**Documento Reporte**

Base de datos de Tienda de Reparación de celulares

**Consideraciones:**

* Sistema Operativo Windows 10
* PostgreSQL verion13.2
* pgAdmin 4 version 5

1. Crear la base de datos **tienda** en PostgreSQL.

* Conectarse a la base de datos PostgreSQL. Comando:

C:\Program Files\PostgreSQL\13\bin>psql -U postgres -W -h localhost

Password:

psql (13.2)

WARNING: Console code page (437) differs from Windows code page (1252)

8-bit characters might not work correctly. See psql reference

page "Notes for Windows users" for details.

Type "help" for help.

postgres=#

* Crear base de datos tienda. Comando:

postgres=# CREATE DATABASE tienda;

CREATE DATABASE

postgres=#

* Verificar la creación de esta. Comando:

postgres=# \l

List of databases

Name | Owner | Encoding | Collate | Ctype | Access privileges

---------------+----------+----------+----------------------------+----------------------------+-----------------------

postgres | postgres | UTF8 | English\_United States.1252 | English\_United States.1252 |

prueba | postgres | UTF8 | English\_United States.1252 | English\_United States.1252 |

supertv | postgres | UTF8 | English\_United States.1252 | English\_United States.1252 |

supertv\_nueva | postgres | UTF8 | English\_United States.1252 | English\_United States.1252 |

template0 | postgres | UTF8 | English\_United States.1252 | English\_United States.1252 | =c/postgres +

| | | | | postgres=CTc/postgres

template1 | postgres | UTF8 | English\_United States.1252 | English\_United States.1252 | =c/postgres +

| | | | | postgres=CTc/postgres

tienda | postgres | UTF8 | English\_United States.1252 | English\_United States.1252 |

(7 rows)

postgres=#

1. Recuperar la base de datos inicial usando el archivo “**tienda.sql**”.

* Se copia el dump tienda.sql en el directorio donde se encuentra el binario pg\_dump.exe
* Se recupera el dump, con el siguiente comando

psql -U postgres -W -h localhost -d tienda < tienda.sql

Output:

C:\Program Files\PostgreSQL\13\bin>psql -U postgres -W -h localhost -d tienda < tienda.sql

Password:

SET

SET

SET

SET

SET

set\_config

------------

(1 row)

SET

SET

SET

CREATE EXTENSION

COMMENT

SET

SET

CREATE TABLE

CREATE SEQUENCE

ALTER SEQUENCE

CREATE TABLE

CREATE SEQUENCE

ALTER SEQUENCE

ALTER TABLE

ALTER TABLE

COPY 0

setval

--------

1

(1 row)

COPY 2

setval

--------

2

(1 row)

ALTER TABLE

ALTER TABLE

C:\Program Files\PostgreSQL\13\bin>

* Verificamos que ahora tengamos tablas y datos en DB tienda recuperada

C:\Program Files\PostgreSQL\13\bin>psql -U postgres -W -h localhost -d tienda

Password:

psql (13.2)

WARNING: Console code page (437) differs from Windows code page (1252)

8-bit characters might not work correctly. See psql reference

page "Notes for Windows users" for details.

Type "help" for help.

tienda=# \dt

List of relations

Schema | Name | Type | Owner

--------+-----------+-------+----------

public | modelos | table | postgres

public | servicios | table | postgres

(2 rows)

tienda=# \d+ modelos

Table "public.modelos"

Column | Type | Collation | Nullable | Default | Storage | Stats target | Description

-------------+-------------------+-----------+----------+-------------------------------------+----------+--------------+-------------

id | integer | | not null | nextval('modelos\_id\_seq'::regclass) | plain | |

descripcion | character varying | | | | extended | |

Indexes:

"modelos\_pkey" PRIMARY KEY, btree (id)

Access method: heap

tienda=# \d+ servicios

Table "public.servicios"

Column | Type | Collation | Nullable | Default | Storage | Stats target | Description

-------------+-------------------+-----------+----------+---------------------------------------+----------+--------------+-------------

id | integer | | not null | nextval('servicios\_id\_seq'::regclass) | plain | |

descripcion | character varying | | | | extended | |

precio | real | | | | plain | |

Indexes:

"servicios\_pkey" PRIMARY KEY, btree (id)

Access method: heap

tienda=# SELECT \* FROM modelos;

id | descripcion

----+-------------

(0 rows)

tienda=# SELECT \* FROM servicios;

id | descripcion | precio

----+-----------------+--------

1 | cambio pantalla | 90

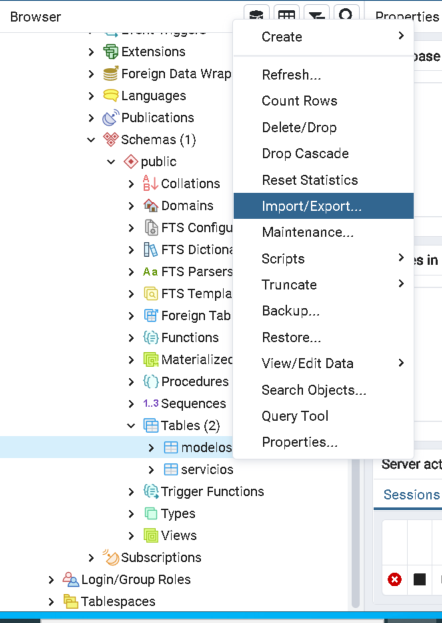
2 | cambio bateria | 50

(2 rows)

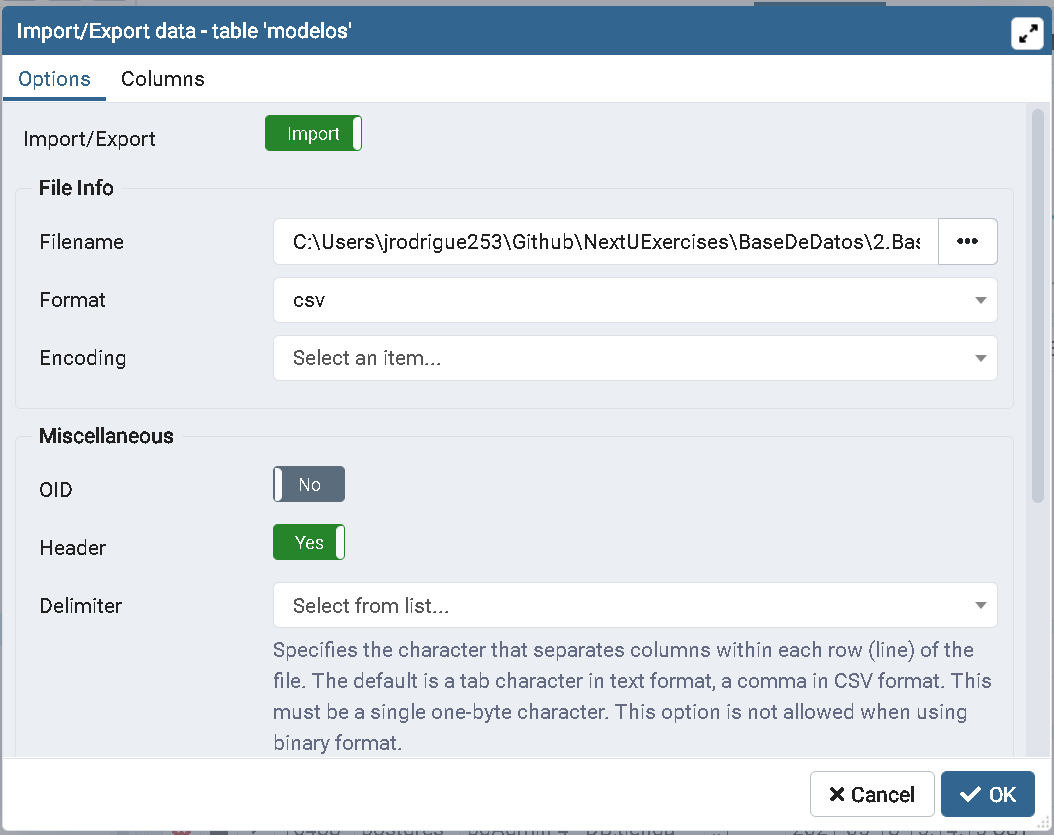
tienda=#

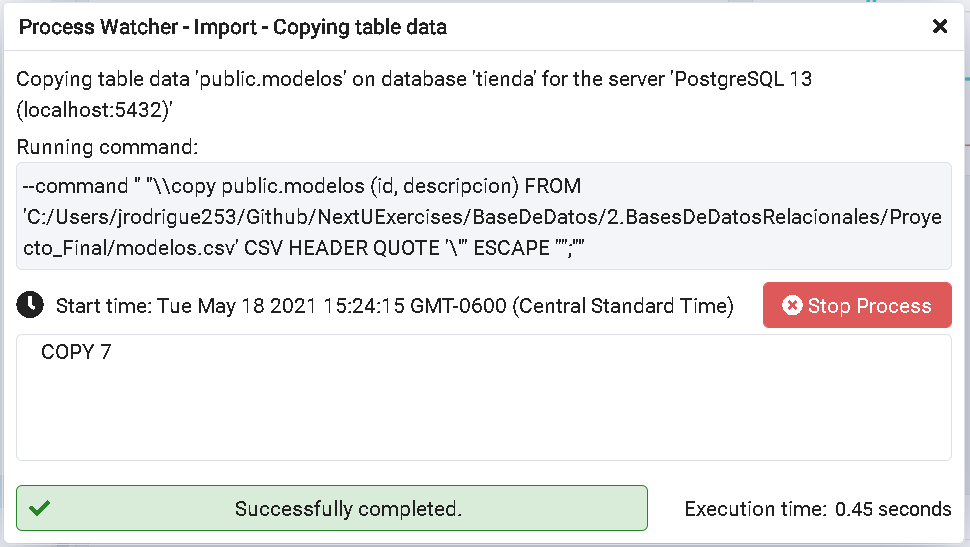
1. Importar el contenido de los modelos usando el archivo “**modelos.csv**”.

* Importación con la herramienta gráfica PgAdmin ubicamos la tabla modelos. Hacemos click derecho y seleccionamos Import/Export



* En la siguiente ventana se selecciona la opción Import, se indica el path donde se encuentra el archivo csv a importar, y finalmente se indica también que el archivo contiene header (para ignorar la primer fila).





El comando ejecutado a través de PgAdmin es:

--command " "\\copy public.modelos (id, descripcion) FROM 'C:/Users/jrodrigue253/Github/NextUExercises/BaseDeDatos/2.BasesDeDatosRelacionales/Proyecto\_Final/modelos.csv' CSV HEADER QUOTE '\"' ESCAPE '''';""

* Se verifica que la tabla contiene los datos:

C:\Users\jrodrigue253>psql -U postgres -W -d tienda

Password:

psql (13.2)

WARNING: Console code page (437) differs from Windows code page (1252)

8-bit characters might not work correctly. See psql reference

page "Notes for Windows users" for details.

Type "help" for help.

tienda=# SELECT \* FROM modelos;

id | descripcion

----+-------------

1 | Noquia A1

2 | Noquia A2

3 | Sansunn S1

4 | Sansunn S2

5 | Aifon 7

6 | Aifon 8

7 | Aifon 9

(7 rows)

tienda=#

1. Crear las tablas clientes y órdenes con los siguientes requerimientos:
2. Usar normalización de los datos.
3. Definir las claves foráneas.
4. Llenar las tablas con los datos proporcionados por el cliente en la tabla anterior.

**Tabla que se debe normalizar**

| **orden** | **cliente(nombre)** | **cliente(teléfono)** | **modelo** | **descripción** | **costo** |
| --- | --- | --- | --- | --- | --- |
| **1** | Luis Torres | 55555555 | Noquia A1 | cambiar pantalla | 90 |
| **2** | Luis Torres | 55555555 | Noquia A2 | cambiar bateria | 50 |
| **3** | Ana | 123123123 | Sansuun S1 | cambiar bateria | 50 |
| **4** | María | 11111111 | Aifon 7 | cambiar bateria | 50 |
| **5** | María | 11111111 | Sansuun S2 | cambiar pantalla | 90 |
| **6** | Pedro | 2222222 | Aifon 8 | cambiar pantalla | 90 |
| **7** | Pedro | 2222222 | Aifon 9 | cambiar pantalla | 90 |

* Se debe crear una tabla clientes con los campos id, cliente, teléfono
* La columna modelo es una llave foránea, la cual asocia el modelo
* La columna descripción (que va a ser reemplazada por servicio para una mejor comprensión) es una llave foránea, la cual asocia la descripción del servicio y el costo.

**Las tablas normalizadas quedarían de la siguiente forma:**

| **id** | **cliente(nombre)** | **cliente(teléfono)** |
| --- | --- | --- |
| **1** | Luis Torres | 55555555 |
| **2** | Ana | 123123123 |
| **3** | María | 11111111 |
| **4** | Pedro | 2222222 |

Tabla **clientes**:

Tabla **modelos**:

| **orden** | **modelo** |
| --- | --- |
| **1** | Noquia A1 |
| **2** | Noquia A2 |
| **3** | Sansuun S1 |
| **4** | Sansuun S2 |
| **5** | Aifon 7 |
| **6** | Aifon 8 |
| **7** | Aifon 9 |

Tabla **servicios**:

| **id** | **descripcion** | **precio** |
| --- | --- | --- |
| **1** | cambio pantalla | 90 |
| **2** | cambio batería | 50 |

Tabla **ordenes**:

| **orden\_id** | **cliente\_id (fk)** | **modelo\_id(fk)** | **servicio\_id (fk)** |
| --- | --- | --- | --- |
| **1** | 1 | 1 | 1 |
| **2** | 1 | 2 | 2 |
| **3** | 2 | 3 | 2 |
| **4** | 3 | 5 | 2 |
| **5** | 3 | 4 | 1 |
| **6** | 4 | 6 | 1 |
| **7** | 4 | 7 | 1 |

**Llaves foráneas en tabla odenes son:**

* + cliente\_id
  + modelo\_id
  + servicio\_id

**Inserción de datos:**

* Se debe crear la tabla clientes:

tienda=# CREATE TABLE clientes (id SERIAL PRIMARY KEY NOT NULL,cliente VARCHAR(256),telefono INT);

CREATE TABLE

tienda=#

* Verificar la tabla creada:

tienda=# \d clientes;

Table "public.clientes"

Column | Type | Collation | Nullable | Default

----------+------------------------+-----------+----------+--------------------------------------

id | integer | | not null | nextval('clientes\_id\_seq'::regclass)

cliente | character varying(256) | | |

telefono | integer | | |

Indexes:

"clientes\_pkey" PRIMARY KEY, btree (id)

* Ingresar los datos de los clientes:

tienda=# INSERT INTO clientes (cliente,telefono) VALUES ('Luis Torres',55555555),('Ana',123123123),('María',11111111),('Pedro',2222222);

INSERT 0 4

tienda=#

* Se verifica que los datos se hayan ingresado correctamente:

tienda=# SELECT \* FROM clientes;

id | cliente | telefono

----+-------------+-----------

1 | Luis Torres | 55555555

2 | Ana | 123123123

3 | María | 11111111

4 | Pedro | 2222222

(4 rows)

* Crear tabla para las órdenes:

tienda=# CREATE TABLE ordenes (orden\_id SERIAL PRIMARY KEY,cliente\_id INT,modelo\_id INT,servicio\_id INT);

CREATE TABLE

tienda=#

* Se verifica si la tabla se ha creado correctamente:

tienda=# \d ordenes;

Table "public.ordenes"

Column | Type | Collation | Nullable | Default

-------------+---------+-----------+----------+-------------------------------------------

orden\_id | integer | | not null | nextval('ordenes\_orden\_id\_seq'::regclass)

cliente\_id | integer | | |

modelo\_id | integer | | |

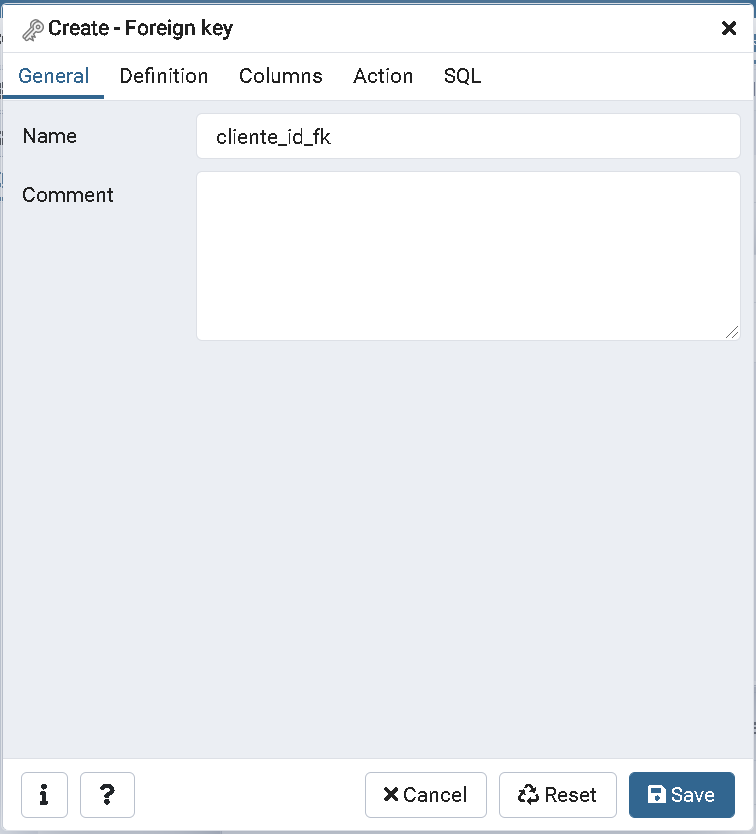
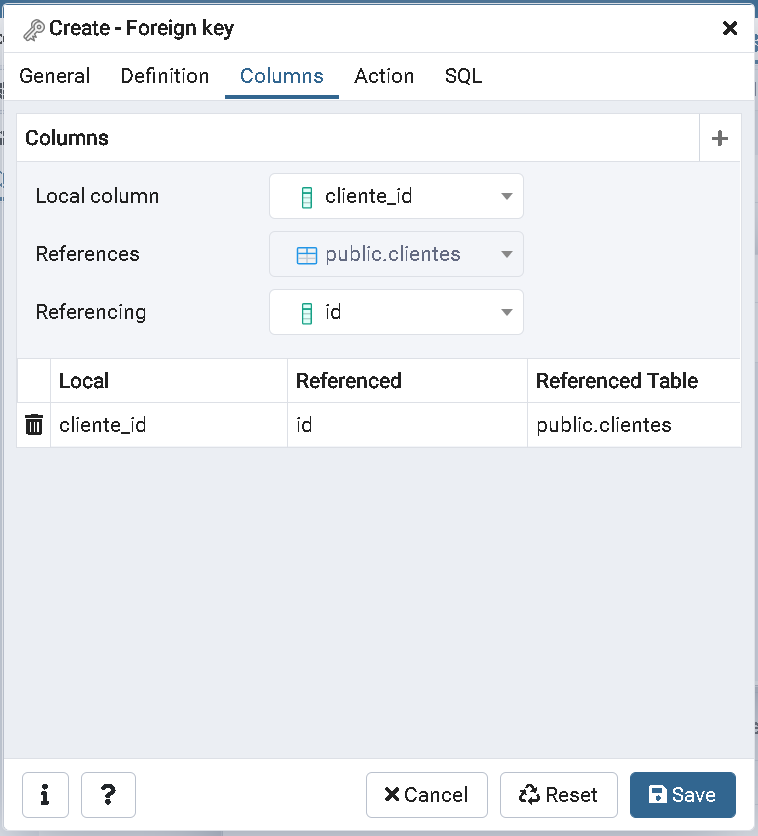
servicio\_id | integer | | |

Indexes:

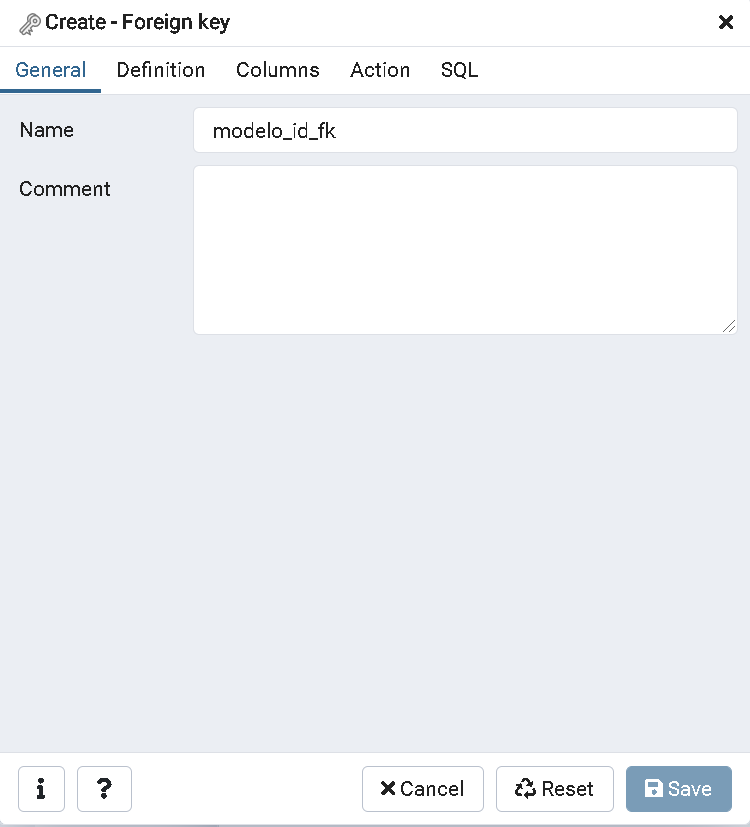
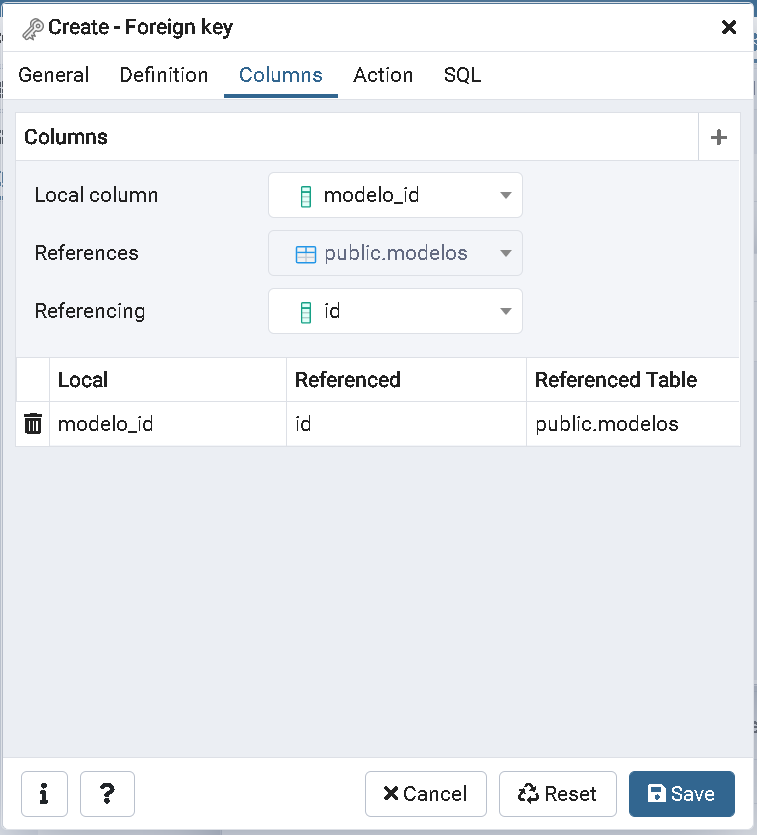
"ordenes\_pkey" PRIMARY KEY, btree (orden\_id)

* Se crean las restricciones (constraints) de las llaves foráneas.

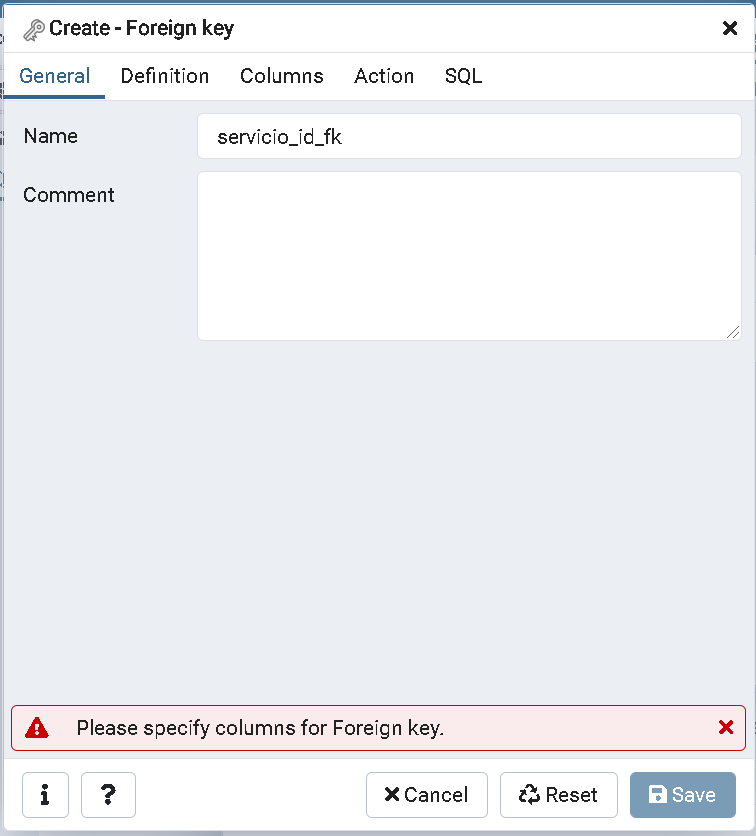
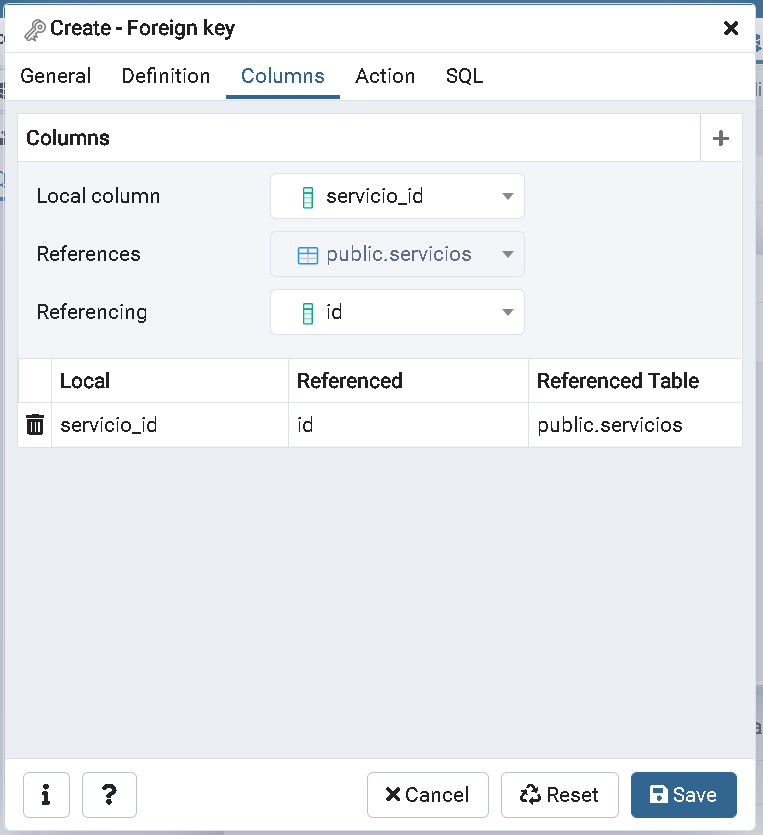
En la tabla ordenes se agrega el constraints de FK para la columna cliente\_id en PgAdmin:

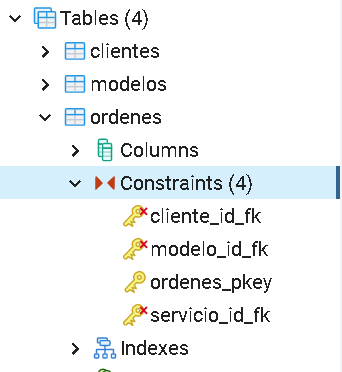
En la tabla ordenes se agrega el constraints de FK para la columna modelo\_id en PgAdmin:

En la tabla ordenes se agrega el constraints de FK para la columna servicio\_id en PgAdmin:

* La tabla órdenes debe tener agregadas las 3 foreign keys:



Esto también se puede verificar vía comando:

tienda=# \d ordenes;

Table "public.ordenes"

Column | Type | Collation | Nullable | Default

-------------+---------+-----------+----------+-------------------------------------------

orden\_id | integer | | not null | nextval('ordenes\_orden\_id\_seq'::regclass)

cliente\_id | integer | | |

modelo\_id | integer | | |

servicio\_id | integer | | |

Indexes:

"ordenes\_pkey" PRIMARY KEY, btree (orden\_id)

"fki\_cliente\_id\_fk" btree (cliente\_id)

"fki\_modelo\_id\_fk" btree (modelo\_id)

"fki\_servicio\_id\_fk" btree (servicio\_id)

Foreign-key constraints:

"cliente\_id\_fk" FOREIGN KEY (cliente\_id) REFERENCES clientes(id) NOT VALID

"modelo\_id\_fk" FOREIGN KEY (modelo\_id) REFERENCES modelos(id) NOT VALID

"servicio\_id\_fk" FOREIGN KEY (servicio\_id) REFERENCES servicios(id) NOT VALID

* Por último, se ingresan los datos en la tabla ordenes, según la tabla ordenes normalizada:

tienda=# INSERT INTO ordenes (cliente\_id,modelo\_id,servicio\_id) VALUES (1,1,1),(1,2,2),(2,3,2),(3,5,2),(3,4,1),(4,6,1),(4,7,1);

INSERT 0 7

tienda=#

* Se verifica los datos de la tabla:

tienda=# SELECT \* FROM ordenes;

orden\_id | cliente\_id | modelo\_id | servicio\_id

----------+------------+-----------+-------------

1 | 1 | 1 | 1

2 | 1 | 2 | 2

3 | 2 | 3 | 2

4 | 3 | 5 | 2

5 | 3 | 4 | 1

6 | 4 | 6 | 1

7 | 4 | 7 | 1

(7 rows)

1. Definir las siguientes consultas:
2. Nombre, Costo total por cliente: El resultado debe ser el nombre del cliente y el costo total de todos sus servicios.

**Consulta:**

tienda=# SELECT c.cliente,

tienda-# SUM(precio) AS costo\_total

tienda-# FROM ordenes o

tienda-# JOIN clientes c ON c.id = o.cliente\_id

tienda-# JOIN modelos m ON m.id = o.modelo\_id

tienda-# JOIN servicios s ON s.id = o.servicio\_id

tienda-# GROUP BY c.cliente;

**Resultado:**

cliente | costo\_total

-------------+-------------

Luis Torres | 140

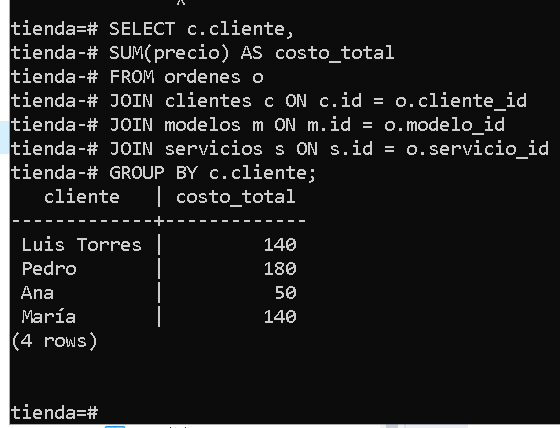
Pedro | 180

Ana | 50

María | 140

(4 rows)

**Comando y salida en imagen:**



1. Costo promedio actual de toda la tienda (que se obtiene promediando los costos de todas las órdenes).

**Consulta:**

tienda=# SELECT SUM(s.precio)/COUNT(orden\_id) AS promedio

tienda-# FROM ordenes o

tienda-# JOIN servicios s ON s.id = o.servicio\_id;

**Resultado:**

promedio

-------------------

72.85714285714286

(1 row)

**Comando y salida en imagen:**

