



**UNIVERSIDAD DE  
COSTA RICA**

Informática Empresarial, Sede Guanacaste  
Curso de Administración de Base de Datos IF5100

### **Caso III: Control de Acceso - Roles y Usuarios**

#### **Manual**

<b>Estudiante</b>	<b>Carné</b>
Dávila Pérez Jose Pablo	B98944
Ruiz Sánchez Minor Slatter	B97027
Segura Rojas Jose Moisés	B97425
Taisigüe Álvarez Cris	B97785
Valdelomar Rojas Rachel	B98023
Vasquez Murillo Erick	B98334

## Tabla de contenido

Introducción	3
Objetivo General	3
Objetivo Específico	4
Manual	4
Ejercicio 2	5
Inserción de datos	5
Tabla Barrio	5
Tabla Cantón	6
Tabla Color	7
Tabla Marca	8
Tabla Método de pago	9
Tabla Modelo	10
Tabla País	10
Tabla Provincia	12
Tabla Distrito	13
Tabla Tipo de licencia	14
Tabla Dirección	15
Tabla Cliente	15
Tabla Insertar Inventario de Vehículo	16
Tabla Rentas	17
Eliminar	18
Tabla Rentas	19
Tabla Inventario Vehículo	19
Tabla Cliente	20
Tabla Dirección	21
Tabla Tipo licencia	22
Tabla Barrio	24
Tabla Barrio	25
Tabla Provincia	27

Tabla Provincia	28
Tabla País	30
Tabla Método Pago	32
Tabla Modelo	32
Tabla Marca	34
Actualizar	37
Tabla Rentas	38
Tabla Inventario Vehículo	39
Tabla Cliente	41
Tabla Tipo Licencia	41
Tabla Modelo	43
Tabla Método Pago	43
Tabla Marca	45
Tabla País	47
Tabla Color	49
Tabla Distrito	50
Tabla Barrio	53
Tabla Provincia	53
Tabla Dirección	55
Ejercicio 3	56
Ejercicio 3.a	56
Ejercicio 3.b	62

### **Introducción**

El siguiente documento pretende explicar la metodología aplicada para la ejecución del tercer caso del curso de Administración de base de datos. Consigo aplicar la metodología de aprendizaje colaborativo para facilitar la repartición del ejercicio y consigo aportar al entendimiento grupal con el fin de lograr el objetivo principal. Todos los miembros del grupo deberán aportar en la elaboración de la evaluación para cumplir con este objetivo.

Se aplicará lo visto en clases, basándose en los recursos que el profesor nos ha brindado como los videos de clase y las notas de clase.

Dado que el ejercicio 1 no cuenta con un valor porcentual, se ha decidido realizar únicamente la explicación por video.

### **Objetivo General**

Explicación de la metodología aplicada para la resolución del tercer caso del curso de Administración de Base de datos, asimismo la elaboración de un manual para un mejor entendimiento.

### **Objetivo Específico**

1. Explicar el proceso las actividades de desarrollo que se realizaron para lograr con la meta.
2. Demostrar por medio del manual la ejecución del caso dos por medio de subtareas.
3. Ejemplificar por medio de imágenes los diferentes pasos a desarrollar para cada ejercicio.

## Manual

### Ejercicio 2

#### Inserción de datos


#### Tabla Barrio

Para gestionar la tabla Barrio, se deben seguir las siguientes instrucciones:

```

3  ---- [Tabla barrio ]
4  SELECT * FROM barrio;
5
6  CREATE OR REPLACE PROCEDURE InsertarBarrio(integer,varchar)
7  AS
8  $$
9      INSERT INTO barrio("id_barrio","nombre")
10     VALUES ($1, $2);
11  $$
12  LANGUAGE SQL;
13
14  CALL InsertarBarrio(2,'El carmen');
15
16  SELECT * FROM barrio;

```



	Data Output	Explain	Messages	Notifications
	id_barrio [PK] integer		nombre character varying (20)	
1	1		El carmen	

Primeramente, se ejecutan las sentencias de las líneas 6 - 12, donde *CREATE OR REPLACE PROCEDURE* son palabras reservadas, *InsertarBarrio* es el nombre del procedimiento, (*Integer, varchar*) sus respectivos parámetros, *INSERT INTO barrio ("id\_barrio","nombre")* la declaración (inserción) de la tabla con cada nombre de sus columnas y *VALUES (\$1, \$2)* se refieren a los parámetros por posición.

Seguidamente, para insertar barrios, se ejecutará la sentencia *CALL InsertarBarrio(parámetros)*, donde *CALL* es una palabra reservada, *InsertarBarrio* es el nombre del procedimiento, (*Integer, varchar*) sus respectivos parámetros. En la evidencia, se ejemplifica esto mediante 1 como identificación del barrio y "El carmen" como variable

de tipo carácter.

Finalmente, al ejecutar la sentencia *SELECT \* FROM barrio*, ocasiona un output con las columnas y los datos insertados.

### Tabla Cantón

Para gestionar la tabla Canton, se deben seguir las siguientes instrucciones:

```

24 CREATE OR REPLACE PROCEDURE InsertarCanton(integer, varchar)
25 AS
26 $$
27     INSERT INTO Canton("id_canton","nombre")
28     VALUES ($1, $2);
29 $$
30 LANGUAGE SQL;
31
32 CALL InsertarCanton(2,'Liberia');
33
34 SELECT * FROM canton;
35

```

	id_canton [PK] integer	nombre character varying (20)
1	1	Tilaran
2	2	Liberia

Primeramente, se ejecutan las sentencias de las líneas 24 - 30, donde *CREATE OR REPLACE PROCEDURE* son palabras reservadas, *InsertarCanton* es el nombre del procedimiento, *(Integer, varchar)* sus respectivos parámetros, *INSERT INTO Canton("id\_canton","nombre")* la declaración (inserción) de la tabla con cada nombre de sus columnas y *VALUES (\$1, \$2)* se refieren a los parámetros por posición.

Seguidamente, para insertar cantones, se ejecutará la sentencia *CALL InsertarCanton(parámetros)*, donde *CALL* es una palabra reservada, *InsertarCanton* es el nombre del procedimiento, *(Integer, varchar)* sus respectivos parámetros. En la evidencia, se ejemplifica esto mediante 1 como identificación del cantón y "Tilarán" como variable de tipo carácter, y 2 como identificación del cantón y "Liberia" como variable de tipo carácter.

Finalmente, al ejecutar la sentencia *SELECT \* FROM canton*, ocasiona un output

con las columnas y los datos insertados.

### Tabla Color

Para gestionar la tabla Color, se deben seguir las siguientes instrucciones:

```

38  ---- [Tabla color ]
39  SELECT * FROM color;
40
41  CREATE OR REPLACE PROCEDURE InsertarColor(integer,varchar)
42  AS
43  $$
44      INSERT INTO Color("id_color","color")
45          VALUES ($1, $2);
46  $$
47  LANGUAGE SQL;
48
49  CALL InsertarColor(2,'Rojo');
50
51  SELECT * FROM color;

```



Data Output	Explain	Messages	Notifications
	id_color [PK] integer	color character varying (20)	
1	1	Azul	
2	2	Rojo	

Primeramente, se ejecutan las sentencias de las líneas 41 - 47, donde *CREATE OR REPLACE PROCEDURE* son palabras reservadas, *InsertarColor* es el nombre del procedimiento, (*Integer*, *varchar*) sus respectivos parámetros, *INSERT INTO* ("id\_color","color") la declaración (inserción) de la tabla con cada nombre de sus columnas y *VALUES* (\$1, \$2) se refieren a los parámetros por posición.

Seguidamente, para insertar colores, se ejecutará la sentencia *CALL InsertarColor*(parámetros), donde *CALL* es una palabra reservada, *InsertarColor* es el nombre del procedimiento, (*Integer*, *varchar*) sus respectivos parámetros. En la evidencia, se ejemplifica esto mediante 1 como identificación del color y "Azul" como variable de tipo carácter, y 2 como identificación del color y "Rojo" como variable de tipo carácter.



Finalmente, al ejecutar la sentencia *SELECT \* FROM color*, ocasiona un output con las columnas y los datos insertados.


### Tabla Marca

Para gestionar la tabla Marca, se deben seguir las siguientes instrucciones:

```

55 ---- [Tabla marca]
56 SELECT * FROM marca;
57
58 CREATE OR REPLACE PROCEDURE InsertarMarca(integer,varchar)
59 AS
60 $$
61     INSERT INTO Marca("id_marca","marca")
62     VALUES ($1, $2);
63 $$
64 LANGUAGE SQL;
65
66 CALL InsertarMarca(2,'suzuki');
67
68 SELECT * FROM marca;

```



	id_marca [PK] integer	marca character varying (20)
1	1	toyota
2	2	suzuki

Primeramente, se ejecutan las sentencias de las líneas 58 - 47, donde *CREATE OR REPLACE PROCEDURE* son palabras reservadas, *InsertarMarca* es el nombre del procedimiento, *(Integer, varchar)* sus respectivos parámetros, *INSERT INTO ("id\_marca","marca")* la declaración (inserción) de la tabla con cada nombre de sus columnas y *VALUES (\$1, \$2)* se refieren a los parámetros por posición.

Seguidamente, para insertar marcas, se ejecutará la sentencia *CALL InsertarMarca(parámetros)*, donde *CALL* es una palabra reservada, *InsertarMarca* es el nombre del procedimiento, *(Integer, varchar)* sus respectivos parámetros. En la evidencia, se ejemplifica esto mediante 1 como identificación de la marca y "Toyota" como variable de tipo carácter, y 2 como identificación de la marca y "Suzuki" como variable de tipo carácter.

Finalmente, al ejecutar la sentencia *SELECT \* FROM marca*, ocasiona un output

con las columnas y los datos insertados.


### Tabla Método de pago

Para gestionar la tabla Metodo\_pago, se deben seguir las siguientes instrucciones:

```

72  ---- [Tabla metodo_pago]
73  SELECT * FROM metodo_pago;
74
75  CREATE OR REPLACE PROCEDURE InsertarMetodoPago(integer,varchar)
76  AS
77  $$
78      INSERT INTO metodo_pago("id_metodo_pago","metodo")
79      VALUES ($1, $2);
80  $$
81  LANGUAGE SQL;
82
83  CALL InsertarMetodoPago(2,'efectivo');
84
85  SELECT * FROM metodo_pago;
86

```



	Data Output	Explain	Messages	Notifications
	id_metodo_pago [PK] integer		metodo character varying (20)	
1		1	tarjeta	
2		2	efectivo	

Primeramente, se ejecutan las sentencias de las líneas 75 - 81, donde *CREATE OR REPLACE PROCEDURE* son palabras reservadas, *InsertarMetodoPago* es el nombre del procedimiento, *(Integer, varchar)* sus respectivos parámetros, *INSERT INTO ("id\_metodo\_pago","metodo")* la declaración (inserción) de la tabla con cada nombre de sus columnas y *VALUES (\$1, \$2)* se refieren a los parámetros por posición.

Seguidamente, para insertar métodos de pago, se ejecutará la sentencia *CALL InsertarMetodoPago(parámetros)*, donde *CALL* es una palabra reservada, *InsertarMetodoPago* es el nombre del procedimiento, *(Integer, varchar)* sus respectivos parámetros. En la evidencia, se ejemplifica esto mediante 1 como identificación del método de pago y "Tarjeta" como variable de tipo carácter, y 2 como identificación del método de pago y "Efectivo" como variable de tipo carácter.

Finalmente, al ejecutar la sentencia *SELECT \* FROM metodo\_pago*, ocasiona un

output con las columnas y los datos insertados.

## Tabla Modelo

Para gestionar la tabla Modelo, se deben seguir las siguientes instrucciones:

```

89  ---- [Tabla modelo]
90  SELECT * FROM modelo;
91
92  CREATE OR REPLACE PROCEDURE InsertarModelo(integer,varchar,varchar)
93  AS
94  $$
95      INSERT INTO modelo("id_modelo","modelo","year_a")
96      VALUES ($1, $2,$3);
97  $$
98  LANGUAGE SQL;
99
100 CALL InsertarModelo(2,'modelo_2','2020');
101
102 SELECT * FROM modelo;

```

Data Output	Explain	Messages	Notifications
Id_modelo [PK] integer	modelo character varying (20)	year_a character varying (20)	
1	1 modelo_1	2015	
2	2 modelo_2	2020	

Primeramente, se ejecutan las sentencias de las líneas **92 - 98**, donde *CREATE OR REPLACE PROCEDURE* son palabras reservadas, **InsertarModelo** es el nombre del procedimiento, (*Integer, varchar, varchar*) sus respectivos parámetros, *INSERT INTO* ("**id\_modelo**", "**modelo**", "**year\_a**") la declaración (inserción) de la tabla con cada nombre de sus columnas y *VALUES* (\$1, \$2, \$3) se refieren a los parámetros por posición.


## Tabla País

Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla país.

```

106 ---- [Tabla pais]
107 SELECT * FROM pais;
108
109 CREATE OR REPLACE PROCEDURE InsertarPais(integer,varchar)
110 AS
111 $$
112     INSERT INTO pais("id_pais","nombre")
113     VALUES ($1, $2);
114 $$
115 LANGUAGE SQL;
116
117 CALL InsertarPais(2,'Panama');
118
119 SELECT * FROM pais;
120

```




	Data Output	Explain	Messages	Notifications
	<div>id_pais</div> <div>[PK] integer</div>		<div>nombre</div> <div>character varying (20)</div>	
1	1	Costa Rica		
2	2	Panama		

## Tabla Provincia

Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla provincia.

```
123 ---- [Tabla provincia]
124 SELECT * FROM provincia;
125
126 CREATE OR REPLACE PROCEDURE InsertarProvincia(integer,varchar)
127 AS
128 $$
129     INSERT INTO provincia("id_provincia","nombre")
130     VALUES ($1, $2);
131 $$
132 LANGUAGE SQL;
133
134 CALL InsertarProvincia(2,'Alajuela');
135
136 SELECT * FROM provincia;
137
```

Data Output		Explain	Messages	Notifications
	id_provincia [PK] integer		nombre character varying (20)	
1	1		Guancaste	
2	2		Alajuela	



## Tabla Distrito

Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla distrito.

```

140 ---- [Tabla distrito]
141 SELECT * FROM distrito;
142
143 CREATE OR REPLACE PROCEDURE InsertarDistrito(integer,varchar)
144 AS
145 $$
146     INSERT INTO distrito("id_distrito","nombre")
147     VALUES ($1, $2);
148 $$
149 LANGUAGE SQL;
150
151 CALL InsertarDistrito(2,'San Carlos');
152
153 SELECT * FROM distrito;

```

Data Output

Explain

Messages

Notifications

	id_distrito [PK] integer	nombre character varying (20)
1	1	liberia
2	2	San Carlos

## Tabla Tipo de licencia

Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla tipo de licencia.

```

140 ---- [Tabla tipo licencia]
141 SELECT * FROM tipo_licencia;
142
143 CREATE OR REPLACE PROCEDURE InsertarTipoLicencia(integer,varchar)
144 AS
145 $$
146     INSERT INTO tipo_licencia("id_tipo_licencia","tipo")
147     VALUES ($1, $2);
148 $$
149 LANGUAGE SQL;
150
151 CALL InsertarTipoLicencia(2,'B2');
152
153 SELECT * FROM tipo_licencia;

```



	Data Output	Explain	Messages	Notifications
	id_tipo_licencia [PK] integer		tipo character varying (2)	
1	1		A1	
2	2		B2	


## Tabla Dirección

Para gestionar la tabla se deben seguir las mismas instrucciones que hemos realizado con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla dirección.

```

175 ---- [Tabla tipo direccion]
176 SELECT * FROM direccion;
177
178 CREATE OR REPLACE PROCEDURE InsertarDireccion(integer,integer,integer,integer,integer,integer)
179 AS
180 $$
181     INSERT INTO direccion("id_direccion","id_pais_fk","id_provincia_fk","id_canton_fk","id_distrito_
182     VALUES ($1, $2,$3,$4,$5,$6);
183 $$
184 LANGUAGE SQL;
185
186 CALL InsertarDireccion(2,2,2,2,2,2);
187
188 SELECT * FROM Direccion;
189

```



	id_direccion [PK] integer	id_pais_fk integer	id_provincia_fk integer	id_canton_fk integer	id_distrito_fk integer	id_barrio_fk integer
1		1	1	1	1	1
2		2	2	2	2	2


## Tabla Cliente

Para gestionar la tabla se deben seguir las mismas instrucciones que hemos realizado con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla cliente.

```

392 ---- [Tabla Cliente]
393 SELECT * FROM Cliente;
394
395 CREATE OR REPLACE PROCEDURE InsertarCliente(varchar,varchar, integer, integer, varchar, integer,
396 AS
397 $$
398     INSERT INTO cliente("cedula","nombre","id_direccion_fk","telefono","mail","edad","id_tipo_lic
399     VALUES ($1,$2,$3,$4,$5,$6,$7,$8,$9);
400 $$
401 LANGUAGE SQL;
402
403 CALL InsertarCliente('583482894','Maria',2,95738186,'Maria@gmail.com',22,2,'2021/11/15',3);
404
405 SELECT * FROM Cliente;
406

```



	cedula [PK] character varying (9)	nombre character varying (25)	id_direccion_fk integer	telefono integer	mail character varying (45)	edad integer	id_tipo_licencia_fk integer	vincimiento_licencia date
1	584340211	Moses		84366119	moses@gmail.com	20		1 2021-12-01
2	583482894	Maria		95738186	Maria@gmail.com	22		2 2021-11-15




## Tabla Insertar Inventario de Vehículo

Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla inventario vehículo.

```

211
212 CREATE OR REPLACE PROCEDURE InsertarInventarioVehiculos(varchar, integer, integer, integer, varchar)
213 AS
214 $$
215     INSERT INTO inventario_vehiculos("placa", "id_marca_fk", "id_modelo_fk", "id_color", "estado")
216     VALUES ($1, $2, $3, $4, $5);
217 $$
218 LANGUAGE SQL;
219
220 CALL InsertarInventarioVehiculos('A900200', 2, 2, 2, 'ESTADO_2');
221
222 SELECT * FROM inventario_vehiculos;
223
224
225

```



The image shows a SQL editor interface with a code editor and a data output table. The code editor contains SQL code for creating a procedure and calling it. The data output table shows the results of the procedure call, with two rows of data.

	placa [PK] character varying (20)	id_marca_fk integer	id_modelo_fk integer	id_color integer	estado character varying (200)
1	500B500	1	1	1	ESTADO_1
2	A900200	2	2	2	ESTADO_2

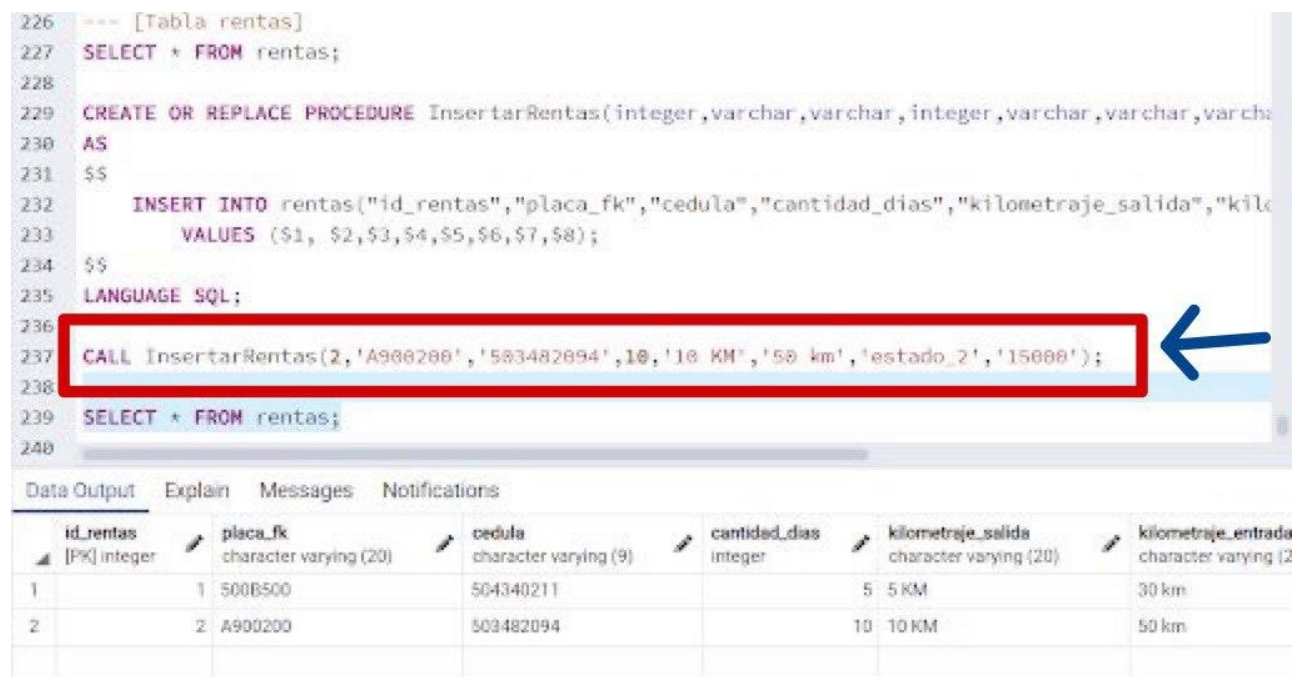
## Tabla Rentas

Para gestionar la tabla Rentas, se deben seguir las siguientes instrucciones:

```

226 --- [Tabla rentas]
227 SELECT * FROM rentas;
228
229 CREATE OR REPLACE PROCEDURE InsertarRentas(integer,varchar,varchar, integer,varchar,varchar,varcha
230 AS
231 $$
232     INSERT INTO rentas("id_rentas","placa_fk","cedula","cantidad_dias","kilometraje_salida","kilo
233     VALUES ($1, $2,$3,$4,$5,$6,$7,$8);
234 $$
235 LANGUAGE SQL;
236
237 CALL InsertarRentas(2,'A900200','503482094',10,'10 KM','50 km','estado_2','15000');
238
239 SELECT * FROM rentas;
240

```



id_rentas [PK] integer	placa_fk character varying (20)	cedula character varying (9)	cantidad_dias integer	kilometraje_salida character varying (20)	kilometraje_entrada character varying (20)
1	1 500B500	504340211	5	5 KM	30 km
2	2 A900200	503482094	10	10 KM	50 km

Primeramente, se ejecutan las sentencias de las líneas **229 - 235**, donde **CREATE OR REPLACE PROCEDURE** son palabras reservadas, **InsertarMetodoPago** es el nombre del procedimiento, *(Integer, varchar)* sus respectivos parámetros, **INSERT INTO ("id\_metodo\_pago","metodo")** la declaración (inserción) de la tabla con cada nombre de sus columnas y **VALUES (\$1, \$2)** se refieren a los parámetros por posición.

Seguidamente, para insertar **métodos de pago**, se ejecutará la sentencia **CALL InsertarMetodoPago(parámetros)**, donde **CALL** es una palabra reservada, **InsertarMetodoPago** es el nombre del procedimiento, *(Integer, varchar)* sus respectivos parámetros. En la evidencia, se ejemplifica esto mediante 1 como identificación del **método de pago** y "Tarjeta" como variable de tipo carácter, y 2 como identificación del **método de pago** y "Efectivo" como variable de tipo carácter.

Finalmente, al ejecutar la sentencia **SELECT \* FROM metodo\_pago**, ocasiona un output con las columnas y los datos insertados.

## Eliminar

En términos generales, la sintaxis es:

```
CREATE OR REPLACE PROCEDURE nombre_del_procedimiento(parámetros)
```

```
AS
```

```
DELETE FROM nombre_de_la_tabla
```

```
WHERE criterio = posición_del_criterio;
```

```
LANGUAGE SQL;
```

```
CALL nombre_del_procedimiento(valor_específico_del_registro);
```

```
SELECT * FROM nombre_de_la_tabla;
```

Primeramente, se ejecutan las sentencias de las líneas **CREATE OR REPLACE PROCEDURE ... LANGUAGE SQL**.

En la sintaxis anterior, **CREATE OR REPLACE PROCEDURE** son palabras reservadas, el nombre del procedimiento será **Eliminar[tabla]** con el propósito de facilitar la coherencia e intuición de los programadores, los parámetros estarán estrictamente vinculados al tipo de datos que se gestionen (**integer, varying character, date, entre otros**), **AS** es una palabra reservada, **DELETE FROM [tabla]** será la declaración para eliminar registros, **WHERE** es una palabra reservada, **criterio = posición\_del\_criterio** será la condición explícita para especificar el registro a eliminar y **LANGUAGE SQL** es una palabra reservada.

Así bien, en la sentencia de llamada, **CALL** es una palabra reservada, el nombre del procedimiento será **Eliminar[tabla]** con el propósito de facilitar la coherencia e intuición de los programadores y (**valor\_específico\_del\_registro**) será un valor explícito para seleccionar pertinentemente el registro.

Finalmente, la sentencia **SELECT \* FROM nombre\_de\_la\_tabla**, ocasiona un output con las columnas y los datos, donde no aparece el registro eliminado.

## Tabla Rentas

Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado en la explicación general, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla rentas.

```

226 --- [Tabla rentas]
227 SELECT * FROM rentas;
228
229 CREATE OR REPLACE PROCEDURE EliminarRentas(integer)
230 AS
231 $$
232     DELETE FROM rentas
233     WHERE id_rentas = 1;
234 $$
235 LANGUAGE SQL;
236
237 CALL EliminarRentas(1);
238
239 SELECT * FROM rentas;
240

```

	id_rentas [PK] integer	placa_fk character varying (20)	cedula character varying (9)	cantidad_dias integer	kilometraje_salida character varying (20)	kilometraje_entrada character varying (20)	estado character varying (200)
1	2	A900200	503482094	10	10 KM	50 km	estado_2
2	3	B900200	503482002	20	20 KM	150 km	Buen estado(Estado_3)
3	1	500B500	103482002	20	20 KM	150 km	Buen estado(Estado_3)

```

226 --- [Tabla rentas]
227 SELECT * FROM rentas;
228
229 CREATE OR REPLACE PROCEDURE EliminarRentas(integer)
230 AS
231 $$
232     DELETE FROM rentas
233     WHERE id_rentas = 1;
234 $$
235 LANGUAGE SQL;
236
237 CALL EliminarRentas(1);
238
239 SELECT * FROM rentas;
240

```

	id_rentas [PK] integer	placa_fk character varying (20)	cedula character varying (9)	cantidad_dias integer	kilometraje_salida character varying (20)	kilometraje_entrada character varying (20)	estado character varying (200)
1	2	A900200	503482094	10	10 KM	50 km	estado_2
2	3	B900200	503482002	20	20 KM	150 km	Buen estado(Estado_3)

## Tabla Inventario Vehículo

Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado

```

210 SELECT * FROM inventario_vehiculos;
211
212 CREATE OR REPLACE PROCEDURE EliminaInventarioVehiculos(varchar)
213 AS
214 $$
215     DELETE FROM inventario_vehiculos
216     WHERE placa = $1;
217 $$
218 LANGUAGE SQL;
219
220 CALL EliminaInventarioVehiculos('B900200');
221
222 SELECT * FROM inventario_vehiculos;
223
224 -----
225
226 --- [Tabla rentas]
227

```

Data Output Explain Messages Notifications

	placa [PK] character varying (20)	id_marca_fk integer	id_modelo_fk integer	id_color integer	estado character varying (200)	
1	500B500	1	1	1	ESTADO_1	
2	A900200	2	2	2	ESTADO_2	

en la explicación general, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla inventario.

## Tabla Cliente

Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado

```

193 SELECT * FROM cliente;
194
195 CREATE OR REPLACE PROCEDURE EliminarCliente(varchar)
196 AS
197 $$
198     DELETE FROM cliente
199     WHERE cedula = $1;
200 $$
201 LANGUAGE SQL;
202
203 CALL EliminarCliente('503482094');
204
205 SELECT * FROM Cliente;
206
207 -----

```

Data Output Explain Messages Notifications

	cedula [PK] character varying (9)	nombre character varying (20)	id_direccion_fk integer	telefono integer	mail character varying (45)	edad integer	id_tipo_licencia_fk integer	vencimiento_licer date
1	503482002	Juan	3	2430186	Juan@gmail.com	25	3	2021-10-15
2	103482002	Messi	1	2430456	Messi@gmail.com	35	2	2021-10-10

en la explicación general, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla cliente.

## Tabla Dirección

Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado

```

174
175 ---- [Tabla tipo direccion]
176 SELECT * FROM direccion;
177
178 CREATE OR REPLACE PROCEDURE EliminarDireccion(integer)
179 AS
180 $$
181     DELETE FROM direccion
182     WHERE id_direccion = $1;
183 $$
184 LANGUAGE SQL;
185
186 CALL EliminarDireccion(1);
187
188 SELECT * FROM Direccion;

```

	id_direccion [PK] integer	id_pais_fk integer	id_provincia_fk integer	id_canton_fk integer	id_distrito_fk integer	id_barrio_fk integer
1		2	2	2	2	2
2		3	3	3	3	3

en la explicación general, simplemente se modifica la tabla a modificar y sus parámetros.

En nuestro caso es la tabla dirección.



## Tabla Tipo licencia

Para gestionar la tabla, se deben seguir las mismas instrucciones qué hemos realizado

```

158 ---- [Tabla tipo licencia]
159 SELECT * FROM tipo_licencia;
160
161 CREATE OR REPLACE PROCEDURE EliminarTipoLicencia(integer)
162 AS
163 $$
164     DELETE FROM tipo_licencia
165     WHERE id_tipo_licencia = $1;
166 $$
167 LANGUAGE SQL;
168
169 CALL EliminarTipoLicencia(4);
170
171 SELECT * FROM tipo_licencia;
172
173 -----
174

```

Data Output	Explain	Messages	Notifications
 <b>id_tipo_licencia</b> [PK] integer	 <b>tipo</b> character varying (2)		
1	1	A1	
2	2	B2	
3	3	B2	

en la explicación general, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla licencia.



**Tabla Barrio**

Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado en la explicación general, simplemente se modifica la tabla a modificar y sus parámetros.




En nuestro caso es la tabla barrio.

```

2  ---- [tabla barrio ]
3  SELECT * FROM barrio;
4
5  CREATE OR REPLACE PROCEDURE EliminarBarrio(integer,varchar)
6  AS
7  $$
8      DELETE FROM barrio;
9          WHERE id_barrio = $1;
10 $$
11 LANGUAGE SQL;
12
13 CALL EliminarBarrio(1,);
14
15 SELECT * FROM barrio;
16
17
18

```

[Data Output](#) [Explain](#) [Messages](#) [Notifications](#)

	 id_barrio [PK] integer 	nombre character varying (20) 	
1	2	invu	
2	3	Santa Cecilia	
3	1	Fatima	

## Tabla Barrio




Para gestionar la tabla, se deben seguir las mismas instrucciones qué hemos realizado en la explicación general, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla barrio.

```

1
2  ---- [Tabla barrio ]
3  SELECT * FROM barrio;
4
5  CREATE OR REPLACE PROCEDURE EliminarBarrio(integer,varchar)
6  AS
7  $$
8      DELETE FROM barrio
9          WHERE id_barrio = $1;
10 $$
11 LANGUAGE SQL;
12
13 CALL EliminarBarrio(1);
14
15 SELECT * FROM barrio;
16
17
18

```

Data Output   Explain   Messages   Notifications

	 id_barrio [PK] integer 	nombre character varying (20) 
1	2 invu	
2	3 Santa Cecilia	

## Tabla Provincia




Para gestionar la tabla, se deben seguir las mismas instrucciones qué hemos realizado

```

123 ---- [Tabla provincia]
124 SELECT * FROM provincia;
125
126 CREATE OR REPLACE PROCEDURE EliminarProvincia(integer)
127 AS
128 $$
129     DELETE FROM provincia
130         WHERE id_provincia = $1;
131 $$
132 LANGUAGE SQL;
133
134 CALL EliminarProvincia(1);
135
136 SELECT * FROM provincia;
137
138 -----

```

Data Output Explain Messages Notifications

	 id_provincia [PK] integer		nombre character varying (20)	
1	2	Alajuela		
2	3	Heredia		
3	1	San Jose		

en la explicación general, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla provincia.

**Tabla Provincia**

Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado en la explicación general, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla provincia.

```

122
123 ---- [Tabla provincia]
124 SELECT * FROM provincia;
125
126 CREATE OR REPLACE PROCEDURE EliminarProvincia(integer)
127 AS
128 $$
129     DELETE FROM provincia
130         WHERE id_provincia = $1;
131 $$
132 LANGUAGE SQL;
133
134 CALL EliminarProvincia(1);
135
136 SELECT * FROM provincia;
137
138 -----

```

Data Output   Explain   Messages   Notifications

	id_provincia [PK] integer		nombre character varying (20)	
1		2	Alajuela	
2		3	Heredia	

## Tabla País

Para gestionar la tabla, se deben seguir las mismas instrucciones qué hemos realizado en la explicación general, simplemente se modifica la tabla a modificar y sus parámetros.




En nuestro caso es la tabla país.

```

100      [tabla país]
107  SELECT * FROM país;
108
109  CREATE OR REPLACE PROCEDURE EliminarPais(integer,varchar)
110  AS
111  $$
112      DELETE FROM país
113          WHERE id_pais = $1;
114  $$
115  LANGUAGE SQL;
116
117  CALL EliminarPais(3);
118
119  SELECT * FROM país;

```

Data Output   Explain   Messages   Notifications

	 id_pais [PK] integer 	nombre character varying (20) 	
1	1	Costa Rica	
2	2	Panama	
3	3	Italia	

```

107 SELECT * FROM pais;
108
109 CREATE OR REPLACE PROCEDURE EliminarPais(integer)
110 AS
111 $$
112     DELETE FROM pais
113     WHERE id_pais = $1;
114 $$
115 LANGUAGE SQL;
116
117 CALL EliminarPais(3);
118
119 SELECT * FROM pais;
120
121 -----
122
123 ---- [Tabla provincia]

```

Data Output   Explain   Messages   Notifications

	id_pais [PK] integer		nombre character varying (20)	
1		1	Costa Rica	
2		2	Panama	



## Tabla Método Pago

Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado

```

72 ---- [Tabla metodo_pago]
73 SELECT * FROM metodo_pago;
74
75 CREATE OR REPLACE PROCEDURE EliminarMetodoPago(integer)
76 AS
77 $$
78     DELETE FROM metodo_pago
79         WHERE id_metodo_pago = $1;
80 $$
81 LANGUAGE SQL;
82
83 CALL EliminarMetodoPago(2;
84
85 SELECT * FROM metodo_pago;

```

Data Output Explain Messages Notifications

	id_metodo_pago [PK] integer		metodo character varying (20)	
1	1		tarjeta	
2	2		efectivo	

en la explicación general, simplemente se modifica la tabla a modificar y sus parámetros.

En nuestro caso es la tabla rentas.

## Tabla Modelo





Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado

```

89  ---- [Tabla modelo]
90  SELECT * FROM modelo;
91
92  CREATE OR REPLACE PROCEDURE EliminarModelo(integer)
93  AS
94  $$
95      DELETE FROM modelo
96          WHERE id_modelo = $1;
97  $$
98  LANGUAGE SQL;
99
100 CALL EliminarModelo(3);
101
102 SELECT * FROM modelo;
103
104 -----

```

Data Output   Explain   Messages   Notifications

		<b>id_modelo</b> [PK] integer 	<b>modelo</b> character varying (20) 	<b>year_a</b> character varying (20) 	
1		1	modelo_1	2015	
2		2	modelo_2	2020	

en la explicación general, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla modelo.

**Tabla Marca**

Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado en la explicación general, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla marca.

```

55 ----- [Tabla  marca]
56 SELECT * FROM marca;
57
58 CREATE OR REPLACE PROCEDURE EliminarMarca(integer)
59 AS
60 $$
61     DELETE FROM marca
62         WHERE id_marca = $1;
63 $$
64 LANGUAGE SQL;
65
66 CALL EliminarMarca(3);
67
68 SELECT * FROM marca;
69
70 -----

```

Data Output Explain Messages Notifications

	id_marca [PK] integer		marca character varying (20)	
1		1	toyota	
2		2	suzuki	
3		3	Ferrati	

```

55 ----- [Tabla  marca]
56 SELECT * FROM marca;
57
58 CREATE OR REPLACE PROCEDURE EliminarMarca(integer)
59 AS
60 $$
61     DELETE FROM marca
62         WHERE id_marca = $1;
63 $$
64 LANGUAGE SQL;
65
66 CALL EliminarMarca(3);
67
68 SELECT * FROM marca;
69
70 -----

```

Data Output   Explain   Messages   Notifications

	id_marca [PK] integer		marca character varying (20)	
1		1	toyota	
2		2	suzuki	

## Actualizar

En términos generales, la sintaxis es:

```
CREATE OR REPLACE PROCEDURE nombre_del_procedimiento(parámetros)
AS
```

```
UPDATE nombre_de_la_tabla
SET criterio = posición_del_criterio
WHERE criterio = posición_del_criterio;
```

```
LANGUAGE SQL;
```

```
CALL nombre_del_procedimiento(parámetros);
SELECT * FROM nombre_de_la_tabla;
```

Primeramente, se ejecutan las sentencias de las líneas CREATE OR REPLACE PROCEDURE ... LANGUAGE SQL.

En la sintaxis anterior, **CREATE OR REPLACE PROCEDURE** son palabras reservadas, el nombre del procedimiento será **Actualizar[tabela]** con el propósito de facilitar la coherencia e intuición de los programadores, los parámetros estarán estrictamente vinculados al tipo de datos que se gestionen (**integer, varying character, date, entre otros**), **AS** es una palabra reservada, **UPDATE [tabela]** será la declaración para actualizar registros, **SET** es una palabra reservada, **criterio = posición\_del\_criterio** será la condición explícita para especificar el registro a actualizar, **WHERE** es una palabra reservada, **criterio = posición\_del\_criterio** será la condición explícita para especificar el registro a actualizar y **LANGUAGE SQL** es una palabra reservada.

Así bien, en la sentencia de llamada, **CALL** es una palabra reservada, el nombre del procedimiento será **Actualizar[tabela]** con el propósito de facilitar la coherencia e intuición de los programadores y (**parámetros**) serán valores explícitos para seleccionar pertinentemente el registro.

Finalmente, la sentencia **SELECT \* FROM nombre\_de\_la\_tabla**, ocasiona un output con las columnas y los datos, donde aparece el registro actualizado.

## Tabla Rentas

Para gestionar la tabla, se deben seguir las mismas instrucciones qué hemos realizado con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla inventario rentas.

```

227 --- [Tabla rentas]
228 SELECT * FROM rentas;
229
230 CREATE OR REPLACE PROCEDURE ActualizarRentas(integer)
231 AS
232 $$
233     UPDATE rentas
234     SET placa_fk = '500B500'
235     WHERE id_rentas = 1;
236 $$
237 LANGUAGE SQL;
238
239 CALL ActualizarRentas(1);
240
241 SELECT * FROM rentas;
242
243

```

Data Output Explain Messages Notifications

	id_rentas [PK] integer	placa_fk character varying (20)	cedula character varying (9)	cantidad_dias integer	kilometraje_salida character varying (20)	kilometraje_entrada character varying (20)	estado character varying (200)
1	1	A900200	503482094	10	10 KM	50 km	estado_2

```

227 --- [Tabla rentas]
228 SELECT * FROM rentas;
229
230 CREATE OR REPLACE PROCEDURE ActualizarRentas(integer,varchar)
231 AS
232 $$
233     UPDATE rentas
234     SET placa_fk = $2
235     WHERE id_rentas = $1;
236 $$
237 LANGUAGE SQL;
238
239 CALL ActualizarRentas(1,'500B500');
240
241 SELECT * FROM rentas;
242
243

```

Data Output Explain Messages Notifications

	id_rentas [PK] integer	placa_fk character varying (20)	cedula character varying (9)	cantidad_dias integer	kilometraje_salida character varying (20)	kilometraje_entrada character varying (20)	estado character varying (200)
1	1	A900200	503482094	10	10 KM	50 km	estado_2

## Tabla Inventario Vehículo

Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla inventario vehículo.

```

227 --- [Tabla rentas]
228 SELECT * FROM rentas;
229
230 CREATE OR REPLACE PROCEDURE ActualizarRentas(integer,varchar)
231 AS
232 $$
233     UPDATE rentas
234     SET placa_fk = $2
235     WHERE id_rentas = $1;
236 $$
237 LANGUAGE SQL;
238
239 CALL ActualizarRentas(1,'500B500');
240
241 SELECT * FROM rentas;
242
243

```

Data Output Explain Messages Notifications

	id_rentas [PK] integer	placa_fk character varying (20)	cedula character varying (9)	cantidad_dias integer	kilometraje_salida character varying (20)	kilometraje_entrada character varying (20)	estado character varying (200)
1	1	500B500	503482094	10	10 KM	50 km	estado_2

```

210 ---- [Tabla inventario_vehiculos]
211 SELECT * FROM inventario_vehiculos;
212
213 CREATE OR REPLACE PROCEDURE ActualizarInventarioVehiculos(character,integer)
214 AS

```



```

210 ---- [Tabla inventario_vehiculos]
211 SELECT * FROM inventario_vehiculos;
212
213 CREATE OR REPLACE PROCEDURE ActualizarInventarioVehiculos(character, integer)
214 AS
215 $$
216     UPDATE inventario_vehiculos
217     SET id_marca_fk = $2
218     WHERE placa = $1;
219 $$
220 LANGUAGE SQL;
221
222 CALL ActualizarInventarioVehiculos('500B500',2);
223
224 SELECT * FROM inventario_vehiculos;
225

```

Data Output Explain Messages Notifications

	placa [PK] character varying (20)	id_marca_fk integer	id_modelo_fk integer	id_color integer	estado character varying (200)
1	A900200	2	2	2	ESTADO_2
2	500B500	2	1	1	ESTADO_1

## Tabla Cliente

Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla cliente.

```

195 CREATE OR REPLACE PROCEDURE Actualizarliente(varchar,integer)
196 AS
197 $$
198     UPDATE cliente
199     SET edad = $2
200     WHERE cedula = $1;
201 $$
202 LANGUAGE SQL;
203
204 CALL Actualizarliente('503482094',25);
205
206
207 SELECT * FROM Cliente;
208
209 -----
210
211 ----- [Tabla_inventario_vehiculos]

```

Data Output Explain Messages Notifications

	cedula [PK] character varying (9)	nombre character varying (20)	id_direccion_fk integer	telefono integer	mail character varying (45)	edad integer	id_tipo_licencia_fk integer	vencimiento_licer date
1	503482094	Juan	2	875575777	juan@gmail.com	22	1	2021-08-11

## Tabla Tipo Licencia

Para gestionar la tabla, se deben seguir las mismas instrucciones qué hemos realizado

```

157
158 ---- [Tabla tipo licencia]
159 SELECT * FROM tipo_licencia;
160
161 CREATE OR REPLACE PROCEDURE ActualizarTipoLicencia(integer,character)
162 AS
163 $$
164     UPDATE tipo_licencia
165     SET tipo = $2
166     WHERE id_tipo_licencia = $1;
167 $$
168 LANGUAGE SQL;
169
170 CALL ActualizarTipoLicencia('A2');
```

Data Output				Explain	Messages	Notifications
	id_tipo_licencia [PK] integer		tipo character varying (2)			
1		1	A1			
2		2	B2			
3		3	B2			

con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla tipo licencia.

## Tabla Modelo

Para gestionar la tabla, se deben seguir las mismas instrucciones qué hemos realizado con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla Modelo.

```

92 CREATE OR REPLACE PROCEDURE ActualizarModelo(integer,character)
93 AS
94 $$
95     UPDATE modelo
96     SET year_a = $2
97     WHERE id_modelo = $1;
98 $$
99 LANGUAGE SQL;
100
101 CALL ActualizarModelo(1,'2009');
102
103 SELECT * FROM modelo;
104
105 -----
106
107 ---- [Tabla pais]
108 SELECT * FROM pais;

```

Data Output Explain Messages Notifications

	 id_modelo [PK] integer 	modelo character varying (20) 	year_a character varying (20) 	
1	1	modelo_1	2015	
2	2	modelo_2	2020	

## Tabla Método Pago

Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado

```

73 SELECT * FROM metodo_pago;
74
75 CREATE OR REPLACE PROCEDURE ActualizarMetodoPago(integer,character)
76 AS
77 $$
78     UPDATE metodo_pago
79     SET metodo = $2
80     WHERE id_metodo_pago = $1;
81 $$
82 LANGUAGE SQL;
83
84 CALL ActualizarMetodoPago(1,'efectivo');
85
86 SELECT * FROM metodo_pago;
87
88 -----

```

Data Output		Explain	Messages	Notifications
	id_metodo_pago [PK] integer		metodo character varying (20)	
1	1		efectivo	

con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla cmetodo pago.

**Tabla Marca**

Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla marca.

```

54
55 ---- [Tabla  marca]
56 SELECT * FROM marca;
57
58 CREATE OR REPLACE PROCEDURE ActuaizarMarca(integer,varchar)
59 AS
60 $$
61     UPDATE marca
62     SET marca = $2
63     WHERE id_marca = $1;
64 $$
65 LANGUAGE SQL;
66
67 CALL ActuaizarMarca(1,'FORD');
68
69 SELECT * FROM marca;

```

Data Output Explain Messages Notifications

	id_marca [PK] integer		marca character varying (20)	
1		2	suzuki	
2		1	FORD	

## Tabla País




Para gestionar la tabla, se deben seguir las mismas instrucciones qué hemos realizado con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla país.

```

110 SELECT * FROM país;
111
112 CREATE OR REPLACE PROCEDURE ActualizarPais(integer,character)
113 AS
114 $$
115     UPDATE país
116     SET nombre = $2
117     WHERE id_pais = $1;
118 $$
119 LANGUAGE SQL;
120
121 CALL ActualizarPais(1,'España');
122
123 SELECT * FROM país;
124
125 -----
126
127 ---- [Tabla provincial]

```

Data Output Explain Messages Notifications

	 id_pais [PK] integer 	nombre character varying (20) 	
1	1	Costa Rica	
2	2	Panama	



```

110 SELECT * FROM pais;
111
112 CREATE OR REPLACE PROCEDURE ActualizarPais(integer,character)
113 AS
114 $$
115     UPDATE pais
116     SET nombre = $2
117     WHERE id_pais = $1;
118 $$
119 LANGUAGE SQL;
120
121 CALL ActualizarPais(1,'España');
122
123 SELECT * FROM pais;
124
125 -----
126
127 ---- [Tabla provincial]

```

Data Output Explain Messages Notifications

	id_pais [PK] integer		nombre character varying (20)	
1	2		Panama	
2	1		España	

## Tabla Color

Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado

```

39 SELECT * FROM color;
40
41 CREATE OR REPLACE PROCEDURE ActualizarColor(integer,varchar)
42 AS
43 $$
44     UPDATE color
45     SET color = $2
46     WHERE id_color = $1;
47 $$
48 LANGUAGE SQL;
49
50 CALL ActualizarColor(2,'negro');
51
52 SELECT * FROM color;
53
54 -----

```

Data Output		Explain	Messages	Notifications
	id_color [PK] integer		color character varying (20)	
1	1		Azul	
2	2		negro	

con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla color.

**Tabla Distrito**




Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla distrito.

```

147 ---- [Tabla distrito]
148 SELECT * FROM distrito;
149
150 CREATE OR REPLACE PROCEDURE ActualizarDistrito(integer,varchar)
151 AS
152 $$
153     UPDATE distrito
154     SET nombre = $2
155     WHERE id_distrito = $1;
156 $$
157 LANGUAGE SQL;
158
159 CALL ActualizarDistrito(1,'San Lupe');
160
161 SELECT * FROM distrito;
162

```

Data Output Explain Messages Notifications

	 id_distrito [PK] integer 	nombre character varying (20) 	
1	2	San Carlos	
2	1	San Lupe	

```

147 ---- [Tabla distrito]
148 SELECT * FROM distrito;
149
150 CREATE OR REPLACE PROCEDURE ActualizarDistrito(integer,varchar)
151 AS
152 $$
153     UPDATE distrito
154     SET nombre = $2
155     WHERE id_distrito = $1;
156 $$
157 LANGUAGE SQL;
158
159 CALL ActualizarDistrito(1,'San Gorge');
160
161 SELECT * FROM distrito;
162

```

Data Output

Explain

Messages

Notifications

	id_distrito [PK] integer	nombre character varying (20)
1	2	San Carlos
2	1	San Gorge

## Tabla Barrio

Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla barrio.

```

1
2 ---- [Tabla barrio ]
3 SELECT * FROM barrio;
4
5 CREATE OR REPLACE PROCEDURE ActulizarBarrio(integer,varchar)
6 AS
7 $$
8     UPDATE barrio
9     SET nombre = $2
10    WHERE id_barrio = $1;
11 $$
12 LANGUAGE SQL;
13
14 CALL ActulizarBarrio(1,'Lupes');
15
16 SELECT * FROM barrio;
17
18

```




Data Output		Explain	Messages	Notifications
	id_barrio [PK] integer		nombre character varying (20)	
1	2		invu	
2	3		Santa Cecilia	

```

3 SELECT * FROM barrio;
4
5 CREATE OR REPLACE PROCEDURE ActulizarBarrio(integer,varchar)
6 AS
7 $$
8     UPDATE barrio
9     SET nombre = $2
10    WHERE id_barrio = $1;
11 $$
12 LANGUAGE SQL;
13
14 CALL ActulizarBarrio(2,'Lupes');
15
16 SELECT * FROM barrio;
17
18

```

Data Output Explain Messages Notifications

	 id_barrio [PK] integer 	nombre character varying (20) 	
1	3	Santa Cecilia	
2	2	Lupes	

## Tabla Provincia




Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla provincia.

```

131 SELECT * FROM provincia;
132
133 CREATE OR REPLACE PROCEDURE ActulizarProvincia(integer,varchar)
134 AS
135 $$
136     UPDATE provincia
137     SET nombre = $2
138     WHERE id_provincia = $1;
139 $$
140 LANGUAGE SQL;
141
142 CALL ActulizarProvincia(2,'Puntarenas');
143
144 SELECT * FROM provincia;
145
146 -----
147

```

Data Output Explain Messages Notifications

	 id_provincia [PK] integer 	nombre character varying (20) 	
1	2	Alajuela	
2	3	Heredia	



## Tabla Dirección








Para gestionar la tabla, se deben seguir las mismas instrucciones que hemos realizado con, simplemente se modifica la tabla a modificar y sus parámetros. En nuestro caso es la tabla dirección.

```

184
185 ---- [Tabla tipo direccion]
186 SELECT * FROM direccion;
187
188 CREATE OR REPLACE PROCEDURE ActualizarDireccion(integer,integer)
189 AS
190 $$
191     UPDATE direccion
192     SET id_direccion = $2
193     WHERE id_direccion = $1;
194 $$
195 LANGUAGE SQL;
196
197 CALL ActualizarDireccion(2,1);
198
199 SELECT * FROM direccion;
200

```

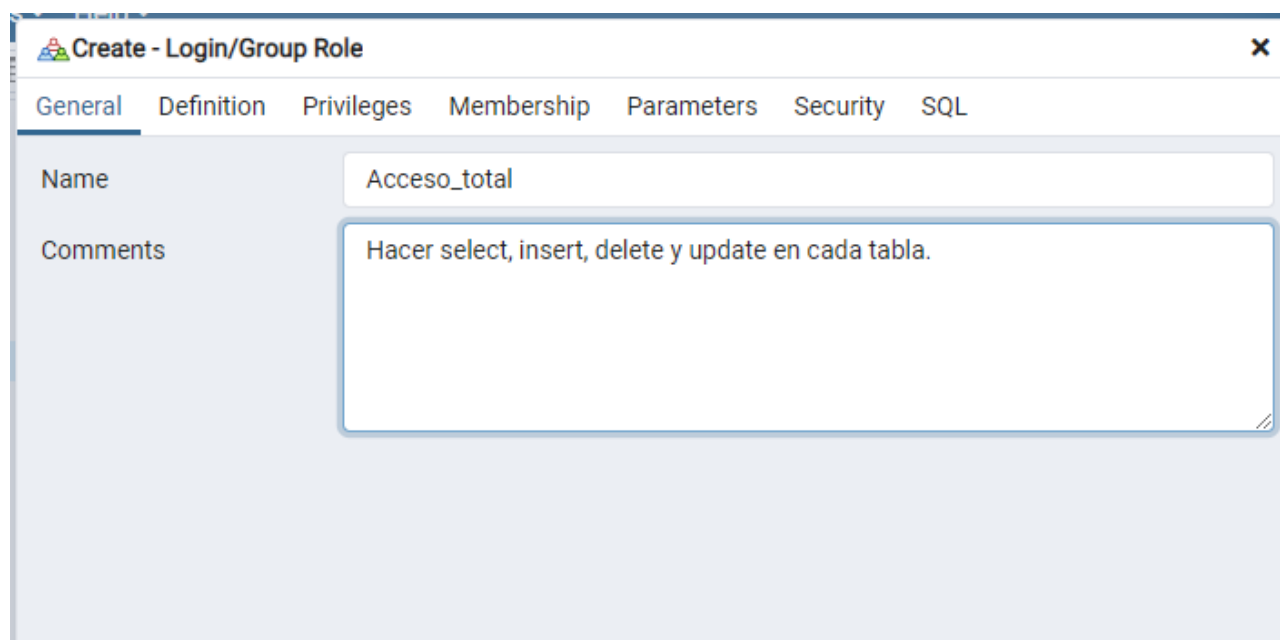
Data Output Explain Messages Notifications

	 id_direccion [PK] integer 	id_pais_fk integer 	id_provincia_fk integer 	id_canton_fk integer 	id_distrito_fk integer 	id_barrio_fk integer 	
1	1	2	2	2	2	2	

## Ejercicio 3

### Ejercicio 3.a

1. Se crean los roles en el apartado roles de login. Se crea el rol de Acceso\_total y con la descripción que muestra en la primera imagen



**Create - Login/Group Role**

General Definition Privileges Membership Parameters Security SQL

Name: Acceso\_total

Comments: Hacer select, insert, delete y update en cada tabla.

2. Para darle los permisos solicitados al rol de Acceso total seleccionamos la tabla las que le daremos el permiso en este caso sería a todas las tablas de esta base de datos



**Asistente de Permisos - Selección de Objetos (paso 1 de 3)**

Please select the objects to grant privileges to from the list below.

Search

<input checked="" type="checkbox"/>	Tipo de Objeto	Esquema	Nombre
<input checked="" type="checkbox"/>	Table	public	barrio
<input checked="" type="checkbox"/>	Table	public	canton
<input checked="" type="checkbox"/>	Table	public	cliente
<input checked="" type="checkbox"/>	Table	public	color
<input checked="" type="checkbox"/>	Table	public	direccion
<input checked="" type="checkbox"/>	Table	public	distrito
<input checked="" type="checkbox"/>	Table	public	inventario_vehiculos
<input checked="" type="checkbox"/>	Table	public	marca
<input checked="" type="checkbox"/>	Table	public	metodo_pago
<input checked="" type="checkbox"/>	Table	public	modelo
<input checked="" type="checkbox"/>	Table	public	pais

? Cancelar Atras Siguiete Terminar

3. Se añade a privilegio el beneficiario Acceso\_total y seleccionamos sus privilegios que serían: INSERT, SELECT, UDATE y DELETE y le damos siguiente después

Asistente de Permisos - Privilege Selection (step 2 of 3)

Please add the required privileges for the selected objects.

Privilegios		
Beneficiario	Privilegios	Cedente
<div>   Acceso_total </div>	<div> <input type="checkbox"/> ALL <input type="checkbox"/> WITH GRANT OPTION </div> <div> <input checked="" type="checkbox"/> INSERT <input type="checkbox"/> WITH GRANT OPTION </div> <div> <input checked="" type="checkbox"/> SELECT <input type="checkbox"/> WITH GRANT OPTION </div> <div> <input checked="" type="checkbox"/> UPDATE <input type="checkbox"/> WITH GRANT OPTION </div> <div> <input checked="" type="checkbox"/> DELETE <input type="checkbox"/> WITH GRANT OPTION </div> <div> <input type="checkbox"/> TRUNCATE <input type="checkbox"/> WITH GRANT OPTION </div> <div> <input type="checkbox"/> REFERENCES <input type="checkbox"/> WITH GRANT OPTION </div> <div> <input type="checkbox"/> TRIGGER <input type="checkbox"/> WITH GRANT OPTION </div>	<div>  postgres </div>

Cancelar
 Atras
 Siguiente
 Terminar

mostraría un SQL y le damos terminar.

- Se crea el rol de Solo\_lectura y con la descripción que muestra en la primera imagen

Create - Login/Group Role

General Definition Privileges Membership Parameters Security SQL

Name Solo\_lectura

Comments Solo podrá hacer select en cada tabla

Asistente de Permisos - Selección de Objetos (paso 1 de 3)

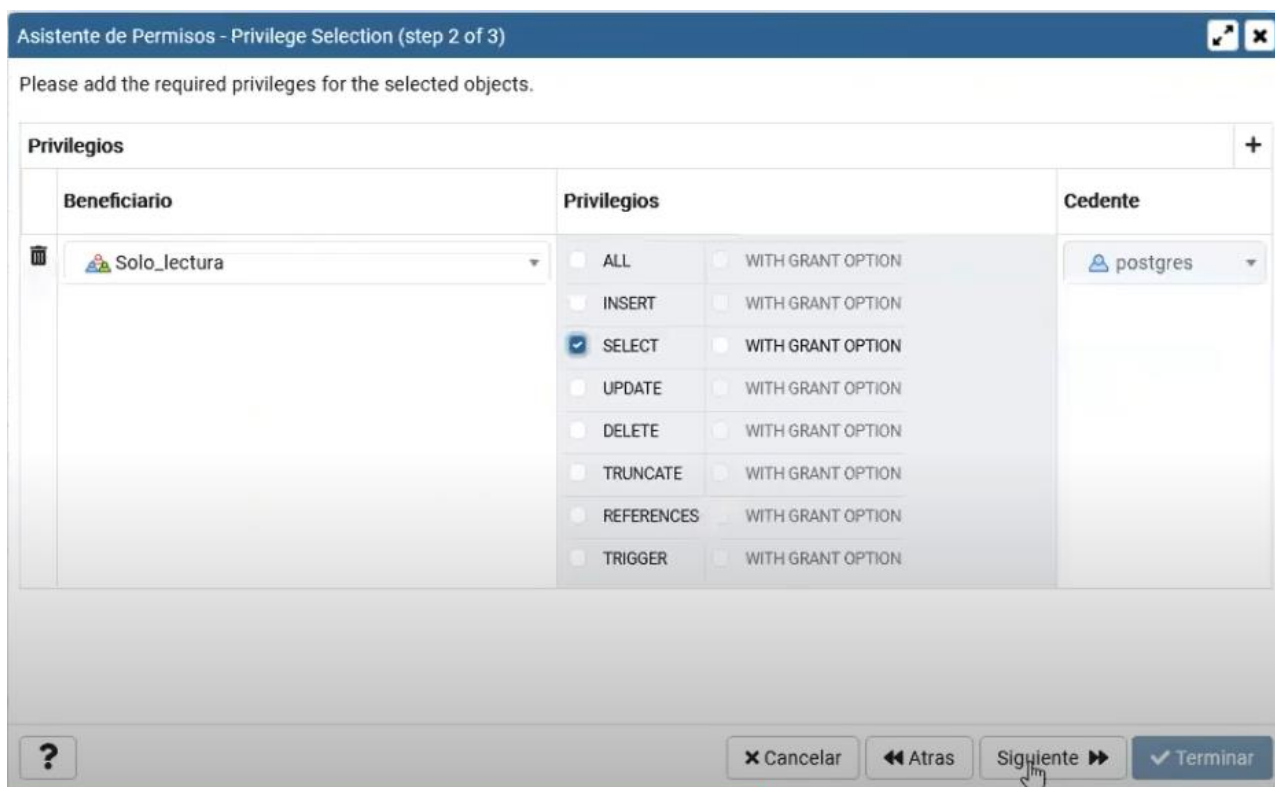
Please select the objects to grant privileges to from the list below.

Search

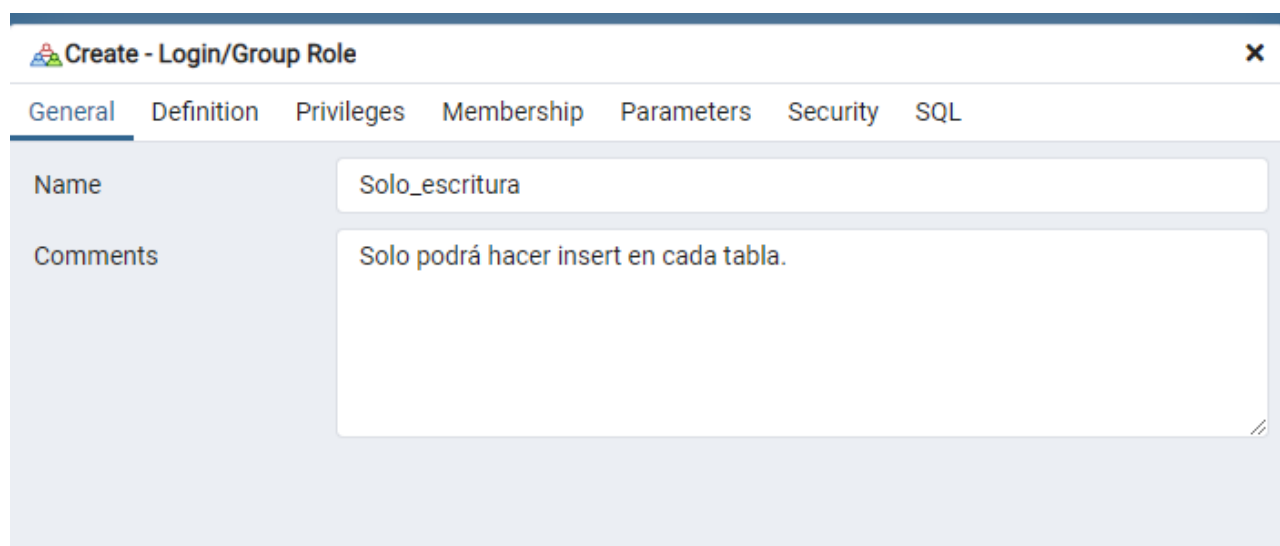
<input checked="" type="checkbox"/>	Tipo de Objeto	Esquema	Nombre
<input checked="" type="checkbox"/>	Table	public	barrio
<input checked="" type="checkbox"/>	Table	public	canton
<input checked="" type="checkbox"/>	Table	public	cliente
<input checked="" type="checkbox"/>	Table	public	color
<input checked="" type="checkbox"/>	Table	public	direccion
<input checked="" type="checkbox"/>	Table	public	distrito
<input checked="" type="checkbox"/>	Table	public	inventario_vehiculos
<input checked="" type="checkbox"/>	Table	public	marca
<input checked="" type="checkbox"/>	Table	public	metodo_pago
<input checked="" type="checkbox"/>	Table	public	modelo
<input checked="" type="checkbox"/>	Table	public	pais

? X Cancelar << Atras Siguiente >> ✓ Terminar

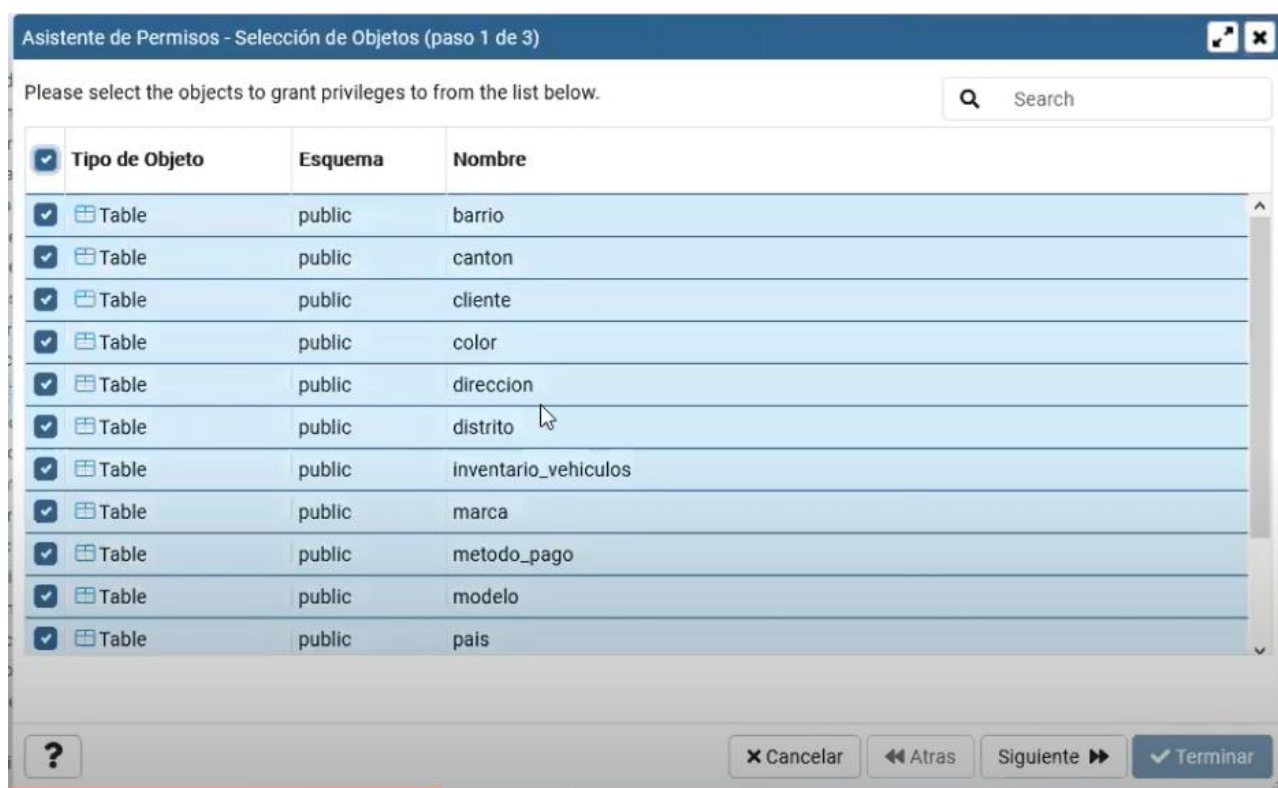
6. Se añade a privilegio el beneficiario Solo\_lectura y seleccionamos sus privilegios que serían: SELECT y le damos siguiente después mostraría un SQL.



7. Se crea el rol de Solo\_escritura y con la descripción que muestra en la primera imagen



8. Para darle los permisos solicitados al rol de Solo\_escritura seleccionamos las tablas las que le daremos el permiso en este caso sería a todas las tablas de esta base de datos



- Se añade a privilegio el beneficiario Solo\_escritura y seleccionamos sus privilegios que serían: INSERT ,y le damos siguiente después mostraría un SQL y le damos terminar.

Asistente de Permisos - Privilege Selection (step 2 of 3)

Please add the required privileges for the selected objects.

Beneficiario	Privilegios	Cedente
Solo_escritura	<input type="checkbox"/> ALL <input checked="" type="checkbox"/> INSERT <input type="checkbox"/> SELECT <input type="checkbox"/> UPDATE <input type="checkbox"/> DELETE <input type="checkbox"/> TRUNCATE <input type="checkbox"/> REFERENCES <input type="checkbox"/> TRIGGER	<input type="checkbox"/> WITH GRANT OPTION <input type="checkbox"/> WITH GRANT OPTION <input type="checkbox"/> WITH GRANT OPTION <input type="checkbox"/> WITH GRANT OPTION <input type="checkbox"/> WITH GRANT OPTION <input type="checkbox"/> WITH GRANT OPTION <input type="checkbox"/> WITH GRANT OPTION
		postgres

? Cancelar Atras Siguiente Terminar

### Ejercicio 3.b

1. Se crea el usuario k\_moraga.

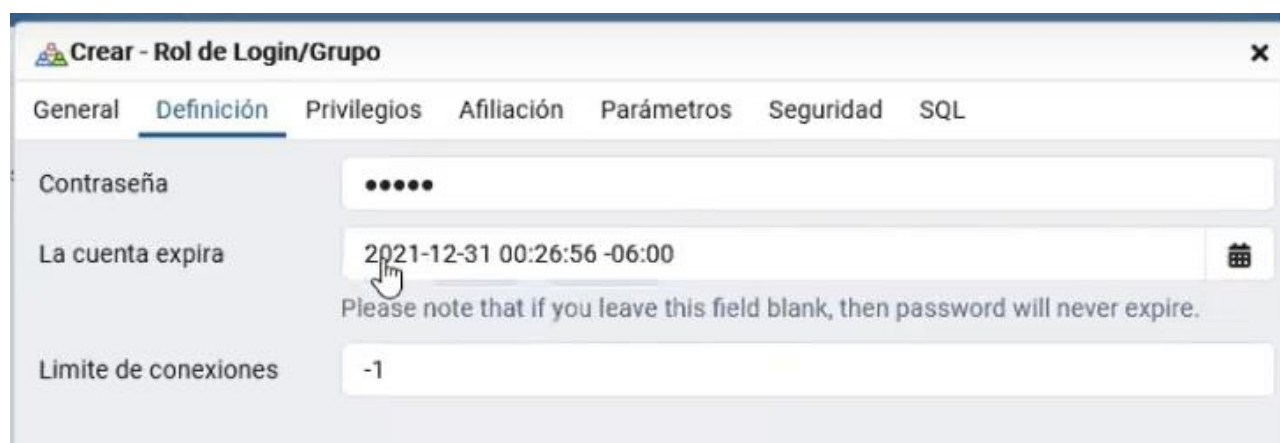
Crear - Rol de Login/Grupo

General Definición Privilegios Afiliación Parámetros Seguridad SQL

Nombre k\_moraga

Comentarios

2. Sé él define la contraseña "linux" y la fecha de expiración



**Crear - Rol de Login/Grupo**

General Definición Privilegios Afiliación Parámetros Seguridad SQL

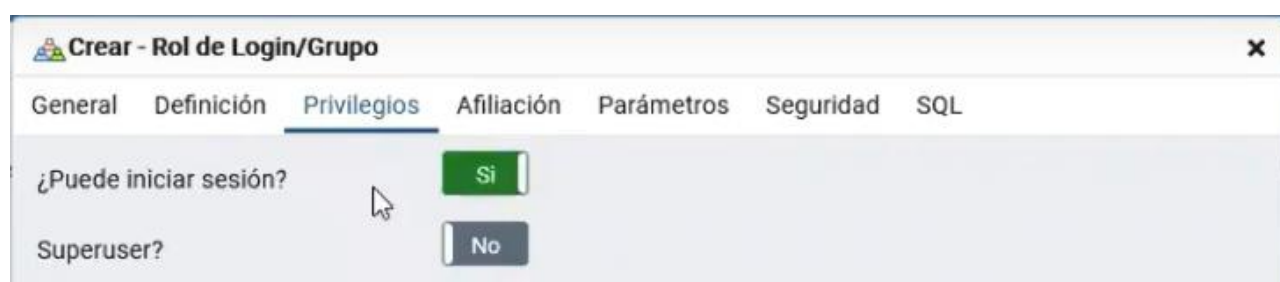
Contraseña: .....

La cuenta expira: 2021-12-31 00:26:56 -06:00

Please note that if you leave this field blank, then password will never expire.

Limite de conexiones: -1

3. Se le da el privilegio de Iniciar sesión y se le asigna la afiliación Acceso\_total.



**Crear - Rol de Login/Grupo**

General Definición Privilegios Afiliación Parámetros Seguridad SQL

¿Puede iniciar sesión? ☒ Si

Superuser? ☐ No



**Crear - Rol de Login/Grupo**

General Definición Privilegios Afiliación Parámetros Seguridad SQL

Roles: Acceso\_total

Seleccione la caja para que el rol incluya WITH ADMIN OPTION

Se crea el usuario m\_cortes.





**Crear - Rol de Login/Grupo**

General Definición Privilegios Afiliación Parámetros Seguridad SQL

Nombre: m\_cortes

Comentarios:

Se le define la contraseña "linux" y la fecha de expiración



**Crear - Rol de Login/Grupo**

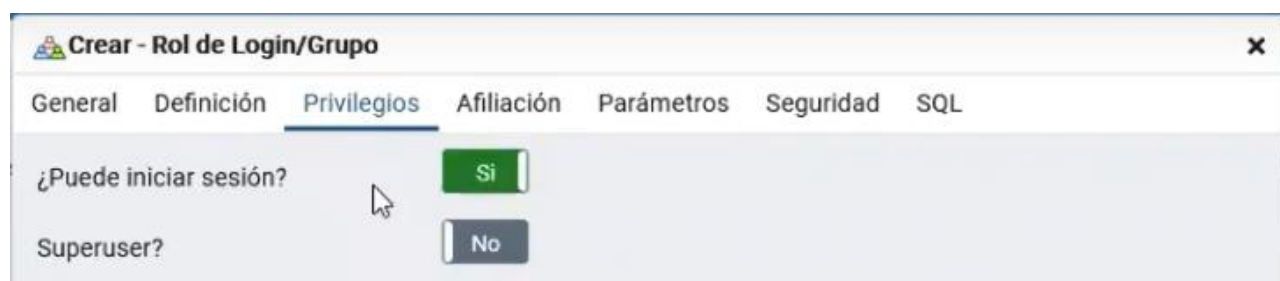
General Definición Privilegios Afiliación Parámetros Seguridad SQL

Contraseña: •••••

La cuenta expira: 2021-12-31 00:28:41 -06:00

Please note that if you leave this field blank, then password will never expire.

Se le da el privilegio de Iniciar sesión.



**Crear - Rol de Login/Grupo**

General Definición Privilegios Afiliación Parámetros Seguridad SQL

¿Puede iniciar sesión? ☒ Si

Superuser? ☐ No

Y se le asigna la afiliación Solo\_lectura.



**Crear - Rol de Login/Grupo**

General Definición Privilegios Afiliación Parámetros Seguridad SQL

Roles: Solo\_lectura

Seleccione la caja para que el rol incluya WITH ADMIN OPTION.

Se crea el usuario a\_cortes.



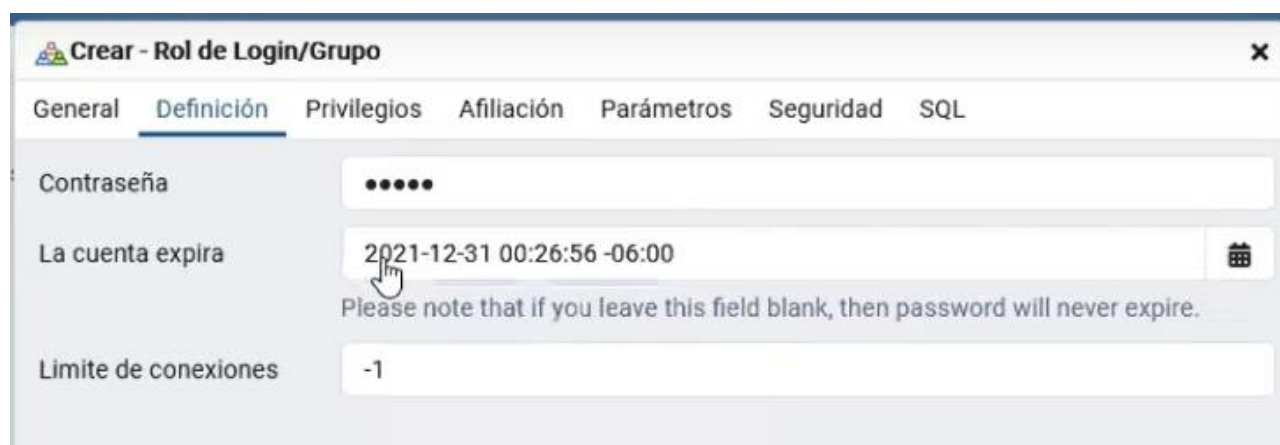
**Crear - Rol de Login/Grupo**

General Definición Privilegios Afiliación Parámetros Seguridad SQL

Nombre: a\_cortes

Comentarios:

Se le define la contraseña "linux" y la fecha de expiración



**Crear - Rol de Login/Grupo**

General Definición Privilegios Afiliación Parámetros Seguridad SQL

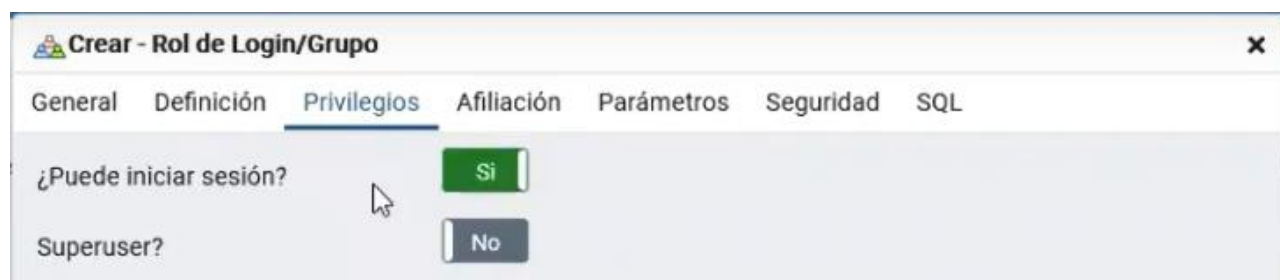
Contraseña: .....

La cuenta expira: 2021-12-31 00:26:56 -06:00

Please note that if you leave this field blank, then password will never expire.

Limite de conexiones: -1

Se le da el privilegio de Iniciar sesión.



**Crear - Rol de Login/Grupo**

General Definición Privilegios Afiliación Parámetros Seguridad SQL

¿Puede iniciar sesión? ☒ Si

Superuser? ☐ No

Se le asigna la afiliación Solo\_escritura.

Crear - Rol de Login/Grupo

General

Definición

Privilegios

Afiliación

Parámetros

Seguridad

SQL

Roles

Solo\_escritura

Seleccione la caja para que el rol incluya WITH ADMIN OPTION.