POSTGRESQL

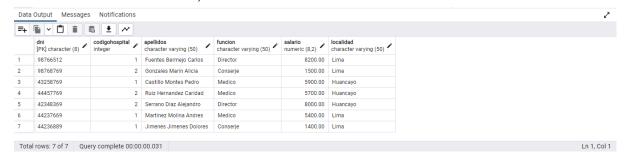
```
1 -- Create database
 2 CREATE DATABASE BdEjemplo;
 3
 4 CREATE TABLE Persona (
 5 dni CHAR(8) NOT NULL,
  6
       CodigoHospital INT NOT NULL,
 7
      apellidos VARCHAR(50) NOT NULL,
 8 funcion VARCHAR(50) NOT NULL,
 9
       salario DECIMAL(8,2) NOT NULL,
 10 localidad VARCHAR(50),
 11 PRIMARY KEY (dni)
 12 );
13
4 CREATE TABLE Persona (
       dni CHAR(8) NOT NULL,
 6 CodigoHospital INT NOT NULL,
      apellidos VARCHAR(50) NOT NULL, funcion VARCHAR(50) NOT NULL,
 7
  8
  9
       salario DECIMAL(8,2) NOT NULL,
 10 localidad VARCHAR(50),
11 PRIMARY KEY (dni)
 12 );
 14 INSERT INTO Persona (dni, CodigoHospital, apellidos, funcion, salario, localidad) VALUES
      ('98766512', 1, 'Fuentes Bermejo Carlos', 'Director', 8200, 'Lima'), ('98768769', 2, 'Gonzales Marin Alicia', 'Conserje', 1500, 'Lima'),
 15
 16
('43258769', 1, 'Castillo Montes Pedro', 'Medico', 5900, 'Huancayo'),
('44457769', 2, 'Ruiz Hernandez Caridad', 'Medico', 5700, 'Huancayo'),
('42348369', 2, 'Serrano Diaz Alejandro', 'Director', 8000, 'Huancayo'),
('44237669', 1, 'Martinez Molina Andres', 'Medico', 5400, 'Lima'),
```

EJERCICIOS

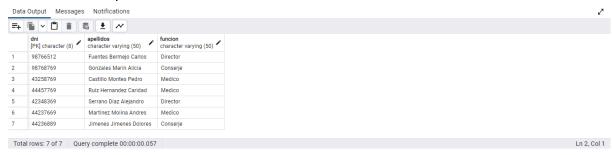
('44236889', 1, 'Jimenes Jimenes Dolores', 'Conserje', 1400, 'Lima');

1.SELECT * FROM Persona;

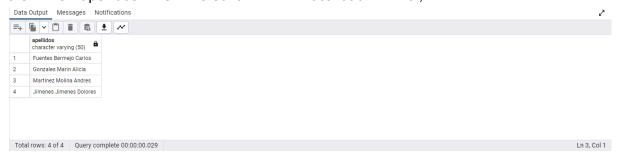
21



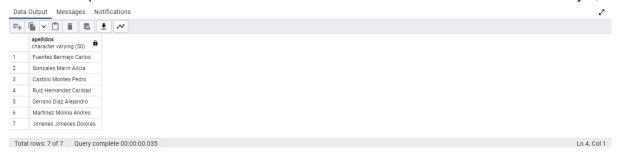
2.SELECT dni, apellidos, funcion FROM Persona;



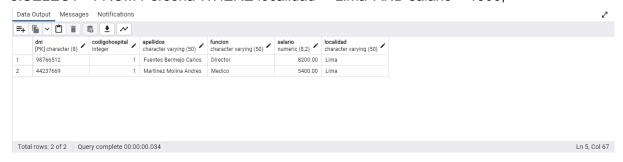
3.SELECT apellidos FROM Persona WHERE localidad = 'Lima';



4.SELECT apellidos FROM Persona WHERE localidad = 'Lima' OR localidad = 'Huancayo';



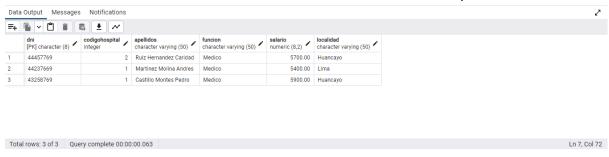
5.SELECT * FROM Persona WHERE localidad = 'Lima' AND salario > 1500;



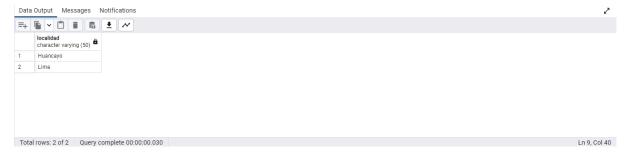
6.SELECT * FROM Persona WHERE localidad = 'Huancayo' AND salario > 1500 AND funcion = 'Director';



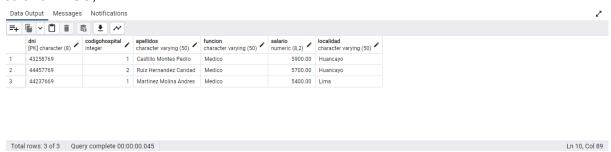
7.SELECT * FROM Persona WHERE funcion = 'Medico' ORDER BY apellidos DESC;



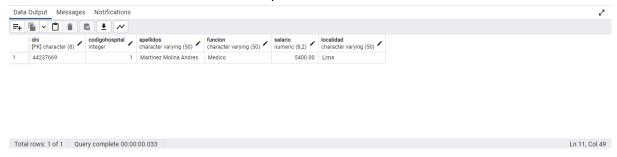
8.SELECT DISTINCT localidad FROM Persona;



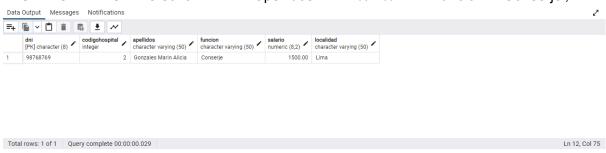
9.SELECT * FROM Persona WHERE salario > 1500 AND funcion = 'Medico' ORDER BY salario DESC;



10.SELECT * FROM Persona WHERE apellidos LIKE 'M%';



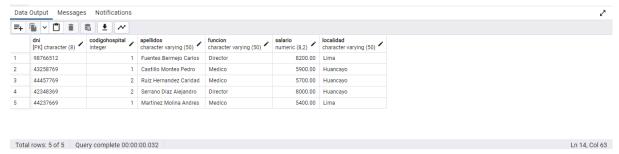
11.SELECT * FROM Persona WHERE apellidos LIKE '%M%' AND funcion = 'Conserje';



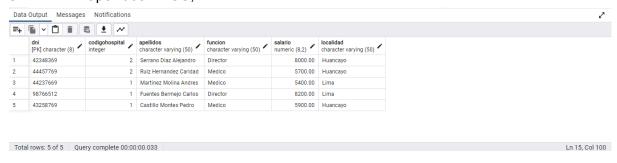
12.SELECT * FROM Persona WHERE salario BETWEEN 1500 AND 2000;



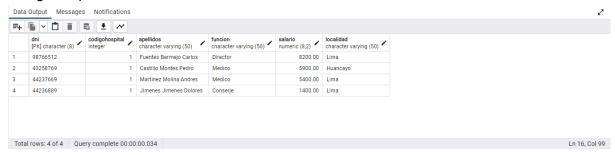
13.SELECT * FROM Persona WHERE funcion IN ('Medico', 'Director');



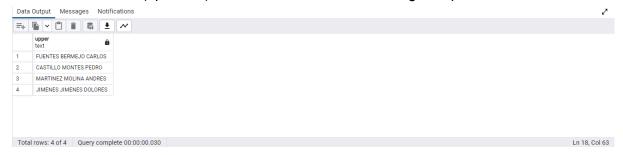
14.SELECT * FROM Persona WHERE funcion NOT IN ('Conserje') AND salario > 1500 ORDER BY apellidos DESC;



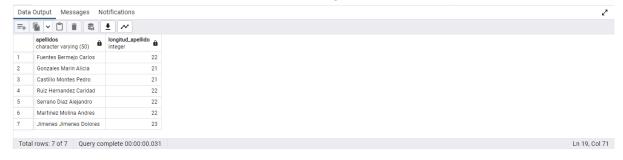
15.SELECT * FROM Persona WHERE (localidad = 'Lima' OR localidad = 'Huancayo') AND CodigoHospital = 1;



16.SELECT UPPER(apellidos) FROM Persona WHERE CodigoHospital = 1;



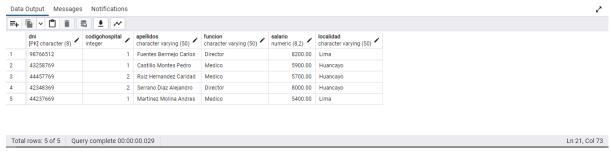
17.SELECT apellidos, LENGTH(apellidos) AS longitud_apellido FROM Persona;



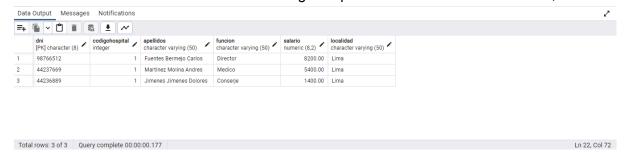
18.SELECT LOWER(apellidos), LOWER(localidad) FROM Persona WHERE CodigoHospital <> 1;



19.SELECT * FROM Persona WHERE CodigoHospital IN (1, 2) AND salario > 1500;



20.SELECT * FROM Persona WHERE CodigoHospital <> 2 AND localidad = 'Lima';



SQLSERVER

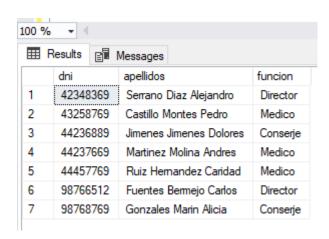
```
SQLQuery1.sql - LA...PC12\USER 17 (54))* → × SQLQuery2.sql - LA...PC12\USER 17 (80))*
     create database BdEjemplo03
     use BdEjemplo03
   □ CREATE TABLE Persona (
      dni char(8) NOT NULL,
     CodigoHospital int not NULL,
      apellidos varchar(50) NOT NULL,
      funcion varchar(50) not null,
      salario decimal(8,2) not null,
      localidad varchar(50) null,
      PRIMARY KEY (dni)
     use BdEjemplo03
   □CREATE TABLE Persona (
      dni char(8) NOT NULL,
      CodigoHospital int not NULL,
      apellidos varchar(50) NOT NULL,
      funcion varchar(50) not null,
      salario decimal(8,2) not null,
      localidad varchar(50) null,
      PRIMARY KEY (dni)
    CREATE TABLE Persona (
       dni char(8) NOT NULL,
       CodigoHospital int not NULL,
       apellidos varchar(50) NOT NULL,
       funcion varchar(50) not null,
       salario decimal(8,2) not null,
       localidad varchar(50) null,
       PRIMARY KEY (dni)
  use bdEjemplo03;
insert into persona values
   ('98766512', 1, 'Fuentes Bermejo Carlos','Director', 8200, 'Lima'),
   ('98768769', 2, 'Gonzales Marin Alicia', 'Conserje', 1500, 'Lima'), ('43258769', 1, 'Castillo Montes Pedro', 'Medico', 5900, 'Huancayo'),
   ('44457769', 2, 'Ruiz Hernandez Caridad', 'Medico', 5700, 'Huancayo'),
   ('42348369', 2, 'Serrano Diaz Alejandro', 'Director', 8000, 'Huancayo'),
   ('44237669', 1, 'Martinez Molina Andres', 'Medico', 5400, 'Lima'),
   ('44236889', 1, 'Jimenes Jimenes Dolores', 'Conserje', 1400, 'Lima')
```

EJERCICIOS

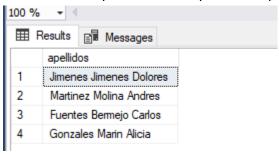
1. Mostrar todos los datos de todas las personas.



2. Obtén el DNI, apellidos y función de todas las personas.



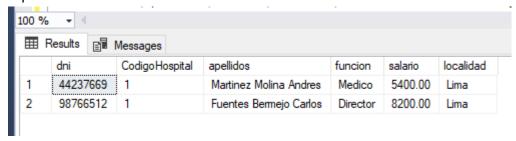
3. Mostrar los apellidos de las personas que vivan en Lima.



4. Mostrar los apellidos de las personas que vivan en lima o Huancayo



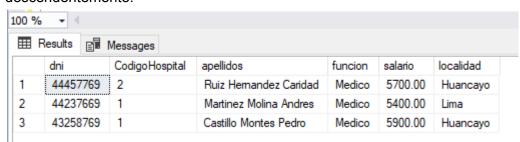
5. Seleccionar los datos de aquellas personas que vivan en Lima y tengan un salario superior a los 1500 soles.



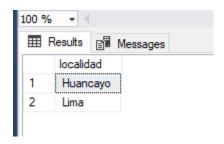
6. Mostrar los datos de las personas que vivan en Huancayo, tengan un salario superior a los 1500 euros y sean directores.



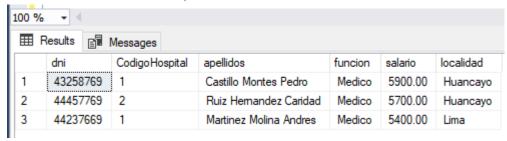
7. Mostrar los datos de las personas cuya función sea Médico ordenados por apellidos descendentemente.



8. Mostrar los datos de todas las localidades que hay en la tabla personas sin repeticiones (debes emplear la cláusula DISTINCT)



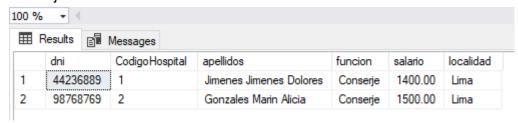
9. Mostrar los datos de las personas que tengan un salario superior a 1500 soles y sean médicos. Ordenar la salida por salario descendentemente.



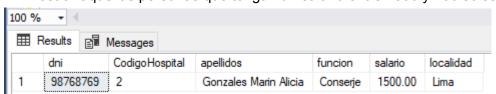
10. Seleccionar aquellas personas cuyo apellido comience por M.



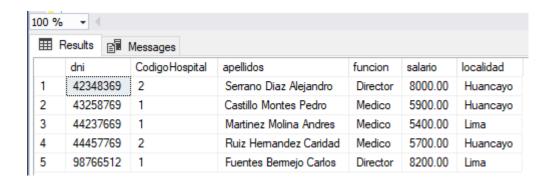
11. Mostrar los datos de las personas que tengan una M en el apellido y cuya función sea Conserje



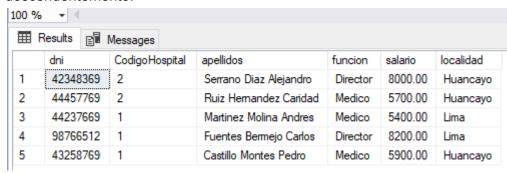
12. Mostrar aquellas personas que tengan un salario entre 1500 y 200 soles.



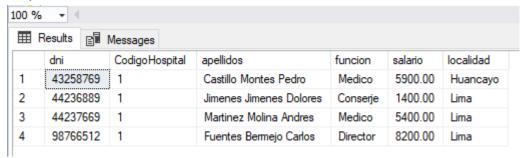
13. Seleccionar los datos de aquellas personas cuya función sea Médico o director (utilizar el operador IN)



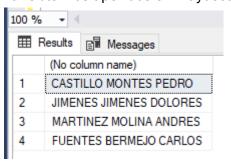
14. Obtener los datos de aquellas personas cuya función no sea Conserje (utilizar el operador NOT IN) y tengan un salario superior a los 1500 soles, ordenados por apellido descendentemente.



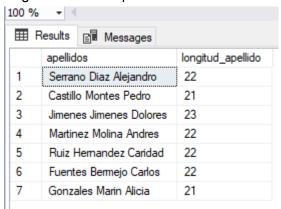
15. Mostrar los datos de las personas que sean de Lima o Huancayo y que pertenezcan al hospital número 1.



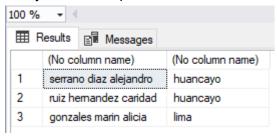
16. Obtén los apellidos en mayúsculas de las personas que trabajen en el hospital número



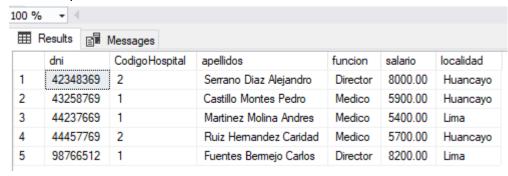
17. Con una consulta devuelve los apellidos de todas las personas. Al lado debe aparecer la longitud de cada apellido.



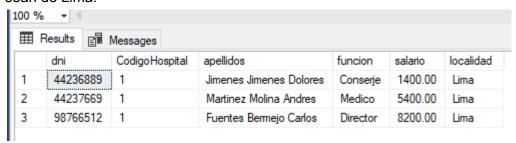
18. Obtener los apellidos y localidad en minúscula de todas aquellas personas que no trabajen en el hospital número 1.



19. Obtener los datos de las personas que trabajen en los hospitales 1 ó 2 y tengan un salario superior a 1500 soles.



20. Visualizar los datos de aquellas personas que no trabajen en el hospital número 2 y que sean de Lima.



DIFERENCIA:

-PostgreSQL es reconocido por su versatilidad y capacidad de personalización, mientras que SQL Server sobresale en términos de integración y en sus herramientas de gestión.