

## EXAMEN BASE DE DATOS SQL

### 1.- CREAR UNA BASE DE DATOS LLAMADA PERSONAL.

En esta ocasión utilice el gestor de bases de datos PostgreSQL y con pgAdmin 4 cree la base de datos en la cual se trabajaría.

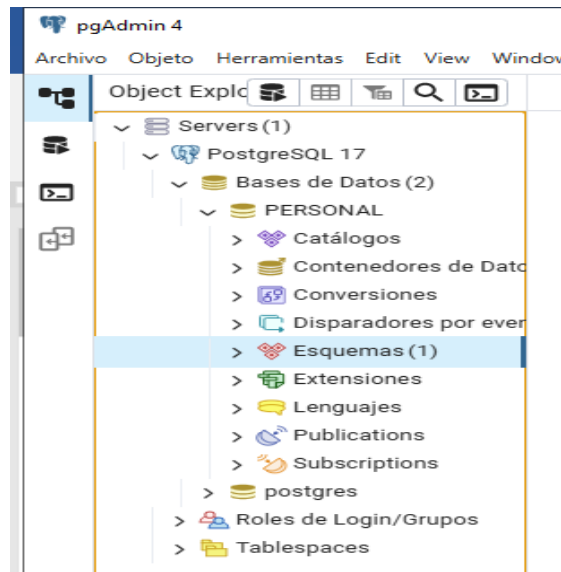


Ilustración 1. CREAR UNA BASE DE DATOS LLAMADA PERSONAL

### 2.- CREAR LAS SIGUIENTES TABLAS DENTRO DE LA BASE

Despues se procedió a crear cada una de las tablas especificadas con sus atributos.

#### 2.1 TABLA PERSONAS

- ESPECIFICACIONES: LOGIN ENTERO.
- ÁREA ENTERO.
- ZONA ENTERO.
- PUESTO ENTERO.
- NO PERMITIR LOGIN DUPLICADOS.
- NOMBRE CADENA DE MÁXIMO 100 CARACTERES.

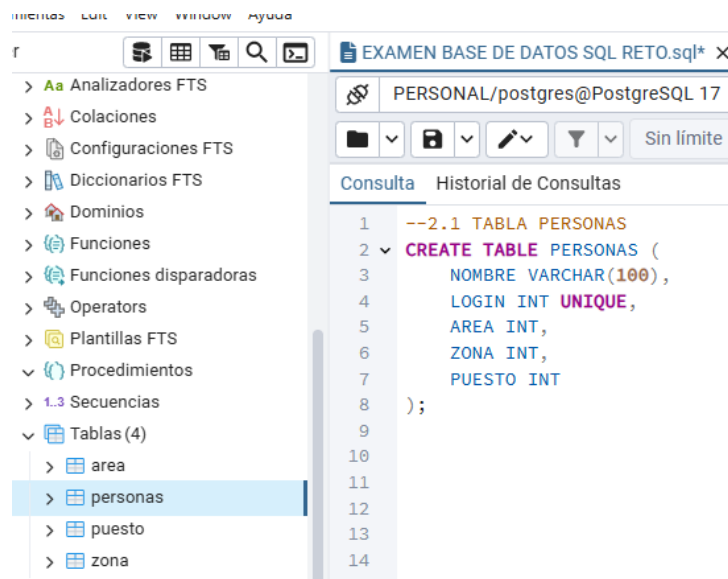


Ilustración 2.Creación de la tabla PERSONAS

## 2.2 TABLA ÁREA

- ESPECIFICACIONES: IDAREA AUTO INCREMENTABLE EN 1 Y COMO CLAVE PRINCIPAL
- DESCRIPCIÓN CADENA NO MÁXIMO DE 50 CARACTERES.

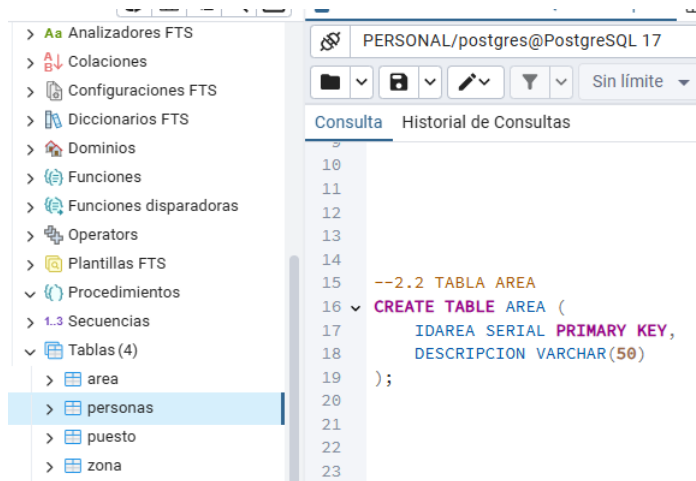


Ilustración 3. TABLA ÁREA

## 2.3 TABLA ZONA

- ESPECIFICACIONES: IDZONA AUTO INCREMENTABLE EN 1 Y COMO CLAVE PRINCIPAL
- DESCRIPCIÓN CADENA NO MÁXIMO DE 50 CARACTERES.

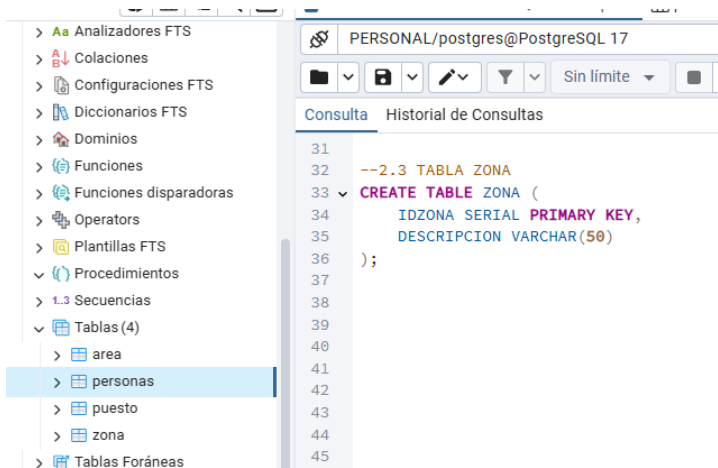


Ilustración 4. TABLA ZONA

## 2.4 TABLA PUESTO

- ESPECIFICACIONES: IDPUESTO AUTO INCREMENTABLE EN 1 Y COMO CLAVE PRINCIPAL.
- DESCRIPCIÓN CADENA NO MÁXIMO DE 50 CARACTERES.

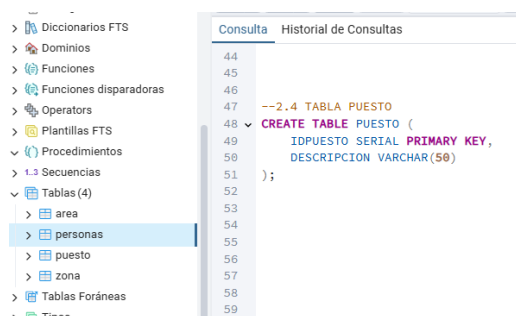


Ilustración 5. TABLA PUESTO

PARA CONTINUAR CON LA CREACIÓN DE QUERYS INGRESE DATOS A LAS TABLAS PREVIAMENTE CREADAS.

```

54
55
56
57 -- Insertar datos en la tabla AREA
58 INSERT INTO AREA (DESCRIPCION) VALUES
59 ('TELEFONIA'),
60 ('IMPLEMENTACION'),
61 ('CAPACITACION'),
62 ('CALIDAD'),
63 ('DESARROLLO');
64 -- Insertar datos en la tabla ZONA
65 INSERT INTO ZONA (DESCRIPCION) VALUES
66 ('DEL VALLE'),
67 ('NARVARTE'),
68 ('RELOX'),
69 ('TORRES');
70 -- Insertar datos en la tabla PUESTO
71 INSERT INTO PUESTO (DESCRIPCION) VALUES
72 ('COORDINADOR'),
73 ('EMPLEADO');
74
75

```

Ilustración 6. INSERCIÓN DE DATOS A LAS TABLAS

```

78
79 -- Insertar datos en la tabla PERSONAS
80 INSERT INTO PERSONAS (NOMBRE, LOGIN, AREA, ZONA, PUESTO) VALUES
81 ('EDGAR', 5654, 1, 1, 1),
82 ('CESAR', 5652, 1, 1, 2),
83 ('URIEL', 5650, 1, 2, 2),
84 ('ISRAEL', 5648, 1, 3, 2),
85 ('JOSUE', 5646, 3, 1, 1),
86 ('FELIPE', 5644, 3, 4, 2),
87 ('GUILLERMO', 5642, 2, 2, 1),
88 ('ALI', 5640, 2, 3, 2),
89 ('MAURO', 5638, 4, 1, 1),
90 ('MIRIAM', 5636, 4, 4, 2),
91 ('RICARDO', 5634, 5, 2, 2),
92 ('LUIS', 5632, 5, 2, 1),
93 ('JOSE', 5630, 5, 3, 2);
94
95
96 --? - QUEPYS
97

```

Ilustración 6.1. INSERCIÓN DE DATOS A LA TABLA PERSONAS

### 3. QUERY:

EN ESTE APARTADO MOSTRARE COMO ESTRUCTURE CADA CONSULTA PARA OBTENER LOS DATOS SOLICITADOS.

#### 3.1 MOSTRAR A TODOS LOS COORDINADORES.

EN ESTE CASO UTILICE DOS CONSULTAS UNA DONDE SOLO DISPONE DE LO QUE SE SOLICITA Y UNA MAS AMPLIA Y DETALLADA.

SE UTILIZA JOIN PARA UNIR LA TABLA PERSONAS CON LA TABLA PUESTO A TRAVÉS DE LA RELACIÓN ENTRE p.PUESTO Y pu.IDPUESTO.

```

94
95 --3.- QUERYS
96 /* 3.1 MOSTRAR A TODOS LOS COORDINADORES
97 Se utiliza JOIN para unir la tabla PERSONAS
98 con la tabla PUESTO a través de la relación entre p.PUESTO y pu.IDPUESTO.*/
99 SELECT NOMBRE
100 FROM PERSONAS
101 JOIN PUESTO ON PUESTO = IDPUESTO
102 /*WHERE filtra los resultados para mostrar
103 solo aquellos registros donde la DESCRIPCION del puesto sea 'COORDINADOR'*/
104 WHERE DESCRIPCION = 'COORDINADOR';
105

```

Data Output

nombre	character varying (100)
1 EDGAR	
2 JOSUE	
3 GUILLERMO	
4 MAURO	
5 LUIS	

Total rows: 5 Query complete 00:00:00.477

Ilustración 7. CONSULTA SIMPLE CON SU RESULTADO

```

107
108 --Con mas atributos
109 SELECT
110 p.NOMBRE,
111 p.LOGIN,
112 a.DESCRIPCION AS AREA,
113 z.DESCRIPCION AS ZONA,
114 pu.DESCRIPCION AS PUESTO
115 FROM
116 PERSONAS p
117 JOIN
118 PUESTO pu ON p.PUESTO = pu.IDPUESTO
119 JOIN
120 AREA a ON p.AREA = a.IDAREA
121 JOIN
122 ZONA z ON p.ZONA = z.IDZONA
123 WHERE
124 pu.DESCRIPCION = 'COORDINADOR';
125

```

Data Output

nombre	login	area	zona	puesto
character varying (100)	integer	character varying (50)	character varying (50)	character varying (50)
1 EDGAR	5654	TELEFONIA	DEL VALLE	COORDINADOR
2 JOSUE	5646	CAPACITACION	DEL VALLE	COORDINADOR
3 GUILLERMO	5642	IMPLEMENTACION	NARVARTE	COORDINADOR
4 MAURO	5638	CALIDAD	DEL VALLE	COORDINADOR
5 LUIS	5632	DESARROLLO	NARVARTE	COORDINADOR

Mensajes Notificaciones

Ejecución exitosa. Tiempo de ejecución total de la consulta: 549 msec.  
5 filas afectadas.

Ilustración 8. CONSULTA CON MAS DETALLES

3.2 MOSTRAR A TODAS LAS PERSONAS QUE LABOREN EN LA ZONA DEL VALLE  
SE UTILIZA JOIN PARA UNIR LA TABLA PERSONAS CON LA TABLA PUESTO  
WHERE FILTRA LOS RESULTADOS PARA QUE SOLO SE MUESTREN LAS PERSONAS QUE TRABAJAN EN LA ZONA DEL VALLE.

```

--3.2 MOSTRAR A TODAS LAS PERSONAS QUE
--LABOREN EN LA ZONA DEL VALLE
1 SELECT
2     p.NOMBRE,
3     p.LOGIN,
4     a.DESCRIPCION AS AREA,
5     z.DESCRIPCION AS ZONA,
6     pu.DESCRIPCION AS PUESTO
7 FROM
8     PERSONAS p
9 JOIN
10    ZONA z ON p.ZONA = z.IDZONA
11 JOIN
12    AREA a ON p.AREA = a.IDAREA
13 JOIN
14    PUESTO pu ON p.PUESTO = pu.IDPUESTO
15 WHERE
16    z.DESCRIPCION = 'DEL VALLE';
  
```

	nombre	login	area	zona	puesto
	character varying (100)	integer	character varying (50)	character varying (50)	character varying (50)
1	EDGAR	5654	TELEFONIA	DEL VALLE	COORDINADOR
2	CESAR	5652	TELEFONIA	DEL VALLE	EMPLEADO
3	JOSUE	5646	CAPACITACION	DEL VALLE	COORDINADOR
4	MAURO	5638	CALIDAD	DEL VALLE	COORDINADOR

Ilustración 9. JOIN PARA UNIR TABLA PERSONAS Y PUESTO

3.3 MOSTRAR A TODOS LOS EMPLEADOS QUE LABOREN EN LA ZONA NARVARTE  
IGUAL QUE EN EL MISMO EJEMPLO UTILICE DOS CONSULTAS UNA SIMPLE Y UNA DETALLADA  
SE UNEN LAS TRES TABLAS (PERSONAS, PUESTO y ZONA).

```

--3.3 MOSTRAR A TODOS LOS EMPLEADOS QUE
--LABOREN EN LA ZONA NARVARTE
1 SELECT p.NOMBRE
2 FROM PERSONAS p
3 JOIN PUESTO pu ON p.PUESTO = pu.IDPUESTO
4 JOIN ZONA z ON p.ZONA = z.IDZONA
5 WHERE pu.DESCRIPCION = 'EMPLEADO' AND z.DESCRIPCION = 'NARVARTE';
  
```

```

SELECT
  p.NOMBRE,
  p.LOGIN,
  a.DESCRIPCION AS AREA,
  z.DESCRIPCION AS ZONA,
  pu.DESCRIPCION AS PUESTO
FROM
  PERSONAS p
JOIN
  PUESTO pu ON p.PUESTO = pu.IDPUESTO
JOIN
  ZONA z ON p.ZONA = z.IDZONA
JOIN
  AREA a ON p.AREA = a.IDAREA
WHERE
  pu.DESCRIPCION = 'EMPLEADO' AND z.DESCRIPCION = 'NARVARTE';
  
```

	nombre	login	area	zona	puesto
	character varying (100)	integer	character varying (50)	character varying (50)	character varying (50)
1	URIEL	5650	TELEFONIA	NARVARTE	EMPLEADO
2	RICARDO	5634	DESARROLLO	NARVARTE	EMPLEADO

Ilustración 10. CONSULTA SIMPLE

Ilustración 11. CONSULTA DETALLADA

3.4 AGRUPAR TODAS LAS PERSONAS POR ÁREA

- Se unen las tablas PERSONAS y AREA.
- GROUP BY a.DESCRIPCIÓN: Agrupa los resultados por el nombre del área.
- COUNT(p.NOMBRE): Cuenta cuántas personas hay en cada grupo (área).

```

--3.4 AGRUPAR TODAS LAS PERSONAS POR AREA
--ESPECIFICA
1 SELECT a.DESCRIPCION AS AREA,
2     COUNT(p.NOMBRE) AS NUMERO_DE_PERSONAS
3 FROM PERSONAS p
4 JOIN AREA a ON p.AREA = a.IDAREA
5 GROUP BY a.DESCRIPCION;
--DETALLADA
1 SELECT
2     a.DESCRIPCION AS AREA,
3     COUNT(p.NOMBRE) AS NUMERO_DE_PERSONAS,
4     STRING_AGG(p.NOMBRE, ',') AS NOMBRES_DE_PERSONAS
5 FROM
6     PERSONAS p
7 JOIN
8     AREA a ON p.AREA = a.IDAREA
9 GROUP BY
10    a.DESCRIPCION;
  
```

area	numero_de_personas
character varying (50)	bigint
1 CAPACITACION	2
2 DESARROLLO	3
3 IMPLEMENTACION	2
4 CALIDAD	2
5 TELEFONIA	4

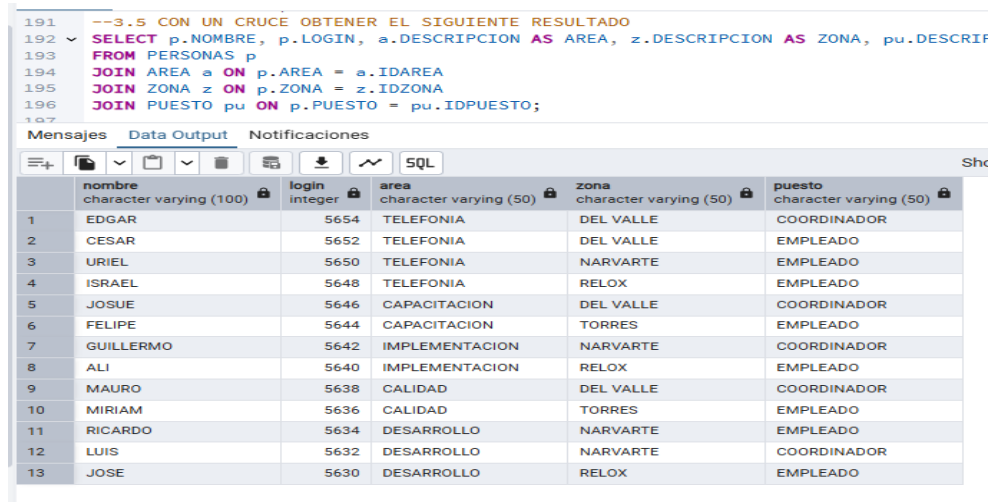
area	numeros_de_personas	nombres_de_personas
character varying (50)	bigint	text
1 CAPACITACION	2	JOSUE FELIPE
2 DESARROLLO	3	RICARDO LUIS, JOSE
3 IMPLEMENTACION	2	GUILLERMO ALI
4 CALIDAD	2	MAURO MIRIAM
5 TELEFONIA	4	EDGAR, CESAR, URIEL, ISRA

Ilustración 12. CONSULTA SIMPLE CON JOIN

Ilustración 13. CONSULTA MAS DETALLADA

### 3.5 CON UN CRUCE OBTENER EL SIGUIENTE RESULTADO

ESTA CONSULTA UTILIZA JOIN PARA COMBINAR TODAS LAS TABLAS (PERSONAS, ÁREA, ZONA, PUESTO) Y MOSTRAR LA DESCRIPCIÓN TEXTUAL DE CADA ID EN LAS COLUMNAS CORRESPONDIENTES.



```
191 --3.5 CON UN CRUCE OBTENER EL SIGUIENTE RESULTADO
192 SELECT p.NOMBRE, p.LOGIN, a.DESCRIPCION AS AREA, z.DESCRIPCION AS ZONA, pu.DESCRIPCION AS PUESTO
193 FROM PERSONAS p
194 JOIN AREA a ON p.AREA = a.IDAREA
195 JOIN ZONA z ON p.ZONA = z.IDZONA
196 JOIN PUESTO pu ON p.PUESTO = pu.IDPUESTO;
```

	nombre character varying (100)	login integer	area character varying (50)	zona character varying (50)	puesto character varying (50)
1	EDGAR	5654	TELEFONIA	DEL VALLE	COORDINADOR
2	CESAR	5652	TELEFONIA	DEL VALLE	EMPLEADO
3	URIEL	5650	TELEFONIA	NARVARTE	EMPLEADO
4	ISRAEL	5648	TELEFONIA	RELOX	EMPLEADO
5	JOSUE	5646	CAPACITACION	DEL VALLE	COORDINADOR
6	FELIPE	5644	CAPACITACION	TORRES	EMPLEADO
7	GUILLERMO	5642	IMPLEMENTACION	NARVARTE	COORDINADOR
8	ALI	5640	IMPLEMENTACION	RELOX	EMPLEADO
9	MAURO	5638	CALIDAD	DEL VALLE	COORDINADOR
10	MIRIAM	5636	CALIDAD	TORRES	EMPLEADO
11	RICARDO	5634	DESARROLLO	NARVARTE	EMPLEADO
12	LUIS	5632	DESARROLLO	NARVARTE	COORDINADOR
13	JOSE	5630	DESARROLLO	RELOX	EMPLEADO

Ilustración 14. CONSULTA CON REFERENCIA A LA IMAGEN PROPORCIONADA

## 4. PROCEDIMIENTOS ALMACENADOS (STORED PROCEDURES)

A CONTINUACIÓN, SE CREAN LOS PROCEDIMIENTOS ALMACENADOS SOLICITADOS.

4.1 CREAR UN SP CON UN PARÁMETRO DE ENTRADA @IDZONA QUE MUESTRE TODOS LAS PERSONAS DE LA ZONA INDICADA EN EL PARÁMETRO.

Explicación:

CREATE OR REPLACE FUNCTION: EN POSTGRESQL, LOS PROCEDIMIENTOS ALMACENADOS SE CREAN COMO FUNCIONES. CREATE OR REPLACE PERMITE CREARLA O ACTUALIZARLA SI YA EXISTE.

SP\_PERSONAS\_POR\_ZONA(ID\_ZONA INT): DEFINE EL NOMBRE DE LA FUNCIÓN Y UN PARÁMETRO DE ENTRADA ID\_ZONA DE TIPO ENTERO.

RETURNS TABLE (...): ESPECIFICA LA ESTRUCTURA DE LA TABLA QUE LA FUNCIÓN DEVOLVERÁ.

RETURN QUERY: LA FUNCIÓN DEVUELVE EL RESULTADO DE LA CONSULTA.

LA CONSULTA SELECCIONA LOS DATOS DE LAS PERSONAS UNIENDO LAS TABLAS, Y EL WHERE FILTRA LOS RESULTADOS BASÁNDOSE EN EL PARÁMETRO DE ENTRADA ID\_ZONA.

```
/*CREAR UN SP CON UN PARAMETRO DE ENTRADA @IDZONA QUE MUESTRE TODOS LAS PERSONAS
DE LA ZONA INDICADA EN EL PARAMETRO.*/
CREATE OR REPLACE FUNCTION sp_personas_por_zona(id_zona INT)
RETURNS TABLE (nombre VARCHAR(100), login INT, area VARCHAR(50), zona VARCHAR(50), puesto VARCHAR(50)) AS $$
BEGIN
    RETURN QUERY
    SELECT per.NOMBRE, per.LOGIN, ar.DESCRIPCION, zo.DESCRIPCION, pue.DESCRIPCION
    FROM PERSONAS AS per
    JOIN AREA AS ar ON per.AREA = ar.IDAREA
    JOIN ZONA AS zo ON per.ZONA = zo.IDZONA
    JOIN PUESTO AS pue ON per.PUESTO = pue.IDPUESTO
    WHERE per.ZONA = id_zona;
END;
$$ LANGUAGE plpgsql;
```

Ilustración 14. CREACIÓN DE STORED PROCEDURES

## 4.2 MUESTRO UN EJEMPLO DE COMO SE USARÍA EL PROCESO ALMACENADO, ESTO NOS AYUDA AHORRAR A ESCRIBIR QUERYS Y HACE MAS OPTIMO LAS CONSULTAS.

```
3 -- Ejemplo de uso:
4 SELECT * FROM sp_personas_por_zona(1);
5
```

nombre	login	area	zona	puesto
character varying	integer	character varying	character varying	character varying
EDGAR	5654	TELEFONIA	DEL VALLE	COORDINADOR
CESAR	5652	TELEFONIA	DEL VALLE	EMPLEADO
JOSUE	5646	CAPACITACION	DEL VALLE	COORDINADOR
MAURO	5638	CALIDAD	DEL VALLE	COORDINADOR

Ilustración 15. EJEMPLO DE COMO USAR UN PROCESO ALMACENADO

## 4.3 CREAR UN SP CON UN PARÁMETRO DE ENTRADA @PUESTO QUE MUESTRE TODAS LAS PERSONAS QUE TENGAN EL PUESTO INDICADO EN EL PARÁMETRO.

PERSONAL/postgres@PostgreSQL 17

```
100 -- Que tengan el puesto indicado en el parametro.*/
101 CREATE OR REPLACE FUNCTION sp_personas_por_puesto(id_puesto INT)
102 RETURNS TABLE (nombre VARCHAR(100), login INT, area VARCHAR(50), zona VARCHAR(50), puesto VARCHAR(50)) AS $$
103 BEGIN
104     RETURN QUERY
105     SELECT per.NOMBRE, per.LOGIN, ar.DESCRIPCION, zo.DESCRIPCION, pue.DESCRIPCION
106     FROM PERSONAS AS per
107     JOIN AREA AS ar ON per.AREA = ar.IDAREA
108     JOIN ZONA AS zo ON per.ZONA = zo.IDZONA
109     JOIN PUESTO AS pue ON per.PUESTO = pue.IDPUESTO
110     WHERE per.PUESTO = id_puesto;
111 END;
112 $$ LANGUAGE plpgsql;
```

nombre	login	area	zona	puesto
character varying	integer	character varying	character varying	character varying
CESAR	5652	TELEFONIA	DEL VALLE	EMPLEADO
URIEL	5650	TELEFONIA	NARVARTE	EMPLEADO
ISRAEL	5648	TELEFONIA	RELOX	EMPLEADO
FELIPE	5644	CAPACITACION	TORRES	EMPLEADO
ALI	5640	IMPLEMENTACION	RELOX	EMPLEADO
MIRIAM	5636	CALIDAD	TORRES	EMPLEADO
RICARDO	5634	DESARROLLO	NARVARTE	EMPLEADO
JOSE	5630	DESARROLLO	RELOX	EMPLEADO

Total rows: 8 Query complete 00:00:00.461

Ilustración 16. PROCESO ALMACENADO

## 4.4 EJEMPLO DE COMO SE USARÍA ESTE PROCESO ALMACENADO

```
104 SELECT * FROM sp_personas_por_puesto(2);
105
```

nombre	login	area	zona	puesto
character varying	integer	character varying	character varying	character varying
CESAR	5652	TELEFONIA	DEL VALLE	EMPLEADO
URIEL	5650	TELEFONIA	NARVARTE	EMPLEADO
ISRAEL	5648	TELEFONIA	RELOX	EMPLEADO
FELIPE	5644	CAPACITACION	TORRES	EMPLEADO
ALI	5640	IMPLEMENTACION	RELOX	EMPLEADO
MIRIAM	5636	CALIDAD	TORRES	EMPLEADO
RICARDO	5634	DESARROLLO	NARVARTE	EMPLEADO
JOSE	5630	DESARROLLO	RELOX	EMPLEADO

Total rows: 8 Query complete 00:00:00.578

Ilustración 17 PROCESO ALMACENADO

#### 4.5 CREAR UN SP PARA OBTENER EL SIGUIENTE RESULTADO

	TELEFONIA	IMPLEMENTACION	CAPACITACION	CALIDAD	DESARROLLO
GENTE ASIGNADA	4	2	2	2	3

##### EXPLICACIÓN:

- ESTA FUNCIÓN NO TOMA PARÁMETROS DE ENTRADA.
- LA CONSULTA UNE PERSONAS E ÁREA, Y USA GROUP BY Y COUNT PARA CONTAR EL NÚMERO DE PERSONAS EN CADA ÁREA. EL RESULTADO SE DEVUELVE EN UNA TABLA CON LAS COLUMNAS (ÁREA Y GENTE ASIGNADA.)

```
6 /*CREAR UN SP PARA OBTENER EL SIGUIENTE RESULTADO*/
7 CREATE OR REPLACE FUNCTION sp_gente_por_area()
8 RETURNS TABLE (area VARCHAR(50), gente_asignada BIGINT) AS $$
9 BEGIN
10     RETURN QUERY
11     SELECT a.DESCRIPCION, COUNT(p.NOMBRE)
12     FROM PERSONAS p
13     JOIN AREA a ON p.AREA = a.IDAREA
14     GROUP BY a.DESCRIPCION;
15 END;
16 $$ LANGUAGE plpgsql;
17
18 -- Ejemplo de uso:
19 SELECT * FROM sp_gente_por_area();
```

mensajes Data Output Notificaciones

area	gente_asignada
TELEFONIA	4
IMPLEMENTACION	2
CAPACITACION	2
CALIDAD	2
DESARROLLO	3

Ilustración 18. PROCESO ALMACENADO