––-- -- a)

-- CREATE USER user1 WITH PASSWORD '111';

-- CREATE USER user2 WITH PASSWORD '222';

-- -- b)

-- CREATE TABLE IF NOT EXISTS tabla1

-- (

-- nrodpto SERIAL PRIMARY KEY,

-- nombdpto VARCHAR(20) UNIQUE NOT NULL

-- );

-- CREATE TABLE IF NOT EXISTS tabla2

-- (

-- id SERIAL PRIMARY KEY,

-- nombre VARCHAR(30) NOT NULL,

-- apellido VARCHAR(30) NOT NULL,

-- sueldo NUMERIC(9,2),

-- nrodpto INTEGER NOT NULL,

-- CONSTRAINT FK\_nrodpto FOREIGN KEY (nrodpto) REFERENCES tabla1(nrodpto)

-- ON UPDATE CASCADE

-- ON DELETE RESTRICT

-- );

-- -- c)

-- GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA PUBLIC TO postgres;

-- GRANT INSERT, SELECT ON tabla1 TO user1

-- GRANT ALL ON tabla2 TO user2

-- -- d)

-- INSERT INTO tabla1 (nombdpto) VALUES

-- ('SANTA CLASA 914'),

-- ('CONGRESO TUCUMAN 19'),

-- ('CEPEDA 125');

-- INSERT INTO tabla2 (nombre, apellido, sueldo, nrodpto) VALUES

-- ('Gonzalo', 'Errandonea', 1534.241, 1),

-- ('Tomas', 'Alaluf', 1, 1),

-- ('Alex', 'Brunetti', 9999.241, 3);

-- -- e)

-- UPDATE tabla1 SET nrodpto = 5 WHERE nrodpto = 1

-- -- Al hacer el update, actualizo el nrodpto 1 a 5, y como la tabla2 tenia el update on cascade, cualquier cambio se que

-- -- produzca en alguna de las cables primarias, como se realizo en este punto, tambien se realizara un cambio el las fk nrodpto de la t

-- --- tabla2

-- DELETE FROM tabla1 WHERE nrodpto = 5

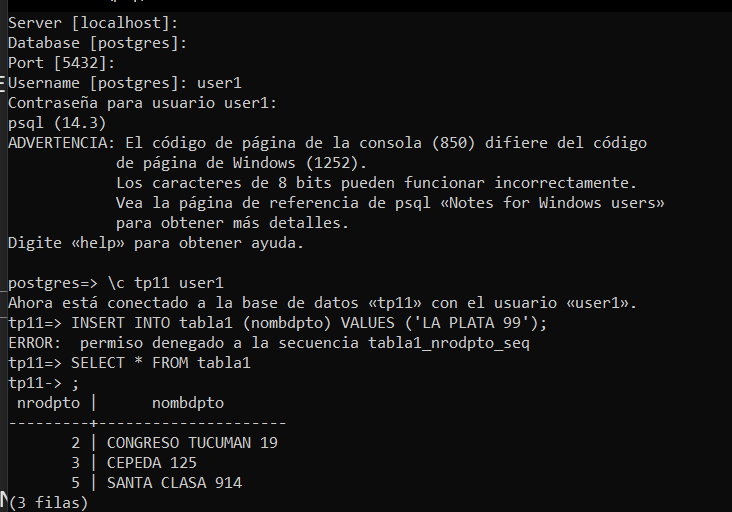
-- -- al hacer un delete nos suelta el siguiente mensaje

-- -- ERROR: update o delete en «tabla1» viola la llave foránea «fk\_nrodpto» en la tabla «tabla2»

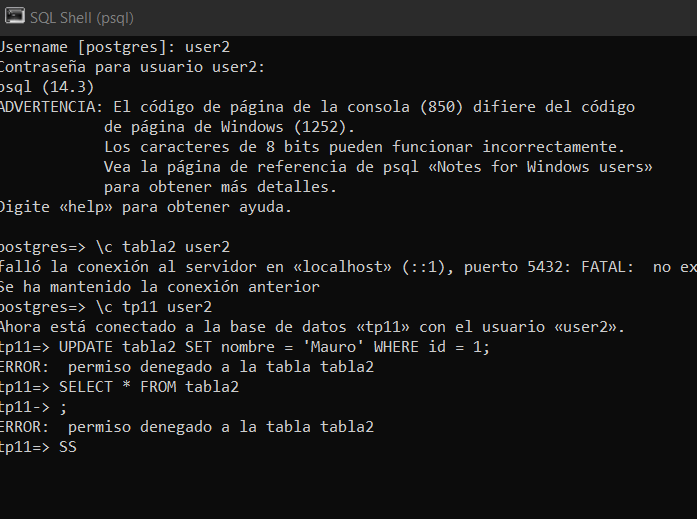
-- -- Esto porque no se puede borrar una clave primaria a la que se le esta haciendo referencia,

-- -- como es el caso de que tabla2 hace referencia con nrodpto a nrodpto de la tabla1

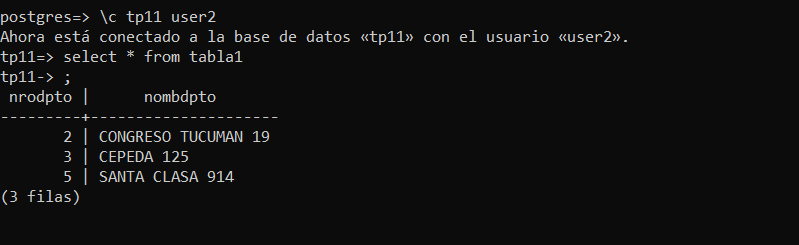
f y g)

1

2



1. GRANT SELECT ON tabla1 TO user2;

j) 

l)

Ls funciones de las que investigue dos fueron las que mas me llamaron la atencion, crypt y pgp\_sym\_encrypt, la diferencias entre estas son muchas, pero las dos mas destacadas son que con crypt no Podemos desencriptar ya que se pierden bytes a la hora de encriptar algo, eso que se cree es una ventaja, realealmente ya no lo es tanto, puesto a que alguien podria romper la seguridad comparando el hash que se genera con la function de mi base de datos con la suya, pgp\_sym\_encrypt en cambio es simetrico, nos permite el poder encriptar como desencriptar con pgp\_sym\_decrypt, lo unico que se necesitan son dos cosas, en el caso de pgp\_sym\_encrypt el elemento a encriptar y una key, un secreto, esto permitira encriptar nuestro elemento en base a esa key, y pgp\_sym\_decrypt solamente pasando el elemento ya encriptado en bytea y nuestra key, ya se desencriptaria, esto nos da la certeza de que lo que hayamos encriptado no sea tan facil de hackear, ya que para desencriptarlo necessitarian la key con la que se encripto.

k)

ALTER TABLE tabla2

DROP COLUMN sueldo;

ALTER TABLE tabla2

ADD sueldo BYTEA NOT NULL;

CREATE EXTENSION IF NOT EXISTS pgcrypto;

CREATE OR REPLACE FUNCTION encrypt(t TEXT) RETURNS BYTEA AS

$funcEncrypt$

BEGIN

RETURN pgp\_sym\_encrypt(t, 'miSecreto123');

END;

$funcEncrypt$ LANGUAGE plpgsql;

CREATE OR REPLACE FUNCTION fdecrypt(t BYTEA) RETURNS NUMERIC AS

$funcFdencrypt$

BEGIN

RETURN (pgp\_sym\_decrypt(t, 'miSecreto123'))::NUMERIC;

END;

$funcFdencrypt$ LANGUAGE plpgsql;

INSERT INTO tabla2 (nombre, apellido, sueldo, nrodpto)

VALUES ('Gonzalo', 'Errandonea', encrypt((123123.213)::text), 2);

SELECT nombre, apellido, fdecrypt(tabla2.sueldo), nrodpto FROM tabla2 WHERE id = 9;

UPDATE tabla2 SET sueldo = encrypt((123.23)::text) WHERE id = 9;

SELECT nombre, apellido, fdecrypt(tabla2.sueldo), nrodpto FROM tabla2 WHERE id = 9;