = students.student_id;
```

**SPECIAL OPERATOR:**

**MySQL IN Condition**

The MySQL IN condition is used to reduce the use of multiple OR conditions in a SELECT, INSERT, UPDATE and DELETE statement.

**Syntax:**

expression IN (value1, value2,.....value\_n);

**Parameters:**

**expression:** It specifies a value to test.

**value1, value2, .....or value\_n:** These are the values to test against expression. If any of these values matches expression, then the IN condition will evaluate to true. This is a quick method to test if any one of the values matches expression.

**Execute the following query:**

```
SELECT *  
  
FROM officers  
WHERE officer_name IN ('Ajeet', 'Vimal', 'Deepika');
```

## **MySQL NOT Condition**

The MySQL NOT condition is opposite of MySQL IN condition. It is used to negate a condition in a SELECT, INSERT, UPDATE or DELETE statement.

### **Syntax:**

NOT condition

### **Parameter:**

**condition:** It specifies the conditions that you want to negate.

## **MySQL NOT Operator with IN condition**

Consider a table "officers", having the following data.

**Execute the following query:**

```
SELECT *  
FROM officers  
WHERE officer_name NOT IN ('Ajeet','Vimal','Deepika');
```

## **MySQL IS NULL Condition**

MySQL IS NULL condition is used to check if there is a NULL value in the expression. It is used with SELECT, INSERT, UPDATE and DELETE statement.

### **Syntax:**

expression IS NULL

### **Parameter:**

**expression:** It specifies a value to test if it is NULL

**Execute the following query:**

```
SELECT *  
FROM officers  
WHERE officer_name IS NULL;
```

## **MySQL IS NOT NULL Condition**

MySQL IS NOT NULL condition is used to check the NOT NULL value in the expression. It is used with SELECT, INSERT, UPDATE and DELETE statements.

### **Syntax:**

expression IS NOT NULL

**Parameter:**

**expression:** It specifies a value to test if it is not NULL value.

**Execute the following query:**

```
SELECT *
FROM officers
WHERE officer_name IS NOT NULL;
```

**SET OPERATORS:**

The Set operator combines the result of 2 queries into a single result. The following are the operators:

☐ Union ☐

Union all

☐ Intersect

☐ Minus

**LAB PRACTICE ASSIGNMENT:**

**Consider the following table structure for this assignment:**

- Location(Location\_Id integer, Reginal\_Group varchar(20))
- Department (Department\_Id, Name, Location\_Id)
- Job(Job\_Id Integer,Function Varchar(30))
- Employee(Employee\_Id, Lastname ,Firstname,Middlename, Job\_Id, Manager\_Id, Hiredate, Salary, Department\_Id)
- Loan(Employee\_Id, Firstname , Loan\_Amount)

**LOCATION TABLE**

LOCATION_IDREGINAL_GROUP	
122	New York
123	Dallas
124	Chicago
167	Boostan

**DEPARTMENT TABLE**

DEPARTMENT_ID	NAME	LOCATION_ID
10	Accounting	122

20	Research	124
30	Sale	123
40	Operation	164

### JOB TABLE

JOB_ID	FUNCTION
667	Cleark
668	Staff
669	Analyst
670	Saleperson
671	Manager
672	President

### EMPLOYEE TABLE

EMPL OYEE_ ID	LAST NAM E	FIRS TNA ME	MIDD LENA ME	JO B_I D	MANA GER_I D	HIR EDA TE	SAL AR Y	DEPART MENT_I D
7369	Smith	Jon	Q	667	7902	17- DEC- 84	800	10
7499	Allen	Kevin	J	670	7698	20- FEB- 85	1600	20
7505	Doyle	Jean	K	671	7839	04- APR- 85	2850	20
7506	Dennis	Lynn	S	671	7839	15- MAY- 85	2750	30
7507	Baker	Leslie	D	671	7839	10- JUN- 85	2200	40
7521	wark	cynthia	D	670	7698	22- FEB- 85	1250	10

**Perform the following queries on the above table:**

- 1) Perform all types of JOIN operations on Employee and Loan tables.

- 2) Perform all types of set operations on Employee and Loan tables.
- 3) Find out no.of employees working in “Sales” department
- 4) Find out the employees who are not working in department 10 or 30.
- 5) List out employee id, last name in descending order based on the salary column.
- 6) How many employees who are working in different departments wise in the organization
- 7) List out the department id having at least four employees
- 8) Display the employee who got the maximum salary.
- 9) Update the employees’ salaries, who are working as Clerk on the basis of 10%.
- 10) Delete the employees who are working in accounting department.
- 11) Find out whose department has not employees.
- 12) List out the department wise maximum salary, minimum salary, average salary of the employees
- 13) How many employees who are joined in 1985.
- 14)** Display the employees who are working in “New York”
- 15) List our employees with their department names

### **Conclusion:**

We have implemented join, set operations, set cardinalities and nested sub queries.

### **Output:**

#### **1. CREATE TABLE:**

##### **a) LOCATION:**

```
CREATE TABLE "Location" ( "Location_ID"  
    INTEGER NOT NULL, "Reginal_Group"  
    VARCHAR(20), PRIMARY  
    KEY("Location_ID")  
);
```

DB Browser for SQLite - D:\Akash Files\A Sem\Sem 5\Practical Assignments\DBMS\Assignment 7\Assignment7.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragmas Execute SQL

SQL 1

```
1 SELECT * FROM Location;
```

	Location_ID	Reginal_Group
1	122	New York
2	123	Dallas
3	124	Chicago
4	164	Boostan

Result: 4 rows returned in 113ms  
At line 1:  
SELECT \* FROM Location;

Edit Database Cell

Mode: Text Import Export Set as NULL

NULL

Type of data currently in cell: NULL  
0 byte(s)

SQL Log

Show SQL submitted by Application Clear

```
292 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM "
293 SELECT "_rowid_",* FROM "main"."Employee" ORDER
294 INSERT INTO "main"."Employee" ("Employee_ID","Le
295 INSERT INTO "main"."Employee"
296 ("Employee_ID", "LastName", "FirstName", "Middl
297 VALUES (7521, 'wark', 'cynthia', 'D', 670, 7698
298 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM "
299 SELECT "_rowid_",* FROM "main"."Employee" ORDER
300 SELECT "_rowid_",* FROM "main"."Employee" ORDER
301 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM "
302 SELECT COUNT(*) FROM (SELECT * FROM Location);
303 SELECT * FROM Location LIMIT 0, 49999;
304 PRAGMA database_list;
```

SQL Log Plot DB Schema Remote UTF-8

**b) JOB:**  
 CREATE TABLE "Job" (  
     "Job\_ID" INTEGER NOT NULL,  
     "Function" TEXT NOT NULL,  
     PRIMARY KEY("Job\_ID")  
 );

DB Browser for SQLite - D:\Akash Files\A Sem\Sem 5\Practical Assignments\DBMS\Assignment 7\Assignment7.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragmas Execute SQL

SQL 1

```
1 SELECT * FROM Job;
```

	Job_ID	Function
1	667	
2	668	Staff
3	669	Analyst
4	670	Saleperson
5	671	Manager
6	672	President

Result: 6 rows returned in 24ms  
At line 1:  
SELECT \* FROM Job;

Edit Database Cell

Mode: Text Import Export Set as NULL

NULL

Type of data currently in cell: NULL  
0 byte(s)

SQL Log

Show SQL submitted by Application Clear

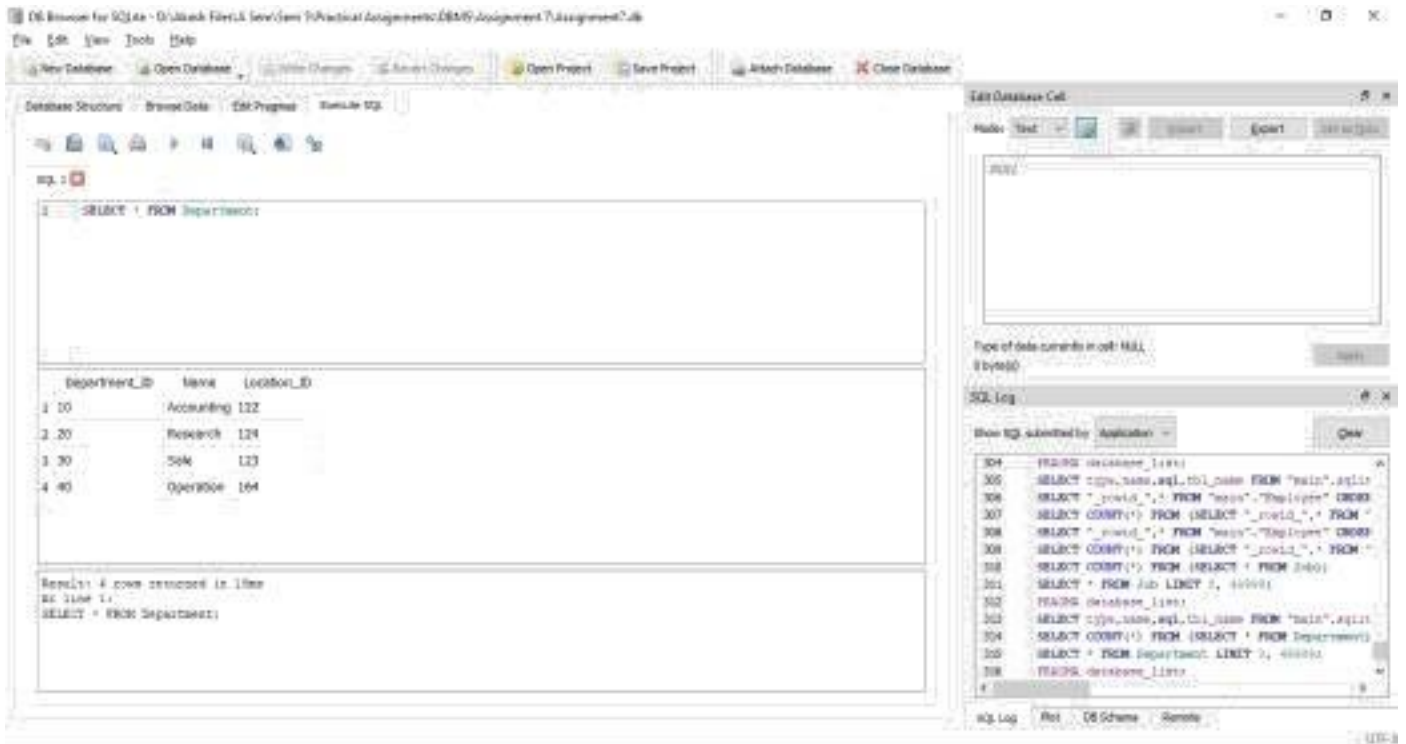
```
300 SELECT "_rowid_",* FROM "main"."Employee" ORDER
301 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM "
302 SELECT COUNT(*) FROM (SELECT * FROM Location);
303 SELECT * FROM Location LIMIT 0, 49999;
304 PRAGMA database_list;
305 SELECT type,name,sql_name FROM "main".sqlite
306 SELECT "_rowid_",* FROM "main"."Employee" ORDER
307 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM "
308 SELECT "_rowid_",* FROM "main"."Employee" ORDER
309 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM "
310 SELECT COUNT(*) FROM (SELECT * FROM Job);
311 SELECT * FROM Job LIMIT 0, 49999;
312 PRAGMA database_list;
```

SQL Log Plot DB Schema Remote UTF-8



### c) DEPARTMENT:

```
CREATE TABLE "Department" (  
    "Department_ID" INTEGER NOT NULL,  
    "Name" TEXT NOT NULL,  
    "Location_ID" INTEGER,  
    PRIMARY KEY("Department_ID"),  
    FOREIGN KEY("Location_ID") REFERENCES Location);
```



### d) EMPLOYEE:

```
CREATE TABLE "Employee" (  
    "Employee_ID" INTEGER NOT NULL,  
    "LastName" TEXT,  
    "FirstName" TEXT NOT NULL,  
    "MiddleName" TEXT,  
    "Job_ID" INTEGER NOT NULL,  
    "Manager_ID" INTEGER NOT NULL,  
    "HireDate" TEXT NOT NULL, "Salary"  
    INTEGER NOT NULL, "Department_ID"  
    INTEGER NOT NULL, PRIMARY  
    KEY("Employee_ID", "Manager_ID"),  
    FOREIGN KEY("Department_ID") REFERENCES "Department"("Department_ID"),  
    FOREIGN KEY("Job_ID") REFERENCES "Job"("Job_ID")  
);
```

DB Browser for SQLite - D:\Akash Files\A Sem\Sem 5\Practical Assignments\DBMS\Assignment 7\Assignment7.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragas Execute SQL

SQL 1

```
1 SELECT * FROM Employee;
```

Employee_ID	LastName	FirstName	MiddleName	Job_ID	Manager_ID	HireDate	Salary	Department_ID
1 7369	Smith	Jon	Q	667	7902	17-DEC-84	800	10
2 7499	Alan	Kevin	J	670	7698	20-FEB-85	1600	20
3 7505	Doyle	Jean	K	671	7839	04-APR-85	2850	20
4 7506	Dennis	Lynn	S	671	7839	15-MAY-85	2750	30
5 7507	Baker	Leslie	D	671	7839	10-JUN-85	2200	40
6 7521	wark	cynthia	D	670	7698	22-FEB-85	1250	10

Result: 6 rows returned in 21ms  
At line 1:  
SELECT \* FROM Employee;

Edit Database Cell

Mode: Text

NULL

Type of data currently in cell: NULL  
0 byte(s)

SQL Log

Show SQL submitted by: Application

```
310 SELECT COUNT(*) FROM (SELECT * FROM Job);
311 SELECT * FROM Job LIMIT 0, 49999;
312 PRAGMA database_list;
313 SELECT type,name,sql,tbl_name FROM "main".sqlite
314 SELECT COUNT(*) FROM (SELECT * FROM Department)
315 SELECT * FROM Department LIMIT 0, 49999;
316 PRAGMA database_list;
317 SELECT type,name,sql,tbl_name FROM "main".sqlite
318 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM "
319 SELECT "_rowid_",* FROM "main"."Employee" ORDER
320 SELECT COUNT(*) FROM (SELECT * FROM Employee);
321 SELECT * FROM Employee LIMIT 0, 49999;
322 PRAGMA database_list;
```

SQL Log Plot DB Schema Remote

UTF-8

## e) LOAN:

CREATE TABLE "Loan" (  
 "Employee\_ID" INTEGER NOT NULL,  
 "FirstName" TEXT NOT NULL,  
 "Loan\_Amount" INTEGER NOT NULL);

DB Browser for SQLite - D:\Akash Files\A Sem\Sem 5\Practical Assignments\DBMS\Assignment 7\Assignment7.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragas Execute SQL

SQL 1

```
1 SELECT * FROM Loan;
```

Employee_ID	FirstName	Loan_Amount
1 7369	Jon	18000
2 7505	Jean	13000
3 7521	cynthia	14500

Result: 3 rows returned in 8ms  
At line 1:  
SELECT \* FROM Loan;

Edit Database Cell

Mode: Text

NULL

Type of data currently in cell: NULL  
0 byte(s)

SQL Log

Show SQL submitted by: Application

```
124 UPDATE "main"."Loan" SET "Loan_Amount"=? WHERE
125 INSERT INTO "main"."Loan" ("Employee_ID","First
126 UPDATE "main"."Loan" SET "Employee_ID"=? WHERE
127 UPDATE "main"."Loan" SET "FirstName"=? WHERE
128 UPDATE "main"."Loan" SET "Loan_Amount"=? WHERE
129 INSERT INTO "main"."Loan" ("Employee_ID","First
130 UPDATE "main"."Loan" SET "Employee_ID"=? WHERE
131 UPDATE "main"."Loan" SET "FirstName"=? WHERE
132 UPDATE "main"."Loan" SET "Loan_Amount"=? WHERE
133 SELECT COUNT(*) FROM (SELECT * FROM Loan);
134 SELECT * FROM Loan LIMIT 0, 49999;
135
```

SQL Log Plot DB Schema Remote

## 2. JOIN:

```
SELECT Employee.FirstName, Loan.FirstName
FROM Employee
LEFT JOIN Loan ON Employee.Employee_ID = Loan.Employee_ID;
```

The screenshot shows the DB Browser for SQLite interface. The main window displays a SQL query in the 'SQL 1' editor. The query is a LEFT JOIN between the 'Employee' and 'Loan' tables. The results are shown in a table with two columns: 'FirstName' and 'FirstName'. The results are as follows:

	FirstName	FirstName
1	Jon	Jon
2	Kevin	NULL
3	Jean	Jean
4	Lynn	NULL
5	Leslie	NULL
6	cynthia	cynthia

Below the table, the results are summarized: 'Result: 6 rows returned in 13ms'. The 'SQL Log' panel on the right shows the executed SQL query.

SQL 1

```
1 SELECT Employee.FirstName, Loan.FirstName
2 FROM Employee
3 LEFT JOIN Loan ON Employee.Employee_ID = Loan.Employee_ID;
```

Result: 6 rows returned in 13ms  
At line 1:  
SELECT Employee.FirstName, Loan.FirstName  
FROM Employee  
LEFT JOIN Loan ON Employee.Employee\_ID = Loan.Employee\_ID;

SQL Log

```
170 PRAGMA recursive_triggers
171 PRAGMA secure_delete
172 PRAGMA synchronous
173 PRAGMA temp_store
174 PRAGMA user_version
175 PRAGMA wal_autocheckpoint
176 SELECT 'x' NOT LIKE 'X'
177 SELECT "_rowid_",* FROM "main"."Employee" ORDER BY
178 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM "
179 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM "
180 SELECT "_rowid_",* FROM "main"."Loan" ORDER BY
181
```

## 3. SET OPERATIONS:

### a) UNION:

```
SELECT * FROM Department
UNION
SELECT * FROM Loan;
```

DB Browser for SQLite - D:\Akash Files\A Sem\Sem 5\Practical Assignments\DBMS\Assignment 7\Assignment7.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragma Execute SQL

SQL 1

```
1 SELECT * FROM Department
2 UNION
3 SELECT * FROM Loan;
```

Department_ID	Name	Location_ID
10	Accounting	122
20	Research	124
30	Sale	123
40	Operation	164
7369	Jon	18000
7505	Jean	13000
7521	cynthia	14500

Result: 7 rows returned in 14ms  
At line 1:  
SELECT \* FROM Department  
UNION  
SELECT \* FROM Loan;

Edit Database Cell

Mode: Text Import Export Set as NULL

NULL

Type of data currently in cell: NULL  
0 byte(s) Apply

SQL Log

Show SQL submitted by Application Clear

```
15 SELECT COUNT(*) FROM (SELECT "rowid",* FROM "m"
16 SELECT "rowid",* FROM "main"."Department" ORDER
17 SELECT COUNT(*) FROM (SELECT "rowid",* FROM "m"
18 SELECT "rowid",* FROM "main"."Job" ORDER BY "
19 SELECT "rowid",* FROM "main"."Loan" ORDER BY "
20 SELECT COUNT(*) FROM (SELECT "rowid",* FROM "m"
21 SELECT COUNT(*) FROM (SELECT * FROM Department
22 UNION
23 SELECT * FROM Loan);
24 SELECT * FROM Department
25 UNION
26 SELECT * FROM Loan LIMIT 0, 49999;
27 PRAGMA database_list;
```

SQL Log Plot DB Schema Remote UTF-8

## b) UNION ALL:

```
SELECT * FROM Job
UNION ALL
SELECT * FROM Location;
```

DB Browser for SQLite - D:\Akash Files\A Sem\Sem 5\Practical Assignments\DBMS\Assignment 7\Assignment7.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragma Execute SQL

SQL 1

```
1 SELECT * FROM Job;
2 UNION ALL
3 SELECT * FROM Location;
```

Job_ID	Function
667	Clerk
668	Staff
669	Analyst
670	Saleperson
671	Manager
672	President
122	New York
123	Dallas
124	Chicago
164	Boston

Result: 10 rows returned in 16ms  
At line 1:  
SELECT \* FROM Job  
UNION ALL  
SELECT \* FROM Location;

Edit Database Cell

Mode: Text Import Export Set as NULL

NULL

Type of data currently in cell: NULL  
0 byte(s) Apply

SQL Log

Show SQL submitted by Application Clear

```
43 SELECT COUNT(*) FROM (SELECT "rowid",* FROM "m"
44 SELECT COUNT(*) FROM (SELECT "rowid",* FROM "m"
45 SELECT "rowid",* FROM "main"."Location" ORDER
46 SELECT COUNT(*) FROM (SELECT "rowid",* FROM "m"
47 SELECT "rowid",* FROM "main"."Job" ORDER BY "
48 SELECT COUNT(*) FROM (SELECT * FROM Job
49 UNION ALL
50 SELECT * FROM Location);
51 SELECT * FROM Job
52 UNION ALL
53 SELECT * FROM Location LIMIT 0, 49999;
54 PRAGMA database_list;
```

SQL Log Plot DB Schema Remote UTF-8

### c) INTERSECT:

```
SELECT Employee.FirstName FROM Employee  
INTERSECT  
SELECT Loan.FirstName FROM Loan;
```

The screenshot shows the DB Browser for SQLite interface. The SQL editor contains the following query:

```
1 SELECT Employee.FirstName FROM Employee  
2 INTERSECT  
3 SELECT Loan.FirstName FROM Loan;  
4
```

The results pane displays the output of the query:

FirstName
1 Jean
2 Jon
3 Cynthia

The status bar indicates "Result: 3 rows returned in 8ms". The SQL Log pane shows the execution details, including the query text and the number of rows returned.

### 4. No. of employees working in sales department:

```
SELECT count(Department_ID) as No_Of_Employees FROM Employee  
WHERE Employee.Department_ID = (SELECT Department_ID FROM  
Department WHERE Name='Sale');
```

The screenshot shows the DB Browser for SQLite interface. The SQL editor contains the following query:

```
1 SELECT count(Department_ID) as No_Of_Employees FROM Employee  
2 WHERE Employee.Department_ID = (SELECT Department_ID FROM Department  
3 WHERE Name='Sale');
```

The results pane displays the output of the query:

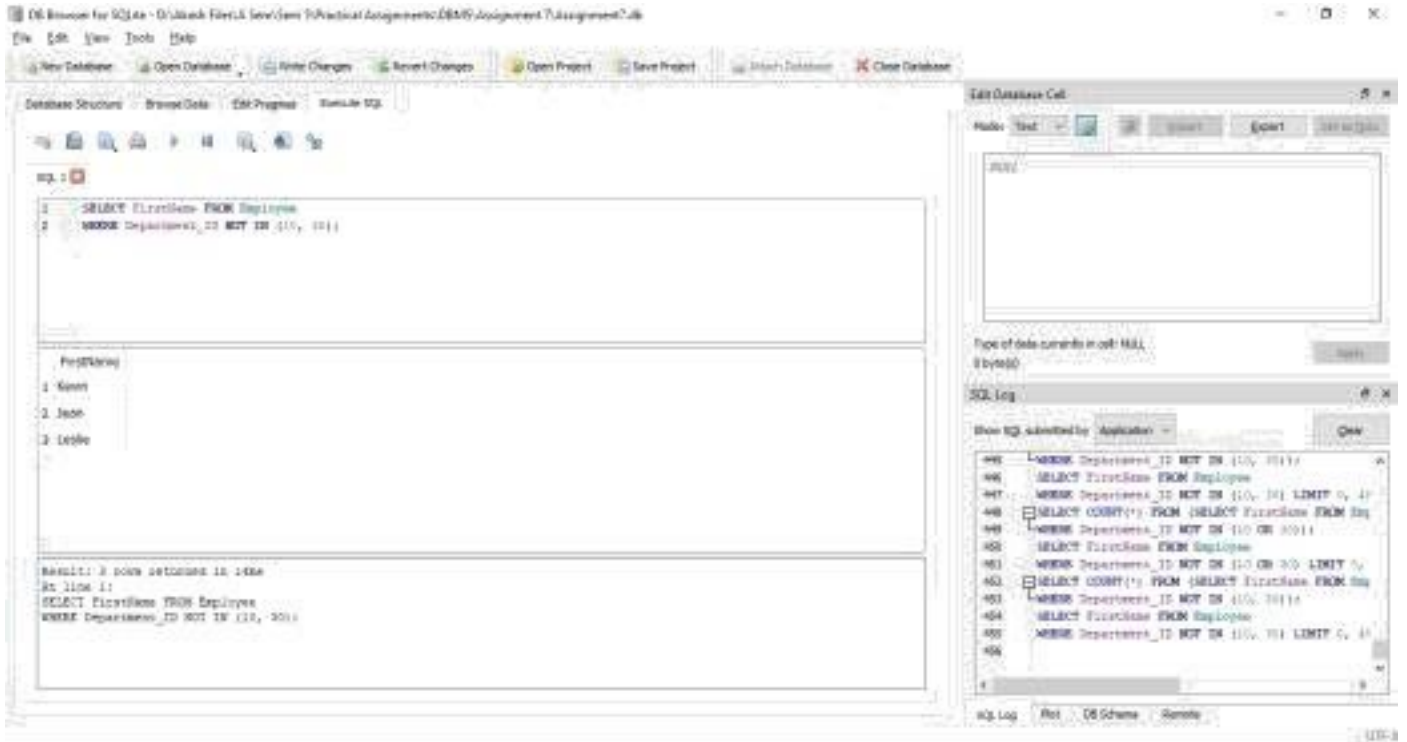
No_Of_Employees
1 1

The status bar indicates "Result: 1 rows returned in 10ms". The SQL Log pane shows the execution details, including the query text and the number of rows returned.



## 5. Find out the employees who are not working in department 10 or 30.

```
SELECT FirstName FROM Employee  
WHERE Department_ID NOT IN (10, 30);
```



## 6. List out employee id, last name in descending order based on the salary column.

```
SELECT Employee_ID, LastName FROM  
Employee ORDER BY Salary DESC;
```

DB Browser for SQLite - D:\Akash Files\A Sem\Sem 5\Practical Assignments\DBMS\Assignment 7\Assignment7.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragma Execute SQL

SQL 1

```
1 SELECT Employee_ID, LastName FROM Employee
2 ORDER BY Salary DESC;
```

Employee_ID	LastName
1 7505	Doyle
2 7506	Dennis
3 7507	Baker
4 7499	Alan
5 7521	wark
6 7369	Smith

Result: 6 rows returned in 22ms  
At line 1:  
SELECT Employee\_ID, LastName FROM Employee  
ORDER BY Salary DESC;

Edit Database Cell

Mode: Text Import Export Set as NULL

NULL

Type of data currently in cell: NULL  
0 byte(s) Apply

SQL Log

Show SQL submitted by Application Clear

```
455 WHERE Department_ID NOT IN (10, 30) LIMIT 0, 45
456 SELECT "_rowid_",* FROM "main"."Employee" ORDER
457 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM "
458 SELECT COUNT(*) FROM (SELECT Employee_ID, LastN
459 ORDER BY Salary DESC);
460 SELECT Employee_ID, LastName FROM Employee
461 ORDER BY Salary DESC LIMIT 0, 49999;
462 SELECT "_rowid_",* FROM "main"."Employee" ORDER
463 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM "
464 SELECT "_rowid_",* FROM "main"."Employee" ORDER
465 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM "
466
```

SQL Log Plot DB Schema Remote UTF-8

## 7. How many employees who are working in different departments wise in the organization

SELECT Department\_ID, count(\*) AS  
No\_of\_employees FROM Employee  
GROUP by Department\_ID;

DB Browser for SQLite - D:\Akash Files\A Sem\Sem 5\Practical Assignments\DBMS\Assignment 7\Assignment7.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragma Execute SQL

SQL 1

```
1 SELECT Department_ID, count(*) AS
2 No_of_employees FROM Employee
3 GROUP by Department_ID;
```

Department_ID	No_of_employees
1 10	2
2 20	2
3 30	1
4 40	1

Result: 4 rows returned in 10ms  
At line 1:  
SELECT Department\_ID, count(\*) AS No\_of\_employees FROM Employee GROUP by Department\_ID;

Edit Database Cell

Mode: Text Import Export Set as NULL

NULL

Type of data currently in cell: NULL  
0 byte(s) Apply

SQL Log

Show SQL submitted by Application Clear

```
609 SELECT count(Department_ID) as Accounting_Emplic
610 FROM Employee WHERE Employee.Department_ID = (
611 SELECT Department_ID FROM Department WHERE Name
612 SELECT COUNT(*) FROM (SELECT Department_ID, cou
613 SELECT Department_ID, count(*) FROM Employee GE
614 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM "
615 SELECT "_rowid_",* FROM "main"."Employee" ORDER
616 SELECT "_rowid_",* FROM "main"."Employee" ORDER
617 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM "
618 SELECT COUNT(*) FROM (SELECT Department_ID, cou
619 SELECT Department_ID, count(*) AS No_of_employe
620
```

SQL Log Plot DB Schema Remote UTF-8

## 8. List out the department id having at least four employees

```
SELECT Department_ID, count(*) AS  
No_of_employees FROM Employee  
GROUP by Department_ID HAVING No_of_employees >= 4;
```

The screenshot shows the DB Browser for SQLite interface. The SQL editor contains the query: `SELECT Department_ID, count(*) AS No_of_employees FROM Employee GROUP by Department_ID HAVING No_of_employees >= 4;`. The results pane shows 0 rows returned in 6ms. The SQL Log pane shows the query execution details.

DB Browser for SQLite - D:\Akash Files\A Sem\Sem 5\Practical Assignments\DBMS\Assignment 7\Assignment7.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragma Execute SQL

SQL 1

```
1 SELECT Department_ID, count(*) AS  
2 No_of_employees FROM Employee  
3 GROUP by Department_ID HAVING No_of_employees >= 4;
```

Result: 0 rows returned in 6ms  
At line 1:  
SELECT Department\_ID, count(\*) AS  
No\_of\_employees FROM Employee  
GROUP by Department\_ID HAVING No\_of\_employees >= 4;

Edit Database Cell

Mode: Text Import Export Set as NULL

NULL

Type of data currently in cell: NULL  
0 byte(s)

SQL Log

Show SQL submitted by Application Clear

```
627 No_of_employees FROM Employee  
628 GROUP by Department_ID HAVING No_of_employees >  
629 SELECT Department_ID, count(*) AS  
630 No_of_employees FROM Employee  
631 GROUP by Department_ID HAVING No_of_employees >  
632 SELECT COUNT(*) FROM (SELECT Department_ID, cot  
633 No_of_employees FROM Employee  
634 GROUP by Department_ID HAVING No_of_employees >  
635 SELECT Department_ID, count(*) AS  
636 No_of_employees FROM Employee  
637 GROUP by Department_ID HAVING No_of_employees >  
638
```

SQL Log Plot DB Schema Remote

UTF-8

## 9. Display the employee who got the maximum salary.

```
SELECT * FROM Employee  
WHERE Salary = (SELECT max(Salary) FROM Employee);
```

The screenshot shows the DB Browser for SQLite interface. The SQL editor contains the query: `SELECT * FROM Employee WHERE Salary = (SELECT max(Salary) FROM Employee);`. The results pane shows 1 row returned in 9ms. The SQL Log pane shows the query execution details.

DB Browser for SQLite - D:\Akash Files\A Sem\Sem 5\Practical Assignments\DBMS\Assignment 7\Assignment7.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragma Execute SQL

SQL 1

```
1 SELECT * FROM Employee  
2 WHERE Salary = (SELECT max(Salary) FROM Employee);
```

Employee_ID	LastName	FirstName	MiddleName	Job_ID	Manager_ID	HireDate	Salary	Department_ID
1 7505	Doyle	Jean	K	671	7839	04-APR-85	2850	20

Result: 1 rows returned in 9ms  
At line 1:  
SELECT \* FROM Employee  
WHERE Salary = (SELECT max(Salary) FROM Employee);

Edit Database Cell

Mode: Text Import Export Set as NULL

NULL

Type of data currently in cell: NULL  
0 byte(s)

SQL Log

Show SQL submitted by Application Clear

```
635 SELECT Department_ID, count(*) AS  
636 No_of_employees FROM Employee  
637 GROUP by Department_ID HAVING No_of_employees >  
638 SELECT COUNT(*) FROM (SELECT max(salary) FROM E  
639 SELECT max(salary) FROM Employee LIMIT 0, 49995  
640 SELECT COUNT(*) FROM (SELECT "rowid",* FROM "  
641 SELECT "rowid",* FROM "main"."Employee" ORDER  
642 SELECT COUNT(*) FROM (SELECT * FROM Employee  
643 WHERE Salary = (SELECT max(Salary) FROM Employee  
644 SELECT * FROM Employee  
645 WHERE Salary = (SELECT max(Salary) FROM Employee  
646
```

SQL Log Plot DB Schema Remote

UTF-8



## 10. Update the employees' salaries, who are working as Clerk on the basis of 10%.

```
UPDATE Employee  
SET Salary = Salary + (Salary*0.1)  
WHERE Job_ID = (  
SELECT Job_ID FROM Job  
WHERE Function = 'Clerk');
```

The screenshot displays the SQL Developer interface. The main window shows the execution of an SQL statement. The statement is as follows:

```
1 UPDATE Employee  
2 SET Salary = Salary + (Salary*0.1)  
3 WHERE Job_ID = (  
4 SELECT Job_ID FROM Job  
5 WHERE Function = 'Clerk');
```

The result pane shows the following message:

```
Result: query executed successfully. Took 116ms, 1 row affected  
At line 1:  
UPDATE Employee  
SET Salary = Salary + (Salary*0.1)  
WHERE Job_ID = (  
SELECT Job_ID FROM Job  
WHERE Function = 'Clerk');
```

The SQL Log pane shows the following statement:

```
658 SELECT COUNT(*) FROM (SELECT "rowid",* FROM "  
659 SELECT COUNT(*) FROM (SELECT "rowid",* FROM "  
660 SELECT "rowid",* FROM "emp"."job" ORDER BY "  
661 SELECT COUNT(*) FROM (SELECT "rowid",* FROM "  
662 SELECT "rowid",* FROM "emp"."job" ORDER BY "  
663 SELECT "rowid",* FROM "emp"."job" ORDER BY "  
664 SELECT "rowid",* FROM "emp"."job" ORDER BY "  
665 SELECT COUNT(*) FROM (SELECT "rowid",* FROM "  
666 SELECT COUNT(*) FROM (SELECT "rowid",* FROM "  
667 SELECT "rowid",* FROM "emp"."job" ORDER BY "  
668 SELECT "rowid",* FROM "emp"."job" ORDER BY "  
669 SELECT COUNT(*) FROM (SELECT "rowid",* FROM "
```

DB Browser for SQLite - D:\Akash Files\A Sem\Sem 5\Practical Assignments\DBMS\Assignment 7\Assignment7.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragas Execute SQL

SQL 1

```
1 SELECT * FROM Employee;
```

	Employee_ID	LastName	FirstName	MiddleName	Job_ID	Manager_ID	HireDate	Salary	Department_ID
1	7369	Smith	Jon	Q	667	7902	17-DEC-84	880	10
2	7499	Alan	Kevin	J	670	7698	20-FEB-85	1600	20
3	7505	Doyle	Jean	K	671	7839	04-APR-85	2850	20
4	7506	Dennis	Lynn	S	671	7839	15-MAY-85	2750	30
5	7507	Baker	Leslie	D	671	7839	10-JUN-85	2200	40
6	7521	wark	cynthia	D	670	7698	22-FEB-85	1250	10

Result: 6 rows returned in 1620ms  
At line 1:  
SELECT \* FROM Employee;

Edit Database Cell

Mode: Text

880

Type of data currently in cell: Text / Numeric  
3 char(s)

SQL Log

Show SQL submitted by Application

```
661 SELECT "_rowid_",* FROM "main"."Job" ORDER BY "
662 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM "
663 SELECT "_rowid_",* FROM "main"."Job" ORDER BY "
664 SELECT "_rowid_",* FROM "main"."Job" ORDER BY "
665 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM "
666 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM "
667 SELECT "_rowid_",* FROM "main"."Job" ORDER BY "
668 SELECT "_rowid_",* FROM "main"."Employee" ORDER
669 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM "
670 SELECT COUNT(*) FROM (SELECT * FROM Employee);
671 SELECT * FROM Employee LIMIT 0, 49999;
672
```

SQL Log Plot DB Schema Remote

UTF-8

## 11. Delete the employees who are working in accounting department.

```
DELETE FROM Employee
WHERE Department_ID = (
SELECT Department_ID FROM Department
WHERE Name = 'Accounting');
```

DB Browser for SQLite - D:\Akash Files\A Sem\Sem 5\Practical Assignments\DBMS\Assignment 7\Assignment7.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragma Execute SQL

SQL 1

```

1 DELETE FROM Employee
2 WHERE Department_ID = (
3 SELECT Department_ID FROM Department
4 WHERE Name = 'Accounting');

```

Result: query executed successfully. Took 33ms, 2 rows affected

At line 1:  
DELETE FROM Employee  
WHERE Department\_ID = (  
SELECT Department\_ID FROM Department  
WHERE Name = 'Accounting');

Edit Database Cell

Mode: Text Import Export Set as NULL

NULL

Type of data currently in cell: NULL  
0 byte(s) Apply

SQL Log

Show SQL submitted by Application Clear

```

679 SELECT "_rowid_",* FROM "main"."Employee" ORDER
680 SELECT "_rowid_",* FROM "main"."Employee" ORDER
681 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM
682 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM
683 SELECT "_rowid_",* FROM "main"."Department" OR
684 SELECT "_rowid_",* FROM "main"."Department" OR
685 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM
686 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM
687 SELECT "_rowid_",* FROM "main"."Department" OR
688 SELECT "_rowid_",* FROM "main"."Employee" ORDER
689 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM
690

```

SQL Log Plot DB Schema Remote

DB Browser for SQLite - D:\Akash Files\A Sem\Sem 5\Practical Assignments\DBMS\Assignment 7\Assignment7.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragma Execute SQL

SQL 1

```

1 SELECT * FROM Employee;

```

Employee_ID	LastName	FirstName	MiddleName	Job_ID	Manager_ID	HireDate	Salary	Department_ID
1 7499	Alan	Kevin	J	670	7698	20-FEB-85	1600	20
2 7505	Doyle	Jean	K	671	7839	04-APR-85	2850	20
3 7506	Dennis	Lynn	S	671	7839	15-MAY-85	2750	30
4 7507	Baker	Leslie	D	671	7839	10-JUN-85	2200	40

Result: 4 rows returned in 12ms

At line 1:  
SELECT \* FROM Employee;

Edit Database Cell

Mode: Text Import Export Set as NULL

NULL

Type of data currently in cell: NULL  
0 byte(s) Apply

SQL Log

Show SQL submitted by Application Clear

```

681 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM
682 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM
683 SELECT "_rowid_",* FROM "main"."Department" OR
684 SELECT "_rowid_",* FROM "main"."Department" OR
685 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM
686 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM
687 SELECT "_rowid_",* FROM "main"."Department" OR
688 SELECT "_rowid_",* FROM "main"."Employee" ORDER
689 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM
690 SELECT COUNT(*) FROM (SELECT * FROM Employee);
691 SELECT * FROM Employee LIMIT 0, 49999;
692

```

SQL Log Plot DB Schema Remote

## 12. Find out whose department has not employees.

```

SELECT * FROM Employee
WHERE NOT EXISTS(
SELECT Department_ID FROM Department
WHERE Department.Department_ID = Employee.Department_ID);

```

DB Browser for SQLite - D:\Akash Files\A Sem\Sem 5\Practical Assignments\DBMS\Assignment 7\Assignment7.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragma Execute SQL

SQL 1

```

1 SELECT * FROM Employee
2 WHERE NOT EXISTS(
3   SELECT Department_ID FROM Department
4   WHERE Department.Department_ID = Employee.Department_ID);

```

Result: 0 rows returned in 8ms

At line 1:  
 SELECT \* FROM Employee  
 WHERE NOT EXISTS(  
 SELECT Department\_ID FROM Department  
 WHERE Department.Department\_ID = Employee.Department\_ID);

Edit Database Cell

Mode: Text Import Export Set as NULL

NULL

Type of data currently in cell: NULL  
 0 byte(s) Apply

SQL Log

Show SQL submitted by: Application Clear

```

155 PRAGMA database_list;
156 SELECT type,name,sql,tbl_name FROM "main".sqlite
157 PRAGMA database_list;
158 SELECT type,name,sql,tbl_name FROM "main".sqlite
159 SELECT COUNT(*) FROM (SELECT * FROM Employee
160 WHERE NOT EXISTS(
161   SELECT Department_ID FROM Department
162   WHERE Department.Department_ID = Employee.Department_ID)
163 SELECT * FROM Employee
164 WHERE NOT EXISTS(
165   SELECT Department_ID FROM Department
166   WHERE Department.Department_ID = Employee.Department_ID)
167 PRAGMA database_list;

```

SQL Log Plot DB Schema Remote UTF-8

### 13. List out the department wise maximum salary, minimum salary, average salary of the employees

SELECT Department\_ID, count(\*) as No\_of\_Employees,  
 max(Salary), min(Salary), avg(Salary)  
 FROM Employee GROUP BY Department\_ID;

DB Browser for SQLite - D:\Akash Files\A Sem\Sem 5\Practical Assignments\DBMS\Assignment 7\Assignment7.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragma Execute SQL

SQL 1

```

1 SELECT Department_ID, count(*) as No_of_Employees,
2 max(Salary), min(Salary), avg(Salary)
3 FROM Employee GROUP BY Department_ID;

```

Department_ID	No_of_Employees	max(Salary)	min(Salary)	avg(Salary)
20	2	2850	1600	2225.0
30	1	2750	2750	2750.0
40	1	2200	2200	2200.0

Result: 3 rows returned in 20ms

At line 1:  
 SELECT Department\_ID, count(\*) as No\_of\_Employees,  
 max(Salary), min(Salary), avg(Salary)  
 FROM Employee GROUP BY Department\_ID;

Edit Database Cell

Mode: Text Import Export Set as NULL

NULL

Type of data currently in cell: NULL  
 0 byte(s) Apply

SQL Log

Show SQL submitted by: Application Clear

```

171 FROM Employee GROUP BY Department_ID);
172 SELECT Department_ID, count(*),
173 max(Salary), min(Salary), avg(Salary)
174 FROM Employee GROUP BY Department_ID LIMIT 0, 3;
175 PRAGMA database_list;
176 SELECT type,name,sql,tbl_name FROM "main".sqlite
177 SELECT COUNT(*) FROM (SELECT Department_ID, count(*)
178   max(Salary), min(Salary), avg(Salary)
179   FROM Employee GROUP BY Department_ID);
180 SELECT Department_ID, count(*) as No_of_Employees,
181 max(Salary), min(Salary), avg(Salary)
182 FROM Employee GROUP BY Department_ID LIMIT 0, 3;
183 PRAGMA database_list;

```

SQL Log Plot DB Schema Remote UTF-8

### 14. How many employees who are joined in 1985.

SELECT count(\*) AS Emp\_Joined\_in\_1985 FROM Employee  
WHERE HireDate like '%85';

DB Browser for SQLite - D:\Akash Files\A Sem\Sem 5\Practical Assignments\DBMS\Assignment 7\Assignment7.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragma Execute SQL

SQL 1

```
1 SELECT count(*) AS Emp_Joined_in_1985 FROM Employee
2 WHERE HireDate like '%85';
```

Emp\_Joined\_in\_1985

Emp_Joined_in_1985
4

Result: 1 rows returned in 23ms  
At line 1:  
SELECT count(\*) AS Emp\_Joined\_in\_1985 FROM Employee  
WHERE HireDate like '%85';

Edit Database Cell

Mode: Text Import Export Set as NULL

NULL

Type of data currently in cell: NULL  
0 byte(s) Apply

SQL Log

Show SQL submitted by Application Clear

```
199 SELECT COUNT(*) FROM (SELECT count(*) FROM Emp_J
200 WHERE HireDate like '%85');
201 SELECT count(*) FROM Employee
202 WHERE HireDate like '%85' LIMIT 0, 49999;
203 PRAGMA database_list;
204 SELECT type,name,sql,tbl_name FROM "main".sqlite
205 SELECT "_rowid_",* FROM "main"."Employee" ORDE
206 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM "
207 SELECT COUNT(*) FROM (SELECT count(*) AS Emp_Jc
208 WHERE HireDate like '%85');
209 SELECT count(*) AS Emp_Joined_in_1985 FROM Emp
210 WHERE HireDate like '%85' LIMIT 0, 49999;
211 PRAGMA database_list;
```

SQL Log Plot DB Schema Remote

UTF-8

## 15. Display the employees who are working in “New York”

SELECT \* FROM Employee  
WHERE Department\_ID = (  
SELECT Department\_ID FROM Department  
WHERE Location\_ID = (  
SELECT Location\_ID FROM Location  
WHERE Reginal\_Group = 'New York'));



DB Browser for SQLite - D:\Akash Files\A Sem\Sem 5\Practical Assignments\DBMS\Assignment 7\Assignment7.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragma Execute SQL

SQL 1

```

1 SELECT * FROM Employee
2 WHERE Department_ID = (
3   SELECT Department_ID FROM Department
4 WHERE Location_ID = (
5   SELECT Location_ID FROM Location
6   WHERE Reginal_Group = 'New York'));

```

Result: 0 rows returned in 5ms

At line 1:  
 SELECT \* FROM Employee  
 WHERE Department\_ID = (  
 SELECT Department\_ID FROM Department  
 WHERE Location\_ID = (  
 SELECT Location\_ID FROM Location  
 WHERE Reginal\_Group = 'New York'));

Edit Database Cell

Mode: Text Import Export Set as NULL

NULL

Type of data currently in cell: NULL  
 0 byte(s) Apply

SQL Log

Show SQL submitted by Application Clear

```

272 SELECT "_rowid_",* FROM "main"."Employee" ORDER BY
273 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM
274 SELECT "_rowid_",* FROM "main"."Job" ORDER BY
275 SELECT "_rowid_",* FROM "main"."Job" ORDER BY
276 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM
277 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM
278 SELECT "_rowid_",* FROM "main"."Employee" ORDER
279 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM
280 SELECT "_rowid_",* FROM "main"."Department" ORC
281 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM
282 SELECT "_rowid_",* FROM "main"."Location" ORDER
283

```

SQL Log Plot DB Schema Remote UTF-8

## 16. List our employees with their department names

SELECT \* FROM Employee, Department  
 WHERE Employee.Department\_ID =  
 Department.Department\_ID GROUP by Name;

DB Browser for SQLite - D:\Akash Files\A Sem\Sem 5\Practical Assignments\DBMS\Assignment 7\Assignment7.db

File Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragma Execute SQL

SQL 1

```

1 SELECT * FROM Employee, Department
2 WHERE Employee.Department_ID = Department.Department_ID
3 GROUP by Name;

```

Employee_ID	LastName	FirstName	MiddleName	Job_ID	Manager_ID	HireDate	Salary	Department_ID	Department_ID	Name
1 7507	Baker	Leslie	D	671	7839	10-JUN-85	2200	40	40	Operation
2 7499	Alan	Kevin	J	670	7698	20-FEB-85	1600	20	20	Research
3 7506	Dennis	Lynn	S	671	7839	15-MAY-85	2750	30	30	Sale

Result: 3 rows returned in 25ms

At line 1:  
 SELECT \* FROM Employee, Department  
 WHERE Employee.Department\_ID = Department.Department\_ID  
 GROUP by Name;

Edit Database Cell

Mode: Text Import Export Set as NULL

7507

Type of data currently in cell: Text / Numeric  
 4 char(s) Apply

SQL Log

Show SQL submitted by Application Clear

```

308 SELECT * FROM Employee, Department
309 WHERE Employee.Department_ID = Department.Dep
310 GROUP by Name LIMIT 0, 49999;
311 PRAGMA database_list;
312 SELECT type,name,sql,tbl_name FROM "main".sql
313 SELECT "_rowid_",* FROM "main"."Department"
314 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM
315 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM
316 SELECT "_rowid_",* FROM "main"."Employee" ORC
317 SELECT COUNT(*) FROM (SELECT "_rowid_",* FROM
318 SELECT "_rowid_",* FROM "main"."Department"
319

```

SQL Log Plot DB Schema Remote UTF-8































































