

ABHISHEK GAJULA

RA2111003010756

1. (Exercise on retrieving records from the table) EMPLOYEES (Employee\_Id, First\_Name, Last\_Name, Email, Phone\_Number, Hire\_Date, Job\_Id, Salary, Commission\_Pct, Manager\_Id, Department\_Id)

( a) Find out the employee id, names, salaries of all the employees

ANSWER:

```
INSERT INTO EMPLOYEES VALUES(1, 'John', 'Doe', 'john.doe@example.com',  
'123456789', '2022-01-01', 'JR001', 5000.00, 0.05, 100, 10);
```

```
INSERT INTO EMPLOYEES VALUES(2, 'Jane', 'Smith', 'jane.smith@example.com',  
'987654321', '2022-02-15', 'JR002', 5500.00, 0.03, 100, 20);
```

```
INSERT INTO EMPLOYEES VALUES(3, 'Michael', 'Johnson',  
'michael.johnson@example.com', '456123789', '2021-11-10', 'JR003', 4800.00,  
0.02, 101, 30);
```

```
INSERT INTO EMPLOYEES VALUES(4, 'Emily', 'Brown',  
'emily.brown@example.com', '789456123', '2022-03-20', 'JR001', 5200.00,  
0.04, 101, 10);
```

```
INSERT INTO EMPLOYEES VALUES(5, 'Sarah', 'Austin',  
'sarah.austin@example.com', '9876543210', '2022-04-10', 'JR002', 6000.00,  
0.03, 100, 20);
```

```
SELECT * FROM EMPLOYEES;
```

```
SELECT Employee_Id, First_Name, Last_Name, Salary FROM EMPLOYEES;
```

The screenshot shows a MySQL query editor with the following SQL script:

```

1 CREATE TABLE EMPLOYEES (
2   Employee_Id INT PRIMARY KEY,
3   First_Name VARCHAR(50),
4   Last_Name VARCHAR(50),
5   Email VARCHAR(100),
6   Phone_Number VARCHAR(20),
7   Hire_Date DATE,
8   Job_Id VARCHAR(10),
9   Salary DECIMAL(10, 2),
10  Commission_Pct DECIMAL(5, 2),
11  Manager_Id INT,
12  Department_Id INT
13 );
14 INSERT INTO EMPLOYEES VALUES(1, 'John', 'Doe', 'john.doe@example.com', '123456789', '2022-01-01', 'JR001', 5000.00, 0.1, 100, 10);
15 INSERT INTO EMPLOYEES VALUES(2, 'Jane', 'Smith', 'jane.smith@example.com', '987654321', '2022-02-15', 'JR002', 5500.00, 0.1, 100, 10);
16 INSERT INTO EMPLOYEES VALUES(3, 'Michael', 'Johnson', 'michael.johnson@example.com', '456123789', '2021-11-10', 'JR003', 4800.00, 0.1, 100, 10);
17 INSERT INTO EMPLOYEES VALUES(4, 'Emily', 'Brown', 'emily.brown@example.com', '789456123', '2022-03-20', 'JR004', 5200.00, 0.1, 100, 10);
18 INSERT INTO EMPLOYEES VALUES(5, 'Sarah', 'Austin', 'sarah.austin@example.com', '9876543210', '2022-04-10', 'JR005', 6000.00, 0.1, 100, 10);
19 SELECT * FROM EMPLOYEES;
20 SELECT Employee_Id, First_Name, Last_Name, Salary FROM EMPLOYEES;
21 SELECT Employee_Id, First_Name, Last_Name FROM EMPLOYEES WHERE Manager_Id = 100;
22 SELECT First_Name, Last_Name FROM EMPLOYEES WHERE Salary >= 4800;
23 SELECT Employee_Id, First_Name, Last_Name FROM EMPLOYEES WHERE Last_Name = 'AUSTIN';
24 SELECT First_Name, Last_Name FROM EMPLOYEES WHERE Department_Id IN (60, 70, 80);
25 SELECT DISTINCT Manager_Id FROM EMPLOYEES;

```

The results of the queries are displayed on the right side of the editor:

Employee_Id	First_Name	Last_Name	Salary
1	John	Doe	5000.00
2	Jane	Smith	5500.00
3	Michael	Johnson	4800.00
4	Emily	Brown	5200.00
5	Sarah	Austin	6000.00

Employee_Id	First_Name	Last_Name
1	John	Doe
2	Jane	Smith
5	Sarah	Austin

First_Name	Last_Name
John	Doe
Jane	Smith
Michael	Johnson
Emily	Brown
Sarah	Austin

Employee_Id	First_Name	Last_Name
1	John	Doe
2	Jane	Smith
5	Sarah	Austin

First_Name	Last_Name
John	Doe
Jane	Smith
Michael	Johnson
Emily	Brown
Sarah	Austin

Employee_Id	First_Name	Last_Name
1	John	Doe
2	Jane	Smith
5	Sarah	Austin

( b) List out the employees who works under manager 100

ANSWER:

SELECT Employee\_Id, First\_Name, Last\_Name FROM EMPLOYEES WHERE Manager\_Id = 100;

The screenshot shows a MySQL query editor with the following SQL script:

```

1 CREATE TABLE EMPLOYEES (
2   Employee_Id INT PRIMARY KEY,
3   First_Name VARCHAR(50),
4   Last_Name VARCHAR(50),
5   Email VARCHAR(100),
6   Phone_Number VARCHAR(20),
7   Hire_Date DATE,
8   Job_Id VARCHAR(10),
9   Salary DECIMAL(10, 2),
10  Commission_Pct DECIMAL(5, 2),
11  Manager_Id INT,
12  Department_Id INT
13 );
14 INSERT INTO EMPLOYEES VALUES(1, 'John', 'Doe', 'john.doe@example.com', '123456789', '2022-01-01', 'JR001', 5000.00, 0.1, 100, 10);
15 INSERT INTO EMPLOYEES VALUES(2, 'Jane', 'Smith', 'jane.smith@example.com', '987654321', '2022-02-15', 'JR002', 5500.00, 0.1, 100, 10);
16 INSERT INTO EMPLOYEES VALUES(3, 'Michael', 'Johnson', 'michael.johnson@example.com', '456123789', '2021-11-10', 'JR003', 4800.00, 0.1, 100, 10);
17 INSERT INTO EMPLOYEES VALUES(4, 'Emily', 'Brown', 'emily.brown@example.com', '789456123', '2022-03-20', 'JR004', 5200.00, 0.1, 100, 10);
18 INSERT INTO EMPLOYEES VALUES(5, 'Sarah', 'Austin', 'sarah.austin@example.com', '9876543210', '2022-04-10', 'JR005', 6000.00, 0.1, 100, 10);
19 SELECT * FROM EMPLOYEES;
20 SELECT Employee_Id, First_Name, Last_Name, Salary FROM EMPLOYEES;
21 SELECT Employee_Id, First_Name, Last_Name FROM EMPLOYEES WHERE Manager_Id = 100;
22 SELECT First_Name, Last_Name FROM EMPLOYEES WHERE Salary >= 4800;
23 SELECT Employee_Id, First_Name, Last_Name FROM EMPLOYEES WHERE Last_Name = 'AUSTIN';
24 SELECT First_Name, Last_Name FROM EMPLOYEES WHERE Department_Id IN (60, 70, 80);
25 SELECT DISTINCT Manager_Id FROM EMPLOYEES;

```

The results of the queries are displayed on the right side of the editor:

Employee_Id	First_Name	Last_Name	Salary
1	John	Doe	5000.00
2	Jane	Smith	5500.00
3	Michael	Johnson	4800.00
4	Emily	Brown	5200.00
5	Sarah	Austin	6000.00

Employee_Id	First_Name	Last_Name
1	John	Doe
2	Jane	Smith
5	Sarah	Austin

First_Name	Last_Name
John	Doe
Jane	Smith
Michael	Johnson
Emily	Brown
Sarah	Austin

Employee_Id	First_Name	Last_Name
1	John	Doe
2	Jane	Smith
5	Sarah	Austin

First_Name	Last_Name
John	Doe
Jane	Smith
Michael	Johnson
Emily	Brown
Sarah	Austin

Employee_Id	First_Name	Last_Name
1	John	Doe
2	Jane	Smith
5	Sarah	Austin

( c) Find the names of the employees who have a salary greater than or equal to 4800

ANSWER:

SELECT First\_Name, Last\_Name FROM EMPLOYEES WHERE Salary >= 4800;

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

```

1 CREATE TABLE EMPLOYEES (
2   Employee_Id INT PRIMARY KEY,
3   First_Name VARCHAR(50),
4   Last_Name VARCHAR(50),
5   Email VARCHAR(100),
6   Phone_Number VARCHAR(20),
7   Hire_Date DATE,
8   Job_Id VARCHAR(10),
9   Salary DECIMAL(10, 2),
10  Commission_Pct DECIMAL(5, 2),
11  Manager_Id INT,
12  Department_Id INT
13 );
14 INSERT INTO EMPLOYEES VALUES(1, 'John', 'Doe', 'john.doe@example.com', '123456789', '2022-01-01', 'JR001', 5000.00, 0.1, 100, 60);
15 INSERT INTO EMPLOYEES VALUES(2, 'Jane', 'Smith', 'jane.smith@example.com', '987654321', '2022-02-15', 'JR002', 5500.00, 0.1, 100, 70);
16 INSERT INTO EMPLOYEES VALUES(3, 'Michael', 'Johnson', 'michael.johnson@example.com', '456123789', '2021-11-10', 'JR003', 4800.00, 0.1, 100, 80);
17 INSERT INTO EMPLOYEES VALUES(4, 'Emily', 'Brown', 'emily.brown@example.com', '789456123', '2022-03-20', 'JR004', 5200.00, 0.1, 100, 60);
18 INSERT INTO EMPLOYEES VALUES(5, 'Sarah', 'Austin', 'sarah.austin@example.com', '9876543210', '2022-04-10', 'JR005', 6000.00, 0.1, 100, 70);
19 SELECT * FROM EMPLOYEES;
20 SELECT Employee_Id, First_Name, Last_Name, Salary FROM EMPLOYEES;
21 SELECT Employee_Id, First_Name, Last_Name FROM EMPLOYEES WHERE Manager_Id = 100;
22 SELECT First_Name, Last_Name FROM EMPLOYEES WHERE Salary >= 4800;
23 SELECT Employee_Id, First_Name, Last_Name FROM EMPLOYEES WHERE Last_Name = 'AUSTIN';
24 SELECT First_Name, Last_Name FROM EMPLOYEES WHERE Department_Id IN (60, 70, 80);
25 SELECT DISTINCT Manager_Id FROM EMPLOYEES;
26
27

```

The results window shows the output of the queries:

Employee_Id	First_Name	Last_Name	Salary
1	John	Doe	5000.00
2	Jane	Smith	5500.00
3	Michael	Johnson	4800.00
4	Emily	Brown	5200.00
5	Sarah	Austin	6000.00

Employee_Id	First_Name	Last_Name
1	John	Doe
2	Jane	Smith
5	Sarah	Austin

First_Name	Last_Name
John	Doe
Jane	Smith
Michael	Johnson
Emily	Brown
Sarah	Austin

Employee_Id	First_Name	Last_Name
5	Sarah	Austin

Manager_Id
100
101

( d) List out the employees whose last name is 'AUSTIN'

ANSWER:

SELECT Employee\_Id, First\_Name, Last\_Name FROM EMPLOYEES WHERE Last\_Name = 'AUSTIN';

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

```

1 CREATE TABLE EMPLOYEES (
2   Employee_Id INT PRIMARY KEY,
3   First_Name VARCHAR(50),
4   Last_Name VARCHAR(50),
5   Email VARCHAR(100),
6   Phone_Number VARCHAR(20),
7   Hire_Date DATE,
8   Job_Id VARCHAR(10),
9   Salary DECIMAL(10, 2),
10  Commission_Pct DECIMAL(5, 2),
11  Manager_Id INT,
12  Department_Id INT
13 );
14 INSERT INTO EMPLOYEES VALUES(1, 'John', 'Doe', 'john.doe@example.com', '123456789', '2022-01-01', 'JR001', 5000.00, 0.1, 100, 60);
15 INSERT INTO EMPLOYEES VALUES(2, 'Jane', 'Smith', 'jane.smith@example.com', '987654321', '2022-02-15', 'JR002', 5500.00, 0.1, 100, 70);
16 INSERT INTO EMPLOYEES VALUES(3, 'Michael', 'Johnson', 'michael.johnson@example.com', '456123789', '2021-11-10', 'JR003', 4800.00, 0.1, 100, 80);
17 INSERT INTO EMPLOYEES VALUES(4, 'Emily', 'Brown', 'emily.brown@example.com', '789456123', '2022-03-20', 'JR004', 5200.00, 0.1, 100, 60);
18 INSERT INTO EMPLOYEES VALUES(5, 'Sarah', 'Austin', 'sarah.austin@example.com', '9876543210', '2022-04-10', 'JR005', 6000.00, 0.1, 100, 70);
19 SELECT * FROM EMPLOYEES;
20 SELECT Employee_Id, First_Name, Last_Name, Salary FROM EMPLOYEES;
21 SELECT Employee_Id, First_Name, Last_Name FROM EMPLOYEES WHERE Manager_Id = 100;
22 SELECT First_Name, Last_Name FROM EMPLOYEES WHERE Salary >= 4800;
23 SELECT Employee_Id, First_Name, Last_Name FROM EMPLOYEES WHERE Last_Name = 'AUSTIN';
24 SELECT First_Name, Last_Name FROM EMPLOYEES WHERE Department_Id IN (60, 70, 80);
25 SELECT DISTINCT Manager_Id FROM EMPLOYEES;
26
27

```

The results window shows the output of the queries:

Employee_Id	First_Name	Last_Name
1	John	Doe
2	Jane	Smith
5	Sarah	Austin

First_Name	Last_Name
John	Doe
Jane	Smith
Michael	Johnson
Emily	Brown
Sarah	Austin

Employee_Id	First_Name	Last_Name
5	Sarah	Austin

Manager_Id
100
101

( e) Find the names of the employees who works in departments 60,70 and 80

SELECT First\_Name, Last\_Name FROM EMPLOYEES WHERE Department\_Id IN (60, 70, 80);

queries.sql + 425xmxu2v NEW MYSQL RUN

```

1 CREATE TABLE EMPLOYEES (
2   Employee_Id INT PRIMARY KEY,
3   First_Name VARCHAR(50),
4   Last_Name VARCHAR(50),
5   Email VARCHAR(100),
6   Phone_Number VARCHAR(20),
7   Hire_Date DATE,
8   Job_Id VARCHAR(10),
9   Salary DECIMAL(10, 2),
10  Commission_Pct DECIMAL(5, 2),
11  Manager_Id INT,
12  Department_Id INT
13 );
14 INSERT INTO EMPLOYEES VALUES(1, 'John', 'Doe', 'john.doe@example.com', '123456789', '2022-01-01', 'JR001', 5000, 0.1, 100, 10);
15 INSERT INTO EMPLOYEES VALUES(2, 'Jane', 'Smith', 'jane.smith@example.com', '987654321', '2022-02-15', 'JR002', 4500, 0.1, 100, 10);
16 INSERT INTO EMPLOYEES VALUES(3, 'Michael', 'Johnson', 'michael.johnson@example.com', '456123789', '2021-11-10', 'JR003', 5500, 0.1, 100, 10);
17 INSERT INTO EMPLOYEES VALUES(4, 'Emily', 'Brown', 'emily.brown@example.com', '789456123', '2022-03-20', 'JR004', 4000, 0.1, 100, 10);
18 INSERT INTO EMPLOYEES VALUES(5, 'Sarah', 'Austin', 'sarah.austin@example.com', '9876543210', '2022-04-10', 'JR005', 5000, 0.1, 100, 10);
19 SELECT * FROM EMPLOYEES;
20 SELECT Employee_Id, First_Name, Last_Name, Salary FROM EMPLOYEES;
21 SELECT Employee_Id, First_Name, Last_Name FROM EMPLOYEES WHERE Manager_Id = 100;
22 SELECT First_Name, Last_Name FROM EMPLOYEES WHERE Salary >= 4000;
23 SELECT Employee_Id, First_Name, Last_Name FROM EMPLOYEES WHERE Last_Name = 'AUSTIN';
24 SELECT First_Name, Last_Name FROM EMPLOYEES WHERE Department_Id IN (60, 70, 80);
25 SELECT DISTINCT Manager_Id FROM EMPLOYEES;
26
27

```

STDIN  
Input for the program ( Optional )

Employee_Id	First_Name	Last_Name
1	John	Doe
2	Jane	Smith
5	Sarah	Austin

First_Name	Last_Name
John	Doe
Jane	Smith
Michael	Johnson
Emily	Brown
Sarah	Austin

Employee_Id	First_Name	Last_Name
5	Sarah	Austin

Manager_Id
100
101

( f ) Display the unique Manager\_Id.

ANSWER:

SELECT DISTINCT Manager\_Id FROM EMPLOYEES;

queries.sql + 425xmxu2v NEW MYSQL RUN

```

1 CREATE TABLE EMPLOYEES (
2   Employee_Id INT PRIMARY KEY,
3   First_Name VARCHAR(50),
4   Last_Name VARCHAR(50),
5   Email VARCHAR(100),
6   Phone_Number VARCHAR(20),
7   Hire_Date DATE,
8   Job_Id VARCHAR(10),
9   Salary DECIMAL(10, 2),
10  Commission_Pct DECIMAL(5, 2),
11  Manager_Id INT,
12  Department_Id INT
13 );
14 INSERT INTO EMPLOYEES VALUES(1, 'John', 'Doe', 'john.doe@example.com', '123456789', '2022-01-01', 'JR001', 5000, 0.1, 100, 10);
15 INSERT INTO EMPLOYEES VALUES(2, 'Jane', 'Smith', 'jane.smith@example.com', '987654321', '2022-02-15', 'JR002', 4500, 0.1, 100, 10);
16 INSERT INTO EMPLOYEES VALUES(3, 'Michael', 'Johnson', 'michael.johnson@example.com', '456123789', '2021-11-10', 'JR003', 5500, 0.1, 100, 10);
17 INSERT INTO EMPLOYEES VALUES(4, 'Emily', 'Brown', 'emily.brown@example.com', '789456123', '2022-03-20', 'JR004', 4000, 0.1, 100, 10);
18 INSERT INTO EMPLOYEES VALUES(5, 'Sarah', 'Austin', 'sarah.austin@example.com', '9876543210', '2022-04-10', 'JR005', 5000, 0.1, 100, 10);
19 SELECT * FROM EMPLOYEES;
20 SELECT Employee_Id, First_Name, Last_Name, Salary FROM EMPLOYEES;
21 SELECT Employee_Id, First_Name, Last_Name FROM EMPLOYEES WHERE Manager_Id = 100;
22 SELECT First_Name, Last_Name FROM EMPLOYEES WHERE Salary >= 4000;
23 SELECT Employee_Id, First_Name, Last_Name FROM EMPLOYEES WHERE Last_Name = 'AUSTIN';
24 SELECT First_Name, Last_Name FROM EMPLOYEES WHERE Department_Id IN (60, 70, 80);
25 SELECT DISTINCT Manager_Id FROM EMPLOYEES;
26
27

```

STDIN  
Input for the program ( Optional )

Employee_Id	First_Name	Last_Name
1	John	Doe
2	Jane	Smith
5	Sarah	Austin

First_Name	Last_Name
John	Doe
Jane	Smith
Michael	Johnson
Emily	Brown
Sarah	Austin

Employee_Id	First_Name	Last_Name
5	Sarah	Austin

Manager_Id
100
101

2. (Exercise on updating records in table) Create Client\_master with the following fields(ClientNO, Name, Address, City, State, bal\_due)

( a ) Insert five records

CREATE TABLE Client\_master (

ClientNO INT PRIMARY KEY,

```
Name VARCHAR(100),  
Address VARCHAR(255),  
City VARCHAR(100),  
State VARCHAR(100),  
bal_due DECIMAL(10, 2)  
);
```

-- Inserting sample records with Indian names into the Client\_master table

```
INSERT INTO Client_master VALUES(1, 'Rahul Kumar', '123 Main St', 'New  
Delhi', 'Delhi', 1000.00);
```

```
INSERT INTO Client_master VALUES(2, 'Priya Sharma', '456 Elm St', 'Mumbai',  
'Maharashtra', 2500.00);
```

```
INSERT INTO Client_master VALUES(3, 'Amit Patel', '789 Oak St', 'Ahmedabad',  
'Gujarat', 1500.00);
```

```
INSERT INTO Client_master VALUES(4, 'Neha Singh', '101 Pine St', 'Bangalore',  
'Karnataka', 2000.00);
```

```
INSERT INTO Client_master VALUES(5, 'Sneha Gupta', '202 Maple St', 'Kolkata',  
'West Bengal', 5500.00);
```

```
SELECT * FROM Client_master;
```



UPDATE Client\_master SET bal\_due = 5100.00 WHERE ClientNO = 2;

```

1 CREATE TABLE Client_master (
2   ClientNO INT PRIMARY KEY,
3   Name VARCHAR(100),
4   Address VARCHAR(255),
5   City VARCHAR(100),
6   State VARCHAR(100),
7   bal_due DECIMAL(10, 2)
8 );
9
10 -- Inserting sample records with Indian names into the Client_master table
11 INSERT INTO Client_master VALUES(1, 'Rahul Kumar', '123 Main St', 'New Delhi', 'Delhi', 1000.00);
12 INSERT INTO Client_master VALUES(2, 'Priya Sharma', '456 Elm St', 'Mumbai', 'Maharashtra', 2500.00);
13 INSERT INTO Client_master VALUES(3, 'Amit Patel', '789 Oak St', 'Ahmedabad', 'Gujarat', 1500.00);
14 INSERT INTO Client_master VALUES(4, 'Neha Singh', '101 Pine St', 'Bangalore', 'Karnataka', 2000.00);
15 INSERT INTO Client_master VALUES(5, 'Sneha Gupta', '202 Maple St', 'Kolkata', 'West Bengal', 5500.00);
16
17 SELECT * FROM Client_master;
18 SELECT Name FROM Client_master WHERE bal_due > 5000;
19 UPDATE Client_master SET bal_due = 5100.00 WHERE ClientNO = 2;
20
21 SELECT * FROM Client_master;
22 ALTER TABLE Client_master RENAME TO Client12;
23 SELECT ClientNO, Name, Address, City, State, bal_due AS BALANCE FROM Client12;
24
25 DESC Client12;
26
27
28
29
30

```

ClientNO	Name	Address	City	State	bal_due
1	Rahul Kumar	123 Main St	New Delhi	Delhi	1000.00
2	Priya Sharma	456 Elm St	Mumbai	Maharashtra	5100.00
3	Amit Patel	789 Oak St	Ahmedabad	Gujarat	1500.00
4	Neha Singh	101 Pine St	Bangalore	Karnataka	2000.00
5	Sneha Gupta	202 Maple St	Kolkata	West Bengal	5500.00

ClientNO	Name	Address	City	State	BALANCE
1	Rahul Kumar	123 Main St	New Delhi	Delhi	1000.00
2	Priya Sharma	456 Elm St	Mumbai	Maharashtra	5100.00

( d ) Change the name of Client\_master to Client12 .

ALTER TABLE Client\_master RENAME TO Client12;

```

1 CREATE TABLE Client_master (
2   ClientNO INT PRIMARY KEY,
3   Name VARCHAR(100),
4   Address VARCHAR(255),
5   City VARCHAR(100),
6   State VARCHAR(100),
7   bal_due DECIMAL(10, 2)
8 );
9
10 -- Inserting sample records with Indian names into the Client_master table
11 INSERT INTO Client_master VALUES(1, 'Rahul Kumar', '123 Main St', 'New Delhi', 'Delhi', 1000.00);
12 INSERT INTO Client_master VALUES(2, 'Priya Sharma', '456 Elm St', 'Mumbai', 'Maharashtra', 2500.00);
13 INSERT INTO Client_master VALUES(3, 'Amit Patel', '789 Oak St', 'Ahmedabad', 'Gujarat', 1500.00);
14 INSERT INTO Client_master VALUES(4, 'Neha Singh', '101 Pine St', 'Bangalore', 'Karnataka', 2000.00);
15 INSERT INTO Client_master VALUES(5, 'Sneha Gupta', '202 Maple St', 'Kolkata', 'West Bengal', 5500.00);
16
17 SELECT * FROM Client_master;
18 SELECT Name FROM Client_master WHERE bal_due > 5000;
19 UPDATE Client_master SET bal_due = 5100.00 WHERE ClientNO = 2;
20
21 SELECT * FROM Client_master;
22 ALTER TABLE Client_master RENAME TO Client12;
23 SELECT ClientNO, Name, Address, City, State, bal_due AS BALANCE FROM Client12;
24
25 DESC Client12;
26
27
28
29
30

```

ClientNO	Name	Address	City	State	bal_due
1	Rahul Kumar	123 Main St	New Delhi	Delhi	1000.00
2	Priya Sharma	456 Elm St	Mumbai	Maharashtra	5100.00
3	Amit Patel	789 Oak St	Ahmedabad	Gujarat	1500.00
4	Neha Singh	101 Pine St	Bangalore	Karnataka	2000.00
5	Sneha Gupta	202 Maple St	Kolkata	West Bengal	5500.00

ClientNO	Name	Address	City	State	BALANCE
1	Rahul Kumar	123 Main St	New Delhi	Delhi	1000.00
2	Priya Sharma	456 Elm St	Mumbai	Maharashtra	5100.00
3	Amit Patel	789 Oak St	Ahmedabad	Gujarat	1500.00
4	Neha Singh	101 Pine St	Bangalore	Karnataka	2000.00
5	Sneha Gupta	202 Maple St	Kolkata	West Bengal	5500.00

Field	Type	Null	Key	Default	Extra
ClientNO	int	NO	PRI	NULL	
Name	varchar(100)	YES		NULL	
Address	varchar(255)	YES		NULL	
City	varchar(100)	YES		NULL	
State	varchar(100)	YES		NULL	
bal_due	decimal(10,2)	YES		NULL	

( e ) Display the bal\_due heading as “BALANCE”.

SELECT ClientNO, Name, Address, City, State, bal\_due AS BALANCE FROM Client12;



STDIN					
Input for the program (Optional)					
ClientNO	Name	Address	City	State	bal_due
1	Rahul Kumar	123 Main St	New Delhi	Delhi	1000.00
2	Priya Sharma	456 Elm St	Mumbai	Maharashtra	5100.00
3	Amit Patel	789 Oak St	Ahmedabad	Gujarat	1500.00
4	Neha Singh	101 Pine St	Bangalore	Karnataka	2000.00
5	Sneha Gupta	202 Maple St	Kolkata	West Bengal	5500.00
ClientNO	Name	Address	City	State	BALANCE
1	Rahul Kumar	123 Main St	New Delhi	Delhi	1000.00
2	Priya Sharma	456 Elm St	Mumbai	Maharashtra	5100.00
3	Amit Patel	789 Oak St	Ahmedabad	Gujarat	1500.00
4	Neha Singh	101 Pine St	Bangalore	Karnataka	2000.00
5	Sneha Gupta	202 Maple St	Kolkata	West Bengal	5500.00
Field	Type	Null	Key	Default	Extra
ClientNO	Int	NO	PRI	NUL	
Name	varchar(100)	YES		NUL	
Address	varchar(255)	YES		NUL	
City	varchar(100)	YES		NUL	
State	varchar(100)	YES		NUL	
bal_due	decimal(10,2)	YES		NUL	

```
-- Insert five records
```



INSERT INTO Teacher (Name, DeptNo, Joining\_Date, DeptName, Location, Salary)

VALUES

('John Smith', 101, '2022-01-15', 'Mathematics', 'Building A', 5000.00),

('Jane Doe', 102, '2022-02-20', 'Commerce', 'Building B', 5500.00),

('Michael Johnson', 103, '2021-12-10', 'Science', 'Building C', 4800.00),

('Emily Brown', 104, '2022-03-25', 'English', 'Building D', 5200.00),

('Sarah Austin', 101, '2022-04-30', 'Mathematics', 'Building A', 6000.00);

```

1  -- Start a transaction
2  START TRANSACTION;
3
4  -- Create the Teacher table
5  CREATE TABLE Teacher (
6      Name VARCHAR(100),
7      DeptNo INT,
8      Joining_Date DATE,
9      DeptName VARCHAR(100),
10     Location VARCHAR(100),
11     Salary DECIMAL(10, 2)
12 );
13
14 -- Insert five records
15 INSERT INTO Teacher (Name, DeptNo, Joining_Date, DeptName, Location, Salary)
16 VALUES
17     ('John Smith', 101, '2022-01-15', 'Mathematics', 'Building A', 5000.00),
18     ('Jane Doe', 102, '2022-02-20', 'Commerce', 'Building B', 5500.00),
19     ('Michael Johnson', 103, '2021-12-10', 'Science', 'Building C', 4800.00),
20     ('Emily Brown', 104, '2022-03-25', 'English', 'Building D', 5200.00),
21     ('Sarah Austin', 101, '2022-04-30', 'Mathematics', 'Building A', 6000.00);
22
23 -- Increment salary by 25% for Mathematics Department
24 UPDATE Teacher
25 SET Salary = Salary * 1.25
26 WHERE DeptName = 'Mathematics';
27 SELECT * FROM Teacher;
28 -- Rollback the transaction
29 ROLLBACK;
30
31 -- Start a new transaction
32 START TRANSACTION;
33
34 -- Insert five records again
35 INSERT INTO Teacher (Name, DeptNo, Joining_Date, DeptName, Location, Salary)
36 VALUES
37     ('John Smith', 101, '2022-01-15', 'Mathematics', 'Building A', 5000.00),
38     ('Jane Doe', 102, '2022-02-20', 'Commerce', 'Building B', 5500.00),
39     ('Michael Johnson', 103, '2021-12-10', 'Science', 'Building C', 4800.00),
40     ('Emily Brown', 104, '2022-03-25', 'English', 'Building D', 5200.00),
41     ('Sarah Austin', 101, '2022-04-30', 'Mathematics', 'Building A', 6000.00);
42

```

STDIN  
Input for the program ( Optional )

Output:

Name	DeptNo	Joining_Date	DeptName	Location	Salary
John Smith	101	2022-01-15	Mathematics	Building A	6250.00
Jane Doe	102	2022-02-20	Commerce	Building B	5500.00
Michael Johnson	103	2021-12-10	Science	Building C	4800.00
Emily Brown	104	2022-03-25	English	Building D	5200.00
Sarah Austin	101	2022-04-30	Mathematics	Building A	7500.00

Name	DeptNo	Joining_Date	DeptName	Location	Salary
John Smith	101	2022-01-15	Mathematics	Building A	6250.00
Jane Doe	102	2022-02-20	Commerce	Building B	6325.00
Michael Johnson	103	2021-12-10	Science	Building C	4800.00
Emily Brown	104	2022-03-25	English	Building D	5200.00
Sarah Austin	101	2022-04-30	Mathematics	Building A	7500.00
John Smith	101	2022-01-15	Mathematics	Building A	5000.00
Jane Doe	102	2022-02-20	Commerce	Building B	6325.00
Michael Johnson	103	2021-12-10	Science	Building C	4800.00
Emily Brown	104	2022-03-25	English	Building D	5200.00
Sarah Austin	101	2022-04-30	Mathematics	Building A	6000.00

( b ) Give Increment of 25% salary for Mathematics Department .

UPDATE Teacher

```

1  -- Start a transaction
2  START TRANSACTION;
3
4  -- Create the Teacher table
5  CREATE TABLE Teacher (
6      Name VARCHAR(100),
7      DeptNo INT,
8      Joining_Date DATE,
9      DeptName VARCHAR(100),
10     Location VARCHAR(100),
11     Salary DECIMAL(10, 2)
12 );
13
14 -- Insert five records
15 INSERT INTO Teacher (Name, DeptNo, Joining_Date, DeptName, Location, Salary)
16 VALUES
17     ('John Smith', 101, '2022-01-15', 'Mathematics', 'Building A', 5000.00),
18     ('Jane Doe', 102, '2022-02-20', 'Commerce', 'Building B', 5500.00),
19     ('Michael Johnson', 103, '2021-12-10', 'Science', 'Building C', 4800.00),
20     ('Emily Brown', 104, '2022-03-25', 'English', 'Building D', 5200.00),
21     ('Sarah Austin', 101, '2022-04-30', 'Mathematics', 'Building A', 6000.00);
22
23 -- Increment salary by 25% for Mathematics Department
24 UPDATE Teacher
25 SET Salary = Salary * 1.25
26 WHERE DeptName = 'Mathematics';
27 SELECT * FROM Teacher;
28 -- Rollback the transaction
29 ROLLBACK;
30
31 -- Start a new transaction
32 START TRANSACTION;
33
34 -- Insert five records again
35 INSERT INTO Teacher (Name, DeptNo, Joining_Date, DeptName, Location, Salary)
36 VALUES
37     ('John Smith', 101, '2022-01-15', 'Mathematics', 'Building A', 5000.00),
38     ('Jane Doe', 102, '2022-02-20', 'Commerce', 'Building B', 5500.00),
39     ('Michael Johnson', 103, '2021-12-10', 'Science', 'Building C', 4800.00),
40     ('Emily Brown', 104, '2022-03-25', 'English', 'Building D', 5200.00),
41     ('Sarah Austin', 101, '2022-04-30', 'Mathematics', 'Building A', 6000.00);
42

```

STDIN  
Input for the program ( Optional )

Output:

Name	DeptNo	Joining_Date	DeptName	Location	Salary
John Smith	101	2022-01-15	Mathematics	Building A	6250.00
Jane Doe	102	2022-02-20	Commerce	Building B	5500.00
Michael Johnson	103	2021-12-10	Science	Building C	4800.00
Emily Brown	104	2022-03-25	English	Building D	5200.00
Sarah Austin	101	2022-04-30	Mathematics	Building A	7500.00

Name	DeptNo	Joining_Date	DeptName	Location	Salary
John Smith	101	2022-01-15	Mathematics	Building A	6250.00
Jane Doe	102	2022-02-20	Commerce	Building B	6325.00
Michael Johnson	103	2021-12-10	Science	Building C	4800.00
Emily Brown	104	2022-03-25	English	Building D	5200.00
Sarah Austin	101	2022-04-30	Mathematics	Building A	7500.00
John Smith	101	2022-01-15	Mathematics	Building A	5000.00
Jane Doe	102	2022-02-20	Commerce	Building B	6325.00
Michael Johnson	103	2021-12-10	Science	Building C	4800.00
Emily Brown	104	2022-03-25	English	Building D	5200.00
Sarah Austin	101	2022-04-30	Mathematics	Building A	6000.00

SET Salary = Salary \* 1.25

WHERE DeptName = 'Mathematics';

SELECT \* FROM Teacher;

```

1  -- Start a transaction
2  START TRANSACTION;
3
4  -- Create the Teacher table
5  CREATE TABLE Teacher (
6      Name VARCHAR(100),
7      DeptNo INT,
8      Joining_Date DATE,
9      DeptName VARCHAR(100),
10     Location VARCHAR(100),
11     Salary DECIMAL(10, 2)
12 );
13
14 -- Insert five records
15 INSERT INTO Teacher (Name, DeptNo, Joining_Date, DeptName, Location, Salary)
16 VALUES
17     ('John Smith', 101, '2022-01-15', 'Mathematics', 'Building A', 5000.00),
18     ('Jane Doe', 102, '2022-02-20', 'Commerce', 'Building B', 5500.00),
19     ('Michael Johnson', 103, '2021-12-10', 'Science', 'Building C', 4800.00),
20     ('Emily Brown', 104, '2022-03-25', 'English', 'Building D', 5200.00),
21     ('Sarah Austin', 101, '2022-04-30', 'Mathematics', 'Building A', 6000.00);
22
23 -- Increment salary by 25% for Mathematics Department
24 UPDATE Teacher
25 SET Salary = Salary * 1.25
26 WHERE DeptName = 'Mathematics';
27 SELECT * FROM Teacher;
28 -- Rollback the transaction
29 ROLLBACK;
30
31 -- Start a new transaction
32 START TRANSACTION;
33
34 -- Insert five records again
35 INSERT INTO Teacher (Name, DeptNo, Joining_Date, DeptName, Location, Salary)
36 VALUES
37     ('John Smith', 101, '2022-01-15', 'Mathematics', 'Building A', 5000.00),
38     ('Jane Doe', 102, '2022-02-20', 'Commerce', 'Building B', 5500.00),
39     ('Michael Johnson', 103, '2021-12-10', 'Science', 'Building C', 4800.00),
40     ('Emily Brown', 104, '2022-03-25', 'English', 'Building D', 5200.00),
41     ('Sarah Austin', 101, '2022-04-30', 'Mathematics', 'Building A', 6000.00);
42

```

STDIN  
Input for the program ( Optional )

Output:

Name	DeptNo	Joining_Date	DeptName	Location	Salary
John Smith	101	2022-01-15	Mathematics	Building A	6250.00
Jane Doe	102	2022-02-20	Commerce	Building B	5500.00
Michael Johnson	103	2021-12-10	Science	Building C	4800.00
Emily Brown	104	2022-03-25	English	Building D	5200.00
Sarah Austin	101	2022-04-30	Mathematics	Building A	7500.00

Name	DeptNo	Joining_Date	DeptName	Location	Salary
John Smith	101	2022-01-15	Mathematics	Building A	6250.00
Jane Doe	102	2022-02-20	Commerce	Building B	6325.00
Michael Johnson	103	2021-12-10	Science	Building C	4800.00
Emily Brown	104	2022-03-25	English	Building D	5200.00
Sarah Austin	101	2022-04-30	Mathematics	Building A	7500.00
John Smith	101	2022-01-15	Mathematics	Building A	5000.00
Jane Doe	102	2022-02-20	Commerce	Building B	6325.00
Michael Johnson	103	2021-12-10	Science	Building C	4800.00
Emily Brown	104	2022-03-25	English	Building D	5200.00
Sarah Austin	101	2022-04-30	Mathematics	Building A	6000.00

( c ) Perform Rollback command

ROLLBACK;

```

1  -- Start a transaction
2  START TRANSACTION;
3
4  -- Create the Teacher table
5  CREATE TABLE Teacher (
6      Name VARCHAR(100),
7      DeptNo INT,
8      Joining_Date DATE,
9      DeptName VARCHAR(100),
10     Location VARCHAR(100),
11     Salary DECIMAL(10, 2)
12 );
13
14 -- Insert five records
15 INSERT INTO Teacher (Name, DeptNo, Joining_Date, DeptName, Location, Salary)
16 VALUES
17     ('John Smith', 101, '2022-01-15', 'Mathematics', 'Building A', 5000.00),
18     ('Jane Doe', 102, '2022-02-20', 'Commerce', 'Building B', 5500.00),
19     ('Michael Johnson', 103, '2021-12-10', 'Science', 'Building C', 4800.00),
20     ('Emily Brown', 104, '2022-03-25', 'English', 'Building D', 5200.00),
21     ('Sarah Austin', 101, '2022-04-30', 'Mathematics', 'Building A', 6000.00);
22
23 -- Increment salary by 25% for Mathematics Department
24 UPDATE Teacher
25 SET Salary = Salary * 1.25
26 WHERE DeptName = 'Mathematics';
27 SELECT * FROM Teacher;
28 -- Rollback the transaction
29 ROLLBACK;
30
31 -- Start a new transaction
32 START TRANSACTION;
33
34 -- Insert five records again
35 INSERT INTO Teacher (Name, DeptNo, Joining_Date, DeptName, Location, Salary)
36 VALUES
37     ('John Smith', 101, '2022-01-15', 'Mathematics', 'Building A', 5000.00),
38     ('Jane Doe', 102, '2022-02-20', 'Commerce', 'Building B', 5500.00),
39     ('Michael Johnson', 103, '2021-12-10', 'Science', 'Building C', 4800.00),
40     ('Emily Brown', 104, '2022-03-25', 'English', 'Building D', 5200.00),
41     ('Sarah Austin', 101, '2022-04-30', 'Mathematics', 'Building A', 6000.00);
42

```

STDIN  
Input for the program ( Optional )

Output:

Name	DeptNo	Joining_Date	DeptName	Location	Salary
John Smith	101	2022-01-15	Mathematics	Building A	6250.00
Jane Doe	102	2022-02-20	Commerce	Building B	5500.00
Michael Johnson	103	2021-12-10	Science	Building C	4800.00
Emily Brown	104	2022-03-25	English	Building D	5200.00
Sarah Austin	101	2022-04-30	Mathematics	Building A	7500.00

Name	DeptNo	Joining_Date	DeptName	Location	Salary
John Smith	101	2022-01-15	Mathematics	Building A	6250.00
Jane Doe	102	2022-02-20	Commerce	Building B	6325.00
Michael Johnson	103	2021-12-10	Science	Building C	4800.00
Emily Brown	104	2022-03-25	English	Building D	5200.00
Sarah Austin	101	2022-04-30	Mathematics	Building A	7500.00
John Smith	101	2022-01-15	Mathematics	Building A	5000.00
Jane Doe	102	2022-02-20	Commerce	Building B	6325.00
Michael Johnson	103	2021-12-10	Science	Building C	4800.00
Emily Brown	104	2022-03-25	English	Building D	5200.00
Sarah Austin	101	2022-04-30	Mathematics	Building A	6000.00

( d ) Give Increment of 15% salary for Commerce Department

START TRANSACTION;

-- Insert five records again

INSERT INTO Teacher (Name, DeptNo, Joining\_Date, DeptName, Location, Salary)

VALUES

('John Smith', 101, '2022-01-15', 'Mathematics', 'Building A', 5000.00),

('Jane Doe', 102, '2022-02-20', 'Commerce', 'Building B', 5500.00),

('Michael Johnson', 103, '2021-12-10', 'Science', 'Building C', 4800.00),

('Emily Brown', 104, '2022-03-25', 'English', 'Building D', 5200.00),

('Sarah Austin', 101, '2022-04-30', 'Mathematics', 'Building A', 6000.00);

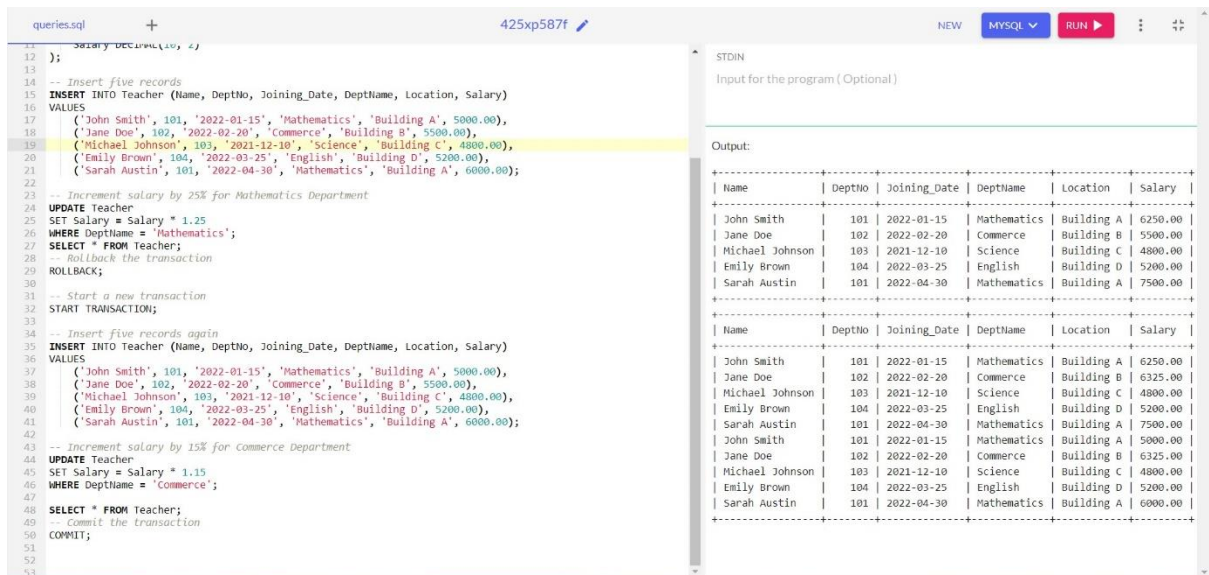
-- Increment salary by 15% for Commerce Department

UPDATE Teacher

SET Salary = Salary \* 1.15

WHERE DeptName = 'Commerce';

SELECT \* FROM Teacher;



The screenshot shows a MySQL IDE window with a query editor on the left and an output pane on the right. The query editor contains SQL code for inserting and updating teacher records. The output pane displays the results of the queries, showing the state of the Teacher table after each operation.

```
queries.sql 425xp587f
11 SET salary DECIMAL(10, 2);
12 );
13
14 -- Insert five records
15 INSERT INTO Teacher (Name, DeptNo, Joining_Date, DeptName, Location, Salary)
16 VALUES
17 ('John Smith', 101, '2022-01-15', 'Mathematics', 'Building A', 5000.00),
18 ('Jane Doe', 102, '2022-02-20', 'Commerce', 'Building B', 5500.00),
19 ('Michael Johnson', 103, '2021-12-10', 'Science', 'Building C', 4800.00),
20 ('Emily Brown', 104, '2022-03-25', 'English', 'Building D', 5200.00),
21 ('Sarah Austin', 101, '2022-04-30', 'Mathematics', 'Building A', 6000.00);
22
23 -- Increment salary by 25% for Mathematics Department
24 UPDATE Teacher
25 SET Salary = Salary * 1.25
26 WHERE DeptName = 'Mathematics';
27 SELECT * FROM Teacher;
28 -- Rollback the transaction
29 ROLLBACK;
30
31 -- Start a new transaction
32 START TRANSACTION;
33
34 -- Insert five records again
35 INSERT INTO Teacher (Name, DeptNo, Joining_Date, DeptName, Location, Salary)
36 VALUES
37 ('John Smith', 101, '2022-01-15', 'Mathematics', 'Building A', 5000.00),
38 ('Jane Doe', 102, '2022-02-20', 'Commerce', 'Building B', 5500.00),
39 ('Michael Johnson', 103, '2021-12-10', 'Science', 'Building C', 4800.00),
40 ('Emily Brown', 104, '2022-03-25', 'English', 'Building D', 5200.00),
41 ('Sarah Austin', 101, '2022-04-30', 'Mathematics', 'Building A', 6000.00);
42
43 -- Increment salary by 15% for Commerce Department
44 UPDATE Teacher
45 SET Salary = Salary * 1.15
46 WHERE DeptName = 'Commerce';
47
48 SELECT * FROM Teacher;
49 -- Commit the transaction
50 COMMIT;
51
52
53
```

Output:

Name	DeptNo	Joining_Date	DeptName	Location	Salary
John Smith	101	2022-01-15	Mathematics	Building A	6250.00
Jane Doe	102	2022-02-20	Commerce	Building B	5500.00
Michael Johnson	103	2021-12-10	Science	Building C	4800.00
Emily Brown	104	2022-03-25	English	Building D	5200.00
Sarah Austin	101	2022-04-30	Mathematics	Building A	7500.00

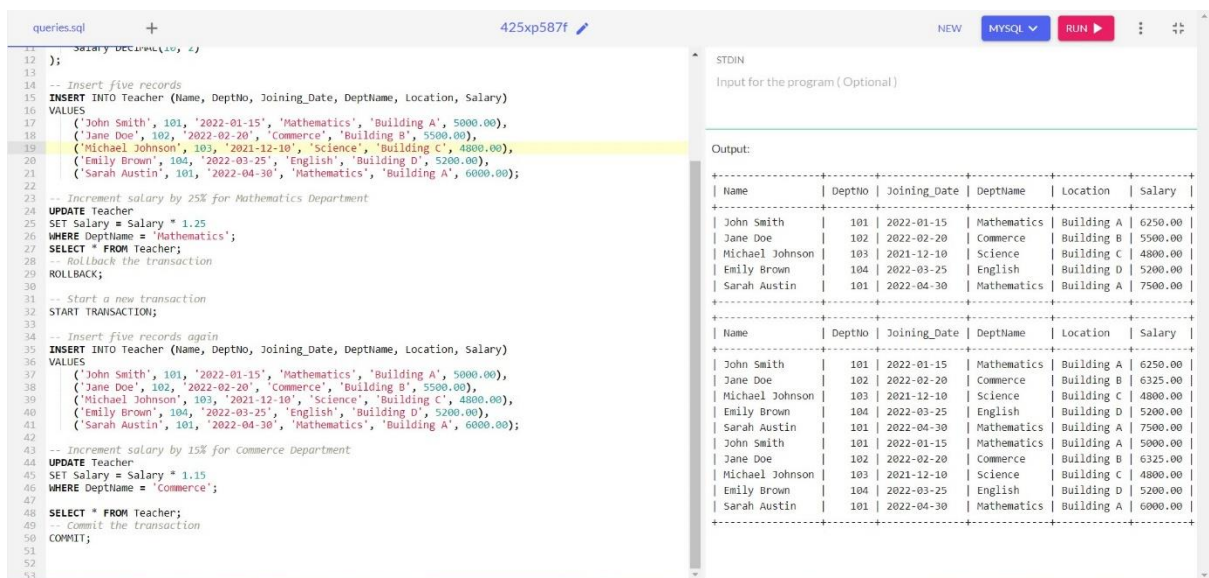
  

Name	DeptNo	Joining_Date	DeptName	Location	Salary
John Smith	101	2022-01-15	Mathematics	Building A	6250.00
Jane Doe	102	2022-02-20	Commerce	Building B	6325.00
Michael Johnson	103	2021-12-10	Science	Building C	4800.00
Emily Brown	104	2022-03-25	English	Building D	5200.00
Sarah Austin	101	2022-04-30	Mathematics	Building A	7500.00
John Smith	101	2022-01-15	Mathematics	Building A	5000.00
Jane Doe	102	2022-02-20	Commerce	Building B	6325.00
Michael Johnson	103	2021-12-10	Science	Building C	4800.00
Emily Brown	104	2022-03-25	English	Building D	5200.00
Sarah Austin	101	2022-04-30	Mathematics	Building A	6000.00

( e ) Perform commit command.

-- Commit the transaction

COMMIT;



The screenshot shows a MySQL IDE window with a query editor on the left and an output pane on the right. The query editor contains SQL code for inserting and updating teacher records. The output pane displays the results of the queries, showing the state of the Teacher table after each operation.

```
queries.sql 425xp587f
11 SET salary DECIMAL(10, 2);
12 );
13
14 -- Insert five records
15 INSERT INTO Teacher (Name, DeptNo, Joining_Date, DeptName, Location, Salary)
16 VALUES
17 ('John Smith', 101, '2022-01-15', 'Mathematics', 'Building A', 5000.00),
18 ('Jane Doe', 102, '2022-02-20', 'Commerce', 'Building B', 5500.00),
19 ('Michael Johnson', 103, '2021-12-10', 'Science', 'Building C', 4800.00),
20 ('Emily Brown', 104, '2022-03-25', 'English', 'Building D', 5200.00),
21 ('Sarah Austin', 101, '2022-04-30', 'Mathematics', 'Building A', 6000.00);
22
23 -- Increment salary by 25% for Mathematics Department
24 UPDATE Teacher
25 SET Salary = Salary * 1.25
26 WHERE DeptName = 'Mathematics';
27 SELECT * FROM Teacher;
28 -- Rollback the transaction
29 ROLLBACK;
30
31 -- Start a new transaction
32 START TRANSACTION;
33
34 -- Insert five records again
35 INSERT INTO Teacher (Name, DeptNo, Joining_Date, DeptName, Location, Salary)
36 VALUES
37 ('John Smith', 101, '2022-01-15', 'Mathematics', 'Building A', 5000.00),
38 ('Jane Doe', 102, '2022-02-20', 'Commerce', 'Building B', 5500.00),
39 ('Michael Johnson', 103, '2021-12-10', 'Science', 'Building C', 4800.00),
40 ('Emily Brown', 104, '2022-03-25', 'English', 'Building D', 5200.00),
41 ('Sarah Austin', 101, '2022-04-30', 'Mathematics', 'Building A', 6000.00);
42
43 -- Increment salary by 15% for Commerce Department
44 UPDATE Teacher
45 SET Salary = Salary * 1.15
46 WHERE DeptName = 'Commerce';
47
48 SELECT * FROM Teacher;
49 -- Commit the transaction
50 COMMIT;
51
52
53
```

Output:

Name	DeptNo	Joining_Date	DeptName	Location	Salary
John Smith	101	2022-01-15	Mathematics	Building A	6250.00
Jane Doe	102	2022-02-20	Commerce	Building B	5500.00
Michael Johnson	103	2021-12-10	Science	Building C	4800.00
Emily Brown	104	2022-03-25	English	Building D	5200.00
Sarah Austin	101	2022-04-30	Mathematics	Building A	7500.00

Name	DeptNo	Joining_Date	DeptName	Location	Salary
John Smith	101	2022-01-15	Mathematics	Building A	6250.00
Jane Doe	102	2022-02-20	Commerce	Building B	6325.00
Michael Johnson	103	2021-12-10	Science	Building C	4800.00
Emily Brown	104	2022-03-25	English	Building D	5200.00
Sarah Austin	101	2022-04-30	Mathematics	Building A	7500.00
John Smith	101	2022-01-15	Mathematics	Building A	5000.00
Jane Doe	102	2022-02-20	Commerce	Building B	6325.00
Michael Johnson	103	2021-12-10	Science	Building C	4800.00
Emily Brown	104	2022-03-25	English	Building D	5200.00
Sarah Austin	101	2022-04-30	Mathematics	Building A	6000.00

4. (Exercise on order by and group by clauses) Create Sales table with the following fields( Sales No, Salesname, Branch, Salesamount, DOB)

( a ) Insert five records

ANSWER:

-- Create the Sales table

```
CREATE TABLE Sales (  
    Sales_No INT PRIMARY KEY,  
    Salesname VARCHAR(100),  
    Branch VARCHAR(100),  
    Salesamount DECIMAL(10, 2),  
    DOB DATE  
);
```

-- Insert five records into the Sales table

```
INSERT INTO Sales (Sales_No, Salesname, Branch, Salesamount, DOB) VALUES  
(1, 'John', 'Branch A', 1500.00, '1990-05-15'),  
(2, 'Alice', 'Branch B', 2000.00, '1988-08-20'),  
(3, 'Bob', 'Branch A', 1800.00, '1995-12-10'),  
(4, 'Emily', 'Branch C', 2200.00, '1992-03-25'),  
(5, 'David', 'Branch B', 1900.00, '1993-12-05');
```

queries.sql
+
425xp587f
NEW
MYSQL
RUN

```

1 -- Create the Sales table
2 CREATE TABLE Sales (
3     Sales_No INT PRIMARY KEY,
4     Salesname VARCHAR(100),
5     Branch VARCHAR(100),
6     Salesamount DECIMAL(10, 2),
7     DOB DATE
8 );
9
10 -- Insert five records into the Sales table
11 INSERT INTO Sales (Sales_No, Salesname, Branch, Salesamount, DOB) VALUES
12     (1, 'John', 'Branch A', 1500.00, '1990-05-15'),
13     (2, 'Alice', 'Branch B', 2000.00, '1988-08-20'),
14     (3, 'Bob', 'Branch A', 1800.00, '1995-12-10'),
15     (4, 'Emily', 'Branch C', 2200.00, '1992-03-25'),
16     (5, 'David', 'Branch B', 1900.00, '1993-12-05');
17
18 -- Calculate total sales amount in each branch
19 SELECT Branch, SUM(Salesamount) AS Total_Sales_Amount
20 FROM Sales
21 GROUP BY Branch;
22
23 -- Calculate average sales amount in each branch
24 SELECT Branch, AVG(Salesamount) AS Average_Sales_Amount
25 FROM Sales
26 GROUP BY Branch;
27
28 -- Display salesmen and DOB born in December
29 SELECT Salesname, DATE_FORMAT(DOB, '%d-%b-%y') AS DOB
30 FROM Sales
31 WHERE MONTH(DOB) = 12;
32
33 -- Display name and DOB of salesmen in alphabetical order of the month
34 SELECT Salesname, DATE_FORMAT(DOB, '%d-%b-%y') AS DOB
35 FROM Sales
36 ORDER BY MONTH(DOB), DAY(DOB), Salesname;
37
38
39

```

STDIN

Input for the program ( Optional )

Branch	Total_Sales_Amount
Branch A	3300.00
Branch B	3900.00
Branch C	2200.00

Branch	Average_Sales_Amount
Branch A	1650.000000
Branch B	1950.000000
Branch C	2200.000000

Salesname	DOB
Bob	10-Dec-95
David	05-Dec-93

Salesname	DOB
Emily	25-Mar-92
John	15-May-90
Alice	20-Aug-88
David	05-Dec-93
Bob	10-Dec-95

( b ) Calculate total salesamount in each branch

SELECT Branch, SUM(Salesamount) AS Total\_Sales\_Amount

FROM Sales

GROUP BY Branch;

queries.sql
+
425xp587f
NEW
MYSQL
RUN

```

1 -- Create the Sales table
2 CREATE TABLE Sales (
3     Sales_No INT PRIMARY KEY,
4     Salesname VARCHAR(100),
5     Branch VARCHAR(100),
6     Salesamount DECIMAL(10, 2),
7     DOB DATE
8 );
9
10 -- Insert five records into the Sales table
11 INSERT INTO Sales (Sales_No, Salesname, Branch, Salesamount, DOB) VALUES
12     (1, 'John', 'Branch A', 1500.00, '1990-05-15'),
13     (2, 'Alice', 'Branch B', 2000.00, '1988-08-20'),
14     (3, 'Bob', 'Branch A', 1800.00, '1995-12-10'),
15     (4, 'Emily', 'Branch C', 2200.00, '1992-03-25'),
16     (5, 'David', 'Branch B', 1900.00, '1993-12-05');
17
18 -- Calculate total sales amount in each branch
19 SELECT Branch, SUM(Salesamount) AS Total_Sales_Amount
20 FROM Sales
21 GROUP BY Branch;
22
23 -- Calculate average sales amount in each branch
24 SELECT Branch, AVG(Salesamount) AS Average_Sales_Amount
25 FROM Sales
26 GROUP BY Branch;
27
28 -- Display salesmen and DOB born in December
29 SELECT Salesname, DATE_FORMAT(DOB, '%d-%b-%y') AS DOB
30 FROM Sales
31 WHERE MONTH(DOB) = 12;
32
33 -- Display name and DOB of salesmen in alphabetical order of the month
34 SELECT Salesname, DATE_FORMAT(DOB, '%d-%b-%y') AS DOB
35 FROM Sales
36 ORDER BY MONTH(DOB), DAY(DOB), Salesname;
37
38
39

```

STDIN

Input for the program ( Optional )

Branch	Total_Sales_Amount
Branch A	3300.00
Branch B	3900.00
Branch C	2200.00

Branch	Average_Sales_Amount
Branch A	1650.000000
Branch B	1950.000000
Branch C	2200.000000

Salesname	DOB
Bob	10-Dec-95
David	05-Dec-93

Salesname	DOB
Emily	25-Mar-92
John	15-May-90
Alice	20-Aug-88
David	05-Dec-93
Bob	10-Dec-95

( c ) Calculate average salesamount in each branch .

SELECT Branch, AVG(Salesamount) AS Average\_Sales\_Amount



FROM Sales

GROUP BY Branch;

The screenshot shows a MySQL IDE with a SQL editor on the left and a results pane on the right. The SQL editor contains the following queries:

```
1 -- Create the Sales table
2 CREATE TABLE Sales (
3   Sales_No INT PRIMARY KEY,
4   Salesname VARCHAR(100),
5   Branch VARCHAR(100),
6   Salesamount DECIMAL(10, 2),
7   DOB DATE
8 );
9
10 -- Insert five records into the Sales table
11 INSERT INTO Sales (Sales_No, Salesname, Branch, Salesamount, DOB) VALUES
12 (1, 'John', 'Branch A', 1500.00, '1990-05-15'),
13 (2, 'Alice', 'Branch B', 2000.00, '1988-08-20'),
14 (3, 'Bob', 'Branch A', 1800.00, '1995-12-10'),
15 (4, 'Emily', 'Branch C', 2200.00, '1992-03-25'),
16 (5, 'David', 'Branch B', 1900.00, '1993-12-05');
17
18 -- Calculate total sales amount in each branch
19 SELECT Branch, SUM(Salesamount) AS Total_Sales_Amount
20 FROM Sales
21 GROUP BY Branch;
22
23 -- Calculate average sales amount in each branch
24 SELECT Branch, AVG(Salesamount) AS Average_Sales_Amount
25 FROM Sales
26 GROUP BY Branch;
27
28 -- Display salesmen and DOB born in December
29 SELECT Salesname, DATE_FORMAT(DOB, '%d-%b-%y') AS DOB
30 FROM Sales
31 WHERE MONTH(DOB) = 12;
32
33 -- Display name and DOB of salesmen in alphabetical order of the month
34 SELECT Salesname, DATE_FORMAT(DOB, '%d-%b-%y') AS DOB
35 FROM Sales
36 ORDER BY MONTH(DOB), DAY(DOB), Salesname;
```

The results pane shows the output of the queries:

Branch	Total_Sales_Amount
Branch A	3300.00
Branch B	3900.00
Branch C	2200.00

Branch	Average_Sales_Amount
Branch A	1650.000000
Branch B	1950.000000
Branch C	2200.000000

Salesname	DOB
Bob	10-Dec-95
David	05-Dec-93

Salesname	DOB
Emily	25-Mar-92
John	15-May-90
Alice	20-Aug-88
David	05-Dec-93
Bob	10-Dec-95

( d ) Display all the salesmen, DOB who are born in the month of December as day in character format i.e. 21-Dec-09

SELECT Salesname, DATE\_FORMAT(DOB, '%d-%b-%y') AS DOB

FROM Sales

WHERE MONTH(DOB) = 12;

The screenshot shows a MySQL IDE with a SQL editor on the left and a results pane on the right. The SQL editor contains the following queries:

```
1 -- Create the Sales table
2 CREATE TABLE Sales (
3   Sales_No INT PRIMARY KEY,
4   Salesname VARCHAR(100),
5   Branch VARCHAR(100),
6   Salesamount DECIMAL(10, 2),
7   DOB DATE
8 );
9
10 -- Insert five records into the Sales table
11 INSERT INTO Sales (Sales_No, Salesname, Branch, Salesamount, DOB) VALUES
12 (1, 'John', 'Branch A', 1500.00, '1990-05-15'),
13 (2, 'Alice', 'Branch B', 2000.00, '1988-08-20'),
14 (3, 'Bob', 'Branch A', 1800.00, '1995-12-10'),
15 (4, 'Emily', 'Branch C', 2200.00, '1992-03-25'),
16 (5, 'David', 'Branch B', 1900.00, '1993-12-05');
17
18 -- Calculate total sales amount in each branch
19 SELECT Branch, SUM(Salesamount) AS Total_Sales_Amount
20 FROM Sales
21 GROUP BY Branch;
22
23 -- Calculate average sales amount in each branch
24 SELECT Branch, AVG(Salesamount) AS Average_Sales_Amount
25 FROM Sales
26 GROUP BY Branch;
27
28 -- Display salesmen and DOB born in December
29 SELECT Salesname, DATE_FORMAT(DOB, '%d-%b-%y') AS DOB
30 FROM Sales
31 WHERE MONTH(DOB) = 12;
32
33 -- Display name and DOB of salesmen in alphabetical order of the month
34 SELECT Salesname, DATE_FORMAT(DOB, '%d-%b-%y') AS DOB
35 FROM Sales
36 ORDER BY MONTH(DOB), DAY(DOB), Salesname;
```

The results pane shows the output of the queries:

Branch	Total_Sales_Amount
Branch A	3300.00
Branch B	3900.00
Branch C	2200.00

Branch	Average_Sales_Amount
Branch A	1650.000000
Branch B	1950.000000
Branch C	2200.000000

Salesname	DOB
Bob	10-Dec-95
David	05-Dec-93

Salesname	DOB
Emily	25-Mar-92
John	15-May-90
Alice	20-Aug-88
David	05-Dec-93
Bob	10-Dec-95



( e ) Display the name and DOB of salesman in alphabetical order of the month.

```
SELECT Salesname, DATE_FORMAT(DOB, '%d-%b-%y') AS DOB
```

```
FROM Sales
```

```
ORDER BY MONTH(DOB), DAY(DOB), Salesname;
```

The screenshot shows a MySQL query editor with a SQL script and its execution results. The script includes creating a 'Sales' table, inserting five records, and three queries: calculating total sales by branch, average sales by branch, and displaying salesmen's names and DOBs in alphabetical order by month.

```
1 -- Create the Sales table
2 CREATE TABLE Sales (
3     Sales_No INT PRIMARY KEY,
4     Salesname VARCHAR(100),
5     Branch VARCHAR(100),
6     Salesamount DECIMAL(10, 2),
7     DOB DATE
8 );
9
10 -- Insert five records into the sales table
11 INSERT INTO Sales (Sales_No, Salesname, Branch, Salesamount, DOB) VALUES
12 (1, 'John', 'Branch A', 1500.00, '1990-05-15'),
13 (2, 'Alice', 'Branch B', 2000.00, '1988-08-20'),
14 (3, 'Bob', 'Branch A', 1800.00, '1995-12-10'),
15 (4, 'Emily', 'Branch C', 2200.00, '1992-03-25'),
16 (5, 'David', 'Branch B', 1900.00, '1993-12-05');
17
18 -- Calculate total sales amount in each branch
19 SELECT Branch, SUM(Salesamount) AS Total_Sales_Amount
20 FROM Sales
21 GROUP BY Branch;
22
23 -- Calculate average sales amount in each branch
24 SELECT Branch, AVG(Salesamount) AS Average_Sales_Amount
25 FROM Sales
26 GROUP BY Branch;
27
28 -- Display salesmen and DOB born in December
29 SELECT Salesname, DATE_FORMAT(DOB, '%d-%b-%y') AS DOB
30 FROM Sales
31 WHERE MONTH(DOB) = 12;
32
33 -- Display name and DOB of salesmen in alphabetical order of the month
34 SELECT Salesname, DATE_FORMAT(DOB, '%d-%b-%y') AS DOB
35 FROM Sales
36 ORDER BY MONTH(DOB), DAY(DOB), Salesname;
```

The results pane shows the output of the queries:

Branch	Total_Sales_Amount
Branch A	3300.00
Branch B	3900.00
Branch C	2200.00

Branch	Average_Sales_Amount
Branch A	1650.000000
Branch B	1950.000000
Branch C	2200.000000

Salesname	DOB
Bob	10-Dec-95
David	05-Dec-93

Salesname	DOB
Emily	25-Mar-92
John	15-May-90
Alice	20-Aug-88
David	05-Dec-93
Bob	10-Dec-95

5. Create an Emp table with the following fields: (EmpNo, EmpName, Job,Basic, DA, HRA,PF, GrossPay, NetPay) (Calculate DA as 30% of Basic and HRA as 40% of Basic)

( a ) Insert Five Records and calculate GrossPay and NetPay.

-- Create the Emp table

```
CREATE TABLE Emp (
```

```
    EmpNo INT PRIMARY KEY,
```

```
    EmpName VARCHAR(100),
```

```
    Job VARCHAR(100),
```

```

    Basic DECIMAL(10, 2),
    DA DECIMAL(10, 2),
    HRA DECIMAL(10, 2),
    PF DECIMAL(10, 2),
    GrossPay DECIMAL(10, 2),
    NetPay DECIMAL(10, 2)
);

-- Insert Five Records and calculate GrossPay and NetPay

INSERT INTO Emp (EmpNo, EmpName, Job, Basic, DA, HRA, PF, GrossPay,
NetPay) VALUES

(1, 'John', 'Manager', 15000.00, 4500.00, 6000.00, 1800.00, NULL, NULL),
(2, 'Alice', 'Clerk', 10000.00, 3000.00, 4000.00, 1200.00, NULL, NULL),
(3, 'Bob', 'Technician', 12000.00, 3600.00, 4800.00, 1440.00, NULL, NULL),
(4, 'Emily', 'Engineer', 18000.00, 5400.00, 7200.00, 2160.00, NULL, NULL),
(5, 'David', 'Analyst', 20000.00, 6000.00, 8000.00, 2400.00, NULL, NULL);

-- Calculate GrossPay and NetPay

UPDATE Emp SET GrossPay = Basic + DA + HRA, NetPay = Basic + DA + HRA - PF;

```

```

queries.sql + 425xp587f NEW MySQL RUN
1 -- Create the Emp table
2 CREATE TABLE Emp (
3     EmpNo INT PRIMARY KEY,
4     EmpName VARCHAR(100),
5     Job VARCHAR(100),
6     Basic DECIMAL(10, 2),
7     DA DECIMAL(10, 2),
8     HRA DECIMAL(10, 2),
9     PF DECIMAL(10, 2),
10    GrossPay DECIMAL(10, 2),
11    NetPay DECIMAL(10, 2)
12 );
13
14 -- Insert Five Records and calculate GrossPay and NetPay
15 INSERT INTO Emp (EmpNo, EmpName, Job, Basic, DA, HRA, PF, GrossPay, NetPay) VALUES
16 (1, 'John', 'Manager', 15000.00, 4500.00, 6000.00, 1800.00, NULL, NULL),
17 (2, 'Alice', 'Clerk', 10000.00, 3000.00, 4000.00, 1200.00, NULL, NULL),
18 (3, 'Bob', 'Technician', 12000.00, 3600.00, 4800.00, 1440.00, NULL, NULL),
19 (4, 'Emily', 'Engineer', 18000.00, 5400.00, 7200.00, 2160.00, NULL, NULL),
20 (5, 'David', 'Analyst', 20000.00, 6000.00, 8000.00, 2400.00, NULL, NULL);
21
22 -- Calculate GrossPay and NetPay
23 UPDATE Emp SET GrossPay = Basic + DA + HRA, NetPay = Basic + DA + HRA - PF;
24
25 -- Display the employees whose Basic is lowest in each department
26 SELECT EmpNo, EmpName, Job, Basic FROM Emp e1
27 WHERE Basic = (SELECT MIN(Basic) FROM Emp e2 WHERE e1.Job = e2.Job);
28
29 -- If NetPay is less than <rs 10,000 add rs.1200 as special allowances
30 UPDATE Emp SET NetPay = NetPay + 1200
31 WHERE NetPay < 10000;
32
33 -- Display the employees whose grosspay lies between 10,000 & 20,000
34 SELECT EmpNo, EmpName, Job, GrossPay
35 FROM Emp
36 WHERE GrossPay BETWEEN 10000 AND 20000;
37
38 -- Display all the employees who earn maximum salary
39 SELECT EmpNo, EmpName, Job, Basic
40 FROM Emp
41 WHERE Basic = (SELECT MAX(Basic) FROM Emp);
42

```

STDIN  
Input for the program ( Optional )

Output:

EmpNo	EmpName	Job	Basic
1	John	Manager	15000.00
2	Alice	Clerk	10000.00
3	Bob	Technician	12000.00
4	Emily	Engineer	18000.00
5	David	Analyst	20000.00

EmpNo	EmpName	Job	GrossPay
2	Alice	Clerk	17000.00

EmpNo	EmpName	Job	Basic
5	David	Analyst	20000.00

( b ) Display the employees whose Basic is lowest in each department .

-- Display the employees whose Basic is lowest in each department

SELECT EmpNo, EmpName, Job, Basic FROM Emp e1

WHERE Basic = (SELECT MIN(Basic) FROM Emp e2 WHERE e1.Job = e2.Job);

```

queries.sql + 425xp587f NEW MySQL RUN
1 -- Create the Emp table
2 CREATE TABLE Emp (
3     EmpNo INT PRIMARY KEY,
4     EmpName VARCHAR(100),
5     Job VARCHAR(100),
6     Basic DECIMAL(10, 2),
7     DA DECIMAL(10, 2),
8     HRA DECIMAL(10, 2),
9     PF DECIMAL(10, 2),
10    GrossPay DECIMAL(10, 2),
11    NetPay DECIMAL(10, 2)
12 );
13
14 -- Insert Five Records and calculate GrossPay and NetPay
15 INSERT INTO Emp (EmpNo, EmpName, Job, Basic, DA, HRA, PF, GrossPay, NetPay) VALUES
16 (1, 'John', 'Manager', 15000.00, 4500.00, 6000.00, 1800.00, NULL, NULL),
17 (2, 'Alice', 'Clerk', 10000.00, 3000.00, 4000.00, 1200.00, NULL, NULL),
18 (3, 'Bob', 'Technician', 12000.00, 3600.00, 4800.00, 1440.00, NULL, NULL),
19 (4, 'Emily', 'Engineer', 18000.00, 5400.00, 7200.00, 2160.00, NULL, NULL),
20 (5, 'David', 'Analyst', 20000.00, 6000.00, 8000.00, 2400.00, NULL, NULL);
21
22 -- Calculate GrossPay and NetPay
23 UPDATE Emp SET GrossPay = Basic + DA + HRA, NetPay = Basic + DA + HRA - PF;
24
25 -- Display the employees whose Basic is lowest in each department
26 SELECT EmpNo, EmpName, Job, Basic FROM Emp e1
27 WHERE Basic = (SELECT MIN(Basic) FROM Emp e2 WHERE e1.Job = e2.Job);
28
29 -- If NetPay is less than <rs 10,000 add rs.1200 as special allowances
30 UPDATE Emp SET NetPay = NetPay + 1200
31 WHERE NetPay < 10000;
32
33 -- Display the employees whose grosspay lies between 10,000 & 20,000
34 SELECT EmpNo, EmpName, Job, GrossPay
35 FROM Emp
36 WHERE GrossPay BETWEEN 10000 AND 20000;
37
38 -- Display all the employees who earn maximum salary
39 SELECT EmpNo, EmpName, Job, Basic
40 FROM Emp
41 WHERE Basic = (SELECT MAX(Basic) FROM Emp);
42

```

STDIN  
Input for the program ( Optional )

Output:

EmpNo	EmpName	Job	Basic
1	John	Manager	15000.00
2	Alice	Clerk	10000.00
3	Bob	Technician	12000.00
4	Emily	Engineer	18000.00
5	David	Analyst	20000.00

EmpNo	EmpName	Job	GrossPay
2	Alice	Clerk	17000.00


EmpNo	EmpName	Job	Basic
5	David	Analyst	20000.00

( c ) If NetPay is less than <rs 10,000 add rs.1200 as special allowances.

-- If NetPay is less than <rs 10,000 add rs.1200 as special allowances

UPDATE Emp SET NetPay = NetPay + 1200

WHERE NetPay < 10000;



The screenshot shows a MySQL query editor with a file named 'queries.sql'. The code defines an 'Emp' table with columns: EmpNo (INT, PRIMARY KEY), EmpName (VARCHAR(100)), Job (VARCHAR(100)), Basic (DECIMAL(10, 2)), DA (DECIMAL(10, 2)), HRA (DECIMAL(10, 2)), PF (DECIMAL(10, 2)), GrossPay (DECIMAL(10, 2)), and NetPay (DECIMAL(10, 2)). It then inserts five records for employees John, Alice, Bob, Emily, and David. Subsequent queries calculate GrossPay and NetPay for each employee based on their Basic salary and various allowances. A specific query is highlighted: 'UPDATE Emp SET NetPay = NetPay + 1200 WHERE NetPay < 10000;'. The output section shows the resulting data for the employees, including their Basic, GrossPay, and NetPay values.

```
1  -- Create the Emp table
2  CREATE TABLE Emp (
3      EmpNo INT PRIMARY KEY,
4      EmpName VARCHAR(100),
5      Job VARCHAR(100),
6      Basic DECIMAL(10, 2),
7      DA DECIMAL(10, 2),
8      HRA DECIMAL(10, 2),
9      PF DECIMAL(10, 2),
10     GrossPay DECIMAL(10, 2),
11     NetPay DECIMAL(10, 2)
12 );
13
14 -- Insert Five Records and calculate GrossPay and NetPay
15 INSERT INTO Emp (EmpNo, EmpName, Job, Basic, DA, HRA, PF, GrossPay, NetPay) VALUES
16 (1, 'John', 'Manager', 15000.00, 4500.00, 6000.00, 1800.00, NULL, NULL),
17 (2, 'Alice', 'Clerk', 10000.00, 3000.00, 4000.00, 1200.00, NULL, NULL),
18 (3, 'Bob', 'Technician', 12000.00, 3600.00, 4800.00, 1440.00, NULL, NULL),
19 (4, 'Emily', 'Engineer', 18000.00, 5400.00, 7200.00, 2160.00, NULL, NULL),
20 (5, 'David', 'Analyst', 20000.00, 6000.00, 8000.00, 2400.00, NULL, NULL);
21
22 -- Calculate GrossPay and NetPay
23 UPDATE Emp SET GrossPay = Basic + DA + HRA, NetPay = Basic + DA + HRA - PF;
24
25 -- Display the employees whose Basic is lowest in each department
26 SELECT EmpNo, EmpName, Job, Basic FROM Emp e1
27 WHERE Basic = (SELECT MIN(Basic) FROM Emp e2 WHERE e1.Job = e2.Job);
28
29 -- If NetPay is less than <rs 10,000 add rs.1200 as special allowances
30 UPDATE Emp SET NetPay = NetPay + 1200
31 WHERE NetPay < 10000;
32
33 -- Display the employees whose grosspay lies between 10,000 & 20,000
34 SELECT EmpNo, EmpName, Job, GrossPay
35 FROM Emp
36 WHERE GrossPay BETWEEN 10000 AND 20000;
37
38 -- Display all the employees who earn maximum salary
39 SELECT EmpNo, EmpName, Job, Basic
40 FROM Emp
41 WHERE Basic = (SELECT MAX(Basic) FROM Emp);
42
```

Output:

EmpNo	EmpName	Job	Basic
1	John	Manager	15000.00
2	Alice	Clerk	10000.00
3	Bob	Technician	12000.00
4	Emily	Engineer	18000.00
5	David	Analyst	20000.00

EmpNo	EmpName	Job	GrossPay
2	Alice	Clerk	17000.00

EmpNo	EmpName	Job	Basic
5	David	Analyst	20000.00

(d) Display the employees whose grosspay lies between 10,000 & 20,000.

-- Display the employees whose grosspay lies between 10,000 & 20,000

SELECT EmpNo, EmpName, Job, GrossPay

FROM Emp

WHERE GrossPay BETWEEN 10000 AND 20000;

queries.sql

+

425xp587f

NEW

MYSQL

RUN

```

1  -- Create the Emp table
2  CREATE TABLE Emp (
3      EmpNo INT PRIMARY KEY,
4      EmpName VARCHAR(100),
5      Job VARCHAR(100),
6      Basic DECIMAL(10, 2),
7      DA DECIMAL(10, 2),
8      HRA DECIMAL(10, 2),
9      PF DECIMAL(10, 2),
10     GrossPay DECIMAL(10, 2),
11     NetPay DECIMAL(10, 2)
12 );
13
14 -- Insert Five Records and calculate GrossPay and NetPay
15 INSERT INTO Emp (EmpNo, EmpName, Job, Basic, DA, HRA, PF, GrossPay, NetPay) VALUES
16     (1, 'John', 'Manager', 15000.00, 4500.00, 6000.00, 1800.00, NULL, NULL),
17     (2, 'Alice', 'Clerk', 10000.00, 3000.00, 4000.00, 1200.00, NULL, NULL),
18     (3, 'Bob', 'Technician', 12000.00, 3600.00, 4800.00, 1440.00, NULL, NULL),
19     (4, 'Emily', 'Engineer', 18000.00, 5400.00, 7200.00, 2160.00, NULL, NULL),
20     (5, 'David', 'Analyst', 20000.00, 6000.00, 8000.00, 2400.00, NULL, NULL);
21
22 -- Calculate GrossPay and NetPay
23 UPDATE Emp SET GrossPay = Basic + DA + HRA, NetPay = Basic + DA + HRA - PF;
24
25 -- Display the employees whose Basic is lowest in each department
26 SELECT EmpNo, EmpName, Job, Basic FROM Emp e1
27 WHERE Basic = (SELECT MIN(Basic) FROM Emp e2 WHERE e1.Job = e2.Job);
28
29 -- If NetPay is less than <rs 10,000 add rs.1200 as special allowances
30 UPDATE Emp SET NetPay = NetPay + 1200
31 WHERE NetPay < 10000;
32
33 -- Display the employees whose grosspay lies between 10,000 & 20,000
34 SELECT EmpNo, EmpName, Job, GrossPay
35 FROM Emp
36 WHERE GrossPay BETWEEN 10000 AND 20000;
37
38 -- Display all the employees who earn maximum salary
39 SELECT EmpNo, EmpName, Job, Basic
40 FROM Emp
41 WHERE Basic = (SELECT MAX(Basic) FROM Emp);
42

```

STDIN

Input for the program (Optional)

Output:

EmpNo	EmpName	Job	Basic
1	John	Manager	15000.00
2	Alice	Clerk	10000.00
3	Bob	Technician	12000.00
4	Emily	Engineer	18000.00
5	David	Analyst	20000.00

EmpNo	EmpName	Job	GrossPay
2	Alice	Clerk	17000.00

EmpNo	EmpName	Job	Basic
5	David	Analyst	20000.00

(e) Display all the employees who earn maximum salary.

-- Display all the employees who earn maximum salary

SELECT EmpNo, EmpName, Job, Basic

FROM Emp

WHERE Basic = (SELECT MAX(Basic) FROM Emp);

queries.sql

+

425xp587f

NEW

MYSQL

RUN

```

1  -- Create the Emp table
2  CREATE TABLE Emp (
3      EmpNo INT PRIMARY KEY,
4      EmpName VARCHAR(100),
5      Job VARCHAR(100),
6      Basic DECIMAL(10, 2),
7      DA DECIMAL(10, 2),
8      HRA DECIMAL(10, 2),
9      PF DECIMAL(10, 2),
10     GrossPay DECIMAL(10, 2),
11     NetPay DECIMAL(10, 2)
12 );
13
14 -- Insert Five Records and calculate GrossPay and NetPay
15 INSERT INTO Emp (EmpNo, EmpName, Job, Basic, DA, HRA, PF, GrossPay, NetPay) VALUES
16     (1, 'John', 'Manager', 15000.00, 4500.00, 6000.00, 1800.00, NULL, NULL),
17     (2, 'Alice', 'Clerk', 10000.00, 3000.00, 4000.00, 1200.00, NULL, NULL),
18     (3, 'Bob', 'Technician', 12000.00, 3600.00, 4800.00, 1440.00, NULL, NULL),
19     (4, 'Emily', 'Engineer', 18000.00, 5400.00, 7200.00, 2160.00, NULL, NULL),
20     (5, 'David', 'Analyst', 20000.00, 6000.00, 8000.00, 2400.00, NULL, NULL);
21
22 -- Calculate GrossPay and NetPay
23 UPDATE Emp SET GrossPay = Basic + DA + HRA, NetPay = Basic + DA + HRA - PF;
24
25 -- Display the employees whose Basic is lowest in each department
26 SELECT EmpNo, EmpName, Job, Basic FROM Emp e1
27 WHERE Basic = (SELECT MIN(Basic) FROM Emp e2 WHERE e1.Job = e2.Job);
28
29 -- If NetPay is less than <rs 10,000 add rs.1200 as special allowances
30 UPDATE Emp SET NetPay = NetPay + 1200
31 WHERE NetPay < 10000;
32
33 -- Display the employees whose grosspay lies between 10,000 & 20,000
34 SELECT EmpNo, EmpName, Job, GrossPay
35 FROM Emp
36 WHERE GrossPay BETWEEN 10000 AND 20000;
37
38 -- Display all the employees who earn maximum salary
39 SELECT EmpNo, EmpName, Job, Basic
40 FROM Emp
41 WHERE Basic = (SELECT MAX(Basic) FROM Emp);
42

```

STDIN

Input for the program (Optional)

Output:

EmpNo	EmpName	Job	Basic
1	John	Manager	15000.00
2	Alice	Clerk	10000.00
3	Bob	Technician	12000.00
4	Emily	Engineer	18000.00
5	David	Analyst	20000.00

EmpNo	EmpName	Job	GrossPay
2	Alice	Clerk	17000.00

EmpNo	EmpName	Job	Basic
5	David	Analyst	20000.00

