

- 1) Select employee details of dept number 10 or 30 .

```
SELECT "EmpNo", "Ename", "Sal", "Hire_Date", "Commission", "Mgr"
FROM public."Emp_Table"
WHERE "DeptNo" = 10 OR "DeptNo" = 30 ;
```

postgres/postgres@PostgreSQL 13 ▾

Query Editor Query History

```
1 SELECT "EmpNo", "Ename", "Sal", "Hire_Date", "Commission", "Mgr"
2 FROM public."Emp_Table"
3 WHERE "DeptNo" = 10 OR "DeptNo" = 30 ;
4
```

Data Output Explain Messages Notifications

	EmpNo [PK] integer	Ename character varying (250)	Sal integer	Hire_Date date	Commission integer	Mgr integer
1	1002	Kapil	15000	1970-01-01	2300	1003
2	1006	David	19000	1985-01-01	2400	1007
3	1004	Williams	9000	2001-01-01	0	1007
4	1005	John	5000	2005-01-01	0	1006

- 2) Write a query to fetch all the dept details with more than 1 Employee.

```
SELECT "Dname" , "Loc"
FROM public."Emp_Table" as e , public."Dept_Table" as d
WHERE e."DeptNo" = d."DeptNo"
GROUP BY d."Dname" , d."Loc"
HAVING COUNT("EmpNo")>1;
```

postgres/postgres@PostgreSQL 13 ▾

Query Editor Query History

```
1 SELECT "Dname" , "Loc"
2 FROM public."Emp_Table" as e , public."Dept_Table" as d
3 WHERE e."DeptNo" = d."DeptNo"
4 GROUP BY d."Dname" , d."Loc"
5 HAVING COUNT("EmpNo")>1;
6
```

Data Output Explain Messages Notifications

	Dname character varying (255)	Loc character varying (255)
1	Accounts	Bangalore
2	IT	Delhi
3	Production	Chennai

- 3) Write a query to fetch employee details whose name starts with the letter "S"

**SELECT \* FROM public."Emp\_Table" WHERE "Ename" LIKE 'S%';**

postgres/postgres@PostgreSQL 13

Query Editor Query History

```
1 SELECT * FROM public."Emp_Table" WHERE "Ename" LIKE 'S%';
```

Data Output Explain Messages Notifications

	EmpNo [PK] integer	Ename character varying (250)	Sal integer	Hire_Date date	Commission integer	DeptNo integer	Mgr integer
1	1001	Sachin	19000	1980-01-01	2100	20	1003
2	1003	Stefen	12000	1990-01-01	500	20	1007

- 4) Select Emp Details Whose experience is more than 2 years.

**SELECT "EmpNo", "Ename", "Sal", "Commission", "DeptNo", "Mgr"  
FROM public."Emp\_Table"  
WHERE ((current\_date - "Hire\_Date")/365) > 2;**

Query Editor Query History

```
1 SELECT "EmpNo", "Ename", "Sal", "Commission", "DeptNo", "Mgr"
2 FROM public."Emp_Table"
3 WHERE ((current_date - "Hire_Date")/365) > 2;
4
```

Data Output Explain Messages Notifications

	EmpNo [PK] integer	Ename character varying (250)	Sal integer	Commission integer	DeptNo integer	Mgr integer
1	1001	Sachin	19000	2100	20	1003
2	1002	Kapil	15000	2300	10	1003
3	1003	Stefen	12000	500	20	1007
4	1006	Dravid	19000	2400	10	1007
5	1007	Martin	21000	1040	[null]	[null]
6	1004	Williams	9000	0	30	1007
7	1005	John	5000	0	30	1006

- 5) Write a SELECT statement to replace the char "a" with "#" in Employee Name ( Ex: **Sachin** as **S#chin**)

**SELECT replace("Ename", 'a', '#') FROM public."Emp\_Table";**

Query Editor		Query History
<pre>1 SELECT replace("Ename", 'a', '#') FROM public."Emp_Table";</pre>		
Data Output		Explain Messages Notifications
	replace text	
1	S#chin	
2	K#pil	
3	Stefen	
4	Dr#vid	
5	M#rtin	
6	Willi#ms	
7	John	

- 6) Write a query to fetch employee name and his/her manager name.

**SELECT e."Ename" , f."Ename" as "Mgr"**  
**FROM public."Emp\_Table" as e,public."Emp\_Table" as f**  
**WHERE e."Mgr" = f."EmpNo";**

Query Editor		Query History
<pre>1 SELECT e."Ename" , f."Ename" as "Mgr" 2 FROM public."Emp_Table" as e,public."Emp_Table" as f 3 WHERE e."Mgr" = f."EmpNo"; 4</pre>		
Data Output		Explain Messages Notifications
	Ename character varying (250)	Mgr character varying (250)
1	Sachin	Stefen
2	Kapil	Stefen
3	Stefen	Martin
4	David	Martin
5	Williams	Martin
6	John	David

7) Fetch Dept Name, Total Salary of the Dept.

```
SELECT "Dname", SUM("Sal") as "Total Sal"
FROM public."Dept_Table" as d, public."Emp_Table" as e
WHERE d."DeptNo" = e."DeptNo"
GROUP BY "Dname";
```

The screenshot shows the DBeaver SQL editor interface. The left sidebar displays the database structure for PostgreSQL 13, with the 'public' schema selected. The main query editor contains the following SQL query:

```
1 SELECT "Dname", SUM("Sal") as "Total Sal"
2 FROM public."Dept_Table" as d, public."Emp_Table" as e
3 WHERE d."DeptNo" = e."DeptNo"
4 GROUP BY "Dname";
```

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

	Dname character varying (255)	Total Sal bigint
1	Accounts	34000
2	IT	31000
3	Production	14000

8) Write a query to fetch ALL the employee details along with department name, department location, irrespective of employee existence in the department.

```
SELECT "EmpNo", "Ename", "Sal", "Hire_Date", "Commission", "Mgr", "Dname", "Loc"
FROM public."Emp_Table" as e, public."Dept_Table" as d
WHERE e."DeptNo" = d."DeptNo";
```

The screenshot shows the DBeaver SQL editor interface. The left sidebar displays the database structure for PostgreSQL 13, with the 'public' schema selected. The main query editor contains the following SQL query:

```
1 SELECT "EmpNo", "Ename", "Sal", "Hire_Date", "Commission", "Mgr", "Dname", "Loc"
2 FROM public."Emp_Table" as e, public."Dept_Table" as d
3 WHERE e."DeptNo" = d."DeptNo";
```

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

	EmpNo integer	Ename character varying (250)	Sal integer	Hire_Date date	Commission integer	Mgr integer	Dname character varying (255)	Loc character varying (255)
1	1006	David	19000	1985-01-01	2400	1007	Accounts	Bangalore
2	1002	Kapil	15000	1970-01-01	2300	1003	Accounts	Bangalore
3	1003	Stefen	12000	1990-01-01	500	1007	IT	Delhi
4	1001	Sachin	19000	1980-01-01	2100	1003	IT	Delhi
5	1005	John	5000	2005-01-01	0	1006	Production	Chennai
6	1004	Williams	9000	2001-01-01	0	1007	Production	Chennai

9) Write an update statement to increase the employee salary by 10 %.

```
UPDATE public."Emp_Table"  
SET "Sal"= "Sal" * 1.1 ;
```

The screenshot shows the PostgreSQL Query Editor interface. The top bar indicates the connection to 'postgres/postgres@PostgreSQL 13'. The 'Query Editor' tab is active, displaying the following SQL query:

```
1 UPDATE public."Emp_Table"
2 SET "Sal"= "Sal" * 1.1 ;
3
```

Below the query editor, the 'Messages' tab shows the execution status: 'UPDATE 7' and 'Query returned successfully in 120 msec.'

The 'Data Output' tab displays the result of the query, which is a table with 7 rows and 9 columns. The columns are: EmpNo (integer), Ename (character varying (250)), Sal (integer), Hire\_Date (date), Commission (integer), DeptNo (integer), Mgr (integer), and two additional columns for Mgr (integer) and an unnamed column (integer). The data is as follows:

EmpNo	Ename	Sal	Hire_Date	Commission	DeptNo	Mgr	Mgr	
1	1001 Sachin	20900	1980-01-01	2100	20	1003		
2	1002 Kapil	16500	1970-01-01	2300	10	1003		
3	1003 Stefan	13200	1990-01-01	500	20	1007		
4	1006 David	20900	1985-01-01	2400	10	1007		
5	1007 Martin	23100	2000-01-01	1040	[null]	[null]		
6	1004 Williams	9900	2001-01-01	0	30	1007		
7	1005 John	5500	2005-01-01	0	30	1006		

10) Write a statement to delete employees belong to Chennai location.

```
DELETE FROM public."Emp_Table"  
WHERE "DeptNo" IN (SELECT "DeptNo"  
FROM public."Dept_Table"  
WHERE "Loc" = 'Chennai');
```

The screenshot shows the PostgreSQL Query Editor interface. The top bar indicates the connection to 'postgres/postgres@PostgreSQL 13'. The 'Query Editor' tab is active, displaying the following SQL query:

```
1 DELETE FROM public."Emp_Table"
2 WHERE "DeptNo" IN (SELECT "DeptNo"
3 FROM public."Dept_Table"
4 WHERE "Loc" = 'Chennai');
5
```

Below the query editor, the 'Messages' tab shows the execution status: 'DELETE 2' and 'Query returned successfully in 164 msec.'

```
1 SELECT * FROM public."Dept_Table";
```

	DeptNo [PK] integer	Dname character varying (255)	Loc character varying (255)
1	10	Accounts	Bangalore
2	20	IT	Delhi
3	30	Production	Chennai
4	40	Sales	Hyderabad
5	50	Adminstration	London

11) Get Employee Name and gross salary (sal + commission).

We have to update column to replace null by 0:

**UPDATE public."Emp\_Table"**

**SET "Commission" = 0**

**WHERE "Commission" IS null;**

postgres/postgres@PostgreSQL 13

Query Editor Query History

```
1 UPDATE public."Emp_Table"
2 SET "Commission" = 0
3 WHERE "Commission" IS null;
```

Data Output Explain Messages Notifications

UPDATE 2

Query returned successfully in 117 msec.

**SELECT "Ename", "Sal" + "Commission" as "Gross Salary"**  
**FROM public."Emp\_Table";**

Query Editor Query History

```
1 SELECT "Ename", "Sal" + "Commission" as "Gross Salary"
2 FROM public."Emp_Table";
3
```

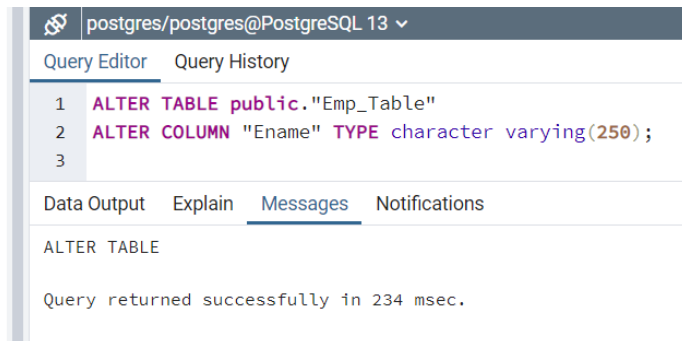
Data Output Explain Messages Notifications

	Ename character varying (250)	Gross Salary integer
1	Sachin	21100
2	Kapil	17300
3	Stefen	12500
4	David	21400
5	Martin	22040
6	Williams	9000
7	John	5000

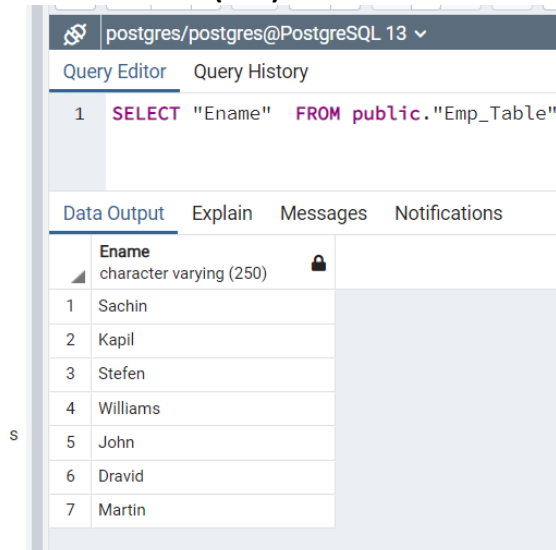
12) Increase the data length of the column Ename of Emp table from 100 to 250 using ALTER statement

**ALTER TABLE public."Emp\_Table"**

**ALTER COLUMN "Ename" TYPE character varying(250);**

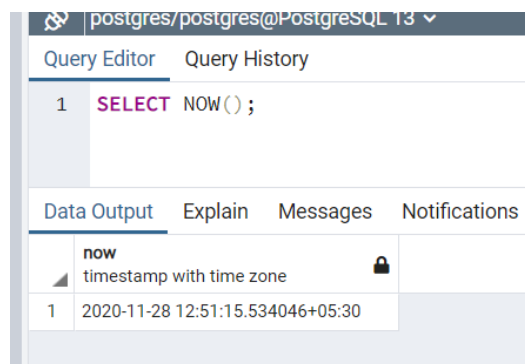


**Ename Varchar(250)**



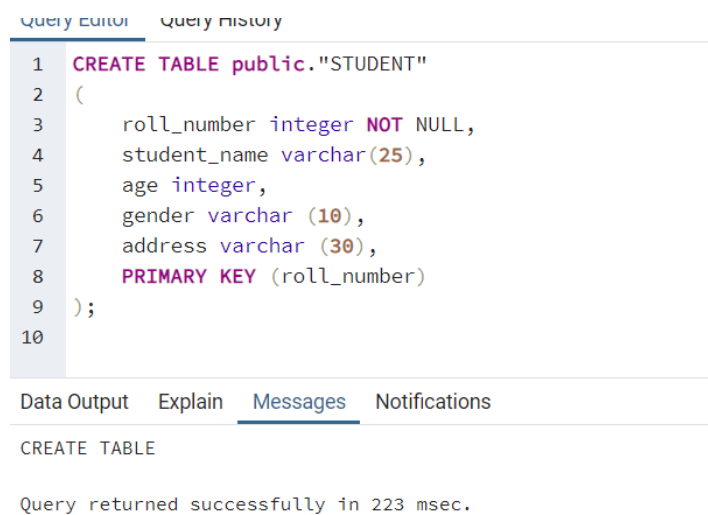
13) Write query to get current datetime.

**SELECT NOW();**



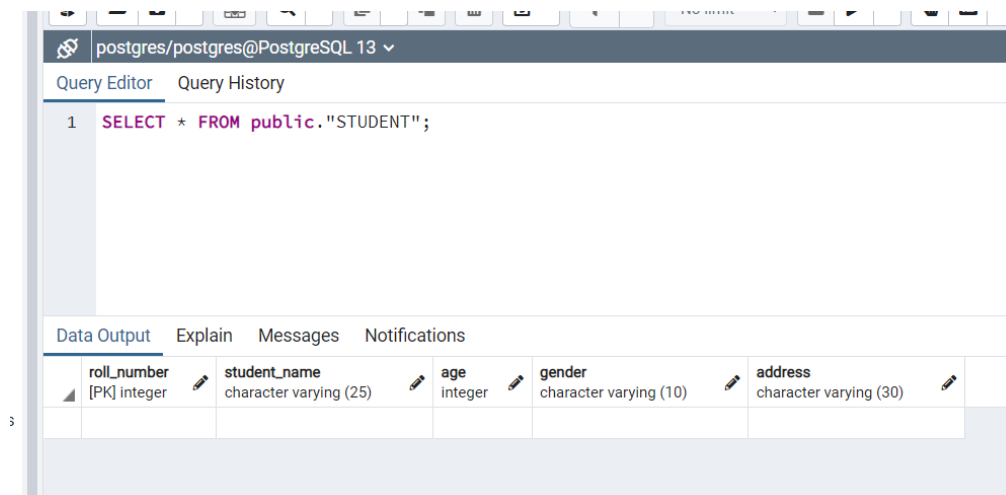
14) Write a statement to create STUDENT table, with related 5 columns

```
CREATE TABLE public."STUDENT"  
(  
    roll_number integer NOT NULL,  
    student_name varchar(25),  
    age integer,  
    gender varchar (10),  
    address varchar (30),  
    PRIMARY KEY (roll_number)  
);
```



The screenshot shows a query editor with the following tabs: Query Editor, Query History, Data Output, Explain, Messages, and Notifications. The Messages tab is selected, displaying the message: "Query returned successfully in 223 msec." The query editor contains the following SQL code:

```
1 CREATE TABLE public."STUDENT"  
2 (  
3     roll_number integer NOT NULL,  
4     student_name varchar(25),  
5     age integer,  
6     gender varchar (10),  
7     address varchar (30),  
8     PRIMARY KEY (roll_number)  
9 );  
10
```



The screenshot shows a query editor with the following tabs: Query Editor, Query History, Data Output, Explain, Messages, and Notifications. The Messages tab is selected, displaying the message: "Query returned successfully in 223 msec." The query editor contains the following SQL code:

```
1 SELECT * FROM public."STUDENT";
```

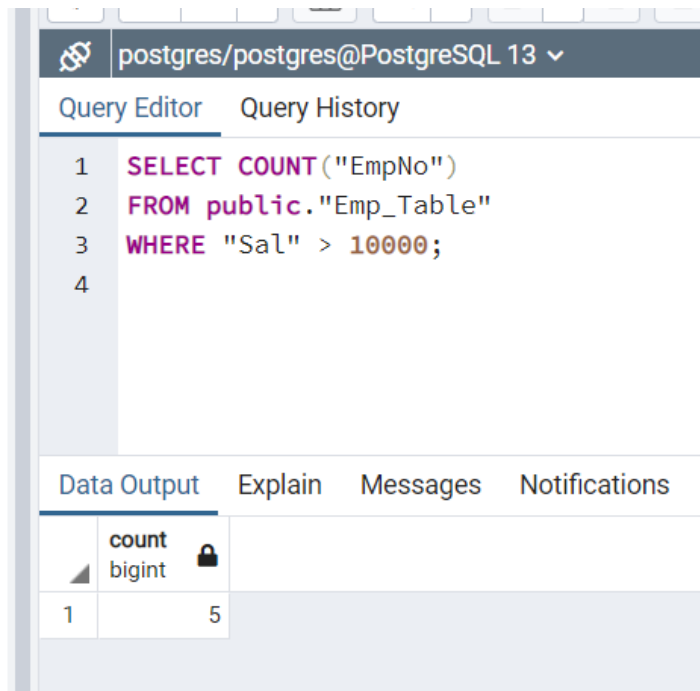
The Data Output tab is selected, displaying the following table:

roll_number	student_name	age	gender	address
[PK] integer	character varying (25)	integer	character varying (10)	character varying (30)



15) Write a query to fetch number of employees in who is getting salary more than 10000.

```
SELECT COUNT("EmpNo")  
FROM public."Emp_Table"  
WHERE "Sal" > 10000;
```



The screenshot shows the PostgreSQL Query Editor interface. The query editor contains the following SQL query:

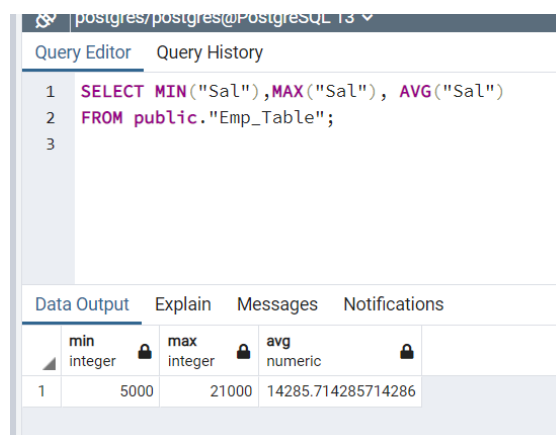
```
1 SELECT COUNT("EmpNo")  
2 FROM public."Emp_Table"  
3 WHERE "Sal" > 10000;  
4
```

Below the query editor, the "Data Output" tab is selected, showing the result of the query. The result is a single row with the following data:

	count
1	5

16) Write a query to fetch minimum salary, maximum salary and average salary from emp table.

```
SELECT MIN("Sal"), MAX("Sal"), AVG("Sal")  
FROM public."Emp_Table";
```

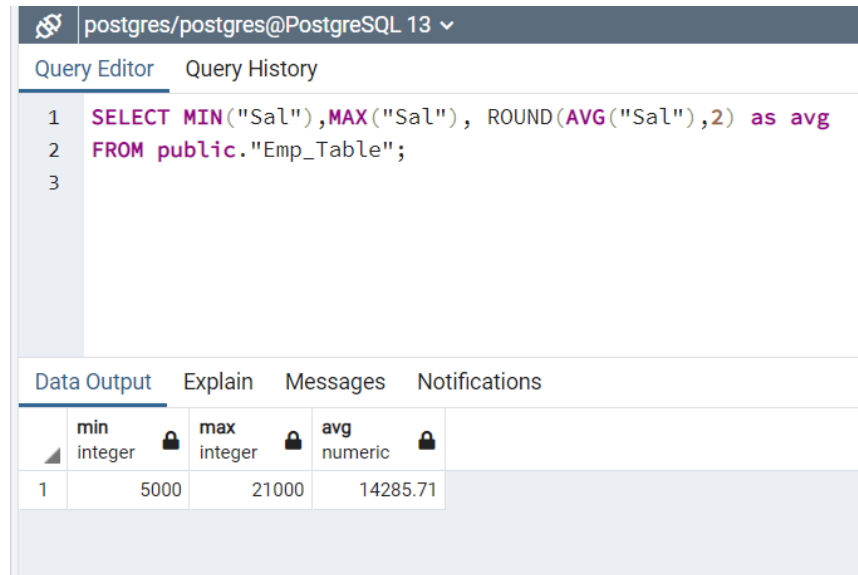


The screenshot shows the PostgreSQL Query Editor interface. The query editor contains the following SQL query:

```
1 SELECT MIN("Sal"), MAX("Sal"), AVG("Sal")  
2 FROM public."Emp_Table";  
3
```

Below the query editor, the "Data Output" tab is selected, showing the result of the query. The result is a single row with the following data:

	min	max	avg
1	5000	21000	14285.714285714286

**AVG in In 2 Decimal Points**


The screenshot shows the PostgreSQL Query Editor interface. The query editor contains the following SQL query:

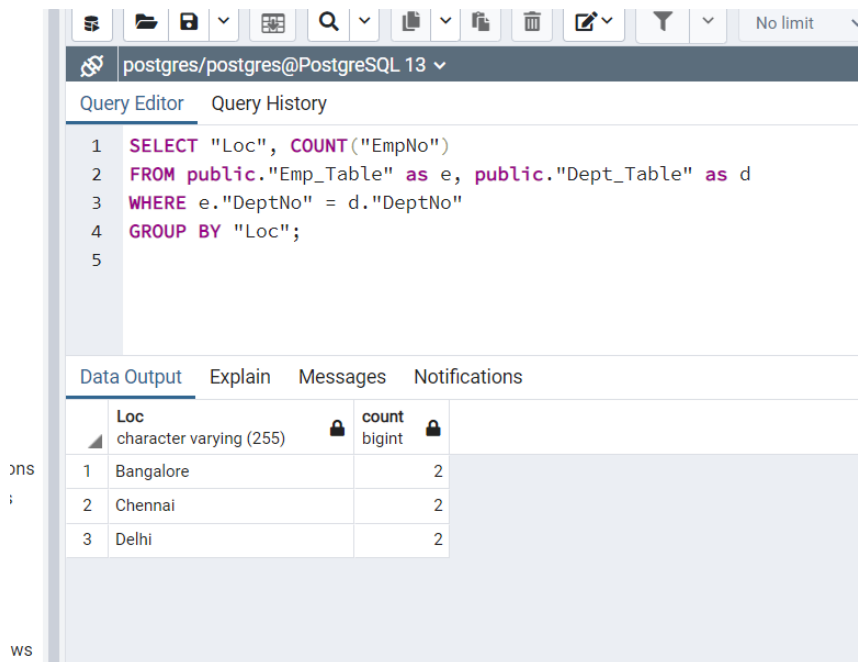
```
1 SELECT MIN("Sal"), MAX("Sal"), ROUND(AVG("Sal"), 2) as avg
2 FROM public."Emp_Table";
3
```

The query is executed, and the results are displayed in the Data Output tab. The results show the minimum salary (5000), maximum salary (21000), and the average salary rounded to 2 decimal points (14285.71).

	min integer	max integer	avg numeric
1	5000	21000	14285.71

17) Write a query to fetch number of employees in each location.

```
SELECT "Loc", COUNT("EmpNo")
FROM public."Emp_Table" as e, public."Dept_Table" as d
WHERE e."DeptNo" = d."DeptNo"
GROUP BY "Loc";
```



The screenshot shows the PostgreSQL Query Editor interface. The query editor contains the following SQL query:

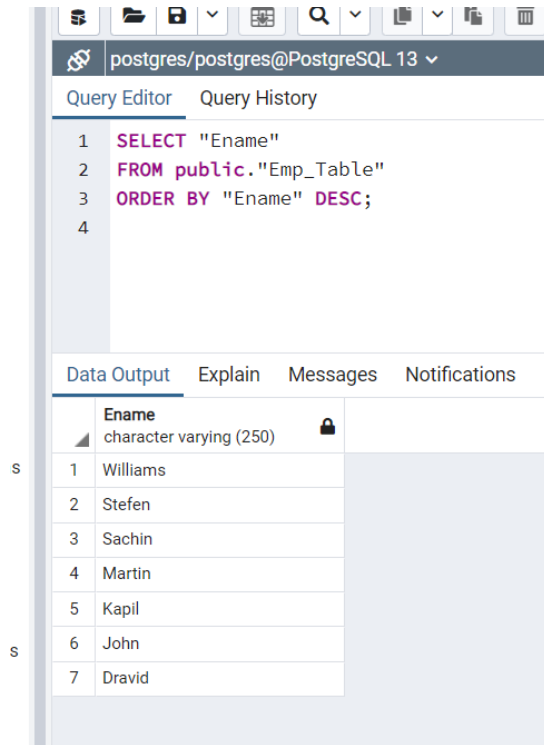
```
1 SELECT "Loc", COUNT("EmpNo")
2 FROM public."Emp_Table" as e, public."Dept_Table" as d
3 WHERE e."DeptNo" = d."DeptNo"
4 GROUP BY "Loc";
5
```

The query is executed, and the results are displayed in the Data Output tab. The results show the location (Loc) and the count of employees (count) for each location.

	Loc character varying (255)	count bigint
1	Bangalore	2
2	Chennai	2
3	Delhi	2

18) Write a query to display employee names in descending order.

```
SELECT "Ename"  
FROM public."Emp_Table"  
ORDER BY "Ename" DESC;
```



The screenshot shows the PostgreSQL Query Editor interface. The query editor displays the following SQL query:

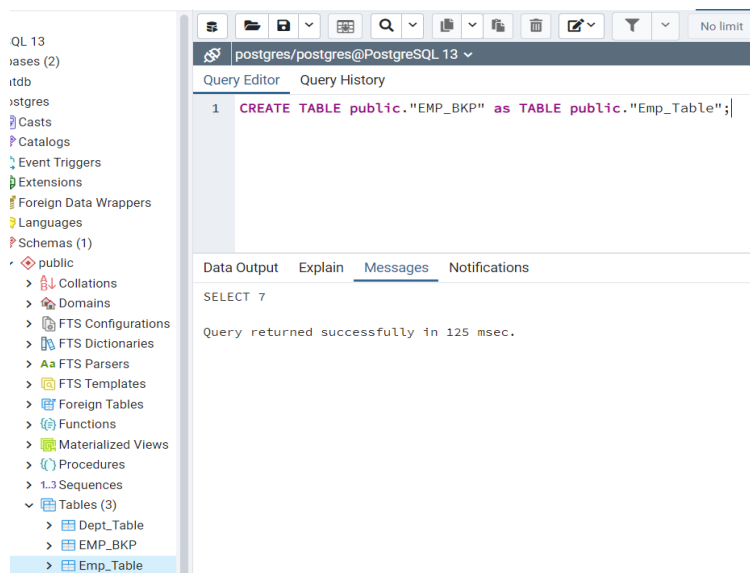
```
1 SELECT "Ename"
2 FROM public."Emp_Table"
3 ORDER BY "Ename" DESC;
4
```

The results are shown in the Data Output tab, displaying a table with 7 rows and 1 column (Ename). The results are ordered in descending order of Ename.

	Ename
1	Williams
2	Stefen
3	Sachin
4	Martin
5	Kapil
6	John
7	Dravid

19) Write a statement to create a new table (EMP\_BKP) from the existing EMP table.

```
CREATE TABLE public."EMP_BKP" as TABLE public."Emp_Table";
```



The screenshot shows the PostgreSQL Query Editor interface. The query editor displays the following SQL query:

```
1 CREATE TABLE public."EMP_BKP" as TABLE public."Emp_Table";
```

The results are shown in the Messages tab, displaying a message indicating that the query was executed successfully.

Query returned successfully in 125 msec.

## VARUN VARMA R

es

its

alogs

rt Triggers

ensions

ign Data Wrappers

iguages

chemas (1)

public

Collations

Domains

FTS Configurations

FTS Dictionaries

Aa FTS Parsers

FTS Templates

Foreign Tables

Functions

Materialized Views

Procedures

Sequences

Tables (3)

Dept\_Table

EMP\_BKP

Emp\_Table

Trigger Functions

Query Editor

Query History

1

SELECT \* FROM public."EMP\_BKP";

Data Output

Explain

Messages

Notifications

	EmpNo integer	ENAME character varying (250)	SAL integer	Hire_Date date	Commission integer	DeptNo integer	Mgr integer
1	1001	Sachin	19000	1980-01-01	2100	20	1003
2	1002	Kapil	15000	1970-01-01	2300	10	1003
3	1003	Stefen	12000	1990-01-01	500	20	1007
4	1004	Williams	9000	2001-01-01	[null]	30	1007
5	1005	John	5000	2005-01-01	[null]	30	1006
6	1006	Dravid	19000	1985-01-01	2400	10	1007
7	1007	Martin	21000	2000-01-01	1040	[null]	[null]

20) Write a query to fetch first 3 characters from employee name appended with salary.

**SELECT LEFT("ENAME",3) || "SAL" FROM public."EMP\_Table";**

Query Editor Query History

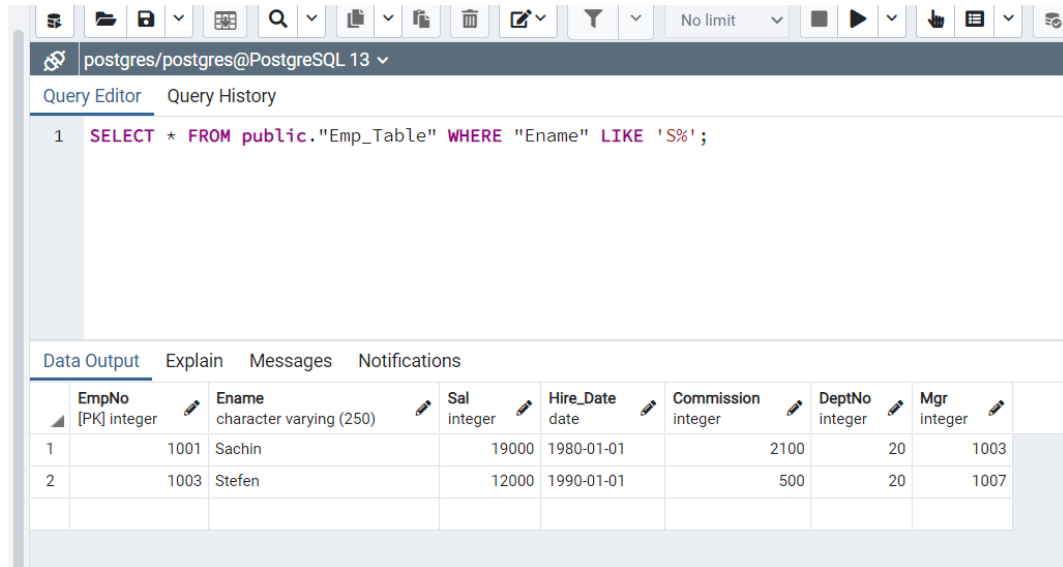
```
1 SELECT LEFT("ENAME",3) || "SAL" FROM public."EMP_Table";
```

Data Output Explain Messages Notifications

	?column? text
1	Sac19000
2	Kap15000
3	Ste12000
4	Wil9000
5	Joh5000
6	Dra19000
7	Mar21000

21) Get the details of the employees whose name starts with S.

**SELECT \* FROM public."Emp\_Table" WHERE "Ename" LIKE 'S%';**



The screenshot shows the PostgreSQL Query Editor interface. The query editor contains the following SQL query:

```
1 SELECT * FROM public."Emp_Table" WHERE "Ename" LIKE 'S%';
```

The results are displayed in the Data Output tab, showing a table with the following columns and data:

	EmpNo [PK] integer	Ename character varying (250)	Sal integer	Hire_Date date	Commission integer	DeptNo integer	Mgr integer
1	1001	Sachin	19000	1980-01-01	2100	20	1003
2	1003	Stefen	12000	1990-01-01	500	20	1007

22) Get the details of the employees who works in Bangalore location.

**SELECT \* FROM public."Emp\_Table"  
WHERE "DeptNo" IN (SELECT "DeptNo"  
FROM public."Dept\_Table"  
WHERE "Loc" = 'Bangalore');**



The screenshot shows the PostgreSQL Query Editor interface. The query editor contains the following SQL query:

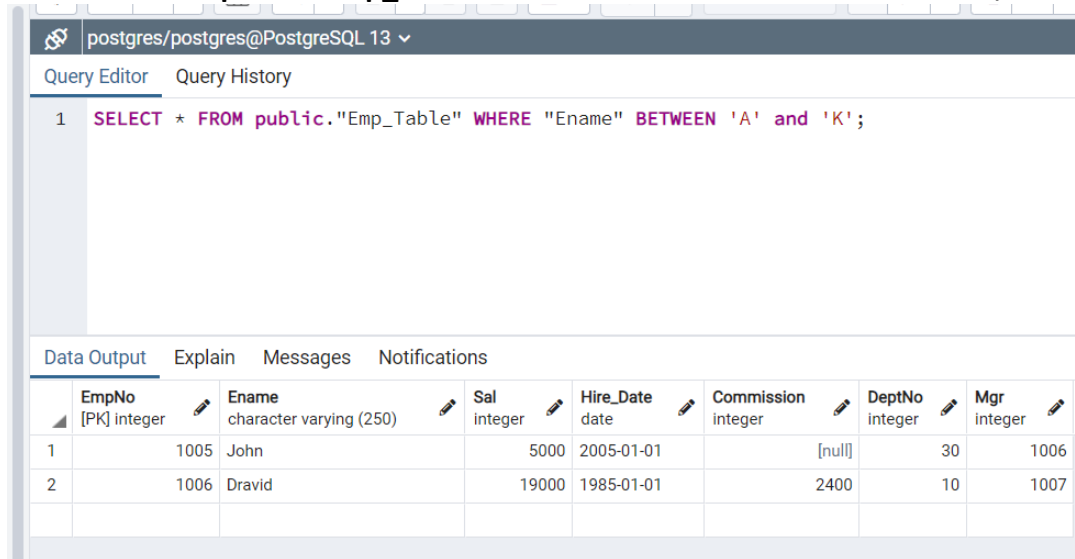
```
1 SELECT * FROM public."Emp_Table"  
2 WHERE "DeptNo" IN (SELECT "DeptNo"  
3 FROM public."Dept_Table"  
4 WHERE "Loc" = 'Bangalore');  
5
```

The results are displayed in the Data Output tab, showing a table with the following columns and data:

	EmpNo [PK] integer	Ename character varying (250)	Sal integer	Hire_Date date	Commission integer	DeptNo integer	Mgr integer
1	1002	Kapil	15000	1970-01-01	2300	10	1003
2	1006	Dravid	19000	1985-01-01	2400	10	1007

23) Write the query to get the employee details whose name started within any letter between A and K.

**SELECT \* FROM public."Emp\_Table" WHERE "Ename" BETWEEN 'A' and 'K';**

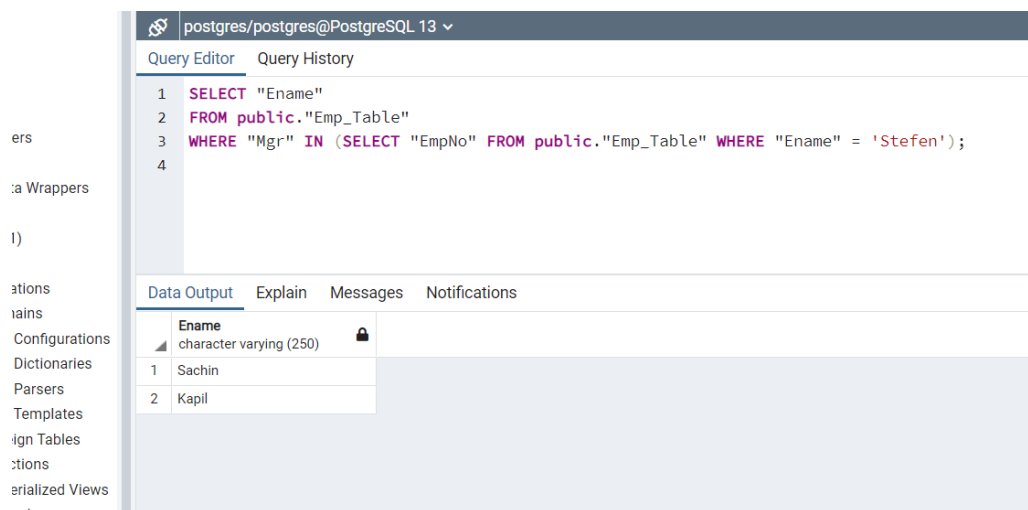


The screenshot shows a PostgreSQL query editor with the query: `SELECT * FROM public."Emp_Table" WHERE "Ename" BETWEEN 'A' and 'K';`. The results are displayed in a table with columns: EmpNo [PK] integer, Ename character varying (250), Sal integer, Hire\_Date date, Commission integer, DeptNo integer, and Mgr integer. The results show two rows: John (EmpNo 1005, Sal 5000, Hire\_Date 2005-01-01, Commission [null], DeptNo 30, Mgr 1006) and David (EmpNo 1006, Sal 19000, Hire\_Date 1985-01-01, Commission 2400, DeptNo 10, Mgr 1007).

	EmpNo [PK] integer	Ename character varying (250)	Sal integer	Hire_Date date	Commission integer	DeptNo integer	Mgr integer
1	1005	John	5000	2005-01-01	[null]	30	1006
2	1006	David	19000	1985-01-01	2400	10	1007

24) Write a query in SQL to display the employees whose manager name is **Stefen**.

**SELECT "Ename"**  
**FROM public."Emp\_Table"**  
**WHERE "Mgr" IN (SELECT "EmpNo" FROM public."Emp\_Table" WHERE "Ename" = 'Stefen');**



The screenshot shows a PostgreSQL query editor with the query: `SELECT "Ename" FROM public."Emp_Table" WHERE "Mgr" IN (SELECT "EmpNo" FROM public."Emp_Table" WHERE "Ename" = 'Stefen');`. The results are displayed in a table with columns: Ename character varying (250). The results show two rows: Sachin and Kapil.

Ename character varying (250)
Sachin
Kapil

25) Write a query in SQL to list the name of the managers who is having maximum number of employees working under him.

```
SELECT f."Ename" as Mngr, COUNT(e."Ename") as Max_Emp
FROM public."Emp_Table" as e, public."Emp_Table" as f
WHERE e."Mgr" = f."EmpNo"
GROUP BY Mngr
ORDER BY Max_Emp DESC
LIMIT 1;
```

The screenshot shows a PostgreSQL query editor interface. The query editor tab is active, displaying the following SQL query:

```
1 SELECT f."Ename" as Mngr, COUNT(e."Ename") as Max_Emp
2 FROM public."Emp_Table" as e, public."Emp_Table" as f
3 WHERE e."Mgr" = f."EmpNo"
4 GROUP BY Mngr
5 ORDER BY Max_Emp DESC
6 LIMIT 1;
```

Below the query editor, the "Data Output" tab is active, showing the results of the query. The results are displayed in a table with two columns: "mngr" (character varying (250)) and "max\_emp" (bigint). The table contains one row of data:

	mngr	max_emp
1	Martin	3