



# **BASES DE DATOS**

Clase 7 – ¿Qué son Bases de Datos?  
Profesora: Erika Gutiérrez Beltrán

# Tema 2: Bases de datos relacionales

---



## Objetivo

Transformar los modelos conceptuales a lógicos. Construir bases de datos para datos estructurados

- Utilización de los modelos en el mundo real
- Transformación del modelado a la base de datos



## Temario

- Clases de entidades
- Arquitectura de los sistemas gestores de Bases de datos, OLTP, OLAP
- Modelo relacional, restricciones de integridad y normalización
- Lenguaje de consulta formal
- Lenguaje de consulta SQL



## Logros

Capturar datos desde una pantalla de usuario real y almacenarlos en una base de datos. Manipulación de estos datos.

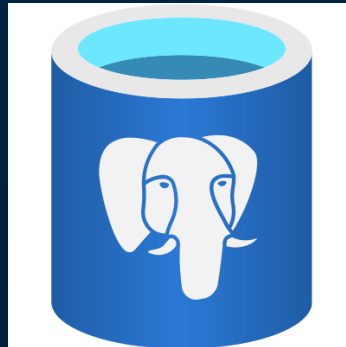


## ¿Qué son las bases de datos?

Las bases de datos registran datos que luego se convertirán en la herramienta de información de una organización. En el caso de las bases de datos relacionales, estas se basan en los modelos para estructurar las tablas, sus atributos, tipos y relaciones.

Un conjunto de filas, conforman una tabla y un conjunto de tablas conforman una base de datos.

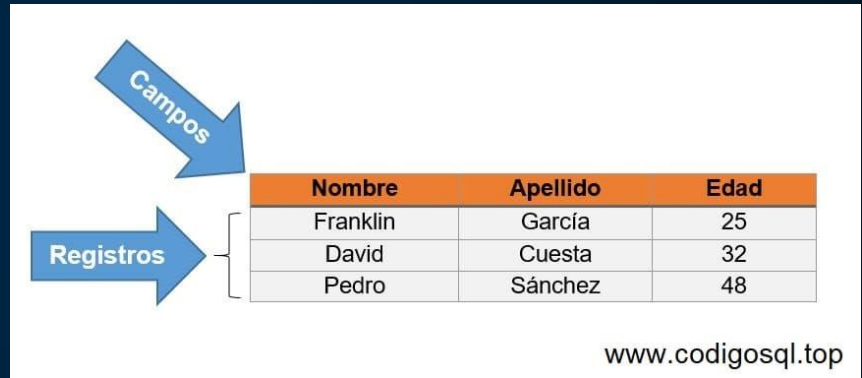
Las bases de datos relaciones se manipulan por medio del lenguaje SQL





## Características

- Las bases de datos poseen un ID único para identificar cada fila ingresada (cada registro), a esto se le conoce como clave primaria. Esto permite más adelante llevar a cabo la creación de relaciones
- Permiten rastrear inventarios, procesar transacciones, administrar información de usuarios
- Las reglas y políticas de las bases de datos relaciones son muy detalladas y estrictas
- Permiten crear y ejecutar procedimientos almacenados, para así ejecutar consultas en masa

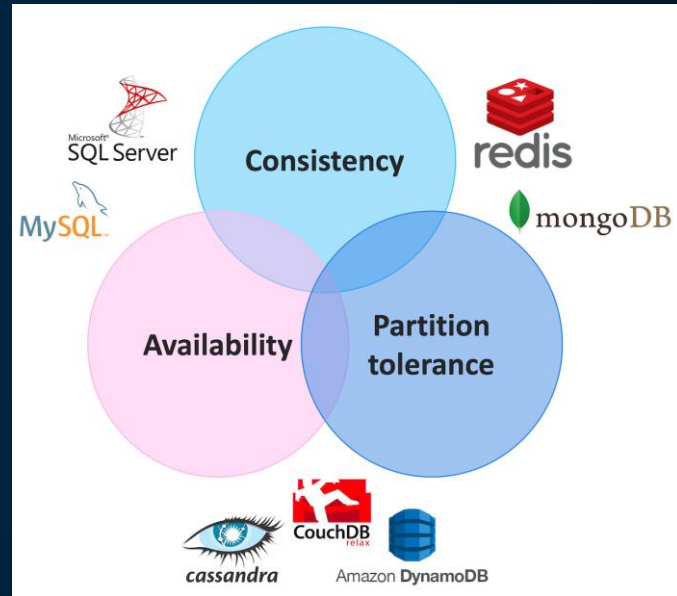




## ¿Qué debería se tener en cuenta al seleccionar una BD relacional?

El tipo de base de datos que se escoja parte de la necesidad que se tengan con la aplicación, por está razón, al momento de seleccionar una base de datos relacional, es necesario realizar las siguientes preguntas:

- ¿Se necesita una base de datos con un gran nivel de precisión en los datos?
- ¿Se necesita escalabilidad?, ¿es posible mantener uniformidad de los datos en todas las instancias?
- ¿Es importante la concurrencia?, ¿cuántos usuarios accederán a los mismos datos al tiempo?
- ¿Qué se requiere en términos de rendimiento y fiabilidad? (SLA)





## Motores de bases de datos

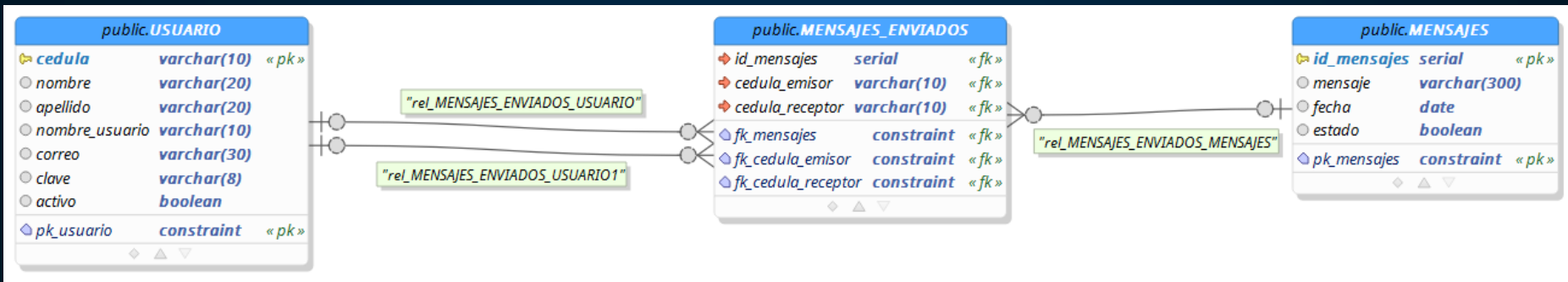


Un motor de base de datos es aquel sistema donde es posible manipular datos no solo a nivel de CRUD (Create, Read, Update, Delete), sino también a nivel de procesamiento, manejo de múltiples usuarios, transacciones, extracción de datos en memoria y mostrar por medio de un lenguaje la información al solicitante por medio de una pantalla.

El motor de base de datos es el punto control de la información.



## ¿Cómo pasar un modelo relacional a una base de datos?



Modelo relacional de mensajes enviados a uno o varios usuarios (PGModeler)



## Tipos de datos

Los datos son representaciones de atributos, los cuales pueden ser:

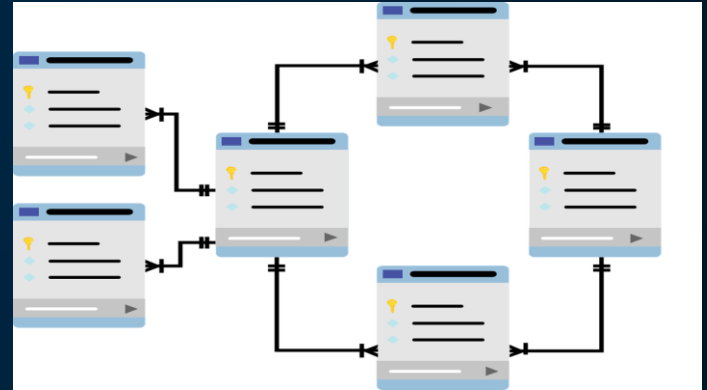
- Mediciones
- Identificadores
- Descripciones
- Cualitativos o cuantitativos

En el lenguaje SQL estos se dividen en tres grandes grupos:

Numéricos: (BIT, TINYINT, BOOL, BOOLEAN, SMALLINT)

Hilos de caracteres: (CHAR, VARCHAR, BINARY)

Fecha y tiempo: (DATE, DATETIME, TIMESTAMP)







## Tipos de datos

### Tipos de datos extendidos en PostgreSQL

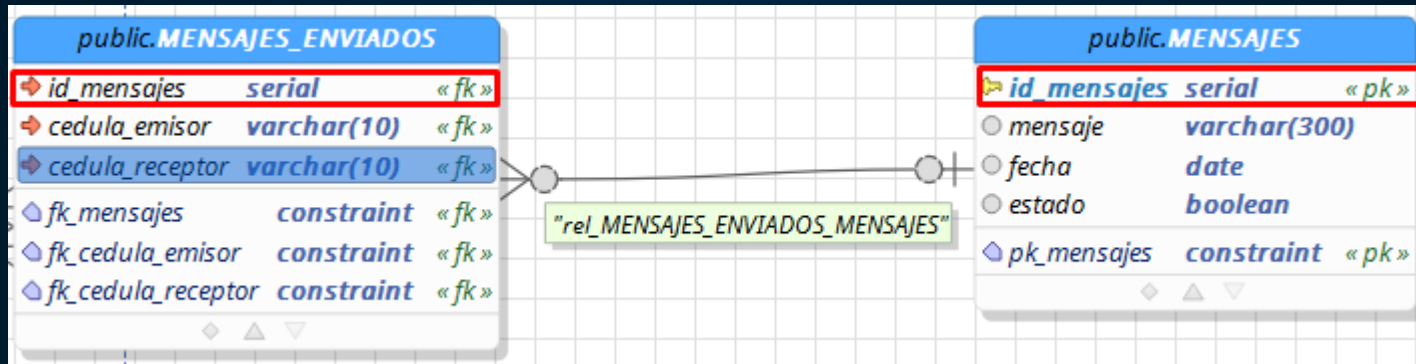
Tipo	Descripción
box	caja rectangular en el plano
cidr	dirección de red o de <i>host</i> en IP versión 4
circle	círculo en el plano
inet	dirección de red o de <i>host</i> en IP versión 4
int8	entero de ocho bytes con signo
line	línea infinita en el plano
lseg	segmento de línea en el plano
path	trayectoria geométrica, abierta o cerrada, en el plano
point	punto geométrico en el plano
polygon	trayectoria geométrica cerrada en el plano
serial	identificador numerico único

### Tipos de datos del estándar SQL3 en PostgreSQL

Tipo en Postgres	Correspondiente en SQL3	Descripción
bool	boolean	valor lógico o booleano (true/false)
char(n)	character(n)	cadena de caracteres de tamaño fijo
date	date	fecha (sin hora)
float4/8	float(86#86)	número de punto flotante con precisión 86#86
float8	real, double precision	número de punto flotante de doble precisión
int2	smallint	entero de dos bytes con signo
int4	int, integer	entero de cuatro bytes con signo
int4	decimal(87#87)	número exacto con 88#88
int4	numeric(87#87)	número exacto con 89#89
money	decimal(9,2)	cantidad monetaria
time	time	hora en horas, minutos, segundos y centésimas
timespan	interval	intervalo de tiempo
timestamp	timestamp with time zone	fecha y hora con zonificación
varchar(n)	character varying(n)	cadena de caracteres de tamaño variable



## Tipos de datos



**Primary key (PK):** identificador único de una fila de una tabla de base de datos, que no puede ser nulo, permite también relacionar entre tablas y puede estar conformada por 1 o más campos.

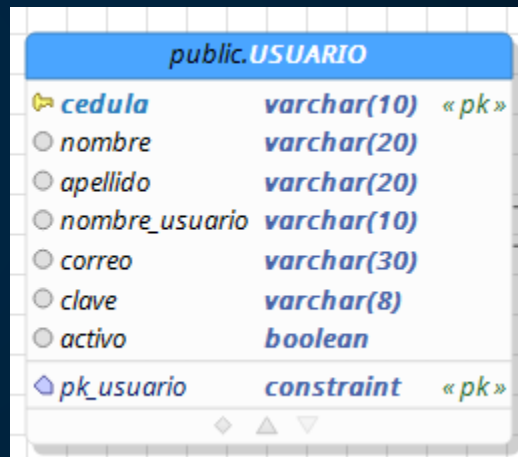
**Foreign Key (FK):** Establece el vínculo entre dos tablas por medio de una PK. La tabla madre es aquella que tiene una PK, y la FK es la tabla hija, deben coincidir en tipo de dato, tamaño y valor. La tabla hija se vuelve dependiente de la otra











## Tablas

Son objetos de bases de datos que contienen todos los datos en filas y columnas. Existen tablas con particiones, repartiendo en unidades la información, facilitando la administración de los datos y haciendo que el proceso de acceder pueda ser más rápida y eficaz. Esto puede ayudar al mantenimiento de sistemas grandes y complejos.

Existen también tablas temporales las cuales permiten restaurar datos en cualquier punto del pasado, funciona como un tipo de historial. En Postgres una tabla temporal puede vivir durante el tiempo de una sesión o durante una transacción. En Oracle es similar, sin embargo, aunque los datos se eliminen la tabla sigue existiendo.



A screenshot of a database interface showing the structure of a table named 'public.USUARIO'. The table has several columns with their respective data types and constraints. The 'cedula' column is marked as the primary key with a yellow key icon and '«pk»'. The 'pk\_usuario' column is marked as a foreign key constraint with a purple diamond icon and '«pk»'. The 'activo' column is of type 'boolean'.

public.USUARIO		
 <b>cedula</b>	<b>varchar(10)</b>	<b>«pk»</b>
 <b>nombre</b>	<b>varchar(20)</b>	
 <b>apellido</b>	<b>varchar(20)</b>	
 <b>nombre_usuario</b>	<b>varchar(10)</b>	
 <b>correo</b>	<b>varchar(30)</b>	
 <b>clave</b>	<b>varchar(8)</b>	
 <b>activo</b>	<b>boolean</b>	
 <b>pk_usuario</b>	<b>constraint</b>	<b>«pk»</b>



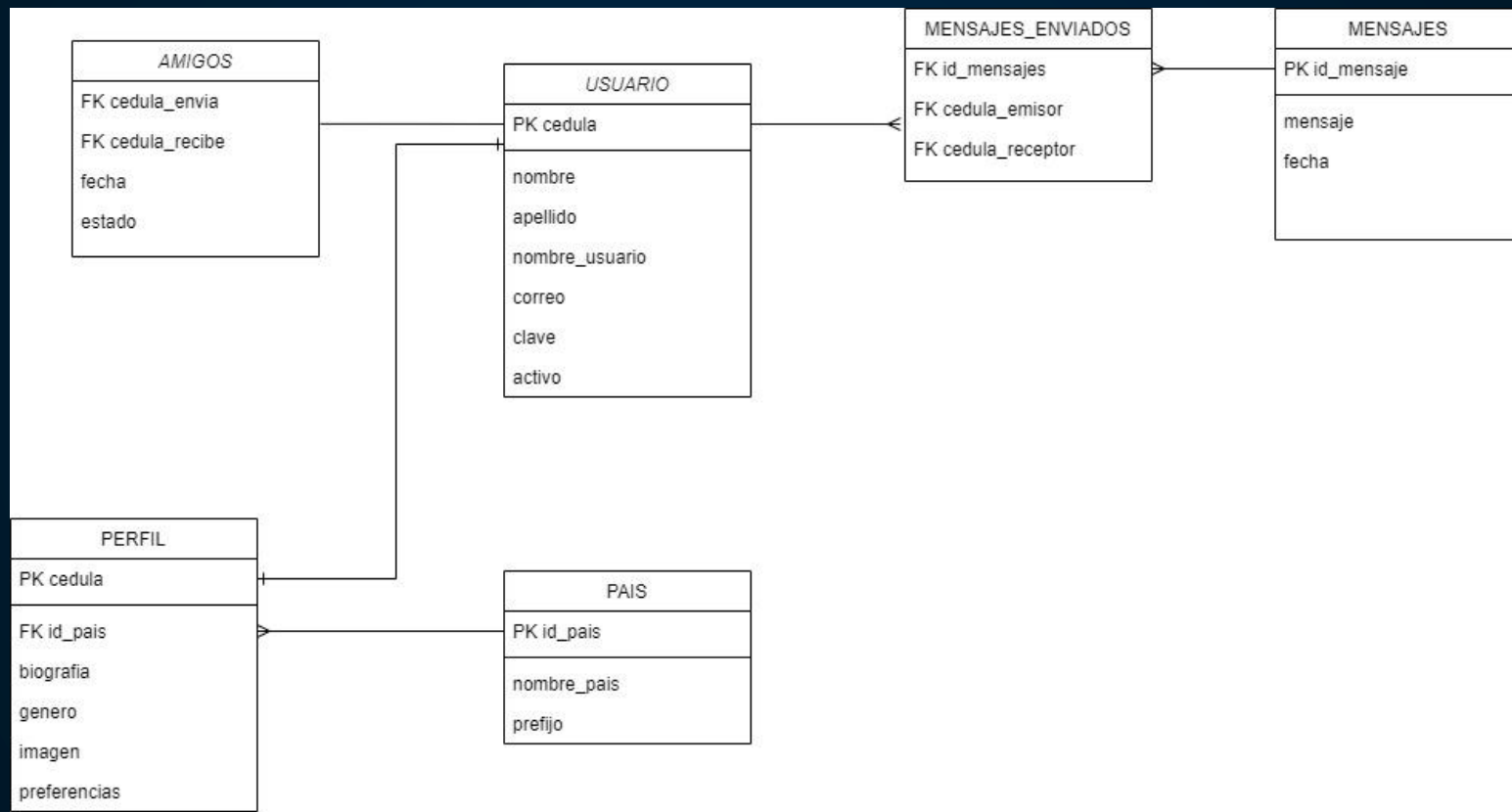
## Asignar tipos

El modelo relacional sobre redes sociales construido en ocasiones anteriores nos permite conocer la relación (uno a muchos, muchos a muchos) y los atributos, más no es posible identificar los tipos que tendrá cada atributo.

Modifiquemos el modelo de forma que sea posible conocer el tipo de los atributos y agreguemos además los atributos que hacen falta, para hacer nuestra base de datos más robusta y permitir más funcionalidades. Pensemos en la necesidad que tiene un usuario en una red social, validemos el modelo existente para así comprobar lo que falta agregar o en su defecto, eliminar.



## Asignar tipos





## Uso de modelos en el mundo real

Los modelos de bases de datos son el primer paso para definir un sistema, ayudan también en la construcción de la arquitectura, interfaz (donde se capturan y se muestran los datos) y componentes de software.

Son de gran utilidad, y deberían hacer parte de la definición inicial de un proyecto, ya que facilitará la construcción de los datos, se puede optimizar antes de implementarse y ayuda a detectar errores.

```
-- Database generated with pgModeler (PostgreSQL Database Modeler).
-- pgModeler version: 0.9.4
-- PostgreSQL version: 13.0
-- Project Site: pgmodeler.io
-- Model Author: ---

-- Database creation must be performed outside a multi lined SQL file.
-- These commands were put in this file only as a convenience.
--
-- object: red_social | type: DATABASE --
-- DROP DATABASE IF EXISTS red_social;
CREATE DATABASE red_social;
-- ddl-end --

-- object: public."USUARIO" | type: TABLE --
-- DROP TABLE IF EXISTS public."USUARIO" CASCADE;
CREATE TABLE public."USUARIO" (
    cedula varchar(10) NOT NULL,
    nombre varchar(20),
    apellido varchar(20),
    nombre_usuario varchar(10),
    correo varchar(30),
    clave varchar(8),
    activo boolean,
    CONSTRAINT pk_usuario PRIMARY KEY (cedula)
);
-- ddl-end --
ALTER TABLE public."USUARIO" OWNER TO postgres;
-- ddl-end --

-- object: public."MENSAJES_ENVIADOS" | type: TABLE --
-- DROP TABLE IF EXISTS public."MENSAJES_ENVIADOS" CASCADE;
CREATE TABLE public."MENSAJES_ENVIADOS" (
    id_mensajes serial,
    cedula_emisor varchar(10),
    cedula_receptor varchar(10)
);
```

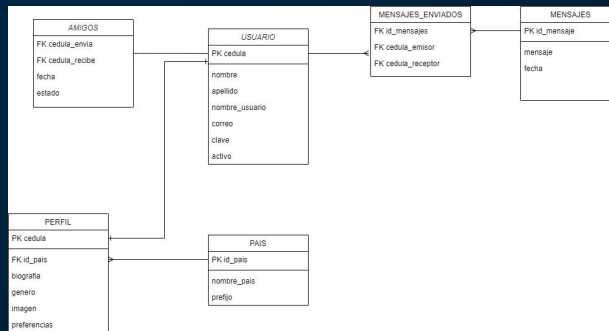
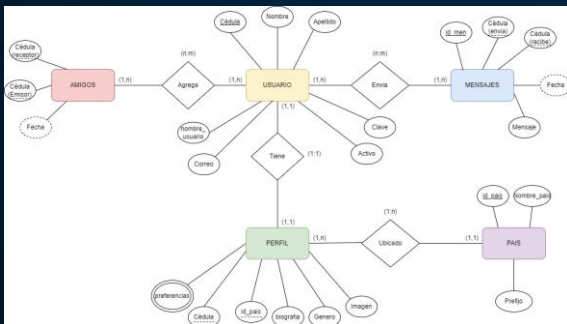


## Paso de modelo conceptual al lógico

El modelo conceptual consiste en el modelo entidad relación, donde definimos por medio de lenguaje natural como serán las tablas, sus posibles atributos y relaciones.

El modelo lógico es el modelo entidad relación, el cual puede ser trasladado al código de manera literal, ya sea utilizando una herramienta o llevando a cabo la ejecución de comandos para ello.

Traslademos el modelo relacional a un sistema gestor de base de datos, para que empecemos a interactuar.





## Paso de modelo conceptual al lógico

**\*Esta práctica puede variar según el motor de base de datos utilizado\***

Para esta práctica se utilizará PostgreSQL, ya que es opensource, muy similar a Oracle y utilizada en la industria con mayor frecuencia.

Comando de gran utilidad

Conectarse por consola a la base de datos:

```
psql nombre_bd usuario
```

Listar todas las bases de datos disponibles

```
\l
```

Cambiarse de base de datos

```
\c nombre_bd
```





## Paso de modelo conceptual al lógico

1. Comando para crear base de datos:

```
CREATE DATABASE red_social;
```

2. Comando para crear tabla

```
CREATE TABLE "USUARIO" (  
    cedula varchar(10) NOT NULL,  
    nombre varchar(20),  
    apellido varchar(20),  
    nombre_usuario varchar(10),  
    correo varchar(30),  
    clave varchar(8),  
    activo boolean,  
    CONSTRAINT pk_usuario PRIMARY KEY (cedula) -- esta constraint permite  
    definir el atributo que será asignado como llave primaria, recordemos que  
    no puede ser null  
);
```



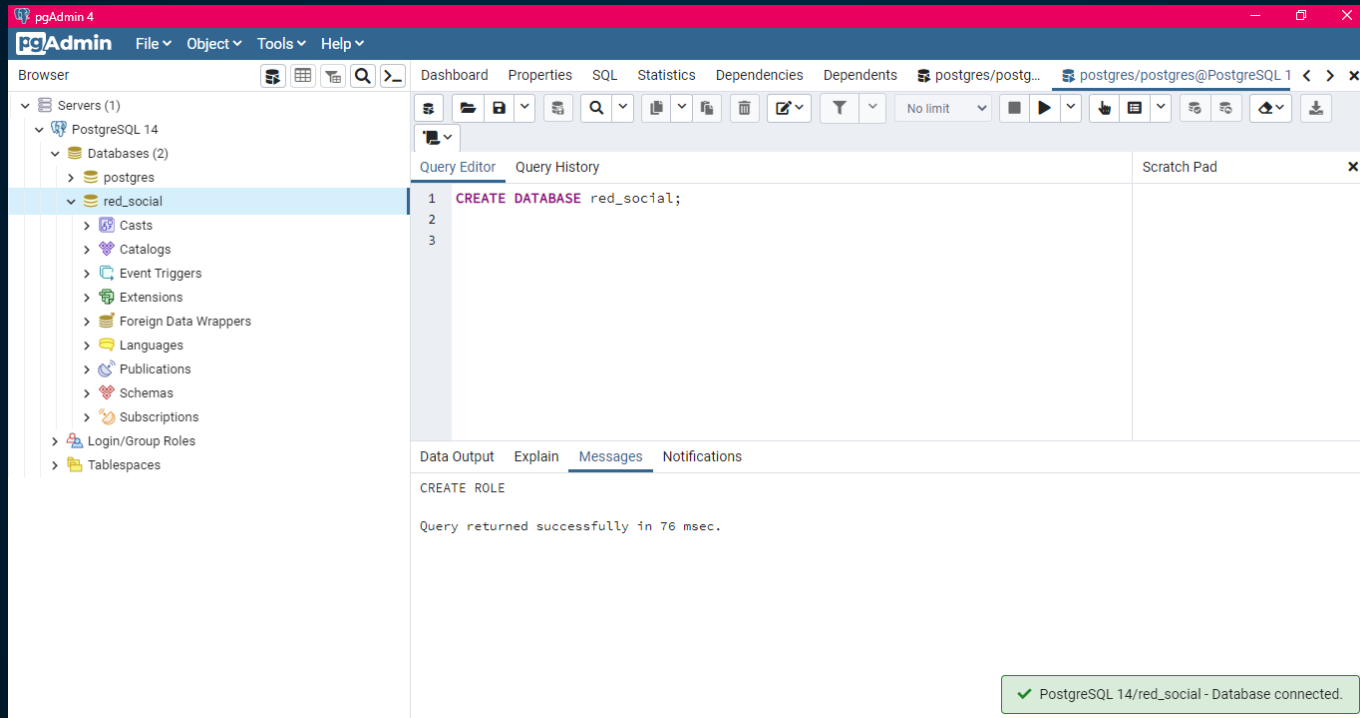
## Paso de modelo conceptual al lógico

3. Asignar llaves foráneas (aquí ya deben estar creadas las demás tablas)

```
ALTER TABLE "MENSAJES_ENVIADOS" ADD CONSTRAINT fk_mensajes FOREIGN KEY  
(id_mensajes)  
REFERENCES "MENSAJES" (id_mensajes) MATCH FULL  
ON DELETE NO ACTION ON UPDATE NO ACTION;
```



## Paso de modelo conceptual al lógico



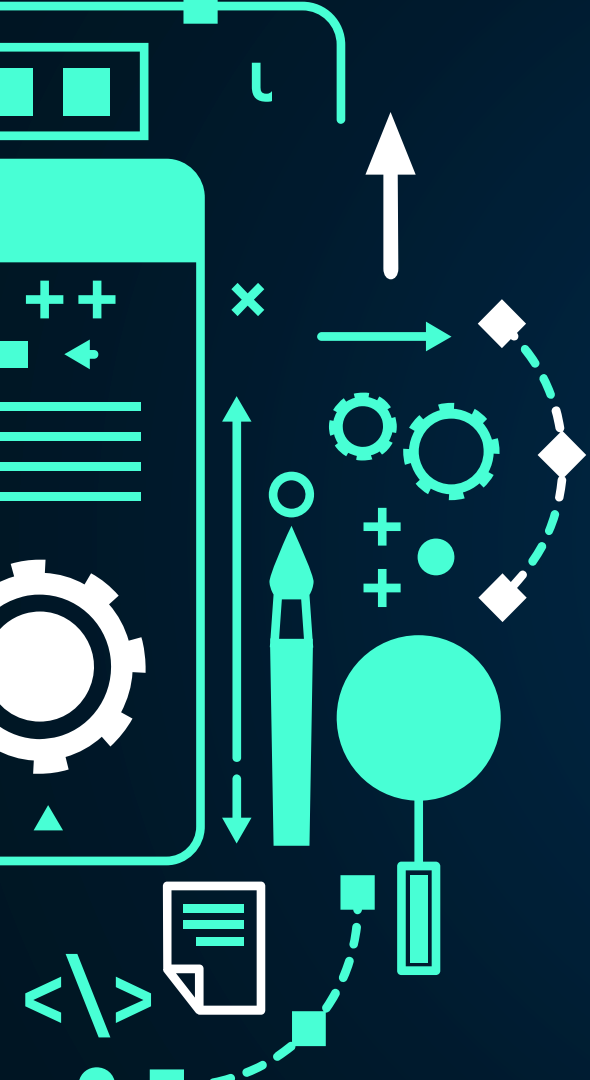
Cliente para PostgreSQL, PgAdmin4



## REFERENCIAS

- Git-Flow cheatsheet: <http://danielkummer.github.io/git-flow-cheatsheet/>
- Flujo de trabajo de Gitflow: <https://www.atlassian.com/es/git/tutorials/comparing-workflows/gitflow-workflow>
- ¿Qué es git?: <https://www.atlassian.com/es/git/tutorials/what-is-git>
- Tipos de datos relevantes en PostgreSQL: <https://www.ibiblio.org/pub/linux/docs/LuCaS/Tutoriales/NOTAS-CURSO-BBDD/notas-curso-BD/node134.html>
- Descargar PostgreSQL: <https://www.postgresql.org/download/>
- Documentación SQL – Tablas: <https://docs.microsoft.com/es-es/sql/relational-databases/tables/tables?view=sql-server-ver15#:~:text=Las%20tablas%20son%20objetos%20de,un%20campo%20dentro%20del%20registro.>
- El significado de base de datos relacional: <https://www.oracle.com/co/database/what-is-a-relational-database/>

-



# Gracias!