

TALLER BASES DE DATOS  
ING. WILLIAM ALEXANDER MATA LLANA PORRAS

**TALLER MYSQL VS POSTGRES QL**

**ASIGNATURA LINEA DE INVESTIGACIÓN III  
JUAN MANUEL NAVARRO CORDOBA  
561220150**

**UNIVERSIDAD DE CUNDINAMARCA EXTENSIÓN CHIA PROGRAMA DE  
INGENIERÍA DE SISTEMAS FACULTAD DE INGENIERÍA**

**2025**

## TALLER: Explorando las diferencias entre MySQL y PostgreSQL con Docker

### Objetivo general

Reconocer las principales diferencias entre los motores de base de datos MySQL y PostgreSQL, ejecutarlos en contenedores Docker, y explorar sus comandos básicos en la terminal interactiva de PostgreSQL, complementando con el uso de un cliente gráfico (pgAdmin o DBeaver).

### Objetivos específicos

- Comprender los conceptos generales de MySQL y PostgreSQL.
- Comparar su estructura, características, ventajas y comandos principales.
- Explorar los comandos básicos desde la terminal interactiva (psql).
- Conectarse a la base de datos con un cliente gráfico externo (pgAdmin o DBeaver).

### Parte 1. Consulta guiada

Investiga los siguientes aspectos sobre MySQL y PostgreSQL, consúltalos en fuentes confiables (documentación oficial, blogs técnicos, etc.) y completa la siguiente tabla. Adjunta una imagen o captura de la fuente consultada o del entorno donde verificaste la información.

Aspecto a comparar	MySQL	PostgreSQL
Tipo de sistema (relacional / mixto)	MySQL es un sistema de bases de datos relacional (RDBMS).	PostgreSQL es un sistema objeto-relacional (ORDBMS), que extiende lo relacional con características de objetos (herencia de tablas, tipos definidos por el usuario, etc.).
Licencia	MySQL usa un modelo de <b>licenciamiento dual</b> : la edición comunitaria (MySQL Community) es de código abierto bajo <b>GPLv2</b> (o compatible), y además Oracle ofrece versiones comerciales con características adicionales.	PostgreSQL se distribuye bajo la <b>Licencia PostgreSQL</b> , una licencia permisiva estilo Berkeley (BSD-like), que permite mayor libertad para uso, redistribución, modificación, incluso en software cerrado.
Enfoque principal	MySQL tiende a optimizar para facilidad de uso, rendimiento en escenarios comunes (web apps), velocidad en consultas simples, soporte amplio de aplicaciones.	PostgreSQL prioriza la robustez, cumplimiento de estándares SQL, extensibilidad, complejidad de consultas, integridad, funciones avanzadas y características que lo hacen más "completo".
Tipos de datos admitidos	MySQL ofrece tipos numéricos (INT, BIGINT, DECIMAL, FLOAT, DOUBLE), cadenas (CHAR, VARCHAR, TEXT), fecha/hora (DATE, DATETIME, TIMESTAMP, TIME, YEAR), tipos binarios, tipos espaciales/geométricos, JSON (a partir de versiones recientes)	PostgreSQL ofrece una gama más amplia y flexible de tipos: tipos numéricos arbitrarios (numeric/decimal de precisión arbitraria), boolean, arrays, tipos compuestos, hstore (clave-valor), JSON/JSONB, tipos geométricos, tipos de red (IP, CIDR), rangos, XML, UUID, herencias, tipos definidos por usuarios.
Integridad referencial	MySQL con el motor InnoDB soporta claves foráneas, constraints (PRIMARY KEY, FOREIGN KEY, UNIQUE, NOT NULL, CHECK) (aunque algunas versiones antiguas o motores diferentes no lo soportaban)	PostgreSQL tiene un sistema fuerte de integridad referencial, con soporte consistente de claves foráneas, constraints (CHECK, UNIQUE, NOT NULL), triggers, reglas, etc. Además, herencia de tablas (aunque con algunas limitaciones) permite modelos

TALLER BASES DE DATOS  
ING. WILLIAM ALEXANDER MATA LLANA PORRAS

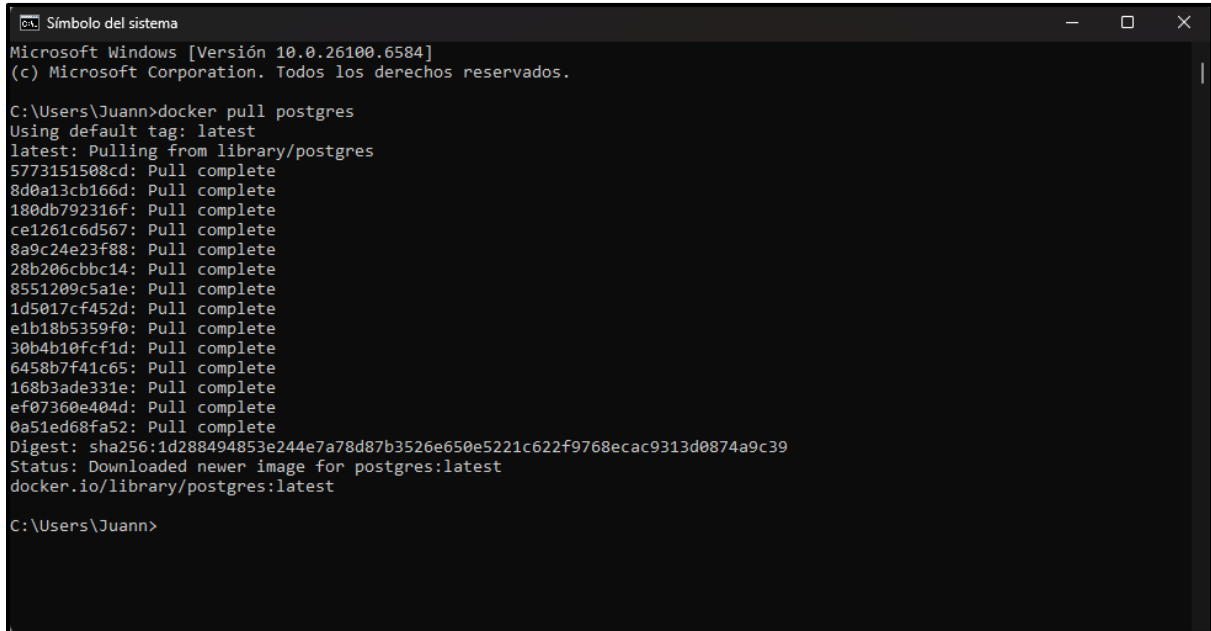
		avanzados.
Soporte de JSON y datos complejos	MySQL soporta tipo <b>JSON</b> desde versiones modernas.	PostgreSQL tiene soporte muy robusto para JSON y JSONB, con funciones, operadores, indexación, capacidad de modificar partes del JSON, consulta eficiente.
Soporte para funciones y procedimientos	MySQL soporta funciones y procedimientos almacenados (stored functions and stored procedures) mediante CREATE FUNCTION, CREATE PROCEDURE, con parámetros IN/OUT, etc.	PostgreSQL tiene un soporte muy poderoso de funciones definidas por el usuario (con PL/pgSQL y otros lenguajes como PL/Perl, PL/Python, etc.), funciones SQL, triggers, procedimientos, agregados definidos por el usuario.
Nivel de cumplimiento del estándar SQL	MySQL es parcialmente compatible con el estándar SQL, no implementa todas las características del estándar, y a veces tiene extensiones propias. PostgreSQL tiene un alto nivel de cumplimiento de estándares SQL y es considerado más cercano al estándar que MySQL.	PostgreSQL tiene un alto nivel de cumplimiento de estándares SQL y es considerado más cercano al estándar que MySQL.
Extensiones disponibles	MySQL tiene un sistema de plugins/almacenaje de motores (storage engines como InnoDB, MyISAM, Memory, etc.), extensiones limitadas comparadas con Postgres.	PostgreSQL tiene un sistema de extensiones muy potente: módulos como PostGIS, pg_trgm, citext, foreign data wrappers (FDW), y muchas extensiones contrib y externas.
Uso recomendado	MySQL es ampliamente usado en aplicaciones web, sistemas CMS (WordPress, Drupal, etc.), escenarios de lectura intensiva, donde la simplicidad, rendimiento y compatibilidad son importantes.	PostgreSQL es adecuado cuando se necesitan consultas complejas, integridad fuerte, extensibilidad, trabajos analíticos, requerimientos de datos más sofisticados (geoespaciales, JSON, tipos personalizados).

## Parte 2. Exploración práctica con Docker y PostgreSQL

Realiza la configuración de tu entorno Docker para levantar una instancia de PostgreSQL. Registra los pasos ejecutados y adjunta un pantallazo de la ejecución exitosa.

Espacio para evidencias:

**Descargar la imagen oficial de PostgreSQL:**

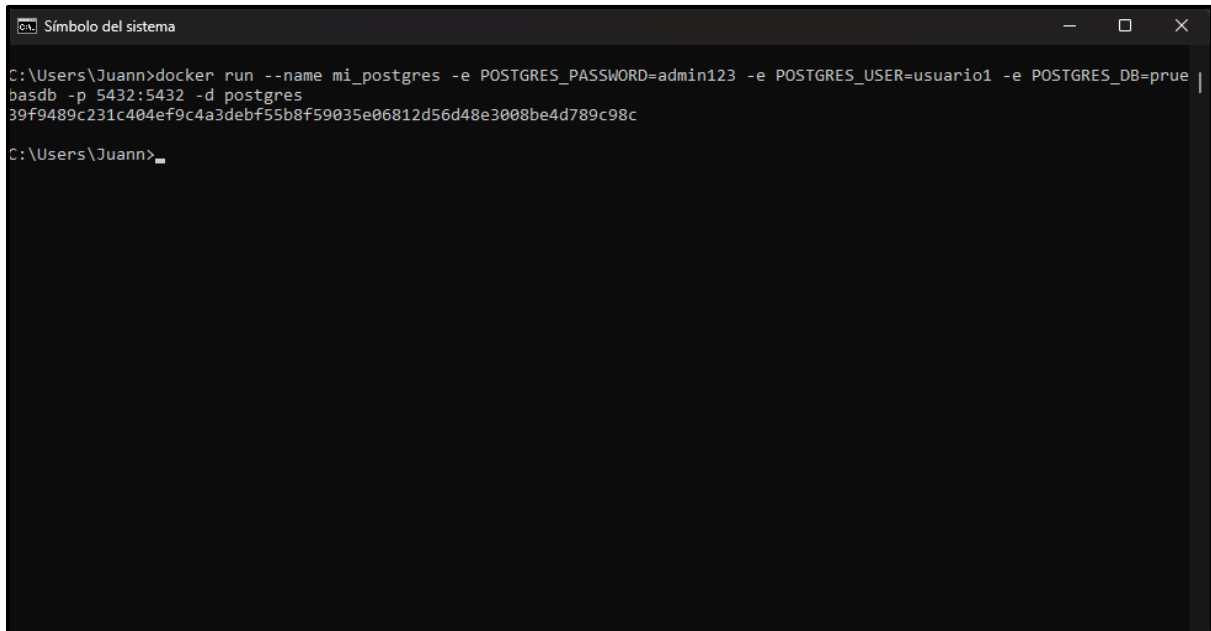


```
Símbolo del sistema
Microsoft Windows [Versión 10.0.26100.6584]
(c) Microsoft Corporation. Todos los derechos reservados.

C:\Users\Juann>docker pull postgres
Using default tag: latest
latest: Pulling from library/postgres
5773151508cd: Pull complete
8d0a13cb166d: Pull complete
180db792316f: Pull complete
ce1261c6d567: Pull complete
8a9c24e23f88: Pull complete
28b206cbbc14: Pull complete
8551209c5a1e: Pull complete
1d5017cf452d: Pull complete
e1b18b5359f0: Pull complete
30b4b10fcf1d: Pull complete
6458b7f41c65: Pull complete
168b3ade331e: Pull complete
ef07360e404d: Pull complete
0a51ed68fa52: Pull complete
Digest: sha256:1d288494853e244e7a78d87b3526e650e5221c622f9768ecac9313d0874a9c39
Status: Downloaded newer image for postgres:latest
docker.io/library/postgres:latest

C:\Users\Juann>
```

**Ejecutar un contenedor de PostgreSQL:**



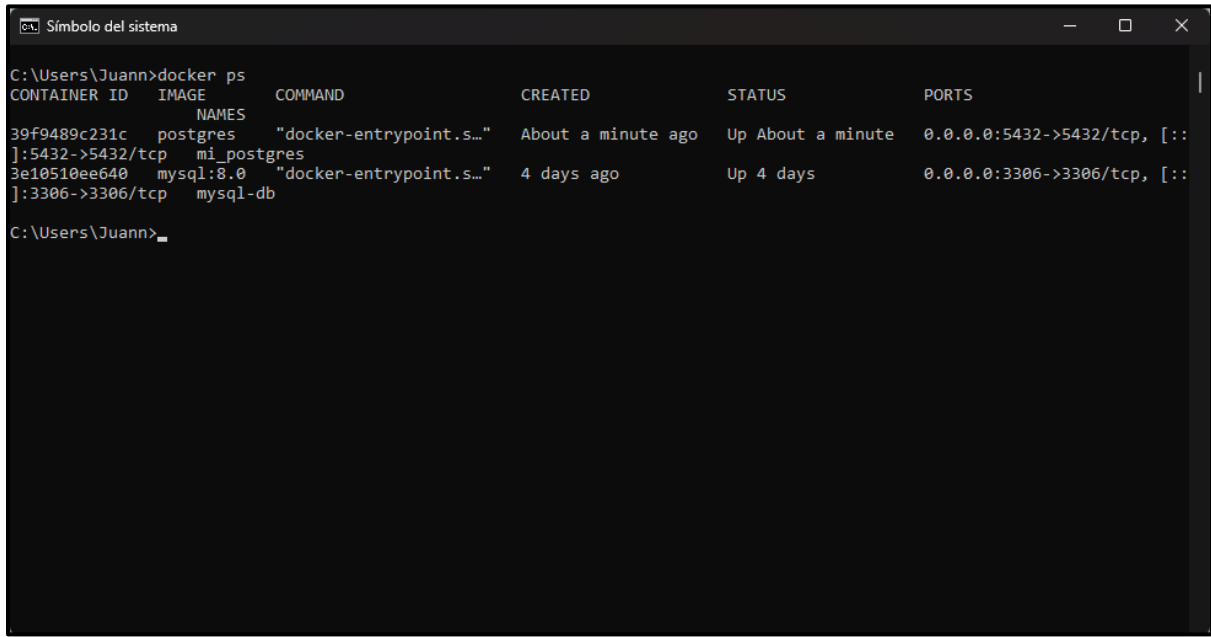
```
Símbolo del sistema

C:\Users\Juann>docker run --name mi_postgres -e POSTGRES_PASSWORD=admin123 -e POSTGRES_USER=usuario1 -e POSTGRES_DB=prue
basdb -p 5432:5432 -d postgres
39f9489c231c404ef9c4a3debf55b8f59035e06812d56d48e3008be4d789c98c

C:\Users\Juann>
```

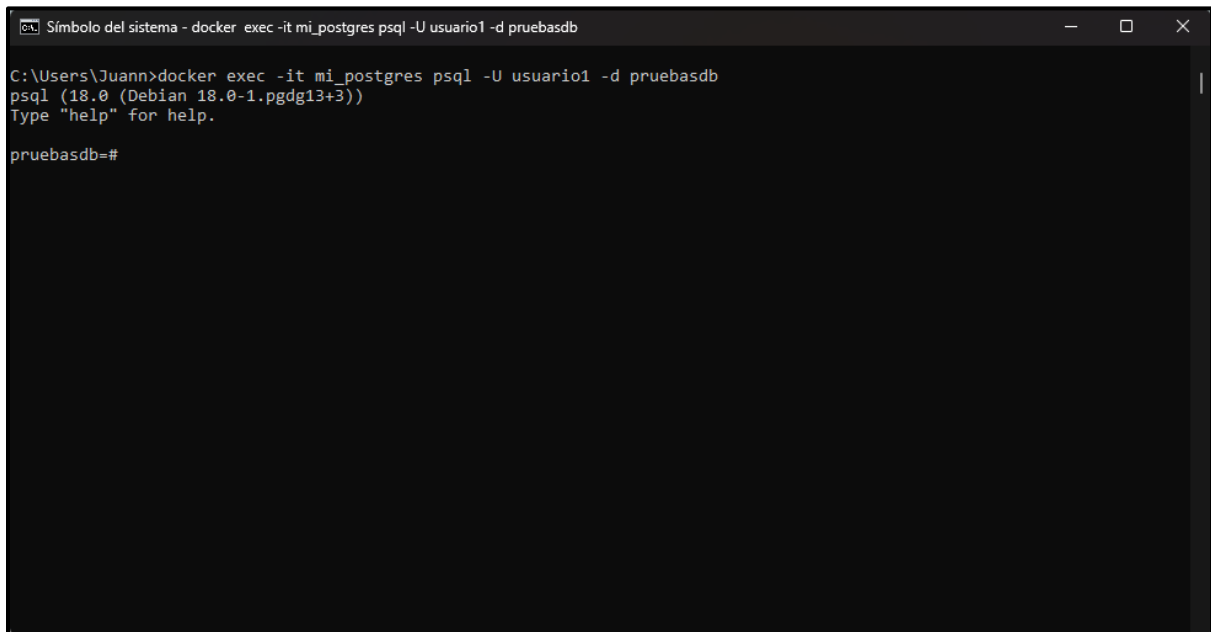
TALLER BASES DE DATOS  
ING. WILLIAM ALEXANDER MATALLANA PORRAS

**Verificar que el contenedor esté corriendo:**



```
Símbolo del sistema
C:\Users\Juann>docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED        STATUS        PORTS
39f9489c231c   postgres  "docker-entrypoint.s..." About a minute ago Up About a minute 0.0.0.0:5432->5432/tcp, [::]:5432->5432/tcp
3e10510ee640   mysql:8.0 "docker-entrypoint.s..." 4 days ago     Up 4 days     0.0.0.0:3306->3306/tcp, [::]:3306->3306/tcp
C:\Users\Juann>
```

**Acceder al contenedor con la terminal interactiva de PostgreSQL (psql):**



```
Símbolo del sistema - docker exec -it mi_postgres psql -U usuario1 -d pruebasdb
C:\Users\Juann>docker exec -it mi_postgres psql -U usuario1 -d pruebasdb
psql (18.0 (Debian 18.0-1.pgdg13+3))
Type "help" for help.

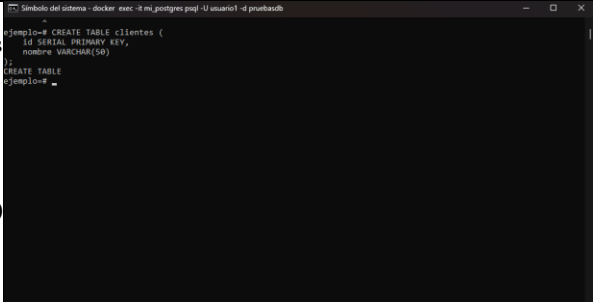
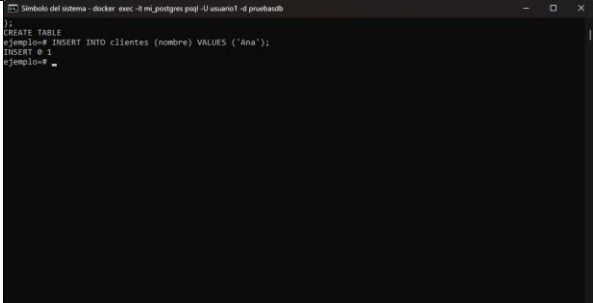
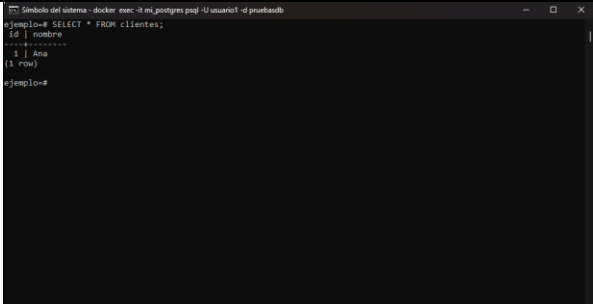
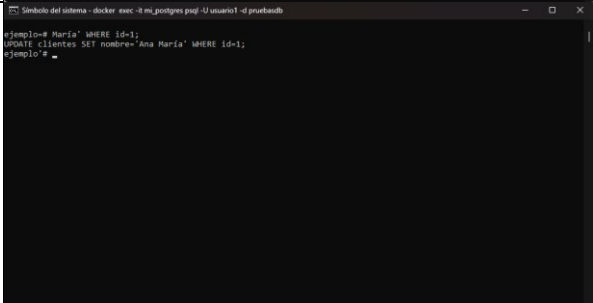
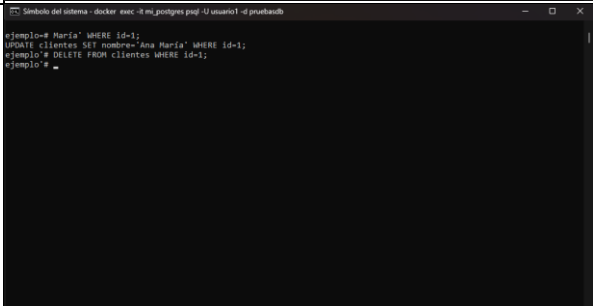
pruebasdb=#
```

Parte 3. Exploración en la terminal interactiva (psql)

Accede al contenedor de PostgreSQL y explora la terminal interactiva. Ejecuta los comandos básicos para gestionar bases de datos, usuarios y tablas. Describe con tus palabras qué hace cada acción y adjunta la evidencia visual (pantallazo de la terminal).

Acción / Comando explorado	MSQL	PostgreSQL	Descripción o evidencia (pantallazo PostgreSQL)
Listar bases de datos	SHOW DATABASES;	l o \list	
Crear una base de datos	CREATE DATABASE ejemplo;	CREATE DATABASE ejemplo;	
Conectarse a una base específica	USE ejemplo;	\c ejemplo	
Listar usuarios	SELECT user, host FROM mysql.user;	\du	
Crear un usuario nuevo	CREATE USER 'juan'@'localhost' IDENTIFIED BY 'clave123';	CREATE USER juan WITH PASSWORD 'clave123';	

TALLER BASES DE DATOS  
ING. WILLIAM ALEXANDER MATALLANA PORRAS

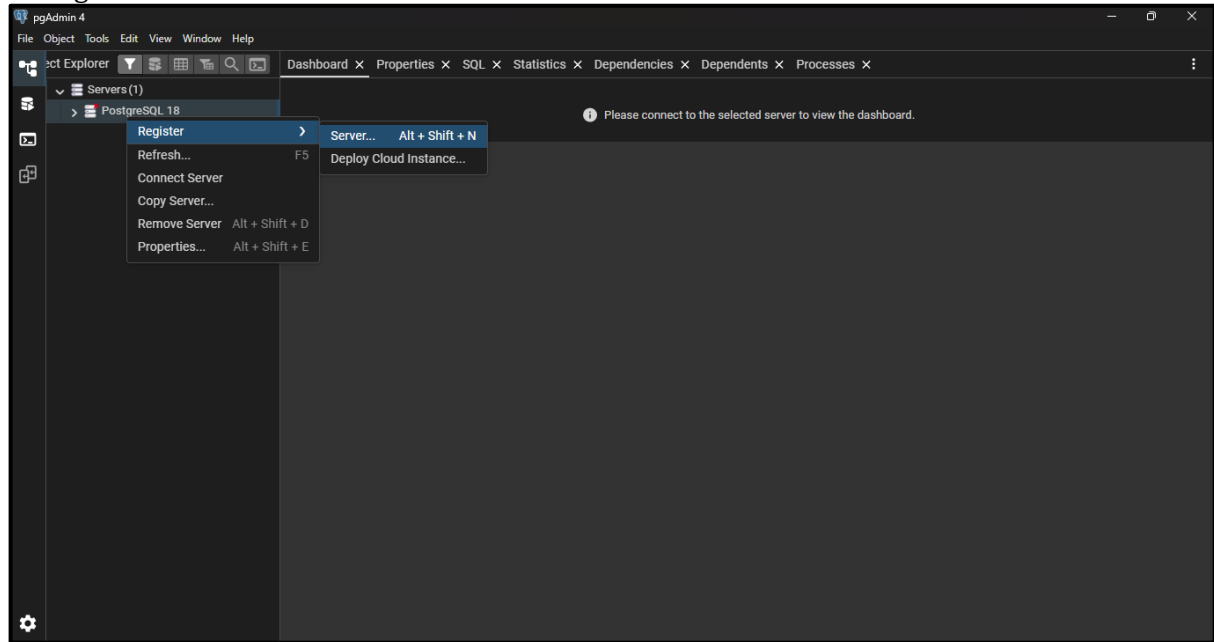
Crear una tabla	sql CREATE TABLE clientes (id INT PRIMARY KEY, nombre VARCHAR(50));	CREATE TABLE clientes ( id SERIAL PRIMARY KEY, nombre VARCHAR(50) );	
Insertar datos en una tabla	INSERT INTO clientes (id, nombre) VALUES (1, 'Ana');	INSERT INTO clientes (nombre) VALUES ( 'Ana'); (usar SERIAL para id autoincremental)	
Consultar registros	SELECT * FROM clientes;	SELECT * FROM clientes;	
Actualizar registros	María' WHERE id=1; UPDATE clientes SET nombre='Ana María' WHERE id=1;	María' WHERE id=1; UPDATE clientes SET nombre='Ana María' WHERE id=1;	
Eliminar registros	DELETE FROM clientes WHERE id=1;	DELETE FROM clientes WHERE id=1;	

#### Parte 4. Conexión con cliente gráfico (pgAdmin o DBeaver)

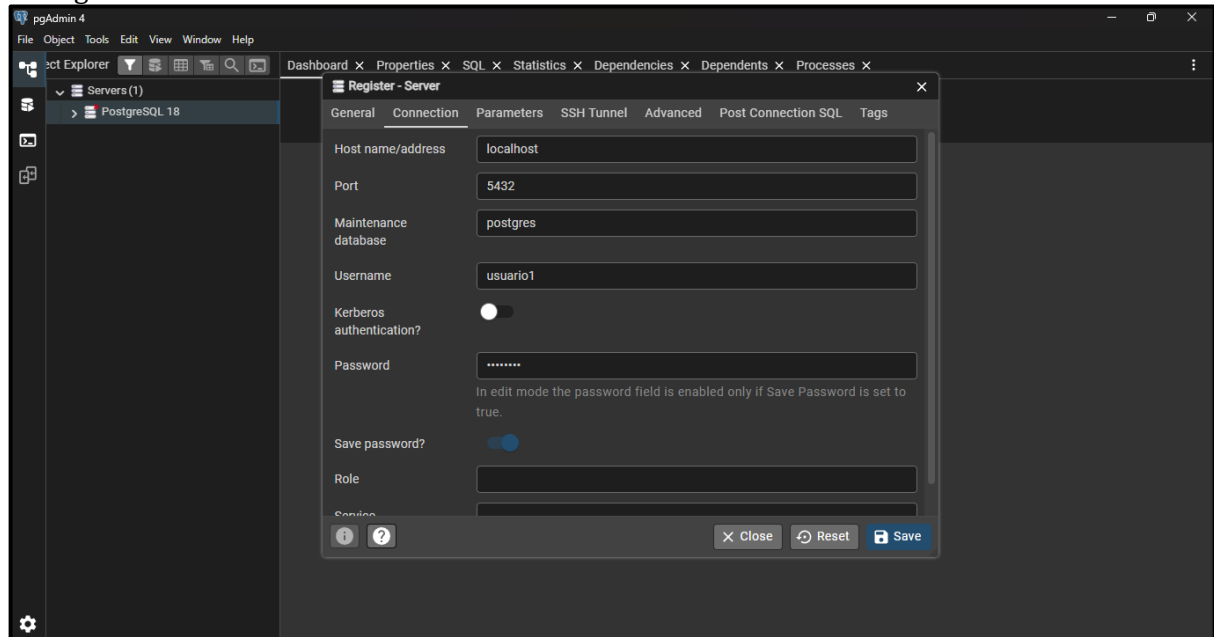
Conéctate a la base de datos PostgreSQL utilizando pgAdmin o DBeaver. Realiza la conexión, crea una base de datos de prueba y explora sus tablas y registros. Anexa pantallazos de la conexión y describe brevemente el proceso.

Evidencia visual:

Configuración del servidor



Configuración del usuario:

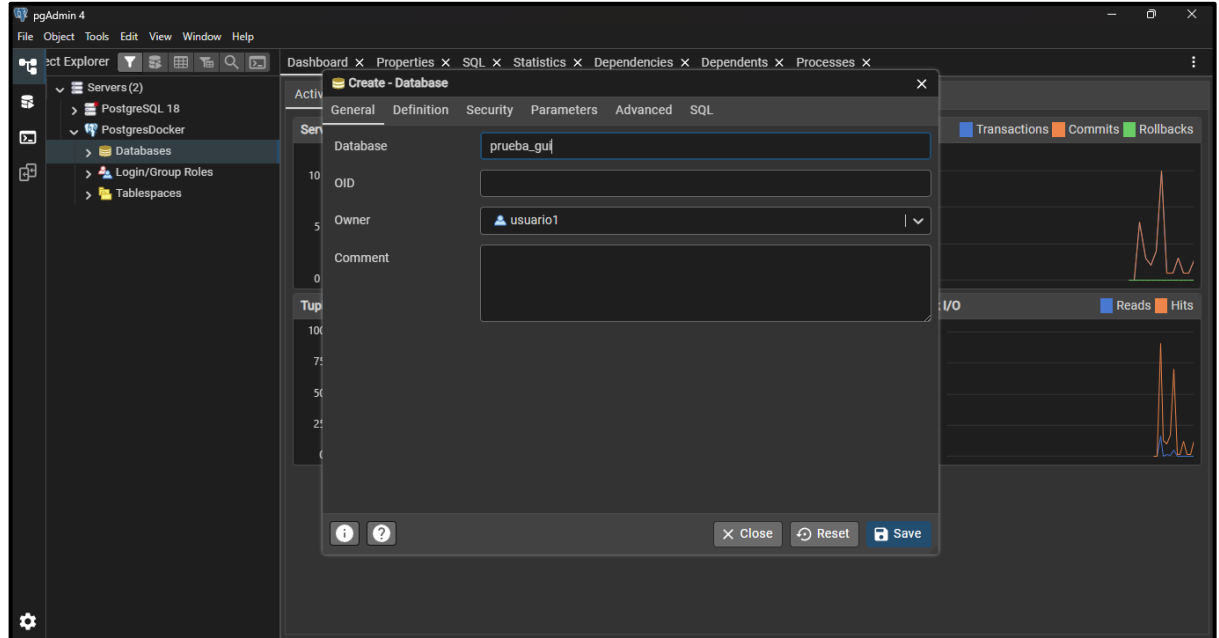




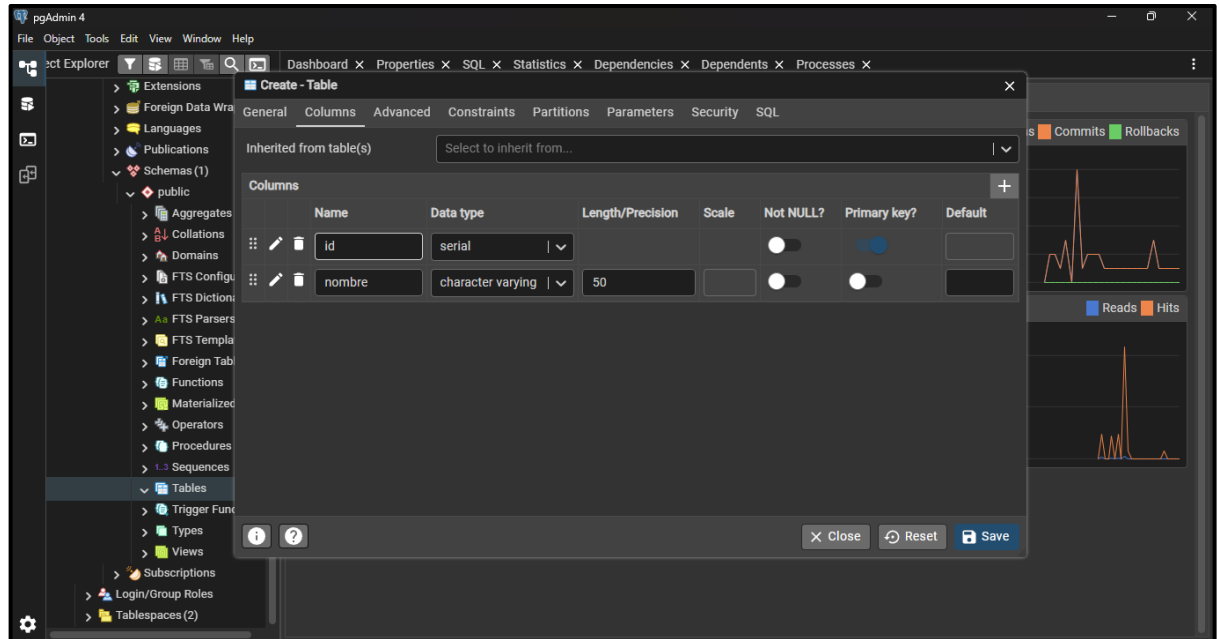
# TALLER BASES DE DATOS

## ING. WILLIAM ALEXANDER MATALLANA PORRAS

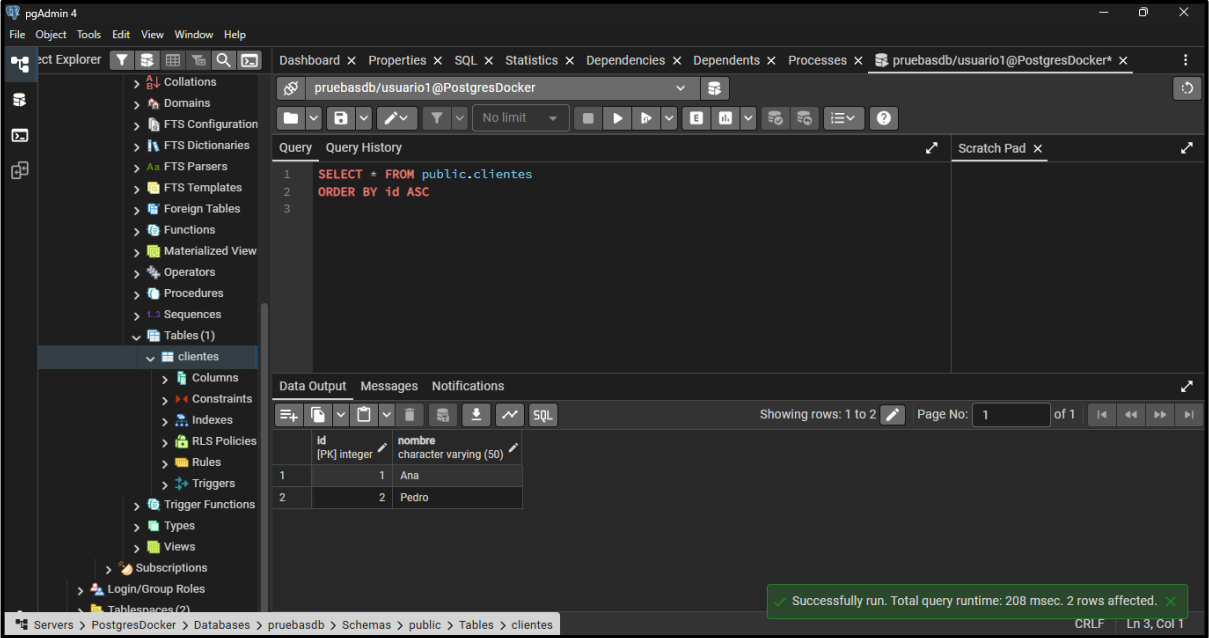
Creación de la base de datos:



Creación de la tabla:



Creación de registros en tabla:



## Parte 5. Actividades de aplicación

Realiza las siguientes tareas en tu entorno PostgreSQL y documenta los resultados con pantallazos o descripciones breves.

### Modelo Relacional (versión reducida)

#### 1. Autor

Campo	Tipo de Dato	Clave	Descripción
id_autor	INT	PK	Identificador del autor
nombre	VARCHAR(120)		Nombre completo del autor
nacionalidad	VARCHAR(80)		Nacionalidad (opcional)

#### 2. Libro

Campo	Tipo de Dato	Clave	Descripción
id_libro	INT	PK	Identificador del libro
titulo	VARCHAR(200)		Título del libro
isbn	VARCHAR(20)		Código ISBN (único)
anio_publicacion	INT		Año de publicación

TALLER BASES DE DATOS  
ING. WILLIAM ALEXANDER MATA LLANA PORRAS

id_autor	INT	FK	Autor del libro (1 autor por libro)
stock	INT		Unidades disponibles para préstamo (>=0)

### 3. Usuario

Campo	Tipo de Dato	Clave	Descripción
id_usuario	INT	PK	Identificador del usuario
documento	VARCHAR(30)		Documento de identidad (único)
nombre	VARCHAR(120)		Nombre completo
email	VARCHAR(120)		Correo electrónico (opcional)

### 4. Prestamo

*(Un registro por libro prestado a un usuario. Si un usuario lleva 2 libros, se crean 2 filas.)*

Campo	Tipo de Dato	Clave	Descripción
id_prestamo	INT	PK	Identificador del préstamo
id_usuario	INT	FK	Usuario que realiza el préstamo
id_libro	INT	FK	Libro prestado
fecha_prestamo	DATE		Fecha del préstamo

fecha_devolucion_max	DATE		Fecha límite de devolución
fecha_devuelto	DATE		Fecha de devolución efectiva (nullable)
estado	VARCHAR(15)		ABIERTO / CERRADO

### Relaciones

#### Autor — Libro

- Relación: Un autor tiene muchos libros.
- Cardinalidad: 1 a N.

#### Libro — Prestamo

- Relación: Un libro puede estar presente en muchos préstamos (una fila por unidad prestada).

- Cardinalidad: 1 a N.
- Regla de negocio sugerida: solo permitir préstamo si stock > 0 y al prestar disminuir stock en 1; al devolver, incrementar stock en 1.

### Usuario — Préstamo

- Relación: Un usuario puede tener muchos préstamos.
- Cardinalidad: 1 a N.

### Consultas a realizar

1. **Libros con su autor**  
Columnas: titulo, nombre\_autor, anio\_publicacion.
2. **Préstamos abiertos con datos del usuario y del libro**  
Columnas: id\_prestamo, usuario, titulo\_libro, fecha\_prestamo, fecha\_devolucion\_max, estado.
3. **Historial de préstamos de un usuario (por documento)**  
Dado un documento, listar titulo\_libro, fecha\_prestamo, fecha\_devuelto, estado.
4. **Top de autores por cantidad de libros registrados**  
Columnas: autor, cantidad\_libros. Ordenar descendente.
5. **Disponibilidad actual de cada libro**  
Columnas: titulo, stock. Filtrar libros con stock = 0 (sin unidades disponibles).
6. **Libros prestados actualmente (no devueltos)**  
Columnas: titulo, usuario, fecha\_prestamo, fecha\_devolucion\_max.  
(Pista: *prestamo.estado='ABIERTO' o fecha\_devuelto IS NULL según tu regla.*)
7. **Usuarios con cantidad de préstamos abiertos**  
Columnas: usuario, prestamos\_abiertos. (Agrupar por usuario.)
8. **Búsqueda por texto (título o autor)**  
Dado un término, mostrar titulo, autor, anio\_publicacion donde el término aparezca en título o nombre de autor (insensible a mayúsculas).
9. **Préstamos vencidos**  
Listar usuario, titulo, dias\_atraso cuando CURRENT\_DATE > fecha\_devolucion\_max y el préstamo siga abierto.
10. **Libros por década de publicación**  
Mostrar década (ej. 1990s, 2000s) y cantidad de libros. (Tip: *agrupa por anio\_publicacion/10.*)

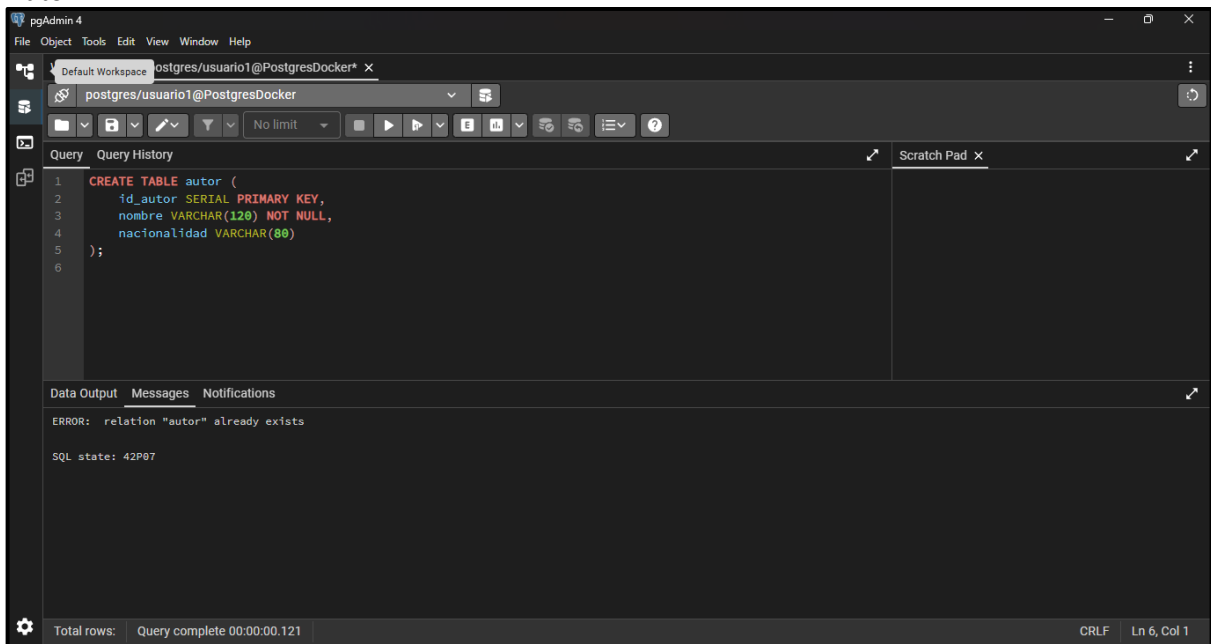
# TALLER BASES DE DATOS

## ING. WILLIAM ALEXANDER MATALLANA PORRAS

Evidencias visuales:

### Creación de las tablas

#### Autor



```
pgAdmin 4
File Object Tools Edit View Window Help

postgres/usuario1@PostgresDocker*
postgres/usuario1@PostgresDocker
No limit

Query Query History Scratch Pad X

1 CREATE TABLE autor (
2   id_autor SERIAL PRIMARY KEY,
3   nombre VARCHAR(120) NOT NULL,
4   nacionalidad VARCHAR(80)
5 );
6

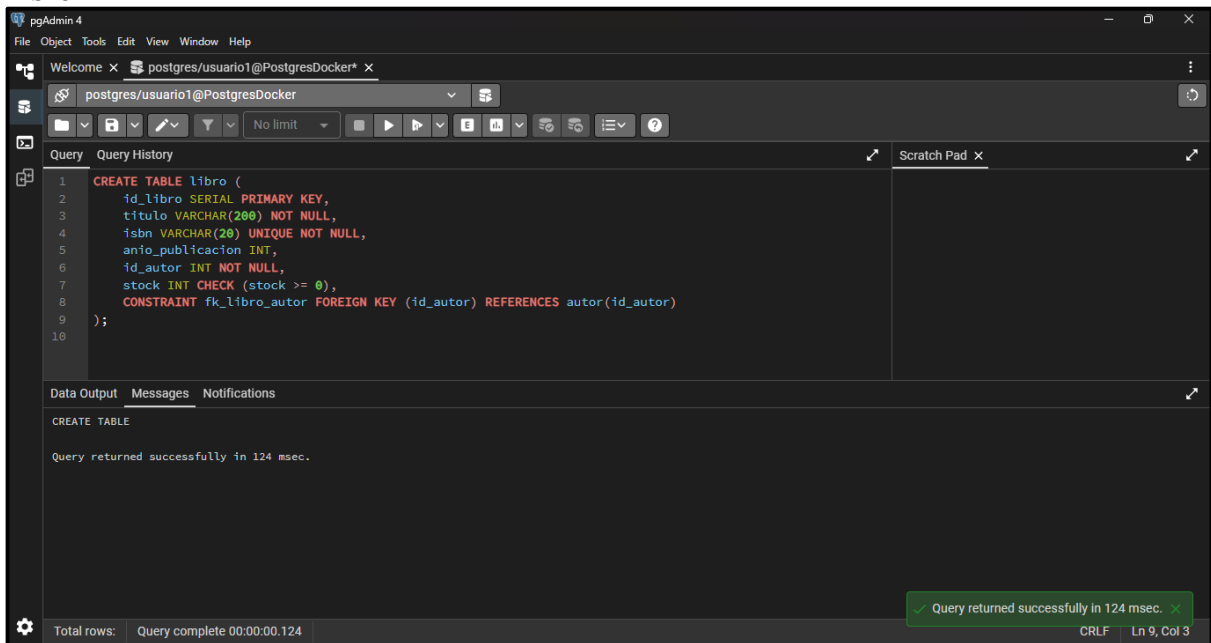
Data Output Messages Notifications

ERROR: relation "autor" already exists

SQL state: 42P07

Total rows: Query complete 00:00:00.121 CRLF Ln 6, Col 1
```

#### Libro



```
pgAdmin 4
File Object Tools Edit View Window Help

Welcome x postgres/usuario1@PostgresDocker*
postgres/usuario1@PostgresDocker
No limit

Query Query History Scratch Pad X

1 CREATE TABLE libro (
2   id_libro SERIAL PRIMARY KEY,
3   titulo VARCHAR(200) NOT NULL,
4   isbn VARCHAR(20) UNIQUE NOT NULL,
5   anio_publicacion INT,
6   id_autor INT NOT NULL,
7   stock INT CHECK (stock >= 0),
8   CONSTRAINT fk_libro_autor FOREIGN KEY (id_autor) REFERENCES autor(id_autor)
9 );
10

Data Output Messages Notifications

CREATE TABLE

Query returned successfully in 124 msec.

Total rows: Query complete 00:00:00.124 CRLF Ln 9, Col 3
```

# TALLER BASES DE DATOS

## ING. WILLIAM ALEXANDER MATALLANA PORRAS

### Usuario

The screenshot shows the pgAdmin 4 interface with the following details:

- Query Editor:** Contains the SQL command:

```
1 CREATE TABLE usuario (  
2     id_usuario SERIAL PRIMARY KEY,  
3     documento VARCHAR(30) UNIQUE NOT NULL,  
4     nombre VARCHAR(120) NOT NULL,  
5     email VARCHAR(120)  
6 );  
7
```
- Data Output:** Displays the message "CREATE TABLE" and "Query returned successfully in 121 msec."
- Status Bar:** Shows "Total rows: Query complete 00:00:00.121" and a green confirmation message "Query returned successfully in 121 msec."

### Préstamo

The screenshot shows the pgAdmin 4 interface with the following details:

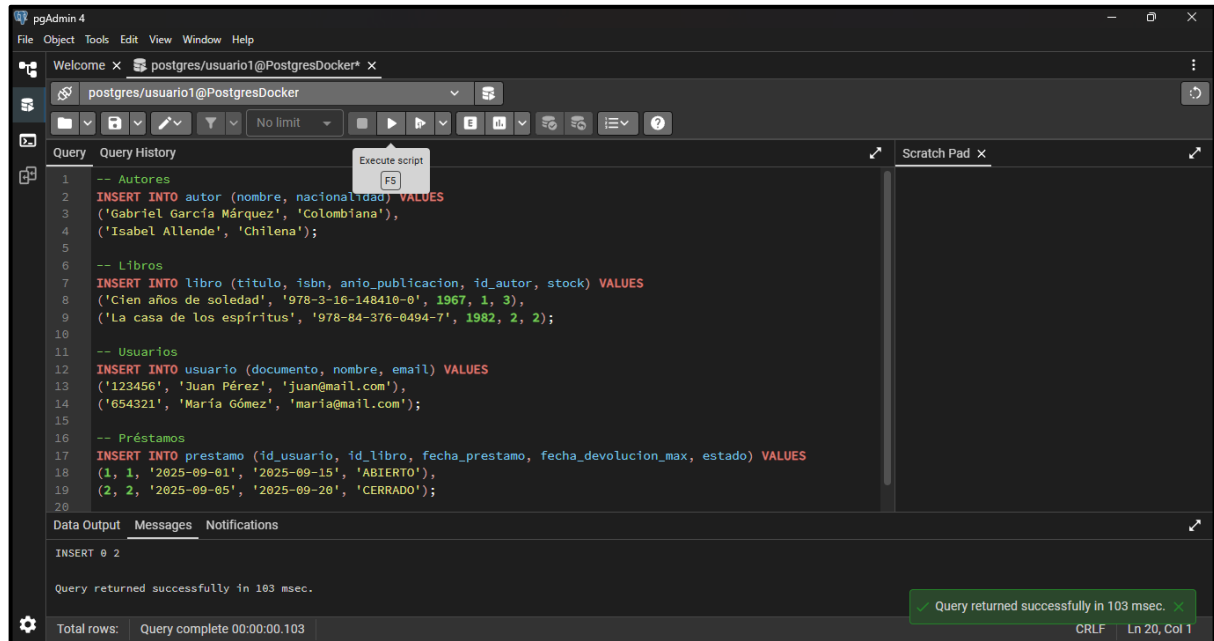
- Query Editor:** Contains the SQL command:

```
1 CREATE TABLE prestamo (  
2     id_prestamo SERIAL PRIMARY KEY,  
3     id_usuario INT NOT NULL,  
4     id_libro INT NOT NULL,  
5     fecha_prestamo DATE NOT NULL,  
6     fecha_devolucion_max DATE NOT NULL,  
7     fecha_devuelto DATE,  
8     estado VARCHAR(15) CHECK (estado IN ('ABIERTO', 'CERRADO')),  
9     CONSTRAINT fk_prestamo_usuario FOREIGN KEY (id_usuario) REFERENCES usuario(id_usuario),  
10    CONSTRAINT fk_prestamo_libro FOREIGN KEY (id_libro) REFERENCES libro(id_libro)  
11 );  
12
```
- Data Output:** Displays the message "CREATE TABLE" and "Query returned successfully in 75 msec."
- Status Bar:** Shows "Total rows: Query complete 00:00:00.075" and a green confirmation message "Query returned successfully in 75 msec."

# TALLER BASES DE DATOS

## ING. WILLIAM ALEXANDER MATALLANA PORRAS

### Insertar datos



```
1 -- Autores
2 INSERT INTO autor (nombre, nacionalidad) VALUES
3 ('Gabriel García Márquez', 'Colombiana'),
4 ('Isabel Allende', 'Chilena');
5
6 -- Libros
7 INSERT INTO libro (titulo, isbn, anio_publicacion, id_autor, stock) VALUES
8 ('Cien años de soledad', '978-3-16-148410-0', 1967, 1, 3),
9 ('La casa de los espíritus', '978-84-376-0494-7', 1982, 2, 2);
10
11 -- Usuarios
12 INSERT INTO usuario (documento, nombre, email) VALUES
13 ('123456', 'Juan Pérez', 'juan@mail.com'),
14 ('654321', 'María Gómez', 'maria@mail.com');
15
16 -- Préstamos
17 INSERT INTO prestamo (id_usuario, id_libro, fecha_prestamo, fecha_devolucion_max, estado) VALUES
18 (1, 1, '2025-09-01', '2025-09-15', 'ABIERTO'),
19 (2, 2, '2025-09-05', '2025-09-20', 'CERRADO');
20
```

Data Output Messages Notifications

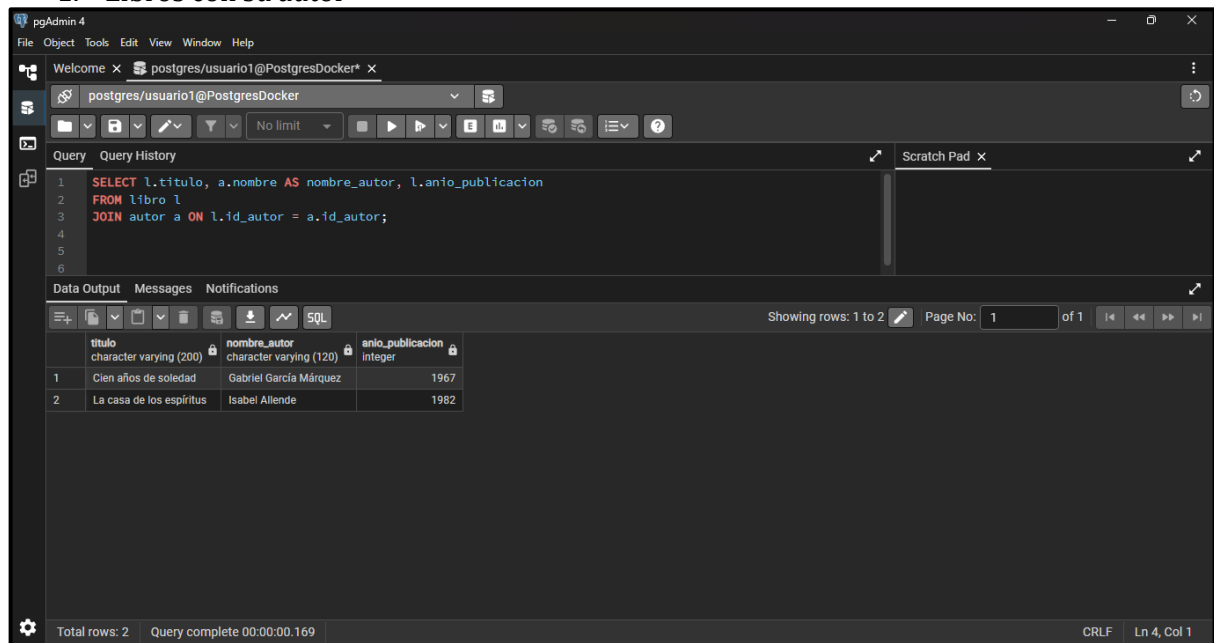
INSERT 0 2

Query returned successfully in 103 msec.

Total rows: Query complete 00:00:00.103

### CONSULTAS A REALIZAR

#### 1. Libros con su autor



```
1 SELECT l.titulo, a.nombre AS nombre_autor, l.anio_publicacion
2 FROM libro l
3 JOIN autor a ON l.id_autor = a.id_autor;
4
5
6
```

Data Output Messages Notifications

	titulo character varying (200)	nombre_autor character varying (120)	anio_publicacion integer
1	Cien años de soledad	Gabriel García Márquez	1967
2	La casa de los espíritus	Isabel Allende	1982

Showing rows: 1 to 2 Page No: 1 of 1

Total rows: 2 Query complete 00:00:00.169

# TALLER BASES DE DATOS

## ING. WILLIAM ALEXANDER MATALLANA PORRAS

### 2. Préstamos abiertos con datos del usuario y del libro

The screenshot shows the pgAdmin 4 interface with a SQL query executed. The query selects loan details along with user and book information for open loans. The results table shows one row of data.

```
1 SELECT p.id_prestamo, u.nombre AS usuario, l.titulo AS titulo_libro,
2       p.fecha_prestamo, p.fecha_devolucion_max, p.estado
3 FROM prestamo p
4 JOIN usuario u ON p.id_usuario = u.id_usuario
5 JOIN libro l ON p.id_libro = l.id_libro
6 WHERE p.estado = 'ABIERTO';
7
```

id_prestamo	usuario	titulo_libro	fecha_prestamo	fecha_devolucion_max	estado
1	Juan Pérez	Cien años de soledad	2025-09-01	2025-09-15	ABIERTO

Successfully run. Total query runtime: 110 msec. 1 rows affected.

### 3. Historial de préstamos de un usuario (por documento)

The screenshot shows the pgAdmin 4 interface with a SQL query executed. The query selects loan details for a specific user, ordered by document number. The results table shows one row of data.

```
1 SELECT l.titulo, p.fecha_prestamo, p.fecha_devuelto, p.estado
2 FROM prestamo p
3 JOIN usuario u ON p.id_usuario = u.id_usuario
4 JOIN libro l ON p.id_libro = l.id_libro
5 WHERE u.documento = '123456';
6
7
```

titulo	fecha_prestamo	fecha_devuelto	estado
Cien años de soledad	2025-09-01	[null]	ABIERTO

Successfully run. Total query runtime: 111 msec. 1 rows affected.



# TALLER BASES DE DATOS

## ING. WILLIAM ALEXANDER MATALLANA PORRAS

### 4. Top de autores por cantidad de libros registrados

The screenshot shows the pgAdmin 4 interface. The query editor contains the following SQL code:

```
1 SELECT a.nombre AS autor, COUNT(l.id_libro) AS cantidad_libros
2 FROM autor a
3 LEFT JOIN libro l ON a.id_autor = l.id_autor
4 GROUP BY a.nombre
5 ORDER BY cantidad_libros DESC;
```

The Data Output tab shows the results of the query:

	autor	cantidad_libros
1	Isabel Allende	1
2	Gabriel García Márquez	1

The status bar at the bottom indicates: Total rows: 2, Query complete 00:00:00.246. A green message box at the bottom right states: Successfully run. Total query runtime: 246 msec. 2 rows affected.

### 5. Disponibilidad actual de cada libro (stock = 0)

The screenshot shows the pgAdmin 4 interface. The query editor contains the following SQL code:

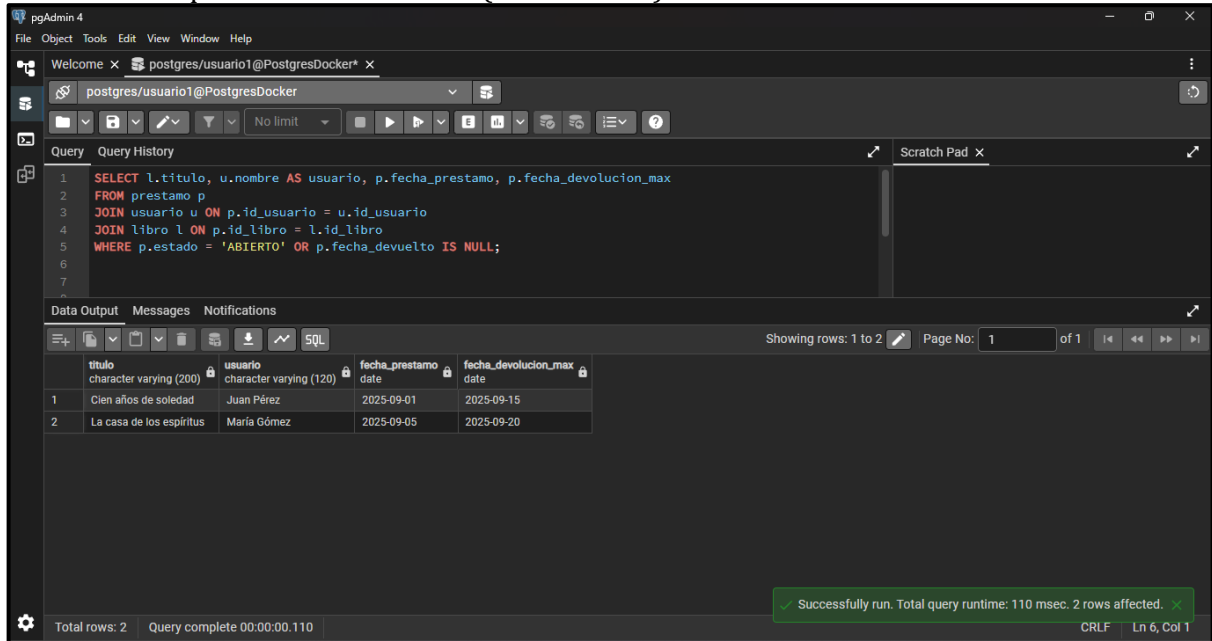
```
1 SELECT titulo, stock
2 FROM libro
3 WHERE stock = 0;
```

The Data Output tab shows the results of the query:

	titulo	stock
--	--------	-------

The status bar at the bottom indicates: Total rows: 0, Query complete 00:00:00.185.

## 6. Libros prestados actualmente (no devueltos)



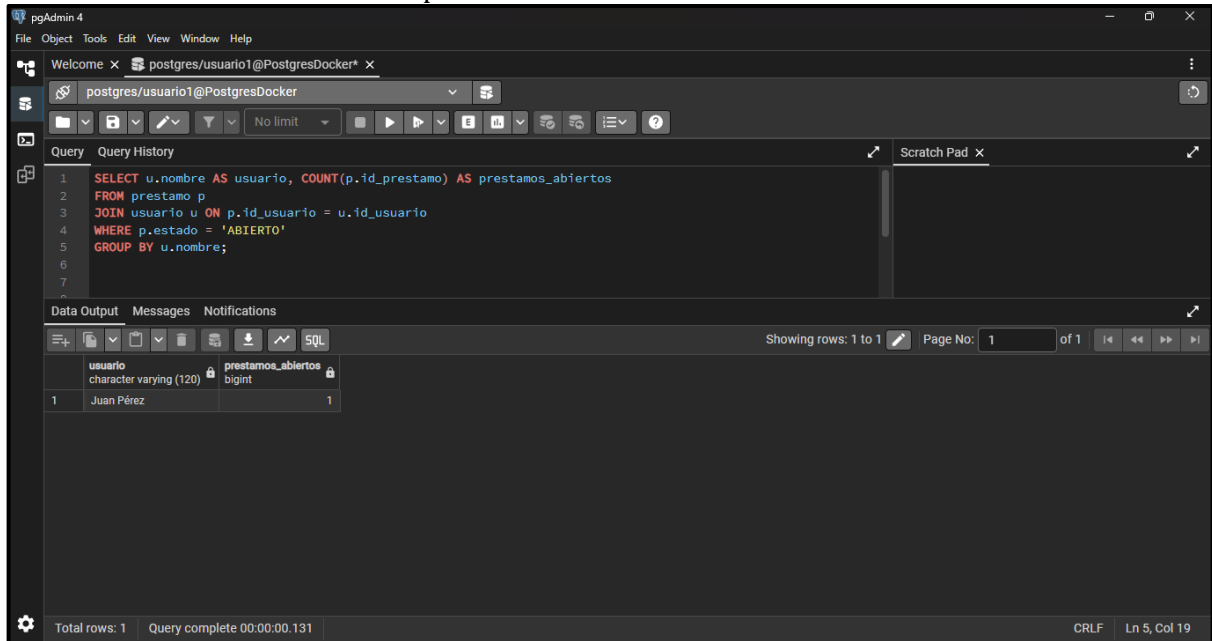
The screenshot shows the pgAdmin 4 interface with a SQL query executed. The query selects book titles, user names, and dates for books that are currently borrowed (not returned). The results are displayed in a table with 2 rows.

```
1 SELECT l.titulo, u.nombre AS usuario, p.fecha_prestamo, p.fecha_devolucion_max
2 FROM prestamo p
3 JOIN usuario u ON p.id_usuario = u.id_usuario
4 JOIN libro l ON p.id_libro = l.id_libro
5 WHERE p.estado = 'ABIERTO' OR p.fecha_devuelto IS NULL;
```

	titulo	usuario	fecha_prestamo	fecha_devolucion_max
1	Cien años de soledad	Juan Pérez	2025-09-01	2025-09-15
2	La casa de los espíritus	María Gómez	2025-09-05	2025-09-20

Successfully run. Total query runtime: 110 msec. 2 rows affected.

## 7. Usuarios con cantidad de préstamos abiertos



The screenshot shows the pgAdmin 4 interface with a SQL query executed. The query counts the number of open loans for each user. The results are displayed in a table with 1 row.

```
1 SELECT u.nombre AS usuario, COUNT(p.id_prestamo) AS prestamos_abiertos
2 FROM prestamo p
3 JOIN usuario u ON p.id_usuario = u.id_usuario
4 WHERE p.estado = 'ABIERTO'
5 GROUP BY u.nombre;
```

	usuario	prestamos_abiertos
1	Juan Pérez	1

# TALLER BASES DE DATOS

## ING. WILLIAM ALEXANDER MATALLANA PORRAS

### 8. Búsqueda por texto (título o autor)

The screenshot shows the pgAdmin 4 interface with a SQL query executed. The query searches for books where the title or author contains the text 'Cien años de soledad'.

```
1 SELECT l.titulo, a.nombre AS autor, l.anio_publicacion
2 FROM libro l
3 JOIN autor a ON l.id_autor = a.id_autor
4 WHERE LOWER(l.titulo) LIKE LOWER('%Cien%')
5 OR LOWER(a.nombre) LIKE LOWER('%Cien%');
6
7
```

The Data Output pane shows the following result:

titulo	autor	anio_publicacion
Cien años de soledad	Gabriel García Márquez	1967

At the bottom, a green status bar indicates: "Successfully run. Total query runtime: 105 msec. 1 rows affected."

### 9. Préstamos vencidos

The screenshot shows the pgAdmin 4 interface with a SQL query executed. The query identifies loans that are overdue and have not been returned.

```
1 SELECT u.nombre AS usuario, l.titulo
2 CURRENT_DATE - p.fecha_devolucion_max AS dias_atraso
3 FROM prestamo p
4 JOIN usuario u ON p.id_usuario = u.id_usuario
5 JOIN libro l ON p.id_libro = l.id_libro
6 WHERE CURRENT_DATE > p.fecha_devolucion_max
7 AND p.estado = 'ABIERTO';
8
9
10
```

The Data Output pane shows the following result:

usuario	titulo	dias_atraso
Juan Pérez	Cien años de soledad	15

At the bottom, a green status bar indicates: "Successfully run. Total query runtime: 111 msec. 1 rows affected."

## 10. Libros por década de publicación

The screenshot shows the pgAdmin 4 interface with a SQL query executed. The query is:

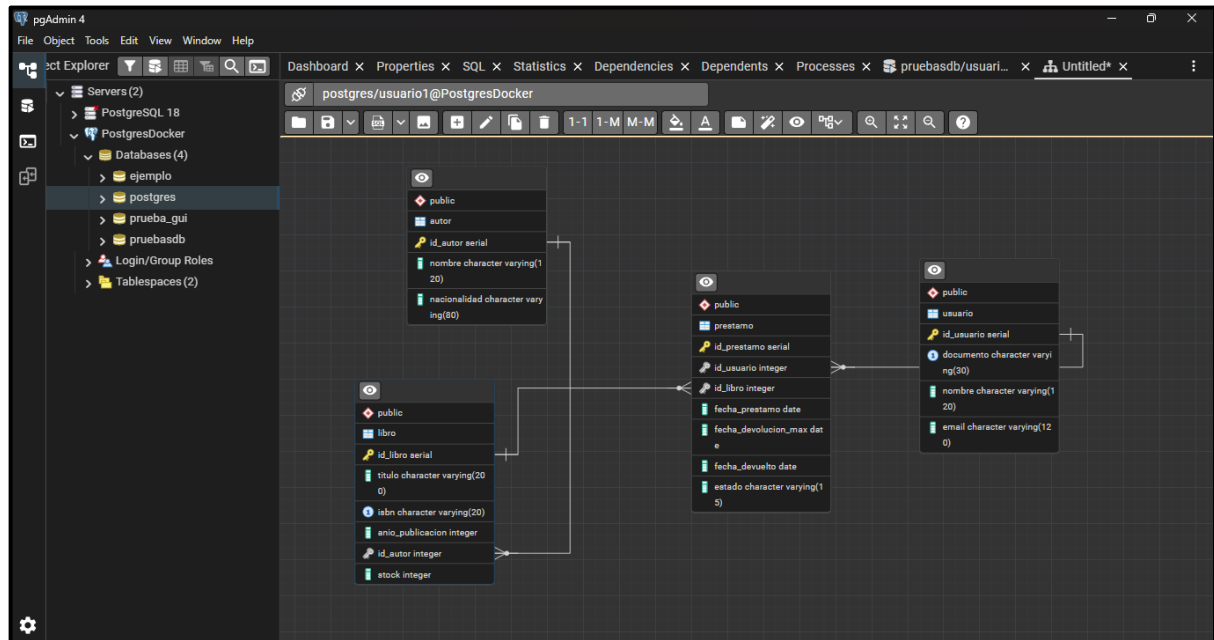
```
SELECT (anio_publicacion/10)*10 AS decada, COUNT(*) AS cantidad_libros
FROM Libro
GROUP BY decada
ORDER BY decada;
```

The results are displayed in a table with two columns: **decada** (integer) and **cantidad\_libros** (bigint). The data shows two decades: 1960 and 1980, each with a count of 1.

decada	cantidad_libros
1960	1
1980	1

A status message at the bottom indicates: "Successfully run. Total query runtime: 109 msec. 2 rows affected."

## Modelo ERD



## Referencias

PostgreSQL Global Development Group. (s. f.). CREATE TABLE [Documentación]. PostgreSQL. <https://www.postgresql.org/docs/current/sql-createtable.html>

## PostgreSQL

PostgreSQL Global Development Group. (s. f.). 2.3. Creating a New Table [Documentación]. PostgreSQL Tutorial. <https://www.postgresql.org/docs/current/tutorial-table.html>

## PostgreSQL

Docker. (2022, 5 de octubre). How to Use the Postgres Docker Official Image. Docker Blog. <https://www.docker.com/blog/how-to-use-the-postgres-docker-official-image/>

## Docker

pgAdmin Development Team. (s. f.). Container Deployment — pgAdmin 4 [Documentación]. pgAdmin. [https://www.pgadmin.org/docs/pgadmin4/latest/container\\_deployment.html](https://www.pgadmin.org/docs/pgadmin4/latest/container_deployment.html)

## pgAdmin

PostgreSQL Global Development Group. (s. f.). Documentation. PostgreSQL. <https://www.postgresql.org/docs/>

## PostgreSQL