



INSTITUTO TECNOLÓGICO SUPERIOR DE JEREZ

INGENIERÍA EN SISTEMAS COMPUTACIONALES

5to Semestre

Fecha de entrega: 14/10/2020

Actividad 2: Ejercicios SQL [Consultas con funciones de agregación]

Tema 2: Lenguaje de manipulación de datos.

Materia: Taller de Base de Datos.

Nombre del Alumno: Marín Ramírez Mario.

Número de Control: S18070186

Correo electrónico: mariomarin502t@gmail.com

Profesor: I.S.C. Salvador Acevedo Sandoval.

Consultas generadas en MySQL

1. Mostrar el salario del empleado que gana más.

```

C:\WINDOWS\system32\CMD.exe - mysql -u root -p
Microsoft Windows [Versión 10.0.18363.1082]
(c) 2019 Microsoft Corporation. Todos los derechos reservados.

C:\Users\marin>mysql -u root -p
Enter password: *****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.19 MySQL Community Server - GPL

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use dreamhome;
Database changed
mysql> SELECT MAX(salary) as Salario_Maximo FROM Staff;
+-----+
| Salario_Maximo |
+-----+
|          30000 |
+-----+
1 row in set (0.00 sec)

```

2. Mostrar el salario del empleado que gana menos.

```

mysql> SELECT MIN(salary) as Salario_Minimo FROM Staff;
+-----+
| Salario_Minimo |
+-----+
|           9000 |
+-----+
1 row in set (0.00 sec)

```

3. Muestre cual es el promedio del salario que perciben los trabajadores.

```

mysql> SELECT AVG(salary) as Salario_Promedio FROM Staff;
+-----+
| Salario_Promedio |
+-----+
|    17000.0000 |
+-----+
1 row in set (0.01 sec)

```

4. Crear una consulta que muestre la cantidad que gasta la empresa en salarios.

```

mysql> SELECT SUM(salary) as Gasto_En_Salarios FROM Staff;
+-----+
| Gasto_En_Salarios |
+-----+
|          102000 |
+-----+
1 row in set (0.00 sec)

```

5. Crear una consulta que muestre la cantidad que gasta la empresa en salarios quincenales (suponiendo que el dato almacenado es mensual).

```
mysql> SELECT SUM(salary)/2 as Gasto_Quincenal FROM Staff;
+-----+
| Gasto_Quincenal |
+-----+
|      51000.0000 |
+-----+
1 row in set (0.01 sec)
```

6. Mostrar cuantas propiedades en renta existen

```
mysql> SELECT COUNT(propertyNo) as Propiedades_En_Renta FROM PropertyForRent;
+-----+
| Propiedades_En_Renta |
+-----+
|             6 |
+-----+
1 row in set (0.00 sec)
```

7. Mostrar cuantas visitas a las propiedades se han hecho.

```
mysql> SELECT COUNT(clientNo) as Visitas FROM Viewing;
+-----+
| Visitas |
+-----+
|       5 |
+-----+
1 row in set (0.00 sec)
```

8. Mostar la cantidad de clientes que atiende la empresa.

```
mysql> SELECT COUNT(clientNo)as Clientes_DeLa_Empresa FROM Client;
+-----+
| Clientes_DeLa_Empresa |
+-----+
|             4 |
+-----+
1 row in set (0.00 sec)
```

9. Mostrar cuantas propiedades en renta que cuesten más de 350 euros existen.

```
mysql> SELECT COUNT(propertyNo) as Renta_Mayor_A_350 FROM PropertyForRent WHERE rent > 350;
+-----+
| Renta_Mayor_A_350 |
+-----+
|          5        |
+-----+
1 row in set (0.00 sec)
```

10. Mostrar cuantas visitas a la propiedad CR56 se han hecho.

```
mysql> SELECT COUNT(clientNo) as Visitas_CR56 FROM Viewing WHERE clientNo = 'CR56';
+-----+
| Visitas_CR56 |
+-----+
|          3    |
+-----+
1 row in set (0.00 sec)
```

11. Mostrar la cantidad de clientes que puedan pagar una renta mayor a 500 euros atiende la empresa

```
mysql> SELECT COUNT(clientNo) as Pagar_Renta_Mayor_A_500 FROM Client WHERE maxRent > 500;
+-----+
| Pagar_Renta_Mayor_A_500 |
+-----+
|          2              |
+-----+
1 row in set (0.00 sec)
```

12. Calcular el promedio de la renta que pueden pagar los clientes.

```
mysql> SELECT AVG(maxRent) as Renta_Promedio_Clientes FROM Client;
+-----+
| Renta_Promedio_Clientes |
+-----+
|          531.2500      |
+-----+
1 row in set (0.00 sec)
```

13. Mostrar el total de rentas recaudadas al mes

```
mysql> SELECT SUM(rent)*2 as Rentas_Al_Mes FROM PropertyForRent;
+-----+
| Rentas_Al_Mes |
+-----+
|          5650  |
+-----+
1 row in set (0.01 sec)
```

14. Mostrar cual es la renta más cara pagada y cuál es la más barata

```
mysql> SELECT MAX(rent) as Renta_Maxima, MIN(rent) as Renta_Minima from propertyforRent;
+-----+-----+
| Renta_Maxima | Renta_Minima |
+-----+-----+
|          650 |          350 |
+-----+-----+
1 row in set (0.00 sec)
```

15. Calcular el promedio de la renta que recibe la empresa.

```
mysql> SELECT AVG(rent) as Promedio_Rentas_Empresa FROM PropertyForRent;
+-----+
| Promedio_Rentas_Empresa |
+-----+
|          470.8333 |
+-----+
1 row in set (0.00 sec)
```

16. Mostrar el total de rentas que pueden pagar los clientes al mes

```
mysql> SELECT SUM(maxRent)*2 as Total_Rentas_Al_Mes FROM Client;
+-----+
| Total_Rentas_Al_Mes |
+-----+
|          4250 |
+-----+
1 row in set (0.00 sec)
```

17. Mostrar el total de rentas recaudadas por rentar CASAS.

```
mysql> SELECT SUM(rent) as Total_Renta_Casas FROM PropertyForRent WHERE type = 'House';
+-----+
| Total_Renta_Casas |
+-----+
|          1250 |
+-----+
1 row in set (0.00 sec)
```

18. EXPLICAR CÓMO FUNCIONA **max** y **min** UTILIZADO EN CAMPOS VARCHAR

Se tomará como ejemplo la Tabla "Branch" con su clave primaria "BranchNo".

```
mysql> SELECT MAX(branchNo) FROM branch;
+-----+
| MAX(branchNo) |
+-----+
| B007          |
+-----+
1 row in set (0.01 sec)

mysql> SELECT MIN(branchNo) FROM branch;
+-----+
| MIN(branchNo) |
+-----+
| B002          |
+-----+
1 row in set (0.00 sec)

mysql> SELECT * FROM Branch;
+-----+
| branchNo | street      | city      | postcode |
+-----+
| B002     | 56 Clover Dr | London    | NW10 6EU |
| B003     | 163 Main St  | Glasgow   | G11 9QX  |
| B004     | 32 Manse Rd  | Bristol   | BS99 1NZ  |
| B005     | 22 Deer Rd   | London    | SW1 4EH  |
| B007     | 16 Argyll St | Aberdeen  | AB2 3SU  |
+-----+
5 rows in set (0.00 sec)

mysql> DESCRIBE Branch;
+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+
| branchNo   | varchar(10)   | NO   | PRI | NULL    |       |
| street     | varchar(30)   | NO   |     | NULL    |       |
| city       | varchar(30)   | NO   |     | NULL    |       |
| postcode   | varchar(10)   | NO   |     | NULL    |       |
+-----+
4 rows in set (0.00 sec)
```

FUNCIÓN MAX

FUNCIÓN MIN

DATOS

TIPOS DE DATOS

Funciona de esta manera: las funciones son expresiones que pueden tomar como argumentos de una cadena. Compara las columnas *ENUM* y *SET* por sus valores de la cadena, en lugar de sus posiciones. En pocas palabras, si la cadena lleva un número, lógicamente obtiene el número más grande o pequeño según el caso. De lo contrario, si es una cadena está sin algún número, los ordena alfabéticamente. Por ejemplo, en una cadena sin números la función *MIN* es la letra **A**, y la función *MAX* la letra **Z**. Ejemplo:

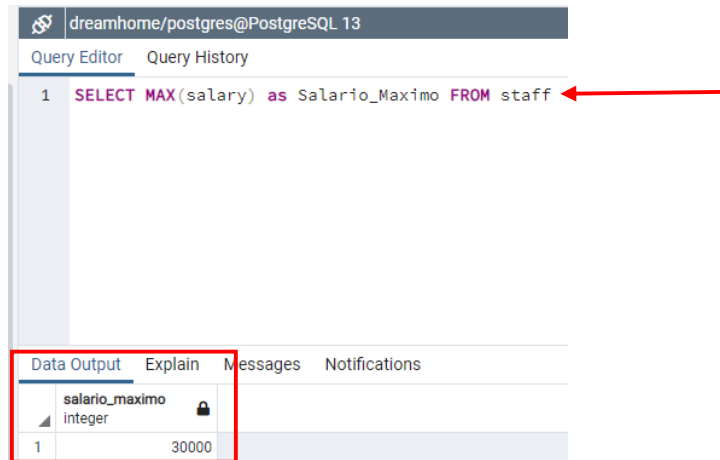
```
mysql> SELECT MIN(CITY) FROM propertyforrent;
+-----+
| MIN(CITY) |
+-----+
| Aberdeen  |
+-----+
1 row in set (0.00 sec)

mysql> SELECT MAX(CITY) FROM propertyforrent;
+-----+
| MAX(CITY) |
+-----+
| London    |
+-----+
1 row in set (0.00 sec)

mysql> DESCRIBE PropertyForRent;
+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+
| propertyNo | char(6)       | NO   | PRI | NULL    |       |
| street     | varchar(30)   | YES  |     | NULL    |       |
| city       | varchar(30)   | YES  |     | NULL    |       |
| postcode   | varchar(10)   | YES  |     | NULL    |       |
| type       | char(10)      | YES  |     | NULL    |       |
| rooms      | tinyint       | YES  |     | NULL    |       |
| rent       | smallint      | YES  |     | NULL    |       |
| ownerNo    | char(6)       | NO   | MUL | NULL    |       |
| staffNo    | varchar(10)   | NO   | MUL | NULL    |       |
| branchNo   | varchar(10)   | NO   | MUL | NULL    |       |
+-----+
10 rows in set (0.00 sec)
```

Consultas generadas en PostgreSQL

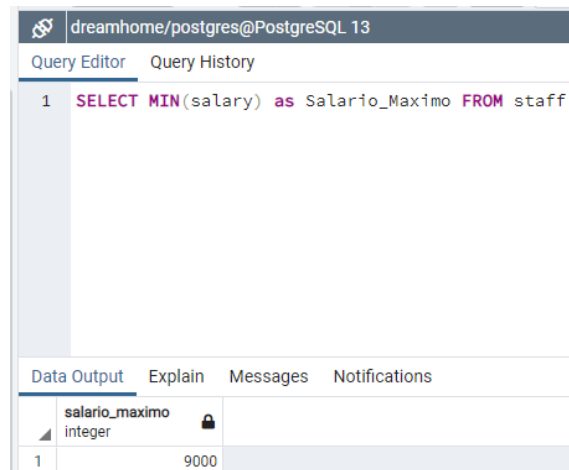
1. Mostrar el salario del empleado que gana más.



The screenshot shows the PostgreSQL Query Editor interface. The query editor contains the following SQL query: `1 SELECT MAX(salary) as Salario_Maximo FROM staff`. A red arrow points to the `FROM staff` part of the query. Below the query editor, the 'Data Output' tab is selected, showing a table with one column named `salario_maximo` of type `integer`. The table contains one row with the value `30000`.

salario_maximo integer
30000

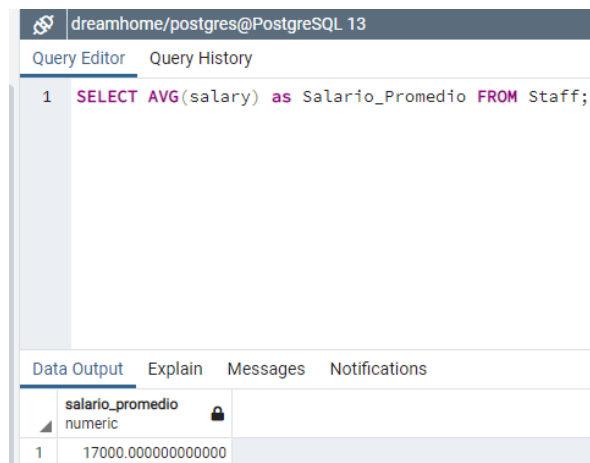
2. Mostrar el salario del empleado que gana menos.



The screenshot shows the PostgreSQL Query Editor interface. The query editor contains the following SQL query: `1 SELECT MIN(salary) as Salario_Maximo FROM staff`. Below the query editor, the 'Data Output' tab is selected, showing a table with one column named `salario_maximo` of type `integer`. The table contains one row with the value `9000`.

salario_maximo integer
9000

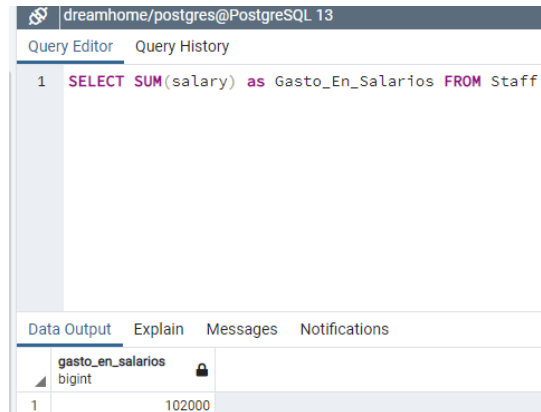
3. Muestre cual es el promedio del salario que perciben los trabajadores.



The screenshot shows the PostgreSQL Query Editor interface. The query editor contains the following SQL query: `1 SELECT AVG(salary) as Salario_Promedio FROM Staff;`. Below the query editor, the 'Data Output' tab is selected, showing a table with one column named `salario_promedio` of type `numeric`. The table contains one row with the value `17000.000000000000`.

salario_promedio numeric
17000.000000000000

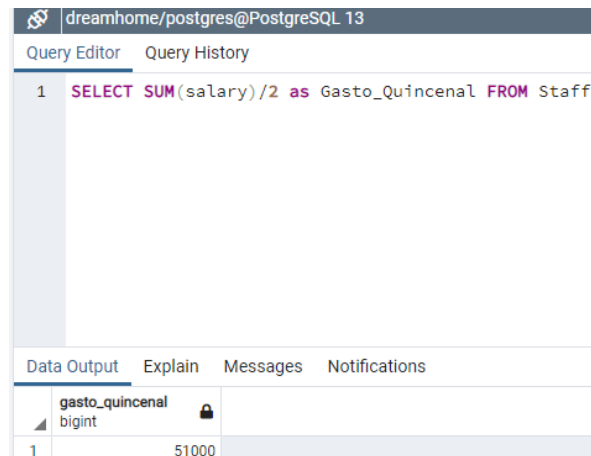
4. Crear una consulta que muestre la cantidad que gasta la empresa en salarios.



The screenshot shows a PostgreSQL query editor interface. The title bar indicates the connection is to 'dreamhome/postgres@PostgreSQL 13'. The 'Query Editor' tab is active, displaying a single SQL query: `1 SELECT SUM(salary) as Gasto_En_Salarios FROM Staff`. Below the query editor, the 'Data Output' tab is selected, showing the results of the query. The results are displayed in a table with two columns: 'gasto_en_salarios' (type 'bigint') and a single row with the value '102000'.

	gasto_en_salarios bigint
1	102000

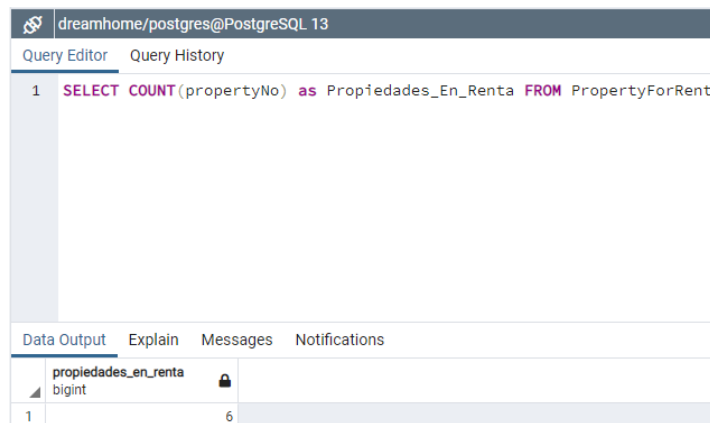
5. Crear una consulta que muestre la cantidad que gasta la empresa en salarios quincenales (suponiendo que el dato almacenado es mensual).



The screenshot shows a PostgreSQL query editor interface. The title bar indicates the connection is to 'dreamhome/postgres@PostgreSQL 13'. The 'Query Editor' tab is active, displaying a single SQL query: `1 SELECT SUM(salary)/2 as Gasto_Quincenal FROM Staff`. Below the query editor, the 'Data Output' tab is selected, showing the results of the query. The results are displayed in a table with two columns: 'gasto_quincenal' (type 'bigint') and a single row with the value '51000'.

	gasto_quincenal bigint
1	51000

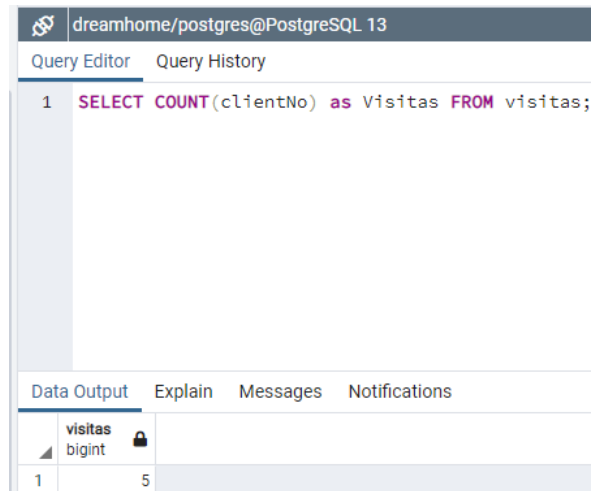
6. Mostrar cuantas propiedades en renta existen.



The screenshot shows a PostgreSQL query editor interface. The title bar indicates the connection is to 'dreamhome/postgres@PostgreSQL 13'. The 'Query Editor' tab is active, displaying a single SQL query: `1 SELECT COUNT(propertyNo) as Propiedades_En_Renta FROM PropertyForRent`. Below the query editor, the 'Data Output' tab is selected, showing the results of the query. The results are displayed in a table with two columns: 'propiedades_en_renta' (type 'bigint') and a single row with the value '6'.

	propiedades_en_renta bigint
1	6

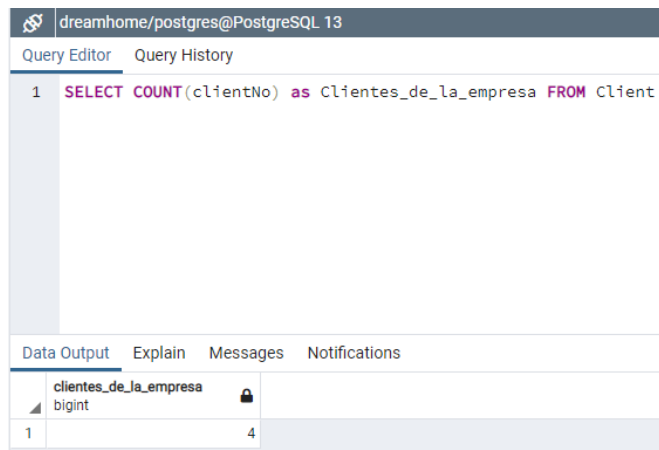
7. Mostrar cuantas visitas a las propiedades se han hecho.



The screenshot shows a PostgreSQL query editor interface. The title bar indicates the connection is to 'dreamhome/postgres@PostgreSQL 13'. The 'Query Editor' tab is active, displaying the SQL query: `1 SELECT COUNT(clientNo) as Visitas FROM visitas;`. Below the query editor, the 'Data Output' tab is selected, showing a table with one row and one column. The column is named 'visitas' and has a data type of 'bigint'. The value in the row is 5.

visitas
5

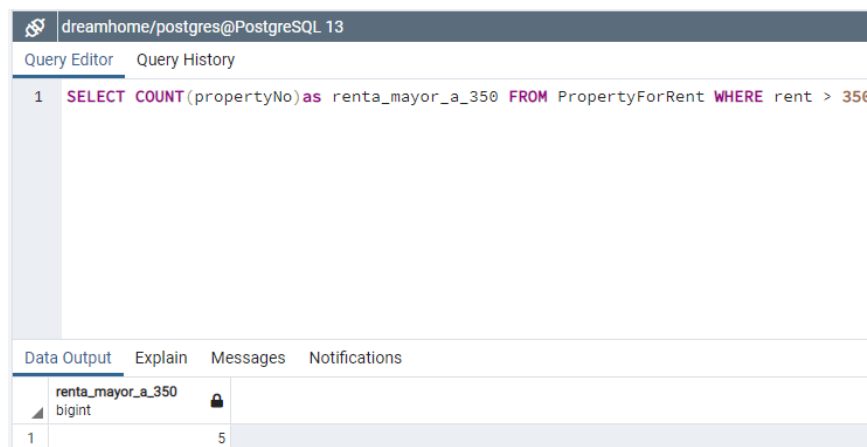
8. Mostrar la cantidad de clientes que atiende la empresa.



The screenshot shows a PostgreSQL query editor interface. The title bar indicates the connection is to 'dreamhome/postgres@PostgreSQL 13'. The 'Query Editor' tab is active, displaying the SQL query: `1 SELECT COUNT(clientNo) as Clientes_de_la_empresa FROM Client`. Below the query editor, the 'Data Output' tab is selected, showing a table with one row and one column. The column is named 'clientes_de_la_empresa' and has a data type of 'bigint'. The value in the row is 4.

clientes_de_la_empresa
4

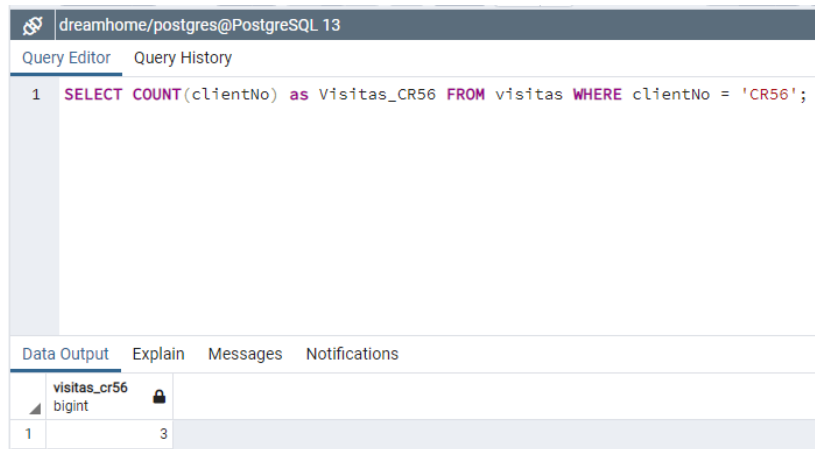
9. Mostrar cuantas propiedades en renta que cuesten más de 350 euros existen.



The screenshot shows a PostgreSQL query editor interface. The title bar indicates the connection is to 'dreamhome/postgres@PostgreSQL 13'. The 'Query Editor' tab is active, displaying the SQL query: `1 SELECT COUNT(propertyNo) as renta_mayor_a_350 FROM PropertyForRent WHERE rent > 350`. Below the query editor, the 'Data Output' tab is selected, showing a table with one row and one column. The column is named 'renta_mayor_a_350' and has a data type of 'bigint'. The value in the row is 5.

renta_mayor_a_350
5

10. Mostrar cuantas visitas a la propiedad CR56 se han hecho



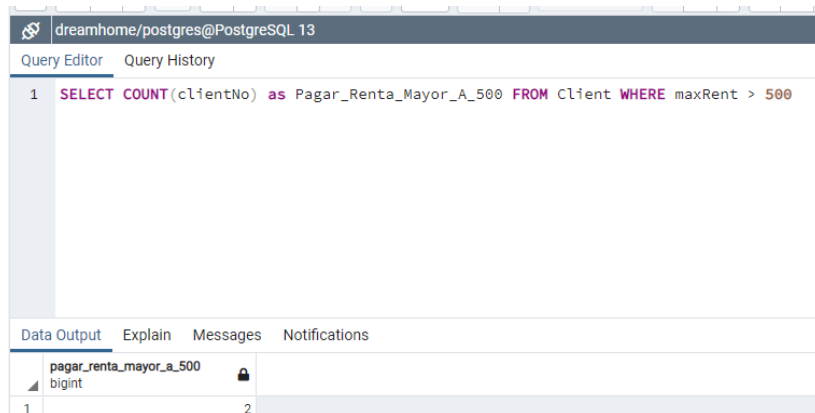
The screenshot shows a PostgreSQL query editor interface. The title bar indicates the connection is to 'dreamhome/postgres@PostgreSQL 13'. The 'Query Editor' tab is active, displaying the following SQL query:

```
1 SELECT COUNT(clientNo) as Visitas_CR56 FROM visitas WHERE clientNo = 'CR56';
```

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

visitas_cr56
3

11. Mostrar la cantidad de clientes que puedan pagar una renta mayor a 500 euros atiende la empresa.



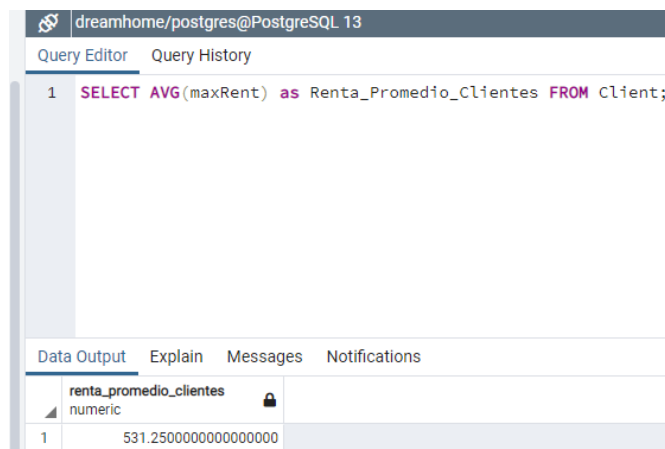
The screenshot shows a PostgreSQL query editor interface. The title bar indicates the connection is to 'dreamhome/postgres@PostgreSQL 13'. The 'Query Editor' tab is active, displaying the following SQL query:

```
1 SELECT COUNT(clientNo) as Pagar_Renta_Mayor_A_500 FROM Client WHERE maxRent > 500
```

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

pagar_renta_mayor_a_500
2

12. Calcular el promedio de la renta que pueden pagar los clientes.



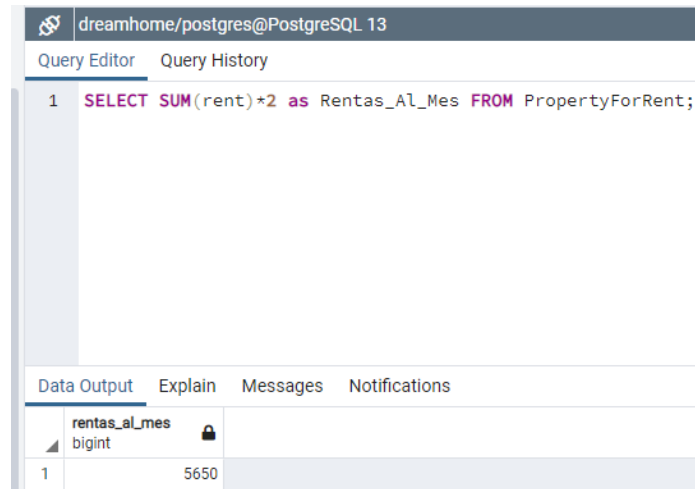
The screenshot shows a PostgreSQL query editor interface. The title bar indicates the connection is to 'dreamhome/postgres@PostgreSQL 13'. The 'Query Editor' tab is active, displaying the following SQL query:

```
1 SELECT AVG(maxRent) as Renta_Promedio_Clientes FROM Client;
```

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

renta_promedio_clientes
531.25000000000000

13. Mostrar el total de rentas recaudadas al mes.



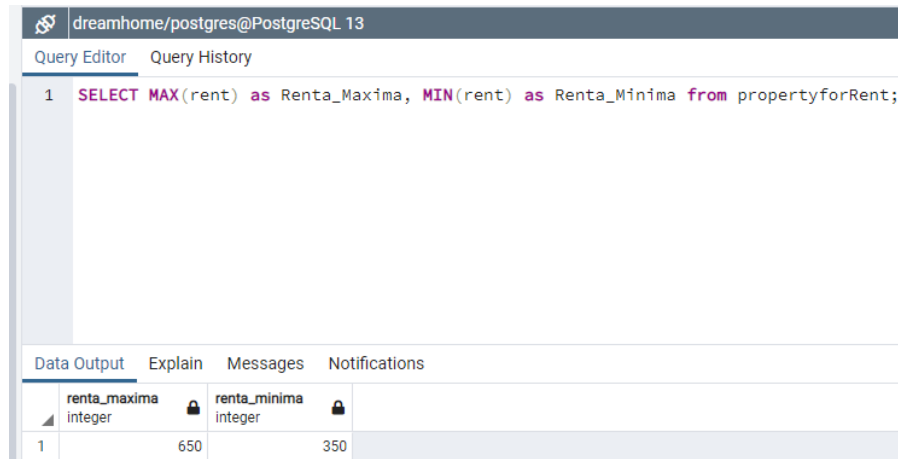
The screenshot shows a PostgreSQL query editor interface. The title bar indicates the connection is to 'dreamhome/postgres@PostgreSQL 13'. The 'Query Editor' tab is active, displaying the following SQL query:

```
1 SELECT SUM(rent)*2 as Rentas_Al_Mes FROM PropertyForRent;
```

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

	rentas_al_mes bigint
1	5650

14. Mostrar cual es la renta más cara pagada y cuál es la más barata.



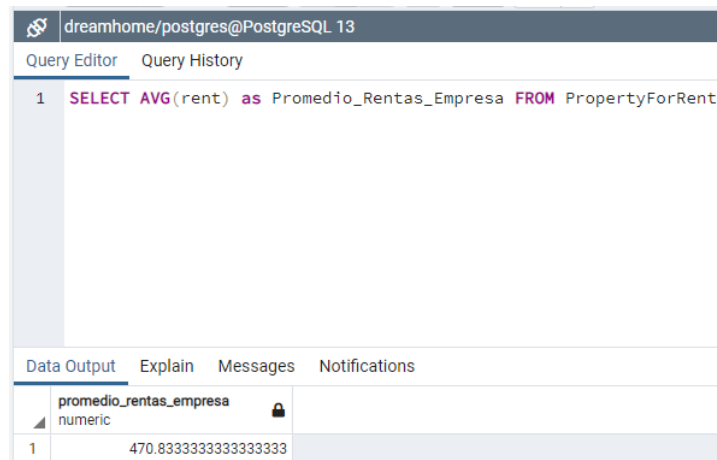
The screenshot shows a PostgreSQL query editor interface. The title bar indicates the connection is to 'dreamhome/postgres@PostgreSQL 13'. The 'Query Editor' tab is active, displaying the following SQL query:

```
1 SELECT MAX(rent) as Renta_Maxima, MIN(rent) as Renta_Minima from propertyforRent;
```

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

	renta_maxima integer	renta_minima integer
1	650	350

15. Calcular el promedio de la renta que recibe la empresa.



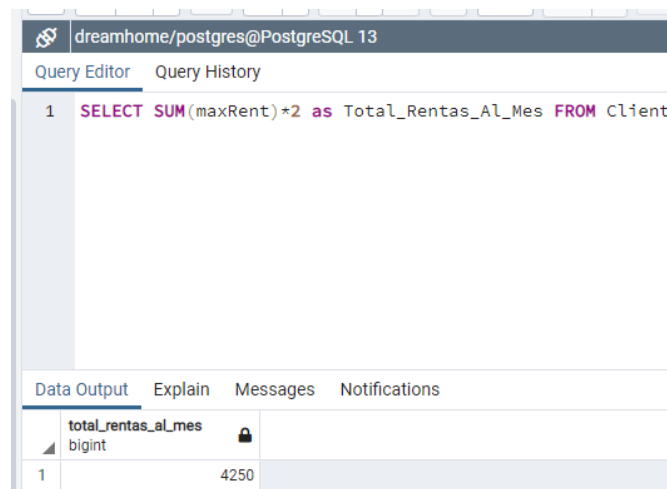
The screenshot shows a PostgreSQL query editor interface. The title bar indicates the connection is to 'dreamhome/postgres@PostgreSQL 13'. The 'Query Editor' tab is active, displaying the following SQL query:

```
1 SELECT AVG(rent) as Promedio_Rentas_Empresa FROM PropertyForRent
```

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

	promedio_rentas_empresa numeric
1	470.83333333333333

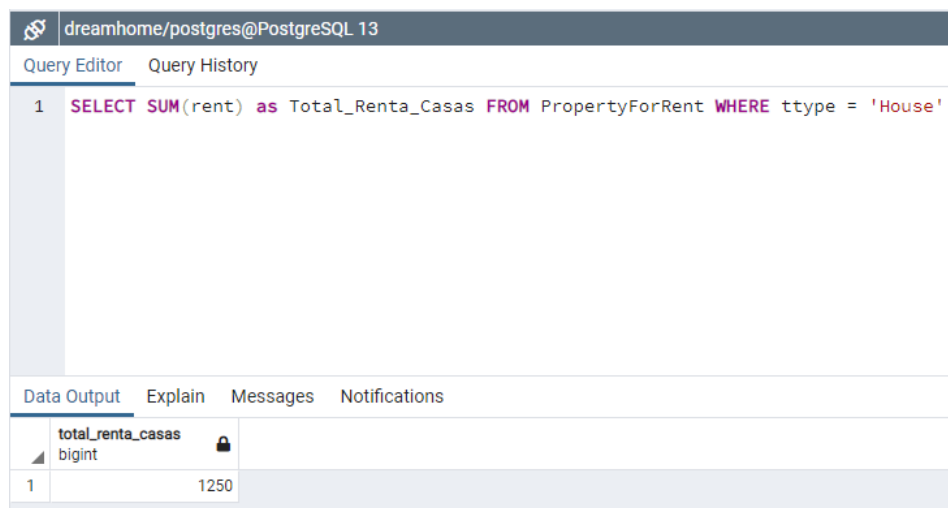
16. Mostrar el total de rentas que pueden pagar los clientes al mes.



The screenshot shows a PostgreSQL query editor interface. The title bar indicates the connection is to 'dreamhome/postgres@PostgreSQL 13'. The 'Query Editor' tab is active, displaying a single SQL query: `1 SELECT SUM(maxRent)*2 as Total_Rentas_Al_Mes FROM Client`. Below the query editor, the 'Data Output' tab is selected, showing the results of the query. The results are presented in a table with one column, 'total_rentas_al_mes', of type 'bigint'. A single row of data is shown with the value 4250.

	total_rentas_al_mes bigint
1	4250

17. Mostrar el total de rentas recaudadas por rentar CASAS



The screenshot shows a PostgreSQL query editor interface. The title bar indicates the connection is to 'dreamhome/postgres@PostgreSQL 13'. The 'Query Editor' tab is active, displaying a single SQL query: `1 SELECT SUM(rent) as Total_Renta_Casas FROM PropertyForRent WHERE ttype = 'House'`. Below the query editor, the 'Data Output' tab is selected, showing the results of the query. The results are presented in a table with one column, 'total_renta_casas', of type 'bigint'. A single row of data is shown with the value 1250.

	total_renta_casas bigint
1	1250