The document is a collection of SQL exercises focused on managing data in an employee management system with two main tables: **Departments** and **Employees**. is a summary of the key points:

1. **Table Creation**:
   - Two tables are created: Departments (with Code, Name, and Budget) and Employees (with SSN, Name, LastName, and Department, which is a foreign key referencing Departments.Code).
2. **Data Insertion**:
   - Sample data is inserted into both tables. The Departments table contains departments such as IT, Accounting, Human Resources, and Research with corresponding budgets. The Employees table contains employees with their names, social security numbers, and associated departments.
3. **SQL Query Exercises**:
   - The exercises include a range of SQL operations, such as:
     - **Basic Select Queries**: Retrieve specific columns, such as last names, or filter employees based on criteria like last name or department.
     - **Aggregations**: Calculate the sum of department budgets and count the number of employees in each department.
     - **Joins**: Combine data from both tables to show employees along with their department information.
     - **Advanced Queries**: Select employees from departments with budgets larger than a specific value, departments with more than two employees, or departments with budgets above the average.
     - **Data Manipulation**: Update department budgets, reassign employees, delete records based on specific conditions, and insert new departments or employees.
4. **Complex Queries**:
   - Some queries involve using subqueries, such as finding departments with a budget greater than the average budget, or identifying employees in the department with the second-lowest budget.
5. **Data Modification Tasks**:
   - Tasks include reducing department budgets by 10%, moving all employees from one department to another, and deleting employees based on their department's budget.

```
-- Create Tables

CREATE TABLE Departments (
 Code INTEGER PRIMARY KEY,
 Name varchar(255) NOT NULL ,
 Budget decimal NOT NULL
);
```

- **Employees Table:**
  - SSN: An integer serving as the unique identifier (primary key) for each employee.
  - Name: The employee's first name.
  - LastName: The employee's last name.
  - Department: An integer representing the department to which the employee belongs. It is a foreign key referencing the Code column in the Departments table.

  ```
  CREATE TABLE Employees (
   SSN INTEGER PRIMARY KEY,
   Name varchar(255) NOT NULL ,
   LastName varchar(255) NOT NULL ,
   Department INTEGER NOT NULL ,
   foreign key (department) references Departments(Code)
  ) ENGINE=INNODB;
  ```

2. Inserting Data:

- Data is inserted into the **Departments** table, representing different departments along with their respective budget allocations.

  ```
  INSERT INTO Departments(Code,Name,Budget) VALUES(14,'IT',65000);
  INSERT INTO Departments(Code,Name,Budget) VALUES(37,'Accounting',15000);
  INSERT INTO Departments(Code,Name,Budget) VALUES(59,'Human Resources',240000);
  INSERT INTO Departments(Code,Name,Budget) VALUES(77,'Research',55000);
  ```

- Data is inserted into the **Employees** table, representing employees and the departments they work for.

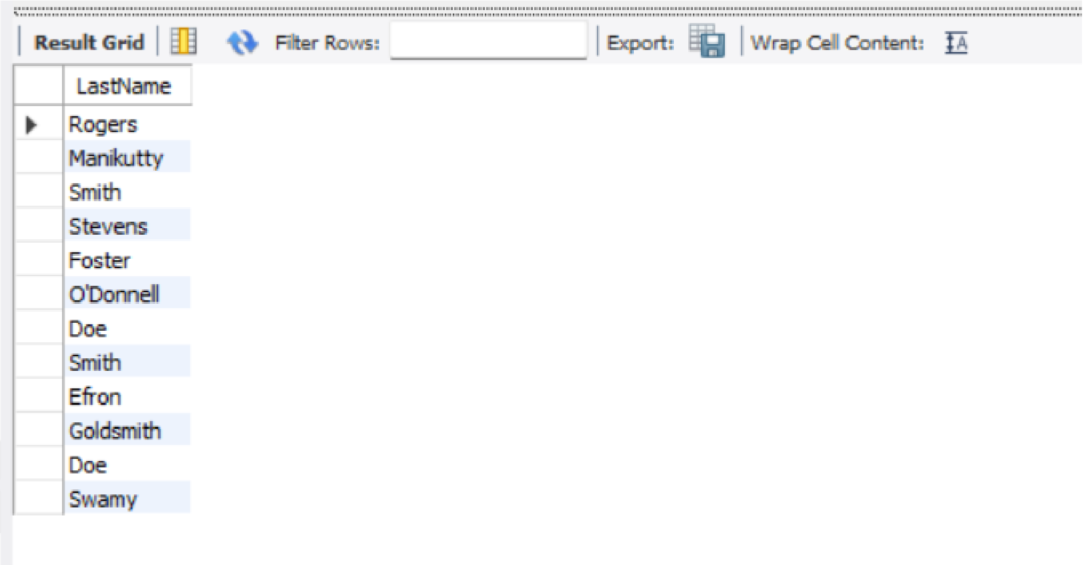
  ```
  INSERT INTO Employees(SSN,Name,LastName,Department)
  VALUES('123234877','Michael','Rogers',14);
  INSERT INTO Employees(SSN,Name,LastName,Department)
  VALUES('152934485','Anand','Manikutty',14);
  INSERT INTO Employees(SSN,Name,LastName,Department)
  VALUES('222364883','Carol','Smith',37);
  INSERT INTO Employees(SSN,Name,LastName,Department)
  VALUES('326587417','Joe','Stevens',37);
  ```

```
INSERT INTO Employees(SSN,Name,LastName,Department) VALUES('332154719','Mary-Anne','Foster',14);

INSERT INTO Employees(SSN,Name,LastName,Department) VALUES('332569843','George','O''Donnell',77);

INSERT INTO Employees(SSN,Name,LastName,Department) VALUES('546523478','John','Doe',59);

INSERT INTO Employees(SSN,Name,LastName,Department) VALUES('631231482','David','Smith',77);

INSERT INTO Employees(SSN,Name,LastName,Department) VALUES('654873219','Zacary','Efron',59);

INSERT INTO Employees(SSN,Name,LastName,Department) VALUES('745685214','Eric','Goldsmith',59);

INSERT INTO Employees(SSN,Name,LastName,Department) VALUES('845657245','Elizabeth','Doe',14);

INSERT INTO Employees(SSN,Name,LastName,Department) VALUES('845657246','Kumar','Swamy',14);
```

Q.1. Select the last name of all employees.

Code:- `SELECT LastName FROM employees;`

Output :

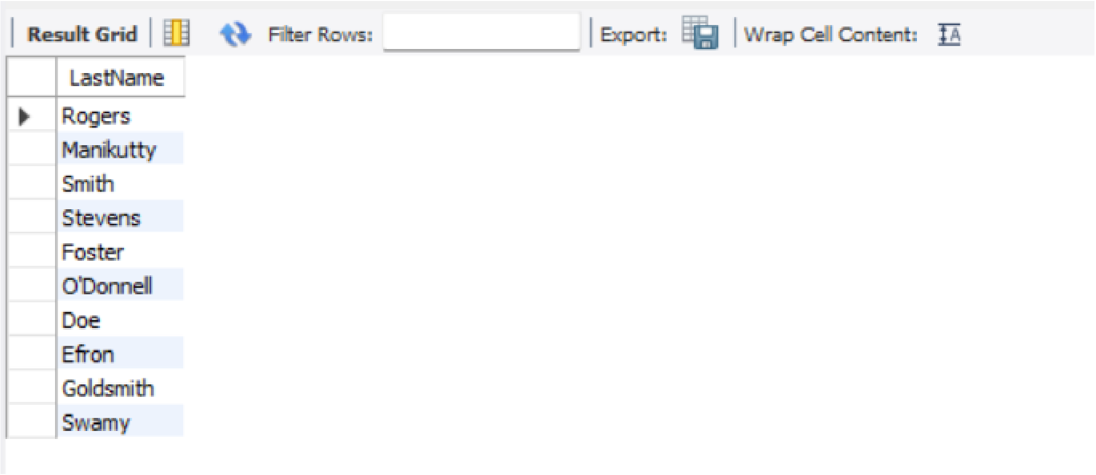


LastName
Rogers
Manikutty
Smith
Stevens
Foster
O'Donnell
Doe
Smith
Efron
Goldsmith
Doe
Swamy

Q.2. Select the last name of all employees, without duplicates.

Code: - **SELECT DISTINCT LastName FROM Employees;**

Output



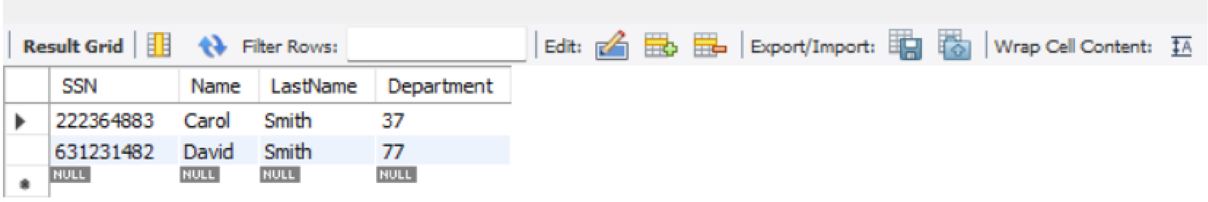
LastName
Rogers
Manikutty
Smith
Stevens
Foster
O'Donnell
Doe
Efron
Goldsmith
Swamy

Q.3. Select all the data of employees whose last name is "Smith".

Code:- SELECT * FROM employees

WHERE LastName='Smith';

Output



SSN	Name	LastName	Department
222364883	Carol	Smith	37
631231482	David	Smith	77
NULL	NULL	NULL	NULL

Q.4. Select all the data of employees whose last name is "Smith" or "Doe".

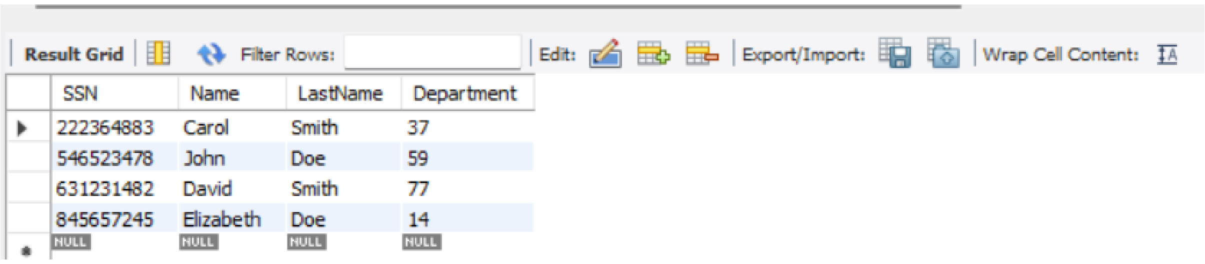
Code:-

/* With OR */

SELECT * FROM Employees

WHERE LastName = 'Smith' OR LastName = 'Doe';

Output:-

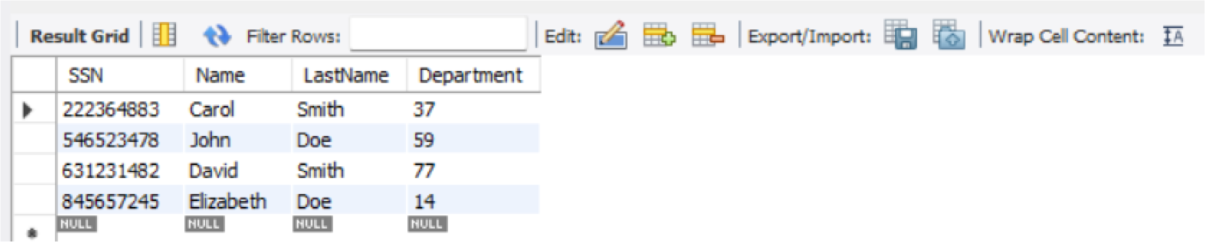


SSN	Name	LastName	Department
222364883	Carol	Smith	37
546523478	John	Doe	59
631231482	David	Smith	77
845657245	Elizabeth	Doe	14
NULL	NULL	NULL	NULL

/* With IN */

```
SELECT * FROM Employees
  WHERE LastName IN ('Smith' , 'Doe');
```

Output:-



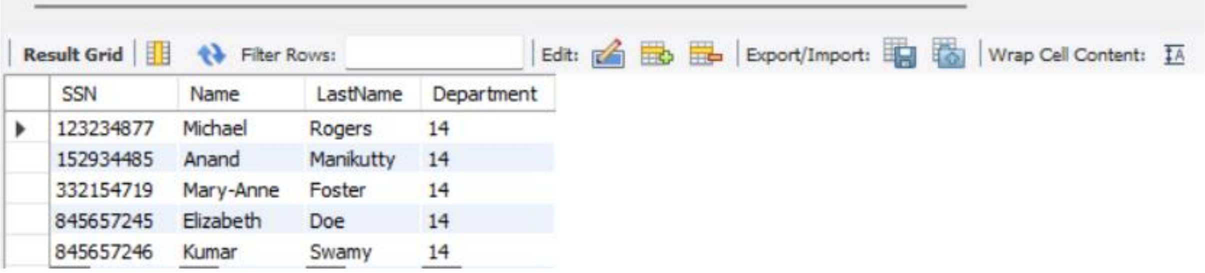
SSN	Name	LastName	Department
222364883	Carol	Smith	37
546523478	John	Doe	59
631231482	David	Smith	77
845657245	Elizabeth	Doe	14
NULL	NULL	NULL	NULL

Q.5. Select all the data of employees that work in department 14.

Code:-

```
SELECT * FROM Employees
  WHERE Department=14;
```

Output:-



SSN	Name	LastName	Department
123234877	Michael	Rogers	14
152934485	Anand	Manikutty	14
332154719	Mary-Anne	Foster	14
845657245	Elizabeth	Doe	14
845657246	Kumar	Swamy	14

Q.6. Select all the data of employees that work in department 37 or department 77.

Code:-

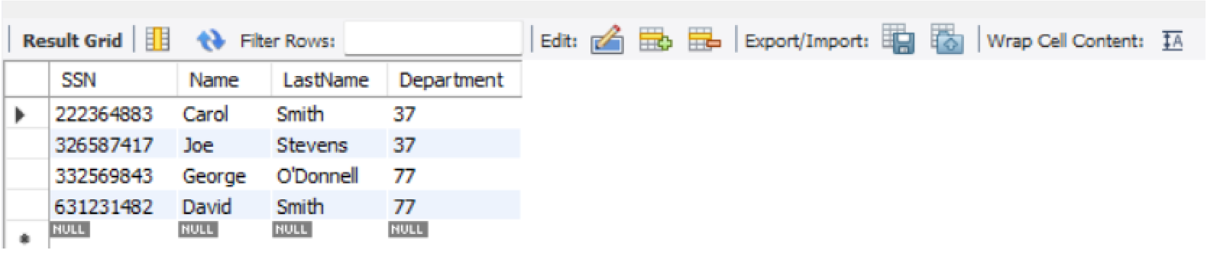
/* With OR */

```
SELECT * FROM Employees
  WHERE Department = 37 OR Department = 77;
```

/* With IN */

```
SELECT * FROM Employees
  WHERE Department IN (37,77);
```

Output:-



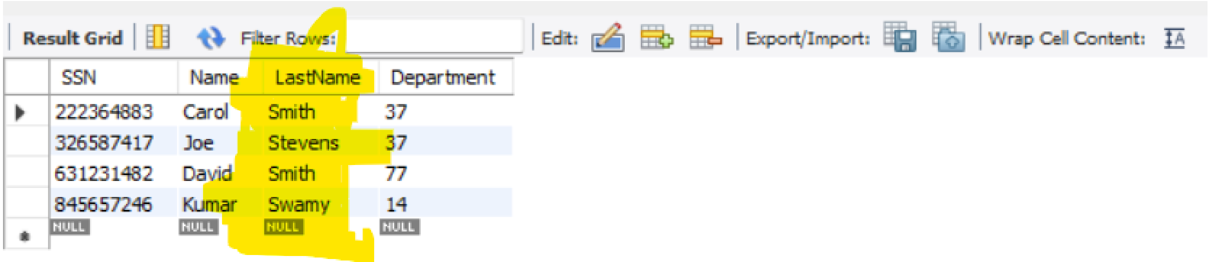
SSN	Name	LastName	Department
222364883	Carol	Smith	37
326587417	Joe	Stevens	37
332569843	George	O'Donnell	77
631231482	David	Smith	77
NULL	NULL	NULL	NULL

Q.7. Select all the data of employees whose last name begins with an "S".

Code:

```
SELECT * FROM Employees

  WHERE LastName LIKE 'S%';
```

Output:

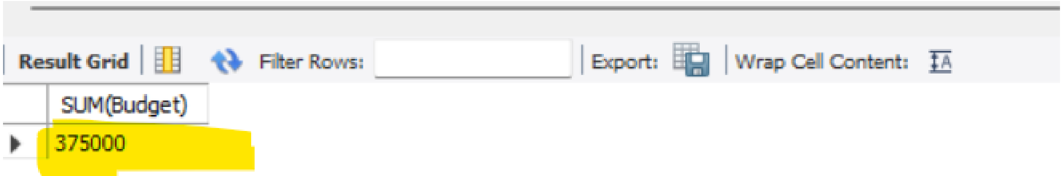


SSN	Name	LastName	Department
222364883	Carol	Smith	37
326587417	Joe	Stevens	37
631231482	David	Smith	77
845657246	Kumar	Swamy	14
NULL	NULL	NULL	NULL

Q.8. Select the sum of all the departments' budgets.

Code: `SELECT SUM(Budget) FROM departments;`

Output:



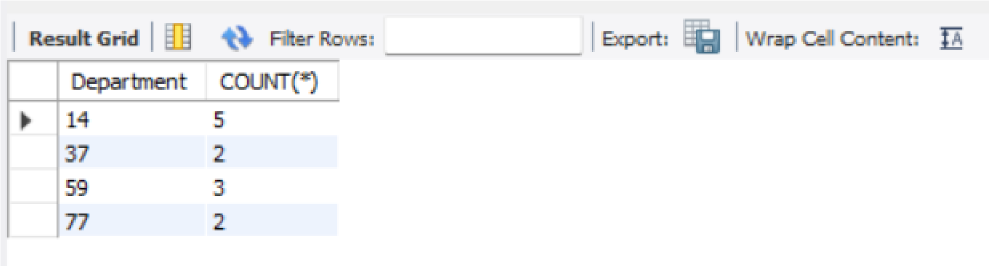
SUM(Budget)
375000

Q.9. Select the number of employees in each department (you only need to show the department code and the number of employees).

Code:

```
SELECT Department, COUNT(*)

  FROM Employees

  GROUP BY Department;
```

Output:



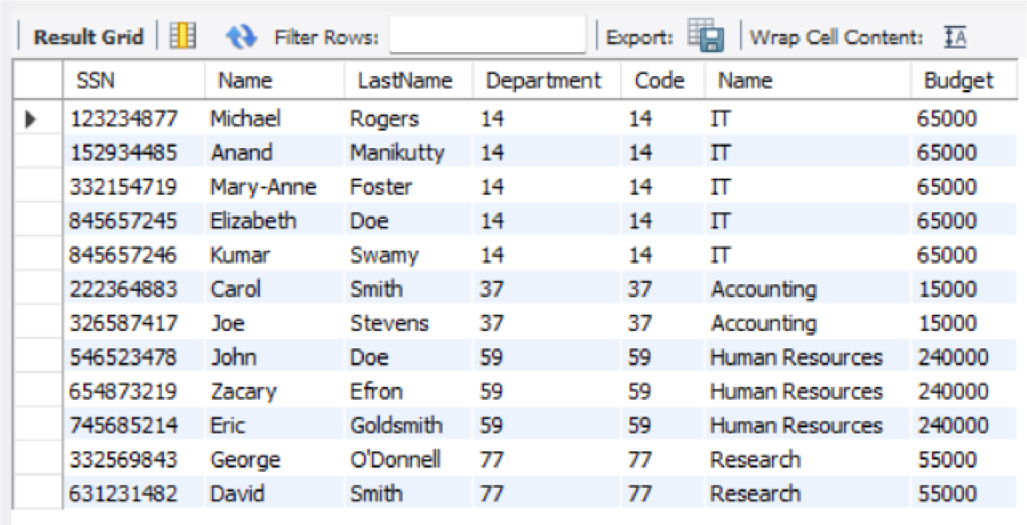
Department	COUNT(*)
14	5
37	2
59	3
77	2

Q.10. Select all the data of employees, including each employee's department's data.

Code:

```
SELECT *
 FROM Employees E INNER JOIN Departments D
 ON E.Department = D.Code;
```

Output:



SSN	Name	LastName	Department	Code	Name	Budget
123234877	Michael	Rogers	14	14	IT	65000
152934485	Anand	Manikutty	14	14	IT	65000
332154719	Mary-Anne	Foster	14	14	IT	65000
845657245	Elizabeth	Doe	14	14	IT	65000
845657246	Kumar	Swamy	14	14	IT	65000
222364883	Carol	Smith	37	37	Accounting	15000
326587417	Joe	Stevens	37	37	Accounting	15000
546523478	John	Doe	59	59	Human Resources	240000
654873219	Zacary	Efron	59	59	Human Resources	240000
745685214	Eric	Goldsmith	59	59	Human Resources	240000
332569843	George	O'Donnell	77	77	Research	55000
631231482	David	Smith	77	77	Research	55000

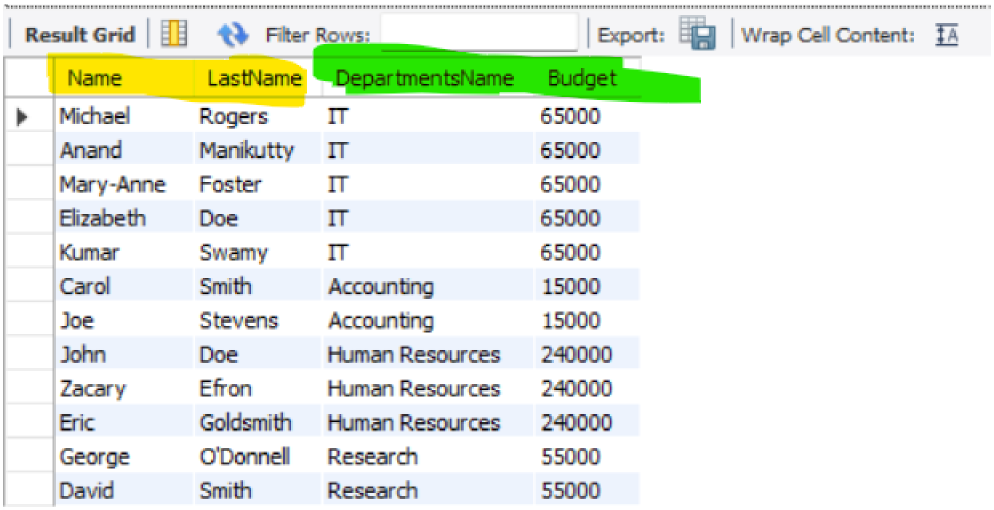
Q.11. Select the name and last name of each employee, along with the name and budget of the employee's department.

Code: /* Without labels */

```
SELECT Employees.Name, LastName, Departments.Name AS DepartmentsName, Budget
  FROM Employees INNER JOIN Departments
  ON Employees.Department = Departments.Code;


/* With labels */
SELECT E.Name, LastName, D.Name AS DepartmentsName, Budget
  FROM Employees E INNER JOIN Departments D
```

```
 ON E.Department = D.Code;
```

Output:



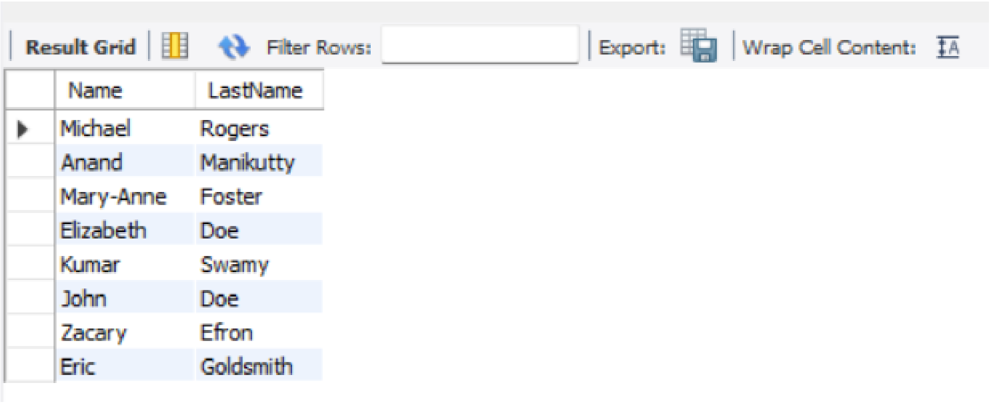
Name	LastName	DepartmentsName	Budget
Michael	Rogers	IT	65000
Anand	Manikutty	IT	65000
Mary-Anne	Foster	IT	65000
Elizabeth	Doe	IT	65000
Kumar	Swamy	IT	65000
Carol	Smith	Accounting	15000
Joe	Stevens	Accounting	15000
John	Doe	Human Resources	240000
Zacary	Efron	Human Resources	240000
Eric	Goldsmith	Human Resources	240000
George	O'Donnell	Research	55000
David	Smith	Research	55000

Q.12. Select the name and last name of employees working for departments with a budget greater than $60,000.

Code:

```
 /* Without subquery */

SELECT Employees.Name, LastName

 FROM Employees INNER JOIN Departments

 ON Employees.Department = Departments.Code

  AND Departments.Budget > 60000;

/* With subquery */

SELECT Name, LastName FROM Employees

 WHERE Department IN

 (SELECT Code FROM Departments WHERE Budget > 60000);
```

Output:



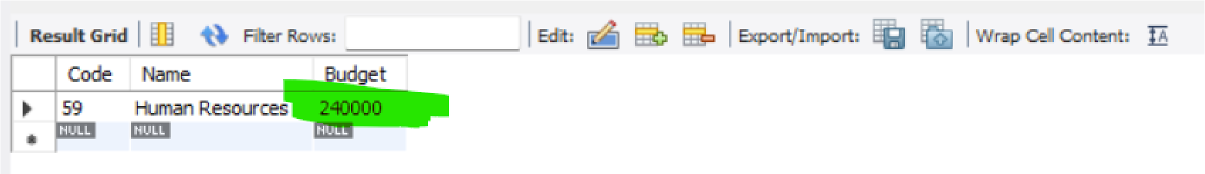
Name	LastName
Michael	Rogers
Anand	Manikutty
Mary-Anne	Foster
Elizabeth	Doe
Kumar	Swamy
John	Doe
Zacary	Efron
Eric	Goldsmith

Q.13. Select the departments with a budget larger than the average budget of all the departments.

Code:

```
SELECT *
 FROM Departments
 WHERE Budget >
 (
   SELECT AVG(Budget)
   FROM Departments
 );
```

Output:



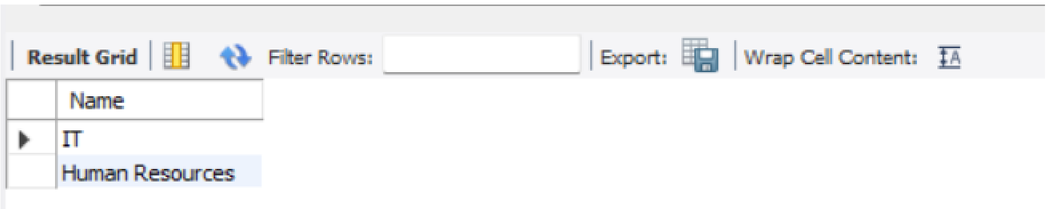
Code	Name	Budget
59	Human Resources	240000
NULL	NULL	NULL

Q.14. Select the names of departments with more than two employees.

Code:

```
/*With subquery*/
SELECT D.Name FROM Departments D
 WHERE 2 <
 (
  SELECT COUNT(*)
    FROM Employees
    WHERE Department = D.Code
 );
/* With IN and subquery */
SELECT Name FROM Departments
 WHERE Code IN
 (
   SELECT Department
     FROM Employees
     GROUP BY Department
    HAVING COUNT(*) > 2
 );
/* With UNION. This assumes that no two departments have
  the same name */
SELECT Departments.Name
 FROM Employees INNER JOIN Departments
 ON Department = Code
 GROUP BY Departments.Name
 HAVING COUNT(*) > 2;
```

Output:



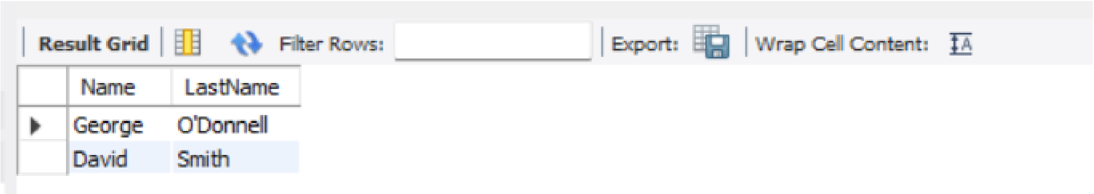
Name
IT
Human Resources

Q.15. Select the name and last name of employees working for departments with second lowest budget.

Code:

```
/* With subquery */
SELECT e.Name, e.LastName
FROM Employees e
WHERE e.Department = (
      SELECT sub.Code
      FROM (SELECT * FROM Departments d ORDER BY d.budget LIMIT 2) sub
      ORDER BY budget DESC LIMIT 1);
```

Output:



Name	LastName
George	O'Donnell
David	Smith

Q.16. Add a new department called "Quality Assurance", with a budget of $40,000 and departmental code 11. Add an employee called "Mary Moore" in that department, with SSN 847-21-9811.

Code:

```
 INSERT INTO Departments

 VALUES ( 11 , 'Quality Assurance' , 40000);

  INSERT INTO employees

 VALUES ( '847219811' , 'Mary' , 'Moore' , 11);
```

Q.17. Reduce the budget of all departments by 10%.

Code: UPDATE Departments SET Budget = Budget * 0.9;

Q.18. Reassign all employees from the Research department (code 77) to the IT department (code 14).

Code: UPDATE Employees SET Department = 14 WHERE Department = 77;

Q.19. Delete from the table all employees in the IT department (code 14)

Code:

```
DELETE

FROM employees

WHERE Department = 14;
```

Q.20. Delete from the table all employees who work in departments with a budget greater than or equal to $60,000.

Code:

```
DELETE FROM Employees

 WHERE Department IN

 (

  SELECT Code FROM Departments

    WHERE Budget >= 60000

 );
```