Name : Jeevan Ravindra Patil
College : National Institute of Technology
M. Tech CSE
Roll No. : 206117010
Sunject : Adavanced Database Lab

Experiment No. 1

- **Consider the following relations**

Employee(employee-name, street, city)
Bank(bank-name, city)
Works(employee-name, bank-name, salary)
Manages(employee-name ,manager-name)

:::::All Tables:::::

```
mysql>show tables;
+--------------+
| Tables_in_db |
+--------------+
| Bank         |
| Employee     |
| Manages      |
| Works        |
+--------------+
```

```
mysql>select * from Employee;
+---------------+--------------+------------+
| employee_name | street       | city       |
+---------------+--------------+------------+
| Jeevan        | Jalgaon Road | Jamner     |
| Abhijith      | Boro Road    | Trivandrum |
| Rahul         | Khoiwal Road | Rajastan   |
| Vishal        | Vish Road    | Hyderabaad |
| Navin         | Nava Road    | Pune       |
| Ashish        | Jalgaon Road | Jamner     |
+---------------+--------------+------------+
```

```
mysql>select * from Bank;
+-------------+------------+
| bank_name   | city       |
+-------------+------------+
| SBI         | Jamner     |
| Indian Bank | Trivandrum |
| BOM         | Pune       |
| Kodak       | Trichy     |
+-------------+------------+
```

```
mysql>select * from Works;
+---------------+-------------+--------+
| employee_name | bank_name   | salary |
+---------------+-------------+--------+
| Jeevan        | SBI         |  25000 |
```

Abhijith	Kodak	2000
Rahul	Indian Bank	14000
Vishal	BOM	18000
Navin	BOM	35000
Ashish	SBI	5000
+---------------+-------------+--------+

---

mysql>select * from Manages;
+---------------+--------------+
| employee_name | manager_name |
+---------------+--------------+
| Vishal | Navin |
| Ashish | Jeevan |
+---------------+--------------+

- **Write the following queries in SQL :**

1. Find the names and cities of residence of all employees who work for State Bank of India.
Query : select Employee.employee_name, Employee.city from Employee, Works where Employee.employee_name = Works.employee_name && bank_name = "SBI";
+---------------+--------+
| employee_name | city |
+---------------+--------+
| Jeevan | Jamner |
| Ashish | Jamner |
+---------------+--------+

2. Find the names, street, address and cities of residence of all employees who work for State Bank of India and earn more than Rs.14, 000.
Query : select Employee.employee_name, Employee.street, Employee.city from Employee, Works where Employee.employee_name = Works.employee_name && bank_name = "SBI" && salary > 14000;
+---------------+--------------+--------+
| employee_name | street | city |
+---------------+--------------+--------+
| Jeevan | Jalgaon Road | Jamner |
+---------------+--------------+--------+

3. Find all the employees in the database who live in the same cities as the banks for which they work.
Query : select Employee.employee_name from Employee, Bank, Works where Employee.employee_name = Works.employee_name && Employee.city = Bank.city && Works.bank_name = Bank.bank_name;
+---------------+
| employee_name |
+---------------+
| Jeevan |
| Navin |

| Ashish |
+---------------+

4. Find all the employees in the database who live in the same cities and on the same streets as do their managers.
Query : select m.employee_name from Employee e, Employee f, Manages m where e.employee_name = f.employee_name and m.manager_name = f.employee_name and e.city = f.city and e.street = f. street;
+---------------+
| employee_name |
+---------------+
| Vishal |
| Ashish |
+---------------+

5. Find all the employees in the database who do not work in State Bank of India.
Query : select e.employee_name from Employee e, Works w where e.employee_name = w.employee_name and w.bank_name <> "SBI";
+---------------+
| employee_name |
+---------------+
| Abhijith |
| Rahul |
| Vishal |
| Navin |
+---------------+

6. Find all the employees in the database who earn more than every employee of Indian Bank.
Query : select e.employee_name from Employee e, Employee f, Works we, Works wf where wf.bank_name = "Indian Bank" and f.employee_name = wf.employee_name and e.employee_name = we.employee_name and we.salary > wf.salary;
+---------------+
| employee_name |
+---------------+
| Jeevan |
| Vishal |
| Navin |
+---------------+

7. Find all employees who earn more than the average salary of all employees of their bank.
Query : select e.employee_name from Employee e, Works w where e.employee_name = w.employee_name and w.salary > (select avg(salary) from Works b where b.bank_name = w.bank_name);
+---------------+
| employee_name |
+---------------+
| Jeevan |
| Navin |
+---------------+

8. Find the bank that has the most employees.
Query : select bank_name, count(employee_name) from Works group by bank_name order by count(employee_name) desc limit 1;

```
+-----------+---------------------+
| bank_name | count(employee_name) |
+-----------+---------------------+
| SBI       |                   2 |
+-----------+---------------------+
```

9. Find the bank that has the smallest payroll.
Query : select bank_name, salary from Works order by salary limit 1;

```
+-----------+--------+
| bank_name | salary |
+-----------+--------+
| Kodak     |   2000 |
+-----------+--------+
```

10. Find those banks whose employees earn a higher salary, on average, than the average salary at State Bank of India.
Query : select bank_name from Works where salary > (select avg(salary) from Works where bank_name = "SBI") group by bank_name;

```
+-----------+
| bank_name |
+-----------+
| BOM       |
| SBI       |
+-----------+
```

11. Find the number of employees working in each bank.
Query : select bank_name, count(employee_name) from Works group by bank_name;

```
+-------------+---------------------+
| bank_name   | count(employee_name) |
+-------------+---------------------+
| BOM         |                   2 |
| Indian Bank |                   1 |
| Kodak       |                   1 |
| SBI         |                   2 |
+-------------+---------------------+
```

Experiment No. 2

- **Question No. 1**

Customer(Cust id : integer, cust_name: string)
Item(item_id: integer, item_name: string, price: integer)
Sales(bill_no: integer, bill_date: date, cust_id: integer, integer)

- **For the above schema, perform the following**

1. Create the tables with the appropriate integrity constraints and insert around 5 records in each of the tables.

Queries :

mysql>create table Customer (cust_id int NOT NULL, cust_name varchar(50), primary key(cust_id));

mysql>create table Item (item_id int NOT NULL, item_name varchar(50), price int, primary key(item_id));

mysql>create table Sales (bill_no int NOT NULL, bill_date date, cust_id int NOT NULL, item_id int NOT NULL, qty_sold int NOT NULL, primary key(bill_no), foreign key(cust_id) references Customer(cust_id), foreign key(item_id) references Item(item_id));

mysql>insert into Customer (cust_id, cust_name) values (1, "Jeevan"), (2, "Abhijith"), (3, "Rahul"), (4, "Vishal"), (5, "Navin");

mysql>insert into Item (item_id, item_name, price) values (101, "Mangoes", 250), (102, "Apples", 200), (103, "Grapes", 150), (104, "Oranges", 175), (105, "Guava", 100);

mysql>insert into Sales (bill_no, bill_date, cust_id, item_id, qty_sold) values (1001, '2018-01-19', 1, 101, 1), (1002, '2018-01-18', 2, 102, 2), (1003, '2018-01-17', 3, 103, 3), (1004, '2018-01-16', 4, 104, 4), (1005, '2018-01-15', 5, 105, 5);

:::::All Tables:::::

```
mysql>show tables;
+---------------+
| Tables_in_db3 |
+---------------+
| Customer      |
| Item          |
| Sales         |
+---------------+
```

```
mysql>select * from Customer;
+---------+-----------+
| cust_id | cust_name |
+---------+-----------+
|       1 | Jeevan    |
```

```
|      2 | Abhijith  |
|      3 | Rahul     |
|      4 | Vishal    |
|      5 | Navin     |
+---------+-----------+
```

```
mysql>select * from Item;
+---------+-----------+-------+
| item_id | item_name | price |
+---------+-----------+-------+
|    101 | Mangoes   |   250 |
|    102 | Apples    |   200 |
|    103 | Grapes    |   150 |
|    104 | Oranges   |   175 |
|    105 | Guava     |   100 |
+---------+-----------+-------+
```

```
mysql>select * from Sales;
+---------+------------+---------+---------+----------+
| bill_no | bill_date  | cust_id | item_id | qty_sold |
+---------+------------+---------+---------+----------+
|   1001 | 2018-01-19 |       1 |     101 |        1 |
|   1002 | 2018-01-18 |       2 |     102 |        2 |
|   1003 | 2018-01-17 |       3 |     103 |        3 |
|   1004 | 2018-01-16 |       4 |     104 |        4 |
|   1005 | 2018-01-15 |       5 |     105 |        5 |
+---------+------------+---------+---------+----------+
```

2. List all the bills for the current date with the customer names and item_id.
Query : select c.cust_name, s.item_id from Customer c, Sales s where c.cust_id = s.cust_id and s.bill_date = curdate();

```
+-----------+---------+
| cust_name | item_id |
+-----------+---------+
| Jeevan    |     101 |
+-----------+---------+
```

3. List the details of the customer who have bought a product which has a price >200.
Query : select c.cust_id, c.cust_name from Customer c, Item i, Sales s where c.cust_id = s.cust_id and s.item_id = i.item_id and i.price > 200;

```
+---------+-----------+
| cust_id | cust_name |
+---------+-----------+
|       1 | Jeevan    |
+---------+-----------+
```

4. Give a count of how many products have been bought by each customer.
Query : select c.cust_id, c.cust_name, count(s.item_id) from Customer c, Sales s where c.cust_id = s.cust_id group by c.cust_id;

```
+---------+-----------+-----------------+
| cust_id | cust_name | count(s.item_id) |
+---------+-----------+-----------------+
|       1 | Jeevan    |               1 |
|       2 | Abhijith  |               1 |
|       3 | Rahul     |               1 |
|       4 | Vishal    |               1 |
|       5 | Navin     |               1 |
+---------+-----------+-----------------+
```

5. Give a list of products bought by a customer having cust_id as 5.
Query : select i.item_id, i.item_name from Item i, Sales s where s.cust_id = 5 and i.item_id = s.item_id;

```
+---------+-----------+
| item_id | item_name |
+---------+-----------+
|     105 | Guava     |
+---------+-----------+
```

6. List the item details which are sold as of today.
Query : select i.item_id, i.item_name from Item i, Sales s where i.item_id = s.item_id group by s.item_id;

```
+---------+-----------+
| item_id | item_name |
+---------+-----------+
|     101 | Mangoes   |
|     102 | Apples    |
|     103 | Grapes    |
|     104 | Oranges   |
|     105 | Guava     |
+---------+-----------+
```

- **Question No. 2**

Student(stud_no: integer, stud_name: string, class: string)
Class(class: string, descrip: string)
Lab(mach_no: integer, Lab_no: integer, description: String)
Allotment(stud_no: integer, mach_no: integer, day_of_week: string)

- **For the above schema, perform the following**

1. Create the tables with the appropriate integrity constraints and insert around 5 records in each of the tables.

Queries :

```
mysql>create table Class (class varchar(50) NOT NULL, descrip varchar(50), primary key(class));

mysql>create table Student (stud_no int NOT NULL, stud_name varchar(50), class varchar(50), primary key(stud_no), foreign key(class) references Class(class));

mysql>create table Lab (mach_no int NOT NULL, lab_no int, description varchar(50), primary key(mach_no));

mysql>create table Allotment (stud_no int NOT NULL, mach_no int NOT NULL, day_of_week varchar(50), foreign key(stud_no) references Student(stud_no), foreign key(mach_no) references Lab(mach_no));

mysql>insert into Class(class, descrip) values ("CSIT", "CSEngg"), ("MECH", "MECH Engg"), ("ECE", "ECE Engg"), ("CIVIL", "CIVIL Engg"), ("DA", "DA Engg");

mysql>insert into Student(stud_no, stud_name, class) values (1, "Jeevan", "CSIT"), (2, "Abhijith", "MECH"), (3, "Rahul", "ECE"), (4, "Vishal", "CIVIL"), (5, "Navin", "DA");

mysql>insert into Lab(mach_no, Lab_no, description) values (101, 501, "CSIT"), (102, 502, "MECH"), (103, 503, "ECE"), (104, 504, "CIVIL"), (105, 505, "DA");

mysql>insert into Allotment (stud_no, mach_no, day_of_week) values (1, 101, "MONDAY"), (2, 102, "TUESDAY"), (3, 103, "WEDNESDAY"), (4, 104, "THURSDAY"), (5, 105, "FRIDAY");

-------------------------------------------------------------------------------------------------------------------------
:::::All Tables:::::
-------------------------------------------------------------------------------------------------------------------------
mysql>show tables;
+---------------+
| Tables_in_db4 |
+---------------+
| Allotment     |
| Class         |
| Lab           |
| Student       |
+---------------+
-------------------------------------------------------------------------------------------------------------------------
mysql>select * from Class;
```

```
+-------+------------+
| class | descrip    |
+-------+------------+
| CSIT  | CSEngg     |
| MECH  | MECH Engg  |
| ECE   | ECE Engg   |
| CIVIL | CIVIL Engg |
| DA    | DA Engg    |
+-------+------------+
```

```
mysql>select * from Student;
+---------+-----------+-------+
| stud_no | stud_name | class |
+---------+-----------+-------+
|       1 | Jeevan    | CSIT  |
|       2 | Abhijith  | MECH  |
|       3 | Rahul     | ECE   |
|       4 | Vishal    | CIVIL |
|       5 | Navin     | DA    |
+---------+-----------+-------+
```

```
mysql>select * from Lab;
+---------+--------+-------------+
| mach_no | lab_no | description |
+---------+--------+-------------+
|     101 |    501 | CSIT        |
|     102 |    502 | MECH        |
|     103 |    503 | ECE         |
|     104 |    504 | CIVIL       |
|     105 |    505 | DA          |
+---------+--------+-------------+
```

```
mysql>select * from Allotment;
+---------+---------+-------------+
| stud_no | mach_no | day_of_week |
+---------+---------+-------------+
|       1 |     101 | MONDAY      |
|       2 |     102 | TUESDAY     |
|       3 |     103 | WEDNESDAY   |
|       4 |     104 | THURSDAY    |
|       5 |     105 | FRIDAY      |
+---------+---------+-------------+
```

2. List all the machine allotments with the student names, lab and machine numbers.

Query : select s.stud_no, s.stud_name, l.* from Student s, Lab l, Allotment a where s.stud_no = a.stud_no and a.mach_no = l.mach_no;

```
+---------+-----------+---------+--------+-------------+
| stud_no | stud_name | mach_no | lab_no | description |
+---------+-----------+---------+--------+-------------+
|       1 | Jeevan    |     101 |    501 | CSIT        |
|       2 | Abhijith  |     102 |    502 | MECH        |
|       3 | Rahul     |     103 |    503 | ECE         |
|       4 | Vishal    |     104 |    504 | CIVIL       |
|       5 | Navin     |     105 |    505 | DA          |
+---------+-----------+---------+--------+-------------+
```

3. Display list of student who has not given any machine.

Query : select s.stud_no, s.stud_name from Student s where stud_no not in (select a.stud_no from Allotment a);

```
+---------+-----------+
| stud_no | stud_name |
+---------+-----------+
|       6 | Boro      |
+---------+-----------+
```

4. Give a count of how many machines have been allocated to the “CSIT” class.

Query : select count(a.mach_no) from Allotment a, Student s where s.stud_no = a.stud_no and s.class = "CSIT";

```
+------------------+
| count(a.mach_no) |
+------------------+
|                1 |
+------------------+
```

5. Count for how many machines have been allocated in Lab_no 1 for the day of the week as “Monday”.

Query : select count(a.mach_no) from Lab l, Allotment a where a.mach_no = l.mach_no and l.lab_no = 501 and a.day_of_week = "MONDAY";

```
+------------------+
| count(a.mach_no) |
+------------------+
|                1 |
+------------------+
```

- **Question No. 3**

employee(emp_id : integer, emp_name: string)
department(dept_id: integer, dept_name:string)
paydetails(emp_id : integer, dept_id: integer, basic: integer, deductions: integer, additions: integer, DOJ: date)
payroll(emp_id : integer, pay_date: date)

- **For the above schema, perform the following**

1. Create the tables with the appropriate integrity constraints and insert around 10 records in each of the tables.
Queries :

```
mysql>create table employee ( emp_id int NOT NULL, emp_name varchar(50), primary key(emp_id));

mysql>create table department ( dept_id int NOT NULL, dept_name varchar(50), primary key(dept_id));

mysql>create table paydetails ( emp_id int NOT NULL, dept_id int NOT NULL, basic int, deductions int, additions int, DOJ date, foreign key(emp_id) references employee(emp_id), foreign key(dept_id) references department(dept_id));

mysql>create table payroll ( emp_id int NOT NULL, pay_date date, foreign key(emp_id) references employee(emp_id));

mysql>insert into employee (emp_id, emp_name) values (1, "A"), (2, "B"), (3, "C"), (4, "D"), (5, "E"), (6, "F"), (7, "G"), (8, "H"), (9, "I"), (10, "J");

mysql>insert into department (dept_id, dept_name) values (101, "DA"), (102, "DB"), (103, "DC"), (104, "DD"), (105, "DE"), (106, "DF"), (107, "DG"), (108, "DH"), (109, "DI"), (110, "DJ");

mysql>insert into paydetails (emp_id, dept_id, basic, deductions, additions, DOJ) values (1, 101, 10000, 1000, 1000, '2017-10-01'), (2, 102, 15000, 1500, 1500, '2017-10-01'), (3, 103, 20000, 2000, 2000, '2017-10-01'), (4, 104, 25000, 2500, 2500, '2017-10-01'), (5, 105, 30000, 3000, 3000, '2017-10-01'), (6, 106, 35000, 3500, 3500, '2017-10-01'), (7, 107, 40000, 4000, 4000, '2017-10-01'), (8, 108, 4500, 450, 450, '2017-10-01'), (9, 109, 5000, 500, 500, '2017-10-01'), (10, 110, 5500, 550, 550, '2017-10-01');

mysql>insert into payroll (emp_id, pay_date) values (1, '2018-01-01'), (2, '2018-01-01'), (3, '2018-01-01'), (4, '2018-01-01'), (5, '2018-01-01'), (6, '2018-01-01'), (7, '2018-01-01'), (8, '2018-01-01'), (9, '2018-01-01'), (10, '2018-01-01');
```

:::::All Tables:::::

```
mysql>show tables;
+---------------+
| Tables_in_db5 |
+---------------+
| department    |
| employee      |
| paydetails    |
| payroll       |
+---------------+
```

```
mysql>select * from employee;
+--------+----------+
| emp_id | emp_name |
+--------+----------+
|      1 | A        |
|      2 | B        |
|      3 | C        |
|      4 | D        |
|      5 | E        |
|      6 | F        |
|      7 | G        |
|      8 | H        |
|      9 | I        |
|     10 | J        |
+--------+----------+
```

```
mysql>select * from department;
+---------+-----------+
| dept_id | dept_name |
+---------+-----------+
|     101 | DA        |
|     102 | DB        |
|     103 | DC        |
|     104 | DD        |
|     105 | DE        |
|     106 | DF        |
|     107 | DG        |
|     108 | DH        |
|     109 | DI        |
|     110 | DJ        |
+---------+-----------+
```

---

mysql>select * from paydetails;

emp_id	dept_id	basic	deductions	additions	DOJ
1	101	10000	1000	9000	2017-10-01
2	102	15000	1500	13500	2017-10-01
3	103	20000	2000	18000	2017-10-01
4	104	25000	2500	22500	2017-10-01
5	105	30000	3000	27000	2017-10-01
6	106	35000	3500	31500	2017-10-01
7	107	40000	4000	36000	2017-10-01
8	108	4500	450	4050	2017-10-01
9	109	5000	500	4500	2017-10-01
10	110	5500	550	4950	2017-10-01

---

mysql>select * from payroll;

emp_id	pay_date
1	2018-01-01
2	2018-01-01
3	2018-01-01
4	2018-01-01
5	2018-01-01
6	2018-01-01
7	2018-01-01
8	2018-01-01
9	2018-01-01
10	2018-01-01

2. List the employee details department wise.

Query : select e.* from employee e, paydetails pd where e.emp_id = pd.emp_id group by pd.dept_id;

emp_id	emp_name
1	A
2	B
3	C
4	D
5	E
6	F
7	G
8	H
9	I
10	J

3. List the details of employees whose basic salary is between 10,000 and 20,000.

Query : select e.* from employee e, paydetails pd where e.emp_id = pd.emp_id and pd.basic >= 10000 and pd.basic <= 20000;

emp_id	emp_name
1	A
2	B
3	C

4. Give a count of how many employees are working in each department.

Query : select d.dept_id, count(pd.emp_id) from paydetails pd, department d where d.dept_id = pd.dept_id group by pd.dept_id;

dept_id	count(pd.emp_id)
101	1
102	1
103	1
104	1
105	1
106	1
107	1
108	1
109	1
110	1

5. Give a names of the employees whose netsalary>10,000.

Query : select e.* from employee e, paydetails pd where e.emp_id = pd.emp_id and (pd.basic - deductions + pd.additions) > 10000;

emp_id	emp_name
2	B
3	C
4	D
5	E
6	F
7	G

Experiment No. 3

- **The following tables form part of a database held in a relational DBMS:**

Hotel (HotelNo, Name, City)
Room (RoomNo, HotelNo, Type, Price)
Booking (HotelNo, GuestNo, DateFrom, DateTo, RoomNo)
Guest (GuestNo, GuestName, GuestAddress)

where,
Hotel contains hotel details and HotelNo is the primary key
Room contains room details for each hotel and (HotelNo, RoomNo) forms the primary key
Booking contains details of the bookings and the primary key comprises (HotelNo, GuestNo and DateFrom)
Guest contains guest details and GuestNo is the primary key.

- **The sample data for the relation is as follows, populate your tables using these data.**

HOTEL
fb01, Grosvenor, London
fb02, Watergate, Paris
ch01, Omni Shoreham, London
ch02, Phoenix Park, London
dc01, Latham, Berlin

ROOM
501, fb01, single, 19
601, fb01, double, 29
701, fb01, family, 39
1001, fb02, single, 58
1101, fb02, double, 86
1001, ch01, single, 29.99
1101, ch01, family, 59.99
701, ch02, single, 10
801, ch02, double, 15
901, dc01, single, 18
1001, dc01, double, 30
1101, dc01, family, 35

GUEST
10001, John Kay, 56 High St, London;
10002, Mike Ritchie, 18 Tain St, London
10003, Mary Tregear, 5 Tarbot Rd, Aberdeen
10004,Joe Keogh, 2 Fergus Dr, Aberdeen
10005, Carol Farrel, 6 Achray St, Glasgow
10006, Tina Murphy, 63 Well St, Glasgow
10007, Tony Shaw, 12 Park Pl, Glasgow

BOOKING
fb01, 10001, 04-04-01, 04-04-08, 501
fb01, 10004, 04-04-15, 04-05-15, 601
fb01, 10005, 04-05-02, 04-05-07, 501
fb01, 10002, 16-05-04, 04-05-29, 601
fb01, 10001, 04-05-01, null, 701
fb02, 10005, 04-05-12, 30-05-04, 1101
ch01, 10006, 04-04-21, null, 1101
ch02, 10002, 04-04-25, 04-05-06, 801
dc01, 10007, 04-05-13, 04-05-15, 1001
dc01, 10003, 04-05-20, null, 1001

1. Using the CREATE TABLE statement, create the Hotel, Room, Booking and Guest tables.
Queries :

```
mysql>create table Hotel(HotelNo varchar(10) not null primary key, Name varchar(25), City varchar(25));

mysql>create table Room(RoomNo int not null, HotelNo varchar(10), Type varchar(10), Price float, foreign key(HotelNo) references Hotel(HotelNo), primary key(RoomNo, HotelNo));

mysql>create table Booking(HotelNo varchar(10), GuestNo int, DateFrom date, DateTo date,RoomNo int, primary key(HotelNo, GuestNo, DateFrom), foreign key(HotelNo) references Hotel(HotelNo), foreign key(RoomNo) references Room(RoomNo));

mysql>create table Guest(GuestNo int primary key, GuestName varchar(25), GuestAddress varchar(50));
```

2. Insert records into each of these tables.
Queries :

```
mysql>insert into Hotel values('fb01', 'Grosvenor', 'London');
mysql>insert into Hotel values('fb02', 'Watergate', 'Paris');
mysql>insert into Hotel values('ch01', 'Omni Shoreham', 'London');
mysql>insert into Hotel values('ch02', 'Phoenix Park', 'London');
mysql>insert into Hotel values('dc01', 'Latham', 'Berlin');
mysql>select * from Hotel;
+---------+---------------+--------+
| HotelNo | Name          | City   |
+---------+---------------+--------+
| fb01    | Grosvenor     | London |
| fb02    | Watergate     | Paris  |
| ch01    | Omni Shoreham | London |
| ch02    | Phoenix Park  | London |
| dc01    | Latham        | Berlin |
+---------+---------------+--------+
mysql>insert into Room values(501, 'fb01', 'single', 19);
mysql>insert into Room values(601, 'fb01', 'double', 29);
mysql>insert into Room values(701, 'fb01', 'family', 39);
mysql>insert into Room values(1001, 'fb02', 'single', 58);
mysql>insert into Room values(1101, 'fb02', 'double', 85);
```

```
mysql>insert into Room values(1001, 'ch01', 'single', 29.99);
mysql>insert into Room values(1101, 'ch01', 'family', 59.99);
mysql>insert into Room values(701, 'ch02', 'single', 10);
mysql>insert into Room values(801, 'ch02', 'double', 15);
mysql>insert into Room values(901, 'dc01', 'single', 18);
mysql>insert into Room values(1001, 'dc01', 'double', 30);
mysql>insert into Room values(1101, 'dc01', 'family', 35);
mysql>select * from Room;
+--------+---------+--------+-------+
| RoomNo | HotelNo | Type   | Price |
+--------+---------+--------+-------+
|    501 | fb01    | single |    19 |
|    601 | fb01    | double |    29 |
|    701 | fb01    | family |    39 |
|   1001 | fb02    | single |    58 |
|   1101 | fb02    | double |    85 |
|   1001 | ch01    | single | 29.99 |
|   1101 | ch01    | family | 59.99 |
|    701 | ch02    | single |    10 |
|    801 | ch02    | double |    15 |
|    901 | dc01    | single |    18 |
|   1001 | dc01    | double |    30 |
|   1101 | dc01    | family |    35 |
+--------+---------+--------+-------+
mysql>insert into Guest values (10001, 'John Kay', '56 High St, London');
mysql>insert into Guest values (10002, 'Mike Ritchie', '18 Tain St, London');
mysql>insert into Guest values (10003, 'Mary Tregear', '5 Tarbot Rd, Aberdeen');
mysql>insert into Guest values (10004, 'Joe Keogh', '2 Fergus Dr, Aberdeen');
mysql>insert into Guest values (10005, 'Carol Farrel', '6 Achray St, Glasgoq');
mysql>insert into Guest values (10005, 'Carol Farrel', '6 Achray St, Glasgow');
mysql>insert into Guest values (10006, 'Tina Murphy', '63 Well St, Glasgow');
mysql>insert into Guest values (10007, 'Tony Shaw', '12 Park Pl, Glasgow');
mysql>select * from Guest;
+---------+-------------+----------------------+
| GuestNo | GuestName   | GuestAddress         |
+---------+-------------+----------------------+
|   10001 | John Kay    | 56 High St, London   |
|   10002 | Mike Ritchie | 18 Tain St, London  |
|   10003 | Mary Tregear | 5 Tarbot Rd, Aberdeen |
|   10004 | Joe Keogh   | 2 Fergus Dr, Aberdeen |
|   10005 | Carol Farrel | 6 Achray St, Glasgow |
|   10006 | Tina Murphy | 63 Well St, Glasgow  |
|   10007 | Tony Shaw   | 12 Park Pl, Glasgow  |
+---------+-------------+----------------------+
mysql>insert into Booking values('fb01', 10001, '04-04-01', '04-04-08', 501);
mysql>insert into Booking values('fb01', 10005, '04-05-02', '04-05-07', 501);
mysql>insert into Booking values('fb01', 10004, '04-04-15', '04-05-15', 601);
mysql>insert into Booking values('fb01', 10002, '16-05-04', '04-05-29', 601);
mysql>insert into Booking values('fb01', 10001, '04-05-01', null , 701);
```

```
mysql>insert into Booking values('fb02', 10005, '04-05-12', '30-05-04' , 1101);
mysql>insert into Booking values('ch01', 10006, '04-04-21', null , 1101);
mysql>insert into Booking values('ch02', 10002, '04-04-25', '04-05-06' , 801);
mysql>insert into Booking values('dc01', 10007, '04-05-13', '04-05-15' , 1001);
mysql>insert into Booking values('dc01', 10003, '04-05-20',null , 1001);
mysql>SELECT * FROM Booking;
+---------+---------+------------+------------+--------+
| HotelNo | GuestNo | DateFrom   | DateTo     | RoomNo |
+---------+---------+------------+------------+--------+
| fb01    |   10001 | 2004-04-01 | 2004-04-08 |    501 |
| fb01    |   10004 | 2004-04-15 | 2004-05-15 |    601 |
| fb01    |   10005 | 2004-05-02 | 2004-05-07 |    501 |
| fb01    |   10002 | 2016-05-04 | 2004-05-29 |    601 |
| fb01    |   10001 | 2004-05-01 | NULL       |    701 |
| fb02    |   10005 | 2004-05-12 | 2030-05-04 |   1101 |
| ch01    |   10006 | 2004-04-21 | NULL       |   1101 |
| ch02    |   10002 | 2004-04-25 | 2004-05-06 |    801 |
| dc01    |   10007 | 2004-05-13 | 2004-05-15 |   1001 |
| dc01    |   10003 | 2004-05-20 | NULL       |   1001 |
+---------+---------+------------+------------+--------+
```

3. Update the price of all rooms by 5%.

Query :

```
update Room set Price = Price*1.05;
```

```
+--------+---------+--------+---------+
| RoomNo | HotelNo | Type   | Price   |
+--------+---------+--------+---------+
|    501 | fb01    | single |   19.95 |
|    601 | fb01    | double |   30.45 |
|    701 | fb01    | family |   40.95 |
|   1001 | fb02    | single |    60.9 |
|   1101 | fb02    | double |   89.25 |
|   1001 | ch01    | single | 31.4895 |
|   1101 | ch01    | family | 62.9895 |
|    701 | ch02    | single |    10.5 |
|    801 | ch02    | double |   15.75 |
|    901 | dc01    | single |    18.9 |
|   1001 | dc01    | double |    31.5 |
|   1101 | dc01    | family |   36.75 |
+--------+---------+--------+---------+
```

4. List all double or family rooms with a price below £40.00 per night, in ascending order of price.

Query :

```
select * from Room where Type = 'double' or Type = 'family' and Price < 40;
```

```
+--------+---------+--------+-------+
| RoomNo | HotelNo | Type   | Price |
+--------+---------+--------+-------+
|    601 | fb01    | double | 30.45 |
|   1101 | fb02    | double | 89.25 |
|    801 | ch02    | double | 15.75 |
|   1001 | dc01    | double |  31.5 |
|   1101 | dc01    | family | 36.75 |
+--------+---------+--------+-------+
```

5. List the bookings for which no date_to has been specified.

Query :

```
select * from Booking where DateTo is null;
+---------+---------+------------+--------+--------+
| HotelNo | GuestNo | DateFrom   | DateTo | RoomNo |
+---------+---------+------------+--------+--------+
| fb01    |   10001 | 2004-05-01 | NULL   |    701 |
| ch01    |   10006 | 2004-04-21 | NULL   |   1101 |
| dc01    |   10003 | 2004-05-20 | NULL   |   1001 |
+---------+---------+------------+--------+--------+
```

6. What is the total income from bookings for the Grosvenor Hotel today?

Query :

```
select sum(r.Price) from Room r, Booking b, Hotel h where DateTo <= 'CURRENT_DATE' and
r.HotelNo = h.HotelNo and r.RoomNo = b.RoomNo and b.HotelNo = r.HotelNo and  h.Name =
'Grosvenor';
+------------------+
| sum(r.Price)     |
+------------------+
| 100.800003051758 |
+------------------+
```

7. What is the lost income from unoccupied rooms at the Grosvenor Hotel?

Query :

```
select sum(price) from Room r, Hotel h where r.HotelNo = h.HotelNo and Name = 'Grosvenor' and
RoomNo in (select b.RoomNo from Booking b, Hotel h where b.Hotelno = h.HotelNo and Name =
'Grosvenor' and DateTo < 'CURRENT_DATE');
+------------------+
| sum(price)       |
+------------------+
| 50.4000015258789 |
+------------------+
```

8. List the number of rooms in each hotel in London.

Query :

```
select Name, count(r.RoomNo) from Hotel h, Room r where h.HotelNo = r.HotelNo and City =
'London' group by Name;
```

```
+---------------+-----------------+
| Name          | count(r.RoomNo) |
+---------------+-----------------+
| Grosvenor     |               3 |
| Omni Shoreham |               2 |
| Phoenix Park  |               2 |
+---------------+-----------------+
```

9. What is the most commonly booked room type for each hotel in London?

Query : select HotelNo, Type, max(y) from (select r.HotelNo, Type, count(type) as y from Hotel h, Room r, Booking b where h.HotelNo = r.HotelNo and b.RoomNo = r.RoomNo and b.HotelNo = h.HotelNo and City = 'London' group by r.HotelNo,Type) as t group by HotelNo,type;

```
+---------+--------+--------+
| HotelNo | Type   | max(y) |
+---------+--------+--------+
| ch01    | family |      1 |
| ch02    | double |      1 |
| fb01    | double |      2 |
| fb01    | family |      1 |
| fb01    | single |      2 |
+---------+--------+--------+
```

Experiment No. 4

- **Working with Procedures & Functions in MySQL**

Consider the
Employee (EmpNo, EmpName, Sex, Salary, Address, DeptNo)
Department (DeptNo, DeptName, Location)

mysql>create table Departmant (DeptNo int NOT NULL, DeptName varchar(20) NOT NULL, Location varchar(20), primary key(DeptNo));

mysql>create table Employee (EmpNo int NOT NULL, EmpName varchar(20) NOT NULL, Sex varchar(10), Salary int, Address varchar(20), DeptNo int NOT NULL, primary key(EmpNo), foreign key(DeptNo) references Departmant(DeptNo));

mysql>insert into Departmant values (101, "CSE", "Pune"), (102, "ECE", "Mumbai"), (103, "MECH", "Mumbai"), (104, "EEE", "Trichy"), (105, "CIVIL", "Trichy");

mysql>insert into Employee values (11, "Jeevan", "Male", 51000, "Pune", 101), (12, "Boro", "Male", 65000, "Mumbai", 102), (13, "Rahul", "Male", 55000, "Trichy", 105), (14, "Vishal", "Male", 45000, "Trichy", 104), (15, "Bhagya", "Female", 49000, "Pune", 101), (16, "Ruchika", "Female", 66000, "Trichy", 105), (17, "Anjali", "Female", 33000, "Mumbai", 103), (18, "Riya", "Female", 35000, "Mumbai", 103), (19, "Amalan", "Male", 47500, "Mumbai", 102), (20, "Sonu", "Female", 56000, "Trichy", 104);

mysql>select * from Employee;

EmpNo	EmpName	Sex	Salary	Address	DeptNo
11	Jeevan	Male	51000	Pune	101
12	Boro	Male	65000	Mumbai	102
13	Rahul	Male	55000	Trichy	105
14	Vishal	Male	45000	Trichy	104
15	Bhagya	Female	49000	Pune	101
16	Ruchika	Female	66000	Trichy	105
17	Anjali	Female	33000	Mumbai	103
18	Riya	Female	35000	Mumbai	103
19	Amalan	Male	47500	Mumbai	102
20	Sonu	Female	56000	Trichy	104

mysql>select * from Departmant;

DeptNo	DeptName	Location
101	CSE	Pune
102	ECE	Mumbai
103	MECH	Mumbai

```
|    104 | EEE      | Trichy   |
|    105 | CIVIL    | Trichy   |
+--------+----------+----------+
```

1. Create a procedure to display the details of an employee record form employee table for a given employee number.
Query : delimiter //

```
create procedure display (in no int)
begin
        select * from Employee where EmpNo = no;
end//

delimiter ;

call display(11);
+-------+---------+------+--------+---------+--------+
| EmpNo | EmpName | Sex  | Salary | Address | DeptNo |
+-------+---------+------+--------+---------+--------+
|    11 | Jeevan  | Male |  51000 | Pune    |    101 |
+-------+---------+------+--------+---------+--------+
```

2. Create a procedure to add details of a new employee into employee table.
Query : delimiter //

```
create procedure add_emp(in no int, in name varchar(10), in sex varchar(10), in salary int, in address varchar(20), in deptno int)
begin
        insert into Employee values (no, name, sex, salary, address, deptno);
end//

delimiter ;

call add_emp(21, "Navin", "Male", 25000, "Pune", 101);
```

3. Write a procedure raise_sal which increases the salary of an employee. It accepts an employee number and salary increase amount. It uses the employee number to find the current salary from the EMPLOYEE table and update the salary.
Query : delimiter //

```
create procedure raise_sal(in no int, in s int)
begin
        update Employee set Salary = Salary + s where EmpNo = no;
end//

delimiter ;

call raise_sal(21, 5000);
```

4. Create a procedure to delete a record form employee table for a given employee name.
Query : delimiter //

```
create procedure del (in name varchar(10))
begin
        delete from Employee where EmpName = name;
end//

delimiter ;

call del("Navin");
```

5. Write a function to display maximum salary of employees from the employee table.
Query : delimiter //

```
create function max()
returns int DETERMINISTIC
begin
        DECLARE sal int;
        select Salary into sal from Employee order by Salary desc limit 1;
        return sal;
end//

delimiter ;

select max ();
+--------+
| max () |
+--------+
|  66000 |
+--------+
```

6. Write a function to display the number of employees working in the Organization.
Query : delimiter //

```
create function cnt()
returns int DETERMINISTIC
begin
        DECLARE cnt int;
        select count(*) into cnt from Employee;
        return cnt;
end//
delimiter ;

select cnt();
+-------+
| cnt() |
+-------+
|    10 |
+-------+
```

7. Write a function to display salary of an employee with the given employee number.
Query : delimiter //

```
create function disp(no int)
returns int DETERMINISTIC
begin
        DECLARE sal int;
        select Salary into sal from Employee where EmpNo = no;
        return sal;
end//

delimiter ;

select disp(11);
+----------+
| disp(11) |
+----------+
|    51000 |
+----------+
```

8. Write a function average which takes DeptNo as input argument and returns the average salary received by the employee in the given department.
Query : delimiter //

```
create function avgsal(dno int)
returns int DETERMINISTIC
begin
        DECLARE sal int;
        select avg(Salary) into sal from Employee where DeptNo = dno;
        return sal;
end//

delimiter ;

select avgsal(101);
+-------------+
| avgsal(101) |
+-------------+
|       50000 |
+-------------+
```

9. Write a procedure which takes the DeptNo as input parameter and lists the names of all employees belonging to that department.
Query : delimiter //

```
create procedure dispdept (in dno int)
begin
        select EmpName from Employee where DeptNo = dno;
end//
```

```
delimiter ;

call dispdept (102);
+---------+
| EmpName |
+---------+
| Boro    |
| Amalan  |
+---------+
```

10. Write procedure that lists the highest salary drawn by an employee in each of the departments. It should make use of a named procedure dept_highest which finds the highest salary drawn by an employee for the given department.

Query : delimiter //

```
create procedure maxdept (in dno int, out sal int)
begin
        select max(Salary) into sal from Employee where DeptNo = dno;
end//

create procedure maxemp ()
begin
        select d.*, e.Salary from Employee e, Departmant d where d.DeptNo = e.DeptNo and e.Salary =
maxdept (d.DeptNo);
end//

delimiter ;
```

11. Write a function that will display the number of employees with salary more than 50k.

Query : delimiter //

```
create function noemp()
returns int DETERMINISTIC
begin
        DECLARE no int;
        select count(*) into no from Employee where Salary > 50000;
        return no;
end//

delimiter ;

select noemp();
+---------+
| noemp() |
+---------+
|       5 |
+---------+
```

12. Write a function that will display the count of the number of employees working in Chennai.
Query : delimiter //

```
create function empcnt()
returns int DETERMINISTIC
begin
        DECLARE no int;
        select count(*) into no from Employee e, Departmant d where e.DeptNo = d.DeptNo and
d.Location = "Pune";
        return no;
end//

delimiter ;

select empcnt();
+----------+
| empcnt() |
+----------+
|        2 |
+----------+
```

Experiment No. 5

- **Working with Triggers**

Employee Schema
Employee (Empid, Lastname, Firstname, Email, Department, designation, DOJ, phone_no)

Table Creation :

CREATE TABLE Employee (Empid INT NOT NULL PRIMARY KEY, Lastname VARCHAR(30), Firstname VARCHAR(30), Email VARCHAR(30), Department VARCHAR(10), designation VARCHAR(10), DOJ VARCHAR(10), phoneno VARCHAR(15));

1. Create a trigger which will calculate the number of rows we have inserted till now.
Query :

```
delimiter //

CREATE TRIGGER no_of_employee_record
AFTER INSERT ON Employee
FOR EACH ROW
BEGIN
        SET @no_of_rows = (SELECT COUNT(*) FROM Employee);
END//

delimiter ;
```

2. Create a trigger that displays a message prior to an insert operation on the Employee table.
3. Create a Trigger that adds “+91” to all Phone numbers in the Employee table.
Test and see if the Trigger works properly by inserting and updating some data in the table.
4. write a trigger to ensure that employee table does not contain duplicate or null values in the Firstname column.
5. Create a trigger that whenever an insert, update, or delete operation occurs on the Employee table, a row is added to the Employeelog table recording the date, user, and action.
6. Create a trigger to insert Employee details into Employee table only if DOJ > 2018.
7. Create a trigger to prevent any Employee named John to be inserted into the table.
8. Create a trigger to raise an exception if the empid is changed.
9. Create a trigger when someone tried to insert a value into a Employees table values are inserted into views.

```
CREATE TABLE employeelog ( date DATE, User VARCHAR(20),Action VARCHAR(15));

delimiter //

CREATE TRIGGER Trigger_Insert
BEFORE INSERT ON Employee
FOR EACH ROW
BEGIN
```

```
	# Question 3
	IF (SUBSTR(NEW.phoneno,1,1) <> '+') THEN
		SET NEW.phoneno = CONCAT('+91 ',NEW.phoneno);
	END IF;

	# Question 6
	IF(NEW.DOJ >= 2018) THEN
		SET @MESSAGE_TEXT = 'Error: DOJ >= 2018';
		SET NEW.Empid = NULL;
	ELSE
		# Question 7
		IF(NEW.Firstname = 'John') THEN
			SET @MESSAGE_TEXT = 'Error: Firstname is John';
			SET NEW.Empid = NULL;
		ELSE
			# Question 4
			IF(NEW.FirstName = '') THEN
				SET @MESSAGE_TEXT = 'Error: Firstname is NULL';
				SET NEW.Empid = NULL;
			ELSE
				# Question 2
				SET @MESSAGE_TEXT = 'Alert: Inserting';
				# Question 5
				INSERT INTO employeelog VALUES(CURDATE(),
CURRENT_USER(), 'Inserting');
			END IF;
		END IF;
	END IF;
END//

CREATE TRIGGER Trigger_Delete
BEFORE DELETE ON Employee
FOR EACH ROW
BEGIN
	INSERT INTO employeelog VALUES(CURDATE(), CURRENT_USER(), 'Deleting');
END//

CREATE TRIGGER Trigger_Update
BEFORE UPDATE ON Employee
FOR EACH ROW
BEGIN
	INSERT INTO employeelog VALUES(CURDATE(), CURRENT_USER(), 'Updating');
	IF(NEW.Empid = OLD.Empid) THEN
		SET @MESSAGE_TEXT = 'Error: Empid is wrong';
		SET NEW.Empid = NULL;
	END IF;
END//

delimiter ;
```

Queries :

```
mysql>INSERT INTO Employee VALUES(1001, 'Patil', 'Jeevan', 'patiljeevanr@gmail.com', 'CSE',
'Developer', '1994', '9876543210');


mysql>select * from Employee;
+-------+----------+-----------+-----------------------+------------+-------------+------------+----------------+
| Empid | Lastname | Firstname | Email                 | Department | designation | DOJ        | phoneno        |
+-------+----------+-----------+-----------------------+------------+-------------+------------+----------------+
|  1001 | Patil    | Jeevan    | patiljeevanr@gmail.com | CSE        | Developer   | 1994 | +91 9876543210 |
+-------+----------+-----------+-----------------------+------------+-------------+------------+----------------+

mysql> select @MESSAGE_TEXT;
+-----------------+
| @MESSAGE_TEXT |
+-----------------+
| Alert: Inserting |
+-----------------+

mysql>INSERT INTO Employee VALUES(1001, 'Patil', 'John', 'patiljeevanr@gmail.com', 'CSE',
'Developer', '1994', '9876543210');
ERROR 1048 (23000): Column 'Empid' cannot be null

mysql>select @MESSAGE_TEXT;
+-------------------------+
| @MESSAGE_TEXT    |
+-------------------------+
| Error: Firstname is John |
+-------------------------+

mysql>INSERT INTO Employee VALUES(1002, 'Patil', 'John', 'patiljeevanr@gmail.com', 'CSE',
'Developer', '2018', '9876543210');
ERROR 1048 (23000): Column 'Empid' cannot be null

mysql>select @MESSAGE_TEXT;
+-------------------+
| @MESSAGE_TEXT|
+-------------------+
| Error: DOJ >= 2018 |
+-------------------+

mysql> INSERT INTO Employee VALUES(1002, 'Patil', 'Akshay', 'akshay@gmail.com', 'CSE',
'Developer', '1995', '9876543210');
```

```
mysql> select * from employeelog;
+------------+----------------+-----------+
| date       | User           | Action    |
+------------+----------------+-----------+
| 2018-04-07 | root@localhost | Inserting |
| 2018-04-07 | root@localhost | Inserting |
+------------+----------------+-----------+

mysql> select @no_of_rows;
+-------------+
| @no_of_rows |
+-------------+
|           2 |
+-------------+

mysql> select * from Employee;
+-------+----------+-----------+-----------------------+------------+-------------+------+----------------+
| Empid | Lastname | Firstname | Email                 | Department | designation | DOJ  | phoneno        |
+-------+----------+-----------+-----------------------+------------+-------------+------+----------------+
|  1001 | Patil    | Jeevan    | patiljeevanr@gmail.com | CSE        | Developer   | 1994 | +91 9876543210 |
|  1002 | Patil    | Akshay    | akshay@gmail.com       | CSE        | Developer   | 1995 | +91 9876543210 |
+-------+----------+-----------+-----------------------+------------+-------------+------+----------------+

mysql> INSERT INTO Employee VALUES(1002, 'Patil', '', 'akshay@gmail.com', 'CSE', 'Developer',
'1995', '9876543210');
ERROR 1048 (23000): Column 'Empid' cannot be null
mysql> select @MESSAGE_TEXT;
+--------------------------+
| @MESSAGE_TEXT            |
+--------------------------+
| Error: Firstname is NULL |
+--------------------------+

mysql> update Employee SET Empid = 1001 where Empid = 1001;
ERROR 1048 (23000): Column 'Empid' cannot be null
mysql> select @MESSAGE_TEXT;
+----------------------+
| @MESSAGE_TEXT        |
+----------------------+
| Error: Empid is wrong |
+----------------------+
```

Experiment No. 6

- **Working with PHP & MySQL**

Design an online E-Banking system for Deposit and withdrawal of funds using HTML, PHP and MySQL. Implement insertion, update and display details module

Customer (acc_no, name, branch-name, balance).

- **HTML Code :**

```
<html>
	<head>
		<title>
			MySQL and PHP connection Assignment
		</title>
	</head>
	<body>
		<form action="phppage.php" method="POST">
			Account No. : <input type="text" name="acc_no" onfocus="this.value=""><br>
			Name : <input type="text" name="name" onfocus="this.value=""><br>
			Branch : <input type="text" name="branch" onfocus="this.value=""><br>
			Balance : <input type="text" name="balance" onfocus="this.value=""><br>
			<input type="submit" value="Insert">
		</form>
		<form action="phppage1.php" method="POST">
			Account No. : <input type="text" name="acc_no" onfocus="this.value=""><br>
			Name : <input type="text" name="name" onfocus="this.value=""><br>
			Branch : <input type="text" name="branch" onfocus="this.value=""><br>
			Balance : <input type="text" name="balance" onfocus="this.value=""><br>
			<input type="submit" value="Update">
		</form>
		<form action="phppage2.php" method="POST">
			Account No. : <input type="text" name="acc_no" onfocus="this.value=""><br>
			<input type="submit" value="Show">
		</form>
		<form action="phppage3.php" method="POST">

			<input type="submit" value="Show All">
		</form>
	</body>
</html>
```

- **php file for insert phppage.php**

```
 <?php
$servername = "localhost";
$username = "root";
```

```
$password = "";
$dbname = "mydb";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
   die("Connection failed: " . $conn->connect_error);
}


$sql = "INSERT INTO Customer VALUES ($_POST[acc_no], '$_POST[name]', '$_POST[branch]',
$_POST[balance])";

if ($conn->query($sql) === TRUE) {
   echo "New record created successfully";
} else {
   echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>
```

- **php file for update phppage1.php**

```
 <?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "mydb";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
   die("Connection failed: " . $conn->connect_error);
}


$sql = "UPDATE Customer SET name = '$_POST[name]', branch_name = '$_POST[branch]', balance
= $_POST[balance] where acc_no = $_POST[acc_no]";

if ($conn->query($sql) === TRUE) {
   echo "Record updated successfully";
} else {
   echo "Error: " . $sql . "<br>" . $conn->error;
}
$conn->close();
?>
```

- **php file for display the given account number phppage2.php**

```
 <?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "mydb";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$query = "select * from Customer where acc_no = $_POST[acc_no];";
$queryResult = $conn->query($query);
echo "<table>";
while ($queryRow = $queryResult->fetch_row()) {
    echo "<tr>";
    for($i = 0; $i < $queryResult->field_count; $i++){
        echo "<td>$queryRow[$i]</td>";
    }
    echo "</tr>";
}
echo "</table>";

$conn->close();
?>
```

- **php file for display all phppage3.php**

```
 <?php
$servername = "localhost";
$username = "root";
$password = "";
$dbname = "mydb";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$query = "select * from Customer limit 100;";
$queryResult = $conn->query($query);
echo "<table>";
while ($queryRow = $queryResult->fetch_row()) {
```

```
    echo "<tr>";
    for($i = 0; $i < $queryResult->field_count; $i++){
        echo "<td>$queryRow[$i]</td>";
    }
    echo "</tr>";
}
echo "</table>";

$conn->close();
?>
```