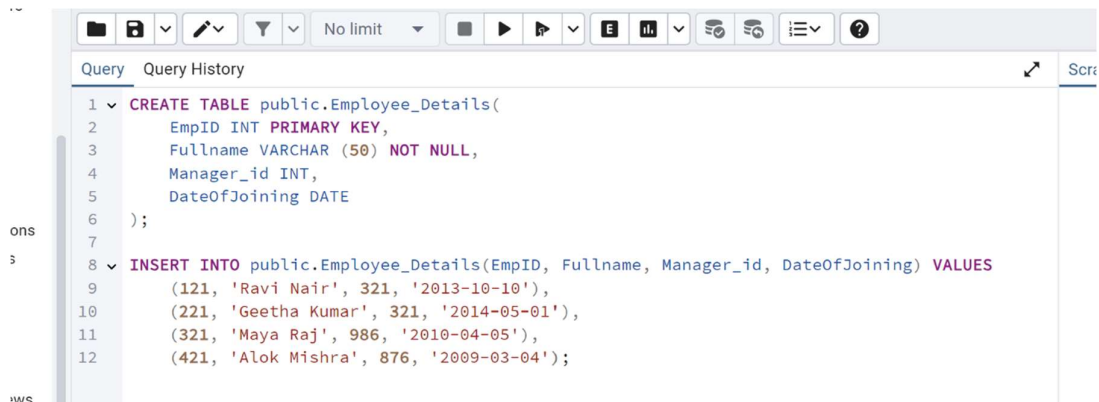


Scientific Computing- Lab 8: DBMS

Write SQL queries based on the table given below

Employee Details			
Emp ID	Fullname	Manager id	Date of joining
121	Ravi Nair	321	10/10/2013
221	Geetha Kumar	321	1/5/2014
321	Maya Raj	986	5/4/2010
421	Alok Mishra	876	4/3/2009

Here we
Create Table
for this value
and then
insert into it:



```
1 CREATE TABLE public.Employee_Details(  
2     EmpID INT PRIMARY KEY,  
3     Fullname VARCHAR (50) NOT NULL,  
4     Manager_id INT,  
5     DateOfJoining DATE  
6 );  
7  
8 INSERT INTO public.Employee_Details(EmpID, Fullname, Manager_id, DateOfJoining) VALUES  
9     (121, 'Ravi Nair', 321, '2013-10-10'),  
10    (221, 'Geetha Kumar', 321, '2014-05-01'),  
11    (321, 'Maya Raj', 986, '2010-04-05'),  
12    (421, 'Alok Mishra', 876, '2009-03-04');
```

Data Output					Messages	Notifications
	empid [PK] integer	fullname character varying (50)	manager_id integer	dateofjoining date		
1	121	Ravi Nair	321	2013-10-10		
2	221	Geetha Kumar	321	2014-05-01		
3	321	Maya Raj	986	2010-04-05		
4	421	Alok Mishra	876	2009-03-04		

Employee Salary

Emp ID	Project	Salary
121	1	50000
221	2	60000
321	1	90000
421	3	55000

Query Query History

```

1 CREATE TABLE public.Employee_Salary(
2     EmpID INT PRIMARY KEY REFERENCES public.Employee_Details(EmpID),
3     Project INT,
4     Salary INT
5 );
6
7 INSERT INTO public.Employee_Salary(EmpID, Project, Salary) VALUES
8     (121, 1, 50000),
9     (221, 2, 60000),
10    (321, 1, 90000),
11    (421, 3, 55000);

```

Data Output Messages Notifications

INSERT 0 4

Query returned successfully in 60 msec.

Data Output Messages Notifications

	empid [PK] integer	project integer	salary integer
1	121	1	50000
2	221	2	60000
3	321	1	90000
4	421	3	55000

- (a) Write a query to fetch employee names and salary records. Return employee details even if the salary record is not present for the employee.

Query Query History

```

1 SELECT e.Fullname, s.Salary
2 FROM public.Employee_Details e
3 LEFT JOIN Employee_Salary s ON e.EmpID = s.EmpID;

```

Data Output Messages Notifications

	fullname character varying (50)	salary integer
1	Ravi Nair	50000
2	Geetha Kumar	60000
3	Maya Raj	90000
4	Alok Mishra	55000

(b) Write a SQL query to fetch all the Employees who are also managers from EmployeeDetails table.

The JOIN condition `e1.EmpID = e2.ManagerID` ensures that we only select employees who are listed as managers for other employees.

`DISTINCT` removes duplicate rows if an employee manages multiple people.

The screenshot shows the PostgreSQL IDE interface. The left sidebar displays the database schema, including the 'public' schema with tables 'employee_details' and 'employee_salary'. The main editor displays the following SQL query:

```
1 SELECT DISTINCT e1.* FROM public.Employee_Details e1
2 JOIN Employee_Details e2 ON e1.EmpID = e2.Manager_id;
```

The 'Data Output' tab shows the result of the query, which is a single row representing an employee who is also a manager:

empid	manager_id	fullname	dateofjoining
1	986	Maya Raj	2010-04-05

(c) Write a SQL query to fetch project wise of count of employees sorted by project's count in descending order.

The screenshot shows the PostgreSQL IDE interface. The left sidebar displays the database schema, including the 'public' schema with tables 'employee_details' and 'employee_salary'. The main editor displays the following SQL query:

```
1 SELECT s.Project, COUNT(s.EmpID) AS EmployeeCount
2 FROM Employee_Salary s
3 GROUP BY s.Project
4 ORDER BY EmployeeCount DESC;
```

The 'Data Output' tab shows the result of the query, which is a table with three rows representing projects and their employee counts:

project	employeecount
1	2
2	1
3	1

(d) Write a SQL query to fetch employee names having salary greater than or equal to 60000 and less than or equal 90000.

The screenshot shows a PostgreSQL query editor interface. On the left is a sidebar with a tree view of the database schema, including 'public' and 'employee_details'. The main area displays a SQL query:

```

1 SELECT e.fullname, s.Salary
2 FROM Employee_Details e
3 JOIN Employee_Salary s ON e.EmpID = s.EmpID
4 WHERE s.Salary BETWEEN 60000 AND 90000;

```

Below the query editor, the 'Data Output' tab is active, showing the results of the query in a table:

	fullname character varying (50)	salary integer
1	Geetha Kumar	60000
2	Maya Raj	90000

Q. 2 Solve the questions for the following table

ID	Country	Official Language(s)	Population (Millions)	GDP (USD Billions)	Founded On
1	United States	English	331	25,462	July 4, 1776
2	China	Mandarin	1,412	18,321	October 1, 1949
3	India	Hindi, English	1,428	3,730	August 15, 1947
4	Japan	Japanese	125	4,231	February 11, 660 BCE
5	Germany	German	84	4,305	January 18, 1871
6	Brazil	Portuguese	214	2,080	September 7, 1822
7	Russia	Russian	143	2,064	June 12, 1990
8	United Kingdom	English	67	3,691	July 12, 927

9	France	French	67	3,000	September 22, 1792
10	Canada	English, French	38	2,139	July 1, 1867

Publications
Schemas (1)
public
Aggregates
Collations
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (3)
countriesgdp
employee_details
employee_salary
Trigger Functions
Types
Views
Subscriptions
ab-9

Query
Query History

```

1 CREATE TABLE public.CountriesGDP (
2   ID SERIAL PRIMARY KEY,
3   Country VARCHAR(50) NOT NULL,
4   OfficialLanguage VARCHAR(100),
5   Population_Millions INT,
6   GDP_USD_billions NUMERIC(12, 2), --precision and scale after decimal points
7   FoundedOn DATE
8 );
9
10 INSERT INTO public.CountriesGDP (ID, Country, OfficialLanguage, Population_Millions, GDP_USD_billions, FoundedOn)
11 VALUES
12 (1, 'United States', 'English', 331, 25462, '1776-07-04'),
13 (2, 'China', 'Mandarin', 1412, 18321, '1949-10-01'),
14 (3, 'India', 'Hindi, English', 1428, 3730, '1947-08-15'),
15 (4, 'Japan', 'Japanese', 125, 4231, '0660-02-11'),
16 (5, 'Germany', 'German', 84, 4305, '1871-01-18'),
17 (6, 'Brazil', 'Portuguese', 214, 2080, '1822-09-07'),
18 (7, 'Russia', 'Russian', 143, 2064, '1990-06-12'),
19 (8, 'United Kingdom', 'English', 67, 3691, '0927-07-12'),
20 (9, 'France', 'French', 67, 3000, '1792-09-22'),
21 (10, 'Canada', 'English, French', 38, 2139, '1867-07-01');
22

```

Data Output
Messages
Notifications

	id [PK] integer	country character varying (50)	officiallanguage character varying (100)	population_millions integer	gdp_usd_billions numeric (12,2)	foundedon date
1	1	United States	English	331	25462.00	1776-07-04
2	2	China	Mandarin	1412	18321.00	1949-10-01
3	3	India	Hindi, English	1428	3730.00	1947-08-15
4	4	Japan	Japanese	125	4231.00	0660-02-11
5	5	Germany	German	84	4305.00	1871-01-18
6	6	Brazil	Portuguese	214	2080.00	1822-09-07
7	7	Russia	Russian	143	2064.00	1990-06-12
8	8	United Kingdom	English	67	3691.00	0927-07-12
9	9	France	French	67	3000.00	1792-09-22
10	10	Canada	English, French	38	2139.00	1867-07-01

Questions:

1. Write a query to fetch country names and their GDP records. Return country details even if the GDP record is not present for the country

The screenshot shows the PostgreSQL GUI interface. On the left, the 'Schemas (1)' tree is expanded to 'public', and 'Tables (3)' is selected, with 'countriesgdp' highlighted. The main query editor contains the following SQL:

```
1 SELECT c.Country, c.GDP_USD_Billions
2 FROM CountriesGDP c
```

The 'Data Output' tab is active, displaying the results of the query in a table:

	country character varying (50)	gdp_usd_billions numeric (12,2)
1	United States	25462.00
2	China	18321.00
3	India	3730.00
4	Japan	4231.00
5	Germany	4305.00
6	Brazil	2080.00
7	Russia	2064.00
8	United Kingdom	3691.00
9	France	3000.00
10	Canada	2139.00

- Write a SQL query to fetch all the countries that have an official language of "English" from the Countries table.

The screenshot shows the PostgreSQL GUI interface. On the left, the 'Schemas (1)' tree is expanded to 'public', and 'Tables (3)' is selected, with 'countriesgdp' highlighted. The main query editor contains the following SQL:

```
1 SELECT c.Country
2 FROM CountriesGDP c
3 WHERE OfficialLanguage LIKE '%English%'
4
```

The 'Data Output' tab is active, displaying the results of the query in a table:

	country character varying (50)
1	United States
2	India
3	United Kingdom
4	Canada

- Write a SQL query to fetch country names that have a population greater than or equal to 100 million and less than or equal to 500 million.

languages
publications
schemas (1)
public
Aggregates
Collations
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (3)
countriesgdp
employee_details
employee_salary
Trigger Functions
Types
Views

Query Query History

```

1 SELECT c.Country, c.Population_Millions
2 FROM public.CountriesGDP c
3 WHERE c.Population_Millions >= 100 AND c.Population_Millions <=500
4

```

Data Output Messages Notifications

	country character varying (50)	population_millions integer
1	United States	331
2	Japan	125
3	Brazil	214
4	Russia	143

4. Write a query to list the countries founded after the year 1900.

foreign data wrappers
languages
publications
schemas (1)
public
Aggregates
Collations
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (3)
countriesgdp
employee_details
employee_salary
Trigger Functions
Types
Views
subscriptions
-9

Query Query History

```

1 SELECT c.Country
2 FROM public.CountriesGDP c
3 WHERE c.FoundedON > '1900-01-01';
4

```

Data Output Messages Notifications

	country character varying (50)
1	China
2	India
3	Russia

5. Write a SQL query to fetch the country names and populations in ascending order of population.

The screenshot shows the PostgreSQL IDE interface. On the left, the 'Schemas (1)' pane is expanded to 'public', and 'Tables (3)' is selected, with 'countriesgdp' highlighted. The main query editor contains the following SQL query:

```
1 SELECT c.Country, c.Population_Millions
2 FROM CountriesGDP c
3 ORDER BY c.Population_Millions ASC;
4
```

The 'Data Output' pane at the bottom displays the results of the query in a table format:

	country	population_millions
	character varying (50)	integer
1	Canada	38
2	France	67
3	United Kingdom	67
4	Germany	84
5	Japan	125
6	Russia	143
7	Brazil	214
8	United States	331
9	China	1412
10	India	1428

6. Write a SQL query to fetch all the countries whose official language is not "English" from the Countries table.

The screenshot shows the PostgreSQL IDE interface. On the left, the 'Schemas (1)' pane is expanded to 'public', and 'Tables (3)' is selected, with 'countriesgdp' highlighted. The main query editor contains the following SQL query:

```
1 SELECT c.Country, c.OfficialLanguage
2 FROM CountriesGDP c
3 WHERE c.OfficialLanguage NOT LIKE '%English%';
```

The 'Data Output' pane at the bottom displays the results of the query in a table format:

	country	officiallanguage
	character varying (50)	character varying (100)
1	China	Mandarin
2	Japan	Japanese
3	Germany	German
4	Brazil	Portuguese
5	Russia	Russian
6	France	French