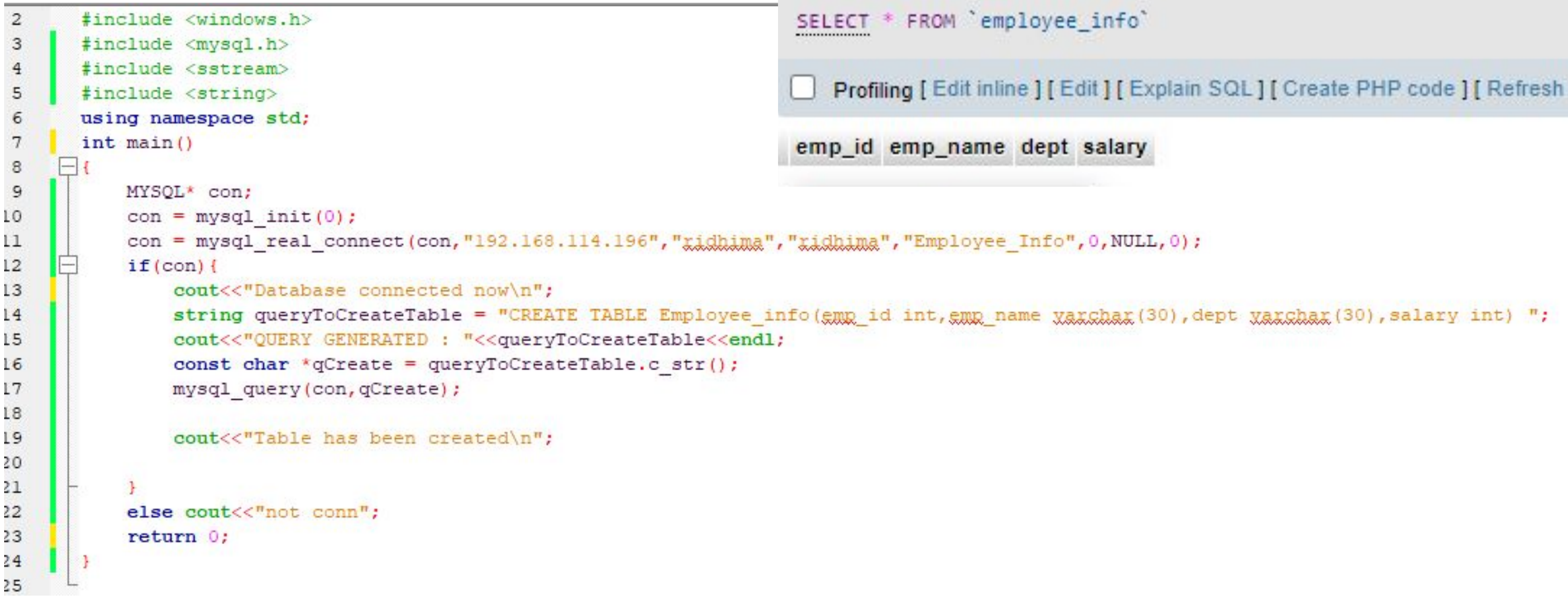
DBMS LAB 1

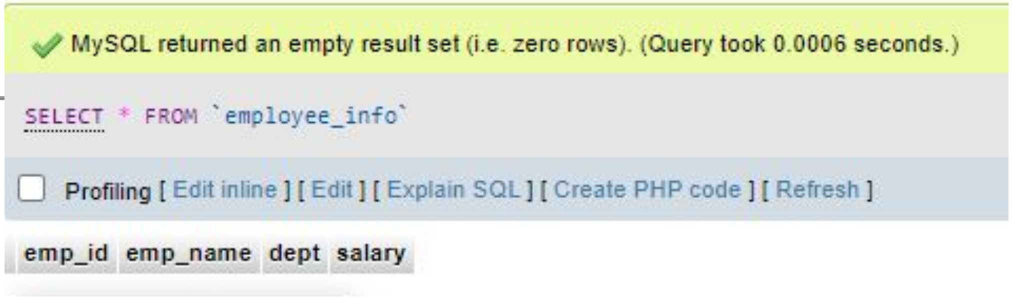
Submitted by
Ridhima Kohli
B19CSE071

Question 1

Table Creation



```cpp
#include <windows.h>
#include <mysql.h>
#include <sstream>
#include <string>
using namespace std;
int main()
{
    MYSQL* con;
    con = mysql_init(0);
    con = mysql_real_connect(con,"192.168.114.196","xidhima","xidhima","Employee_Info",0,NULL,0);
    if(con){
        cout<<"Database connected now\n";
        string queryToCreateTable = "CREATE TABLE Employee_info(emp_id int,emp_name varchar(30),dept varchar(30),salary int)  ";
        cout<<"QUERY GENERATED : "<<queryToCreateTable<<endl;
        const char *qCreate = queryToCreateTable.c_str();
        mysql_query(con,qCreate);

        cout<<"Table has been created\n";

    }
    else cout<<"not conn";
    return 0;
}
```

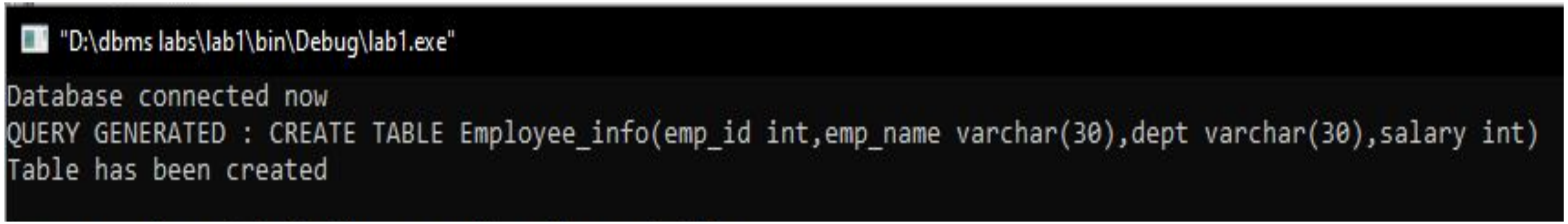


✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0006 seconds.)

```sql
SELECT * FROM `employee_info`
```

☐ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

emp_id	emp_name	dept	salary

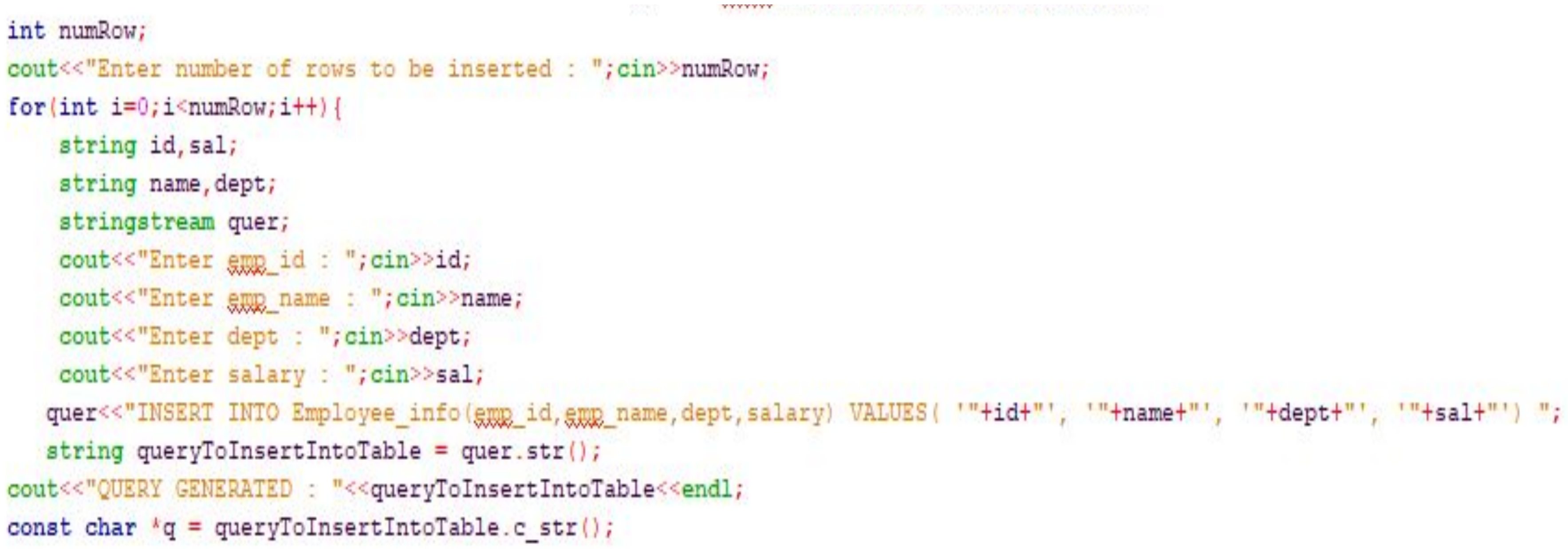


"D:\dbms labs\lab1\bin\Debug\lab1.exe"

```
Database connected now
QUERY GENERATED : CREATE TABLE Employee_info(emp_id int,emp_name varchar(30),dept varchar(30),salary int)
Table has been created
```

Question 1

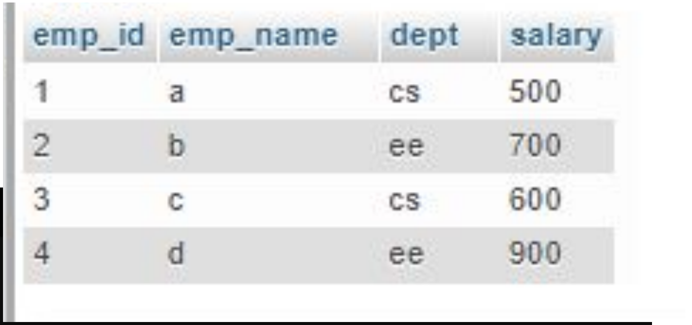
Insertion of values



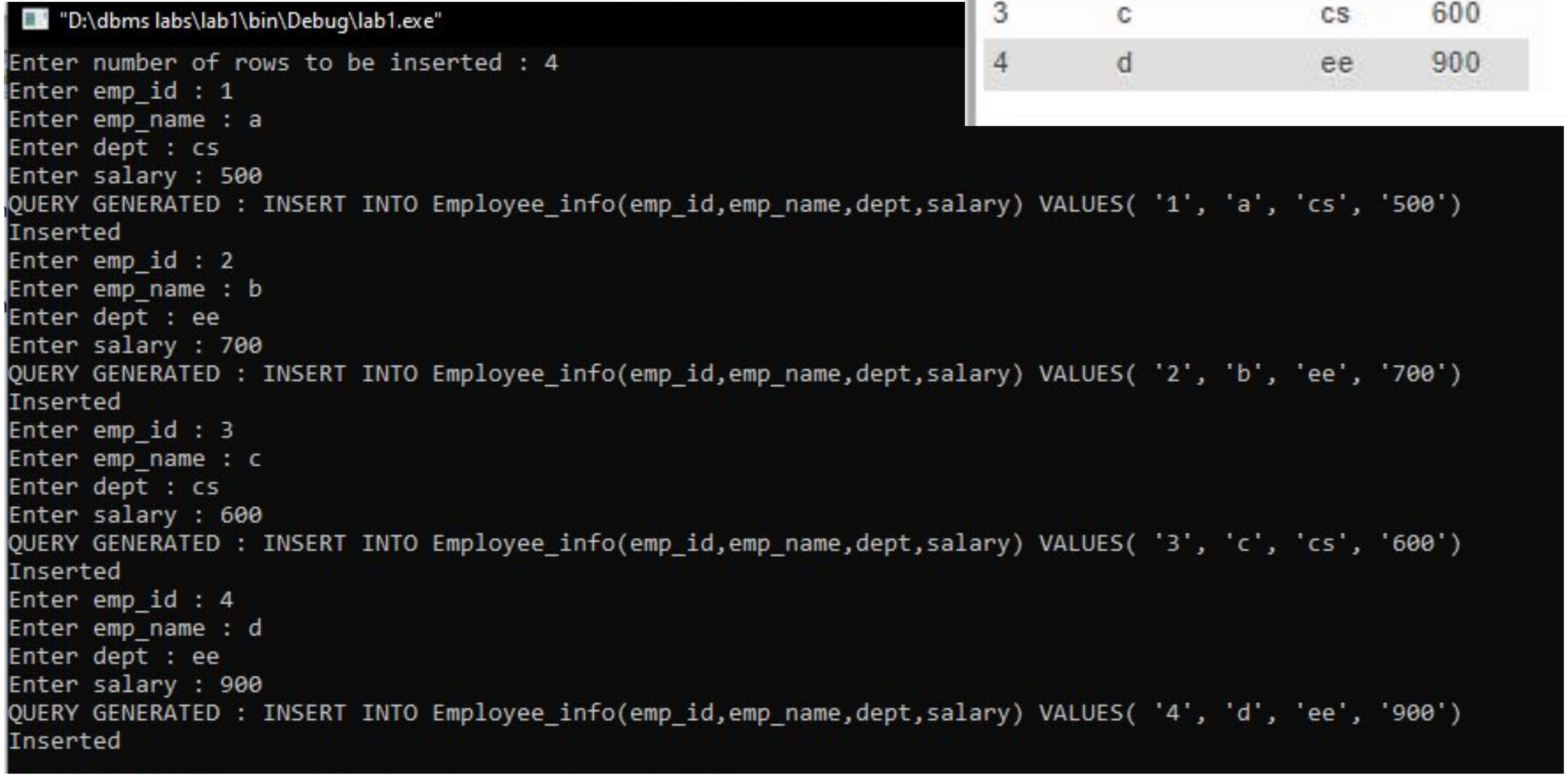
```
int numRow;
cout<<"Enter number of rows to be inserted : ";cin>>numRow;
for(int i=0;i<numRow;i++){
    string id,sal;
    string name,dept;
    stringstream quer;
    cout<<"Enter emp_id : ";cin>>id;
    cout<<"Enter emp_name : ";cin>>name;
    cout<<"Enter dept : ";cin>>dept;
    cout<<"Enter salary : ";cin>>sal;
   quer<<"INSERT INTO Employee_info(emp_id,emp_name,dept,salary) VALUES( '"+id+"', '"+name+"', '"+dept+"', '"+sal+"') ";
   string queryToInsertIntoTable = quer.str();
cout<<"QUERY GENERATED : "<<queryToInsertIntoTable<<endl;
const char *q = queryToInsertIntoTable.c_str();
```

Question 1

Insertion of values



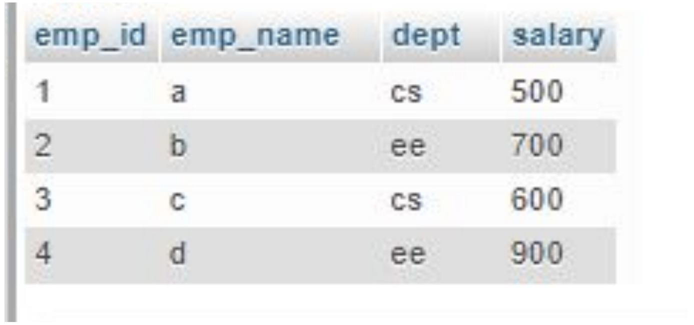
emp_id	emp_name	dept	salary
1	a	cs	500
2	b	ee	700
3	c	cs	600
4	d	ee	900



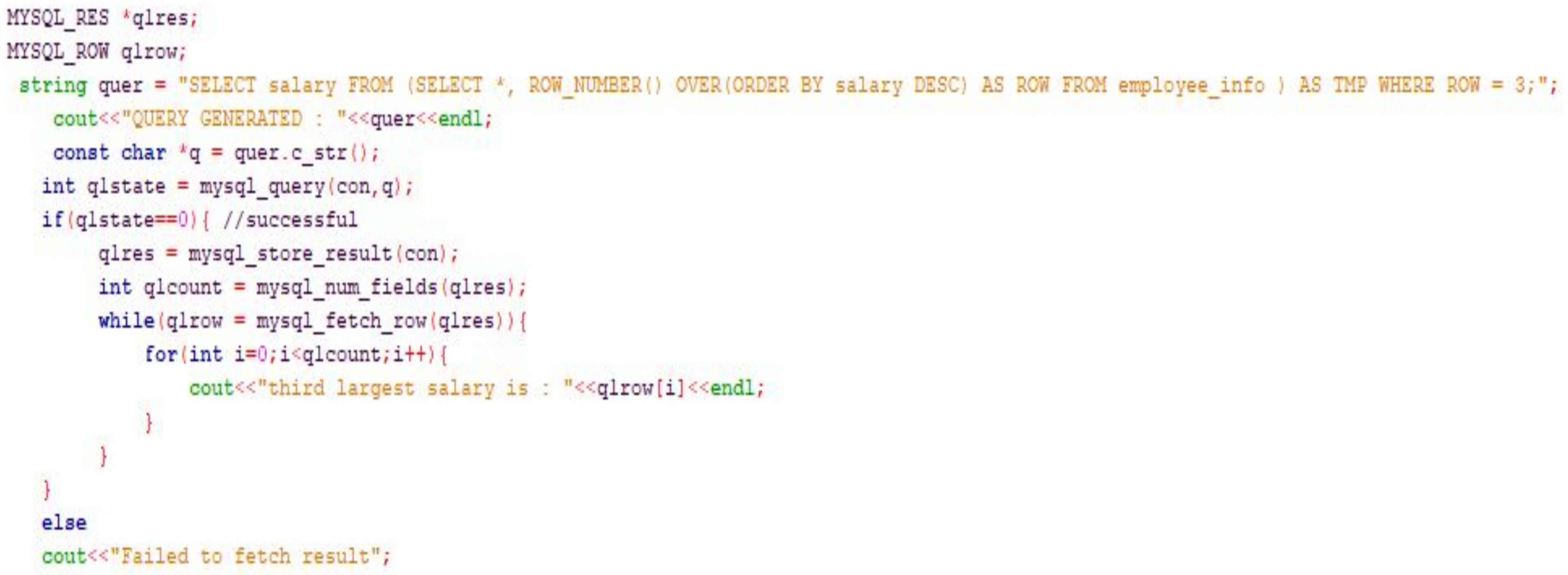
```
"D:\dbms labs\lab1\bin\Debug\lab1.exe"
Enter number of rows to be inserted : 4
Enter emp_id : 1
Enter emp_name : a
Enter dept : cs
Enter salary : 500
QUERY GENERATED : INSERT INTO Employee_info(emp_id,emp_name,dept,salary) VALUES( '1', 'a', 'cs', '500')
Inserted
Enter emp_id : 2
Enter emp_name : b
Enter dept : ee
Enter salary : 700
QUERY GENERATED : INSERT INTO Employee_info(emp_id,emp_name,dept,salary) VALUES( '2', 'b', 'ee', '700')
Inserted
Enter emp_id : 3
Enter emp_name : c
Enter dept : cs
Enter salary : 600
QUERY GENERATED : INSERT INTO Employee_info(emp_id,emp_name,dept,salary) VALUES( '3', 'c', 'cs', '600')
Inserted
Enter emp_id : 4
Enter emp_name : d
Enter dept : ee
Enter salary : 900
QUERY GENERATED : INSERT INTO Employee_info(emp_id,emp_name,dept,salary) VALUES( '4', 'd', 'ee', '900')
Inserted
```

Question 1

a) Find the third_highest salary from the Employee_Info table



emp_id	emp_name	dept	salary
1	a	cs	500
2	b	ee	700
3	c	cs	600
4	d	ee	900

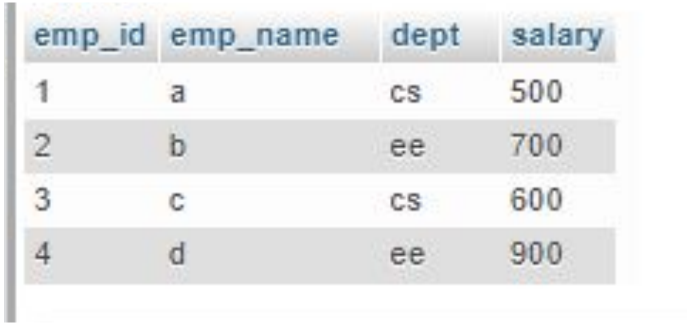


```
MYSQL_RES *qlres;
MYSQL_ROW qlrow;
 string quer = "SELECT salary FROM (SELECT *, ROW_NUMBER() OVER(ORDER BY salary DESC) AS ROW FROM employee_info ) AS TMP WHERE ROW = 3;";
    cout<<"QUERY GENERATED : "<<quer<<endl;
    const char *q = quer.c_str();
   int qlstate = mysql_query(con,q);
   if(qlstate==0){ //successful
        qlres = mysql_store_result(con);
        int qlcount = mysql_num_fields(qlres);
        while(qlrow = mysql_fetch_row(qlres)){
            for(int i=0;i<qlcount;i++){
                cout<<"third largest salary is : "<<qlrow[i]<<endl;
            }
        }
   }
   else
   cout<<"Failed to fetch result";
```

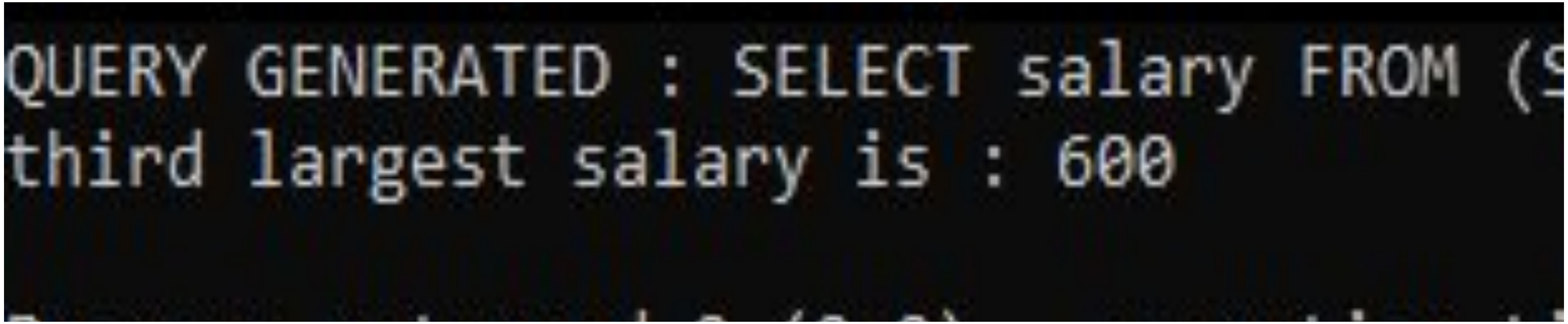
Question 1

a) Find the third_highest salary from the Employee_Info table

Output : -



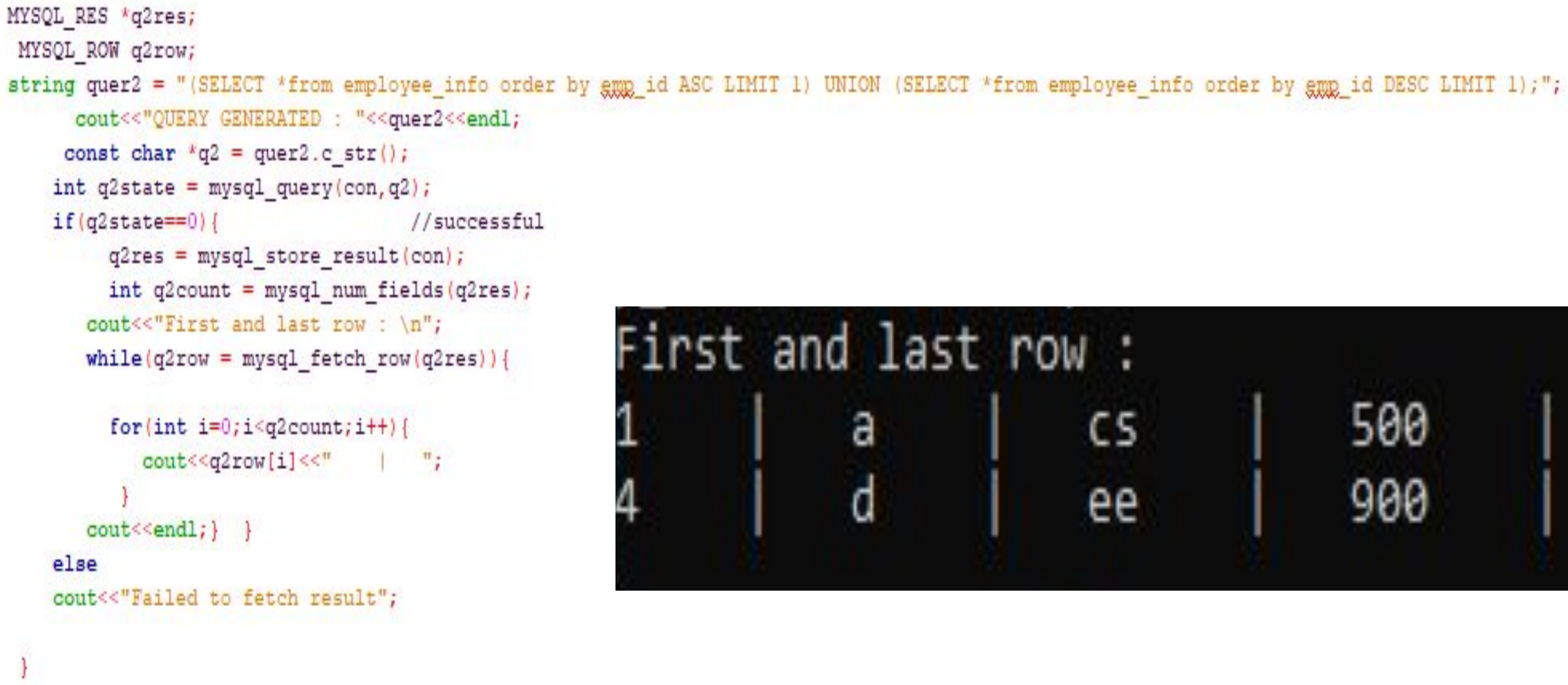
emp_id	emp_name	dept	salary
1	a	cs	500
2	b	ee	700
3	c	cs	600
4	d	ee	900



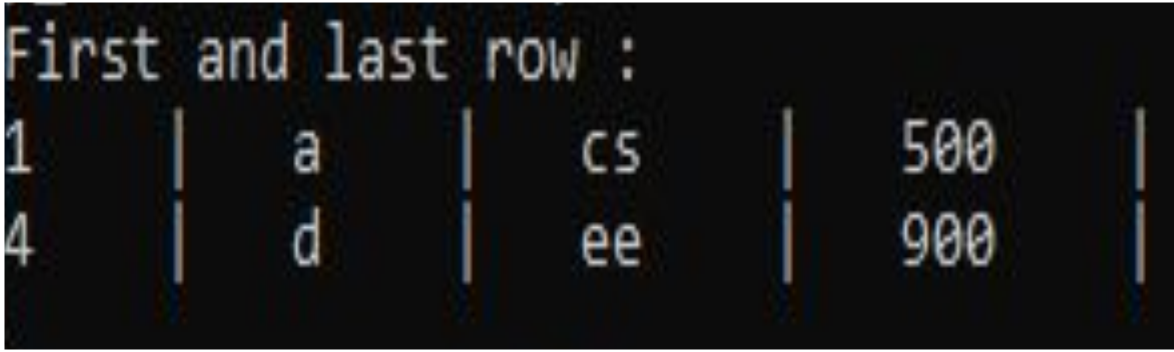
```
QUERY GENERATED : SELECT salary FROM (S
third largest salary is : 600

[illegible]
```

Question 1

a. b) Display the first and last record from the Employee_Info table



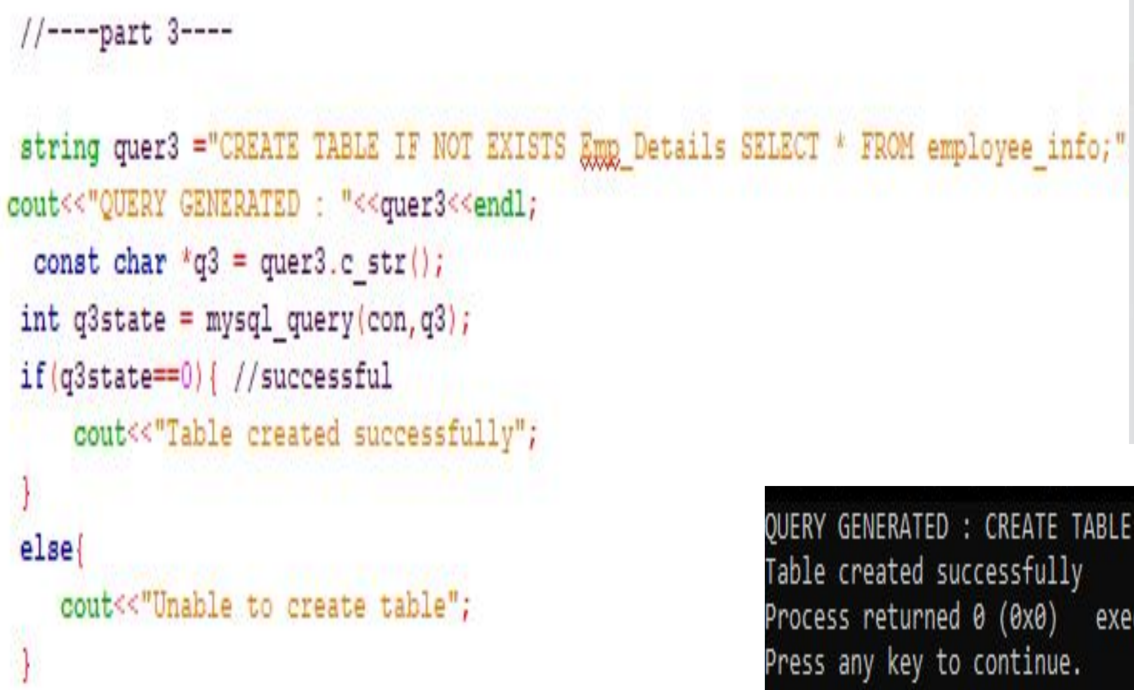
```
MYSQL_RES *q2res;
 MYSQL_ROW q2row;
string quer2 = "(SELECT *from employee_info order by emp_id ASC LIMIT 1) UNION (SELECT *from employee_info order by emp_id DESC LIMIT 1);";
      cout<<"QUERY GENERATED : "<<quer2<<endl;
     const char *q2 = quer2.c_str();
    int q2state = mysql_query(con,q2);
    if(q2state==0){                  //successful
         q2res = mysql_store_result(con);
         int q2count = mysql_num_fields(q2res);
       cout<<"First and last row : \n";
       while(q2row = mysql_fetch_row(q2res)){

         for(int i=0;i<q2count;i++){
            cout<<q2row[i]<<"    |    ";
          }
       cout<<endl;}  }
    else
    cout<<"Failed to fetch result";

 }
```



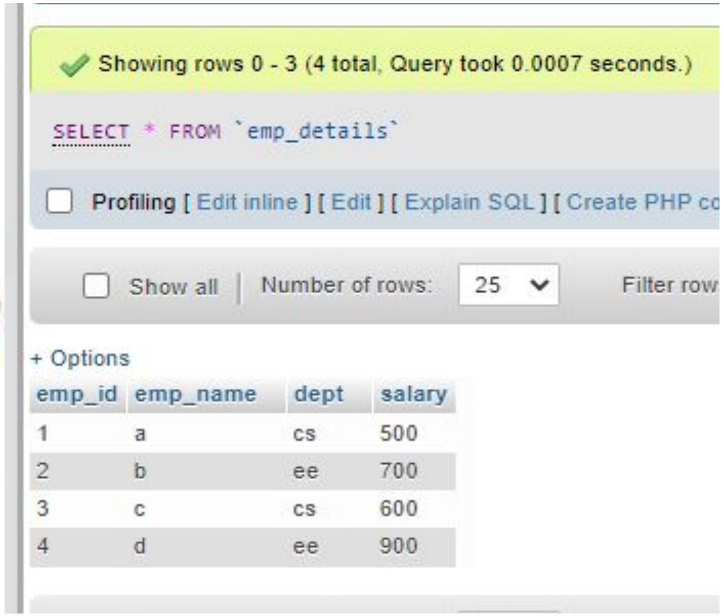
```
First and last row :
1     |    a    |    cs    |    500    |
4     |    d    |    ee    |    900    |
```

Question 1

c) Copy all rows of the Employee_Info table into another new table Emp_Details



```
//----part 3----

string quer3 ="CREATE TABLE IF NOT EXISTS Emp_Details SELECT * FROM employee_info;"
cout<<"QUERY GENERATED : "<<quer3<<endl;
 const char *q3 = quer3.c_str();
int q3state = mysql_query(con,q3);
if(q3state==0){ //successful
    cout<<"Table created successfully";
}
else{
   cout<<"Unable to create table";
}
```



✓ Showing rows 0 - 3 (4 total, Query took 0.0007 seconds.)

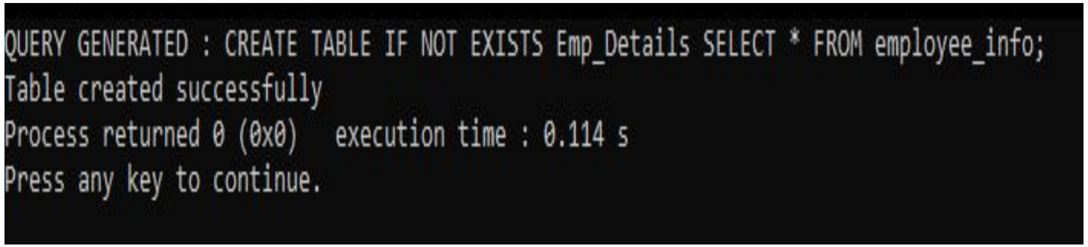
```
SELECT * FROM `emp_details`
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP co

Show all | Number of rows: 25 Filter row

+ Options

emp_id	emp_name	dept	salary
1	a	cs	500
2	b	ee	700
3	c	cs	600
4	d	ee	900



```
QUERY GENERATED : CREATE TABLE IF NOT EXISTS Emp_Details SELECT * FROM employee_info;
Table created successfully
Process returned 0 (0x0)   execution time : 0.114 s
Press any key to continue.
```

Question 2. Create the following four tables: **(10 points)** **(Table creation in next slide)**
Employee(emp_name, street, city)
Works(emp_name, company_name, salary)
Company(company_name, city)
Managers(emp_name, manager_name)

a. Identify the Primary Key-Foreign Key relationships between the tables. You may design your own Primary Keys, if required. Clearly state assumptions, if any. **(5 points)**

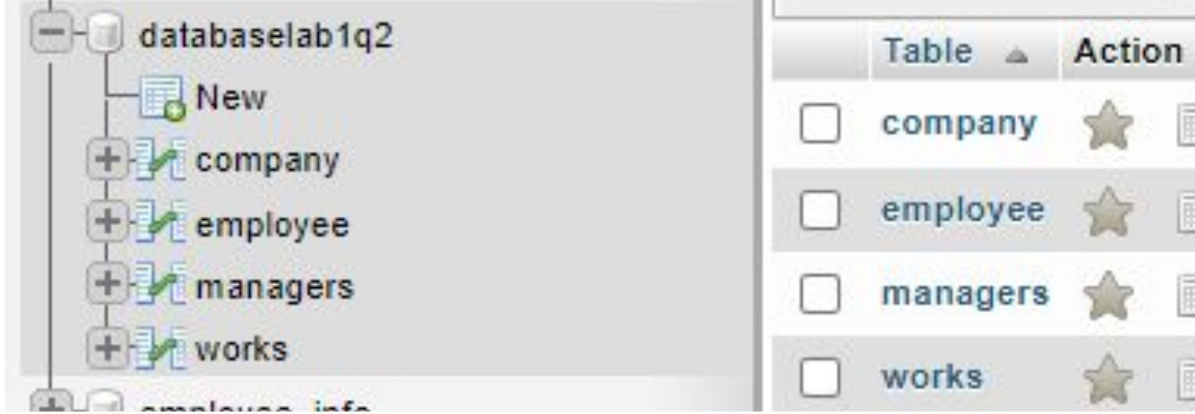
Answer : We create primary key e_id and c_id which refer to employee id and company id in order to uniquely identify the employee/company record since their names might be common.

Following table shows the primary key and foreign keys of these four tables

Table	**Primary Key**	**Foreign Key**
Employee	e_id	-
Works	e_id+c_id	c_id,e_id
Company	c_id	-
Managers	e_id	-

Question 2 : Queries to create tables

Just like question 1 , we can use the functions in c++ and execute the given queries



```
CREATE TABLE Employee(e_id int NOT NULL,emp_name varchar(30) NOT NULL,street int,city varchar(30),
PRIMARY KEY (e_id))
```

```
CREATE TABLE Company(c_id int NOT NULL,company_name varchar(30) NOT NULL,city varchar(30),
PRIMARY KEY (c_id))
```

```
CREATE TABLE Works(e_id int NOT NULL,emp_name varchar(30) NOT NULL,company_name varchar(30),c_id int,
FOREIGN KEY(e_id) REFERENCES Employee(e_id), FOREIGN KEY(c_id) REFERENCES Company(c_id),PRIMARY KEY
(e_id,c_id))
```

```
CREATE TABLE Managers(e_id int NOT NULL,emp_name varchar(30) NOT NULL, company_name
varchar(30),FOREIGN KEY(e_id) REFERENCES Employee(e_id),PRIMARY KEY (e_id))
```

Question 2.

B) Write SQL queries for the following: **(4 * 5 = 20 points)**

1. Find **names** of all employees who work for SBI

```
SELECT emp_name FROM Employee WHERE e_id IN
(SELECT DISTINCT e_id FROM Works WHERE
company_name = “SBI” )
```

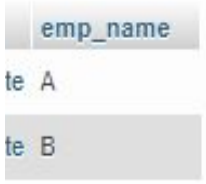
2. Find **cities of residence** of all employees who work for SBI

```
SELECT DISTINCT city FROM Employees WHERE e_id IN
(SELECT DISTINCT e_id FROM Works WHERE
company_name = “SBI” )
```

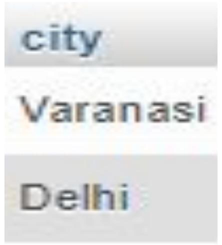
3. Find **names** of all employees who don’t work for SBI

```
SELECT e_name FROM Employee WHERE e_id NOT IN
(SELECT DISTINCT e_id FROM Works WHERE
company_name = ‘SBI’ )
```

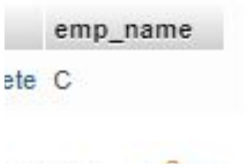
Outputs



	emp_name
te	A
te	B



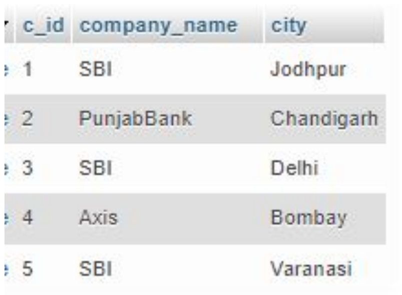
city
Varanasi
Delhi



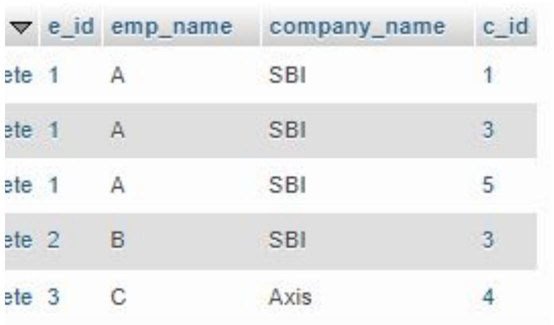
	emp_name
ete	C

4 Find names of all employees who have worked for all branches of SBI

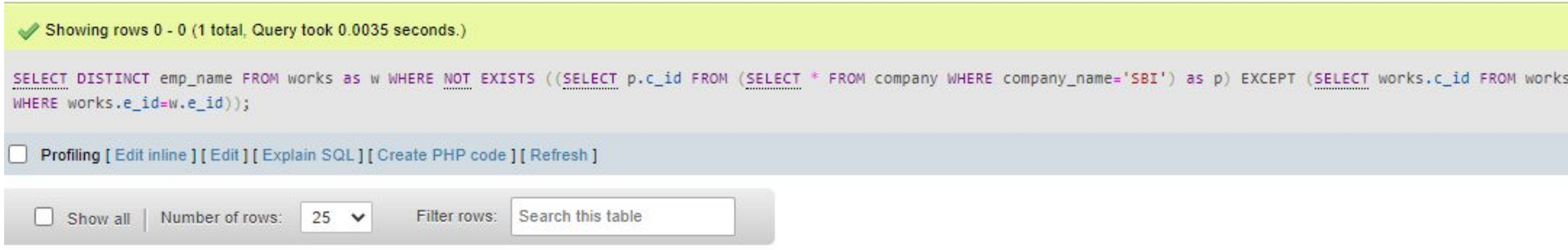
SELECT DISTINCT emp_name FROM works as w WHERE NOT EXISTS ((SELECT p.c_id FROM (SELECT * FROM company WHERE company_name='SBI') as p) EXCEPT (SELECT works.c_id FROM works WHERE works.e_id=w.e_id))



	c_id	company_name	city
:	1	SBI	Jodhpur
:	2	PunjabBank	Chandigarh
:	3	SBI	Delhi
:	4	Axis	Bombay
:	5	SBI	Varanasi



	e_id	emp_name	company_name	c_id
ete	1	A	SBI	1
ete	1	A	SBI	3
ete	1	A	SBI	5
ete	2	B	SBI	3
ete	3	C	Axis	4

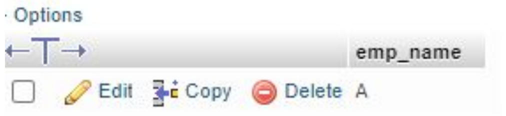


Showing rows 0 - 0 (1 total, Query took 0.0035 seconds.)

```
SELECT DISTINCT emp_name FROM works as w WHERE NOT EXISTS ((SELECT p.c_id FROM (SELECT * FROM company WHERE company_name='SBI') as p) EXCEPT (SELECT works.c_id FROM works
WHERE works.e_id=w.e_id));
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table



Options

←T→	emp_name
Edit Copy Delete	A

For part 4 we use Relational Division with results as shown

Question 2.

c) Simulate examples of various anomalies like Insertion, Deletion, and Update on referenced as well as referencing relations in the aforementioned database. Specify what anomalies would violate the Referential integrity constraint and what could be a potential solution for the same. **(10 points)**

Insertion anomaly : If we try to insert a value into a table whose foreign key value is not present as primary key in parent table , then it would not be inserted . This is insertion anomaly.

Deletion/Updation anomaly : When we try to update/delete from parent but the attribute is used as foreign key in some other relation , then the operation won't be allowed.

In the given database, violation of Referential Integrity can occur in these ways

- Insertion anomaly : If foreign keys are not provided and we add an emp_name or company_name in works or managers table who are not present in Employee and Company relation

  Example :

  If foreign keys are not used , we might insert a record ("Ram","Axis",2000) in Works table. However employee names Ram is not in the main Employee database , hence creates an anomaly

- Updation/Deletion anomaly: If we don't provide an ON UPDATE and ON DELETE constraint (example cascade) then incase an employee's data is deleted from Employee relation , its values won't be deleted from Works and Managers relations.

Example :

If we don't put constraints , then on deleting record (1,A,25,Varanasi) from employee , the works table wouldn't know what to do with e_id 1

Hence to avoid this , we use foreign keys e_id and c_id as shown.
For avoiding updation and deletion anomalies , we need to put constraints of ON UPDATE CASCADE , ON DELETE CASCADE , etc

Employee(e_id,emp_name, street, city)
Works(e_id,emp_name,company_name, c_id,salary)
Company(c_id,company_name, city)
Managers(e_id,emp_name, manager_name)

QUESTION 3

Create the following tables: **(10 points)** **(Queries on next slide)**

Employee(emp_name, email, contact_no., department)

Department(emp_name, salary, emp_designation)

Awardee(emp_name, email, department, experience)

a. Identify Primary Key-Foreign Key relationships between the tables. You may design your own Primary Keys, if required. Clearly state assumptions, if any. **(5 points)**

Assumption : **Every employee works in separate department. The department names are different within that company whose database is being made. The emp_names,experience and salary values can be common/same. Since the emails and contact numbers cannot be same , so either of them can be primary key or foreign key. But we are assuming that not all employees provide their email or contact and hence these values can be NULL. Therefore we add an extra key e_id which is the employee id and use it as primary key as well as foreign key for these relations as shown : -**

Table	Primary Key	Foreign Key
Employee	e_id	-
Department	e_id	e_id
Awardee	e_id	e_id

QUESTION 3

Create the following tables:

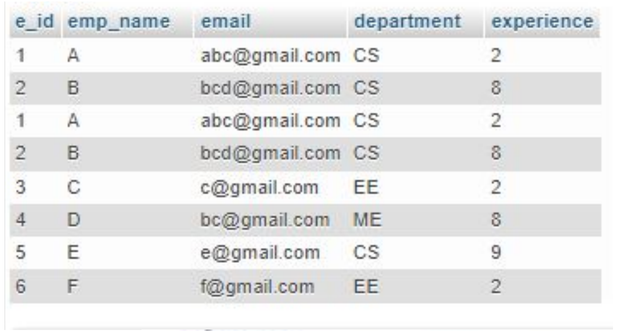
Employee(emp_name, email, contact_no., department)
Department(emp_name, salary, emp_designation)
Awardee(emp_name, email, department, experience)

```
CREATE TABLE Employee(e_id int NOT NULL,emp_name VARCHAR(30) NOT NULL, email
VARCHAR(30),contact_no CHAR(10) ,department VARCHAR(30),PRIMARY KEY (e_id))

CREATE TABLE Department(e_id int NOT NULL,emp_name VARCHAR(30) NOT NULL,salary
int,emp_designation VARCHAR(30),PRIMARY KEY (e_id),FOREIGN KEY e_id REFERENCES
Employee(e_id))


CREATE TABLE Awardee(e_id int NOT NULL,emp_name VARCHAR(30) NOT NULL, email
VARCHAR(30),department VARCHAR(30),experience int,PRIMARY KEY (e_id),FOREIGN KEY
e_id REFERENCES Employee(e_id))
```

b. Write SQL queries for the following: **(2 * 5 = 10 points)**

1. If an update is made on an entry for a particular employee (X) in one table, then it will automatically update corresponding values for X in all other associated tables

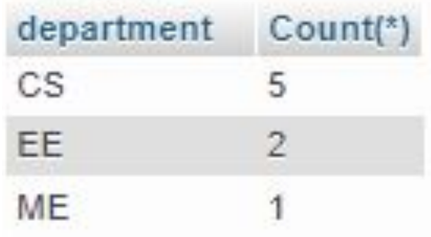
```
ALTER TABLE Department  WITH CHECK ADD  CONSTRAINT update_cons_fk FOREIGN KEY(e_id)
 REFERENCES Employee (e_id )
 ON UPDATE CASCADE

ALTER TABLE Awardee WITH CHECK ADD  CONSTRAINT update_cons_fk FOREIGN KEY(e_id)
 REFERENCES Employee (e_id )
 ON UPDATE CASCADE
```



e_id	emp_name	email	department	experience
1	A	abc@gmail.com	CS	2
2	B	bcd@gmail.com	CS	8
1	A	abc@gmail.com	CS	2
2	B	bcd@gmail.com	CS	8
3	C	c@gmail.com	EE	2
4	D	bc@gmail.com	ME	8
5	E	e@gmail.com	CS	9
6	F	f@gmail.com	EE	2

2. Find no. of awardees from each department

```
SELECT department,Count(*) FROM Awardee GROUP BY Department
```



department	Count(*)
CS	5
EE	2
ME	1

Some Observations

- During table creation in question 1, if datatype is not specified , the compiler doesn’t throw any error nor does the query return non zero output. But the table is not created. Table is created when data type for fields is specified
- For question 2 b , part 4 there can be two methods - one using Relational division and other by equating the count of sbi branches with sbi branches count grouped by employee id. However former method is used as second is more programming oriented , the former fulfills the learning objective of assignment