



EXPLORANDO LAS DIFERENCIAS ENTRE MYSQL Y
POSTGRESQL CON DOCKER

PRESENTADO POR:
Neissy Alejandra Medina Cubillos

PRESENTADO A:
William Alexander Matallana

LINEA PROFUNDIZACION III
PROGRAMA DE INGENIERÍA DE SISTEMAS
UNIVERSIDAD DE CUNDINAMARCA
EXTENSIÓN CHIA
OCTUBRE 2025



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)	Relacional con capacidades JSON; variantes como MySQL HeatWave amplían analítica; motor InnoDB por defecto	Relacional y objeto-relacional con extensiones, tipos avanzados y ecosistema de extensiones; enfoque en estándares
Licencia	GPLv2 (edición comunitaria), ediciones comerciales disponibles por Oracle.	Licencia PostgreSQL (similar a BSD/MIT), muy permisiva
Enfoque principal	Alto rendimiento en lecturas, simplicidad operativa, ecosistema LAMP, gran adopción web	Conformidad SQL, extensibilidad, tipos complejos, integridad y funciones avanzadas (CTE, ventanas, etc.)
Tipos de datos admitidos	INT, VARCHAR, DATE, DATETIME, DECIMAL, JSON, ENUM/SET, BLOB/TEXT; variaciones por motor	INT, VARCHAR, DATE/TIMESTAMP, NUMERIC, JSON/JSONB, ARRAY, RANGE, HSTORE, UUID, tipos geométricos; tipos definidos por el usuario
Integridad referencial	Soportada en InnoDB	Soporte robusto de FK,

	con FK, ON DELETE/UPDATE; otros motores pueden variar	restricciones CHECK avanzadas y diferibles, políticas RLS
Soporte de JSON y datos complejos	JSON nativo con funciones JSON; sin índices GIN nativos para JSON.	JSON y JSONB con índices GIN/BTREE, operadores y consultas eficientes
Soporte para funciones y procedimientos	Stored Procedures/Functions (SQL/PSM), triggers, eventos programados	Funciones y procedimientos en múltiples lenguajes (PL/pgSQL, SQL, PL/Python, etc.), triggers, DO blocks
Nivel de cumplimiento del estándar SQL	Parcial con extensiones propias; ciertas diferencias en comportamiento SQL	Alta conformidad con SQL y amplio soporte de CTE, ventanas, subconsultas correlacionadas
Extensiones disponibles	Plugins y features; ecosistema menor de extensiones en comparación	Amplio ecosistema: PostGIS, pg_trgm, fuzzystrmatch, unaccent, citext, FDW (foreign data wrappers).
Uso recomendado	Aplicaciones web tradicionales, CMS, e-commerce, cargas OLTP simples a moderadas	Aplicaciones con lógica SQL compleja, analítica ligera, geoespacial (PostGIS), integridad estricta, datos semiestructurados.

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.



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

C:\Users\ale-j>docker pull postgres:latest
latest: Pulling from library/postgres
8c7716127147: Pull complete
a585c5f82f15: Pull complete
2433c366ca00: Pull complete
edd90ab5059f: Pull complete
af60ce4418c9: Pull complete
751039babae5: Pull complete
f0d70120d9e2: Pull complete
203b16f56a7d: Pull complete
1014e14b3351: Pull complete
9a68d6020eab: Pull complete
eed0ac863490: Pull complete
dd6d7b9d8ba8: Pull complete
f69a7c424b50: Pull complete
f5af7533693a: Pull complete
Digest: sha256:073e7c8b84e2197f94c8083634640ab37105effe1bc853ca4d5fbece3219b0e8
Status: Downloaded newer image for postgres:latest
docker.io/library/postgres:latest
```

```
C:\Users\ale-j>
```

```
C:\Users\ale-j>docker run -d --name postgres-dev -p 5433:5432 -e POSTGRES_PASSWORD=1234 -e POSTGRES_USER=admin -v C:/Users/ale-j/Desktop/PRACTICA_BD/MyPostgresVolumen postgres:latest
704f7fedfb8264e06b1b1229624006b38fe492721189f5f585c1aff2d682cb1

C:\Users\ale-j>
```

```
C:\Users\ale-j>docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                                                                 NAMES
704f7fedfb82   postgres:latest "docker-entrypoint.s..." 16 minutes ago Up 16 minutes 0.0.0.0:5433->5432/tcp, [::]:5433->5432/tcp   postgres-dev
b00fcac1e488   mysql:latest   "docker-entrypoint.s..." 7 days ago    Up 59 minutes 0.0.0.0:3315->3306/tcp, [::]:3315->3306/tcp   PracticaBD

C:\Users\ale-j>
```

Containers / PracticaBD

PracticaBD

b00fcac1e488

mysql:latest

3315:3306

STATUS

Running (43 seconds ago)

Logs

Inspect

Bind mounts

Exec

Files

Stats

2025-09-24T00:27:28.798135Z

0

[System]

[MY-011323]

[Server]

X Plugin ready for connections. Bind-address: '::' port: 33060, socket: /var/run/mysqld/mysqld.sock

2025-09-24T00:27:28.798332Z

0

[System]

[MY-010931]

[Server]

/usr/sbin/mysqld: ready for connections. Version: '9.4.0' socket: '/var/run/mysqld/mysqld.sock' port: 3306 MySQL Community Server - GPL.

2025-09-27T01:25:43.797690Z

0

[System]

[MY-013172]

[Server]

Received SHUTDOWN from user <via user signal>. Shutting down mysqld (Version: 9.4.0).

2025-09-27T01:25:45.925320Z

0

[Warning]

[MY-010909]

[Server]

/usr/sbin/mysqld: Forcing close of thread 10 user: 'root'.

2025-09-27T01:25:47.702879Z

0

[System]

[MY-010910]

[Server]

/usr/sbin/mysqld: Shutdown complete (mysqld 9.4.0) MySQL Community Server - GPL.

2025-09-27T01:25:47.702935Z

0

[System]

[MY-015016]

[Server]

MySQL Server - end.

2025-10-01 22:48:16+00:00

0

[Note]

[Entrypoint]

Entrypoint script for MySQL Server 9.4.0-1.el9 started.

2025-10-01 22:48:17+00:00

0

[Note]

[Entrypoint]

Switching to dedicated user 'mysql'

2025-10-01 22:48:17+00:00

0

[Note]

[Entrypoint]

Entrypoint script for MySQL Server 9.4.0-1.el9 started.

/var/lib/mysql/mysqld.sock' -> '/var/run/mysqld/mysqld.sock'

2025-10-01T22:48:18.383520Z

0

[System]

[MY-015015]

[Server]

MySQL Server - start.

2025-10-01T22:48:18.801249Z

0

[System]

[MY-010116]

[Server]

/usr/sbin/mysqld (mysqld 9.4.0) starting as process 1

2025-10-01T22:48:18.801293Z

0

[System]

[MY-015590]

[Server]

MySQL Server has access to 8 logical CPUs.

2025-10-01T22:48:18.801317Z

0

[System]

[MY-015590]

[Server]

MySQL Server has access to 3989012480 bytes of physical memory.

2025-10-01T22:48:18.810370Z

0

[Warning]

[MY-010159]

[Server]

Setting lower_case_table_names=2 because file system for /var/lib/mysql/ is case insensitive

2025-10-01T22:48:18.899176Z

1

[System]

[MY-013576]

[InnoDB]

InnoDB initialization has started.

2025-10-01T22:48:21.276291Z

1

[System]

[MY-013577]

[InnoDB]

InnoDB initialization has ended.

2025-10-01T22:48:22.654880Z

0

[Warning]

[MY-010068]

[Server]

CA certificate ca.pem is self signed.

2025-10-01T22:48:22.655922Z

0

[System]

[MY-013602]

[Server]

Channel mysql_main configured to support TLS. Encrypted connections are now supported for this channel.

2025-10-01T22:48:22.744813Z

0

[Warning]

[MY-011010]

[Server]

Insecure configuration for --pid-file: Location '/var/run/mysqld' in the path is accessible to all OS users. Consider choosing a different directory.

2025-10-01T22:48:22.821974Z

0

[System]

[MY-011323]

[Server]

X Plugin ready for connections. Bind-address: '::' port: 33060, socket: /var/run/mysqld/mysqld.sock

2025-10-01T22:48:22.822542Z

0

[System]

[MY-010931]

[Server]

/usr/sbin/mysqld: ready for connections. Version: '9.4.0' socket: '/var/run/mysqld/mysqld.sock' port: 3306 MySQL Community Server - GPL.

RAM 1.39 GB

CPU 0.25%

Disk: 3.04 GB used (limit 1006.85 GB)

Terminal

New version available

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	Imagen1
Crear una base de datos	create database <nombre>;	create database <nombre>;	Imagen2
Conectarse a una base específica	use database <nombre>	\c <nombre>	Imagen3
Listar usuarios	SELECT User, Host FROM mysql.user;	\du	Imagen4
Crear un usuario nuevo	CREATE USER 'nuevo_usuario'@'%' IDENTIFIED BY 'mi_password'; GRANT ALL PRIVILEGES ON *.* TO 'nuevo_usuario'@'%' WITH GRANT OPTION; FLUSH PRIVILEGES;	CREATE USER nuevo_usuario WITH PASSWORD 'mi_password';	Imagen5
Crear una tabla	CREATE TABLE persona (id INT AUTO_INCREMENT PRIMARY KEY, nombre VARCHAR(100) NOT NULL, edad INT, email VARCHAR(150) UNIQUE, fecha_creacion TIMESTAMP DEFAULT CURRENT_TIMESTAMP);	CREATE TABLE persona (id SERIAL PRIMARY KEY, nombre VARCHAR(100) NOT NULL, edad INT, email VARCHAR(150) UNIQUE, fecha_creacion TIMESTAMP DEFAULT CURRENT_TIMESTAMP);	Imagen6
Insertar datos en una tabla	INSERT INTO persona (nombre, edad,	INSERT INTO persona (nombre,	Imagen7

	email) VALUES ('Juan Pérez', 30, 'juan@example.com');	edad, email) VALUES ('Juan Pérez', 30, 'juan@example.co m');	
Consultar registros	select * from <tabla>;	select * from tabla	Imagen8
Actualizar registros	UPDATE persona SET edad = 31, email = 'juan.perez@examp le.com' WHERE id = 1;	UPDATE persona SET edad = 31, email = 'juan.perez@examp le.com' WHERE id = 1;	Imagen9
Eliminar registros	DELETE FROM persona WHERE id = 1;	DELETE FROM persona WHERE id = 1;	Imagen10

Imagen1

```
admin=# \l
                                List of databases
  Name  | Owner  | Encoding | Locale Provider | Collate  | Ctype    | Locale | ICU Rules | Access privileges
-----+-----+-----+-----+-----+-----+-----+-----+-----
admin   | admin  | UTF8     | libc            | en_US.utf8 | en_US.utf8 |         |          | =c/admin          +
postgres | admin  | UTF8     | libc            | en_US.utf8 | en_US.utf8 |         |          | admin=CTc/admin  +
template0 | admin  | UTF8     | libc            | en_US.utf8 | en_US.utf8 |         |          | =c/admin          +
template1 | admin  | UTF8     | libc            | en_US.utf8 | en_US.utf8 |         |          | admin=CTc/admin
(4 rows)
admin=# |
```

Imagen2

```
admin=# create database prueba;
CREATE DATABASE
admin=# |
```

Imagen3

```
admin=# \c prueba
You are now connected to database "prueba" as user "admin".
prueba=# |
```

Imagen4

```
prueba=# \du
                                List of roles
Role name |                               Attributes
-----+-----
admin     | Superuser, Create role, Create DB, Replication, Bypass RLS
prueba=# |
```

Imagen5

```
prueba=# CREATE USER nuevo_usuario WITH PASSWORD 'password';
CREATE ROLE
prueba=#
```

Imagen6

```
prueba=# CREATE TABLE persona (
    id SERIAL PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    edad INT,
    email VARCHAR(150) UNIQUE,
    fecha_creacion TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
CREATE TABLE
prueba=# |
```

Imagen7

```
prueba=# INSERT INTO persona (nombre, edad, email)
VALUES ('Juan Pérez', 30, 'juan@example.com');
INSERT 0 1
prueba=# |
```

Imagen8

```
prueba=# select * from persona;
 id | nombre | edad | email | fecha_creacion
-----+-----
  1 | Juan Pérez | 30 | juan@example.com | 2025-09-30 00:34:39.846464
(1 row)
prueba=# |
```

Imagen9:

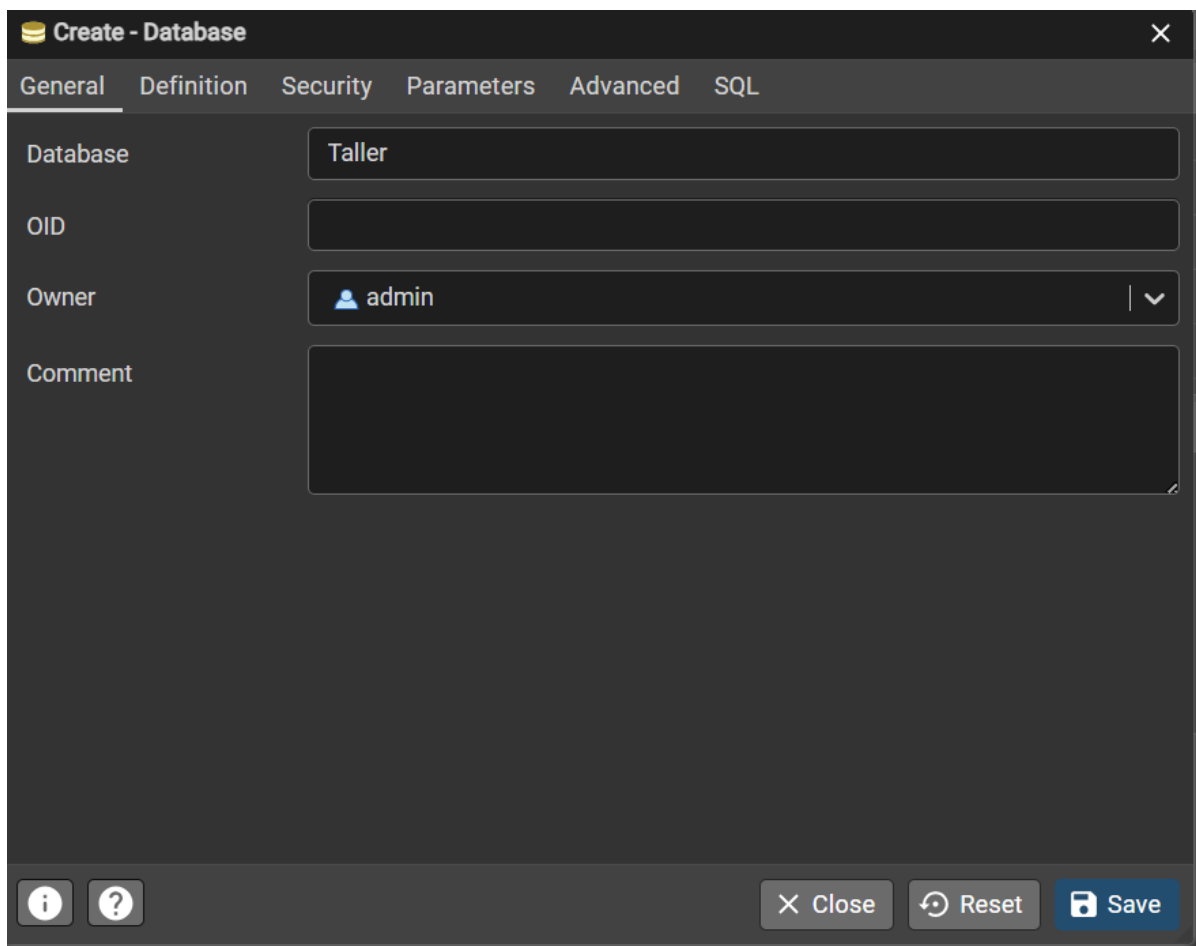
```
prueba=# UPDATE persona
SET edad = 31, email = 'juan.perez@example.com'
WHERE id = 1;
UPDATE 1
prueba=# |
```

Imagen10

```
prueba=# DELETE FROM persona
WHERE id = 1;
DELETE 1
prueba=# |
```

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.



Create - Database

General Definition Security Parameters Advanced SQL

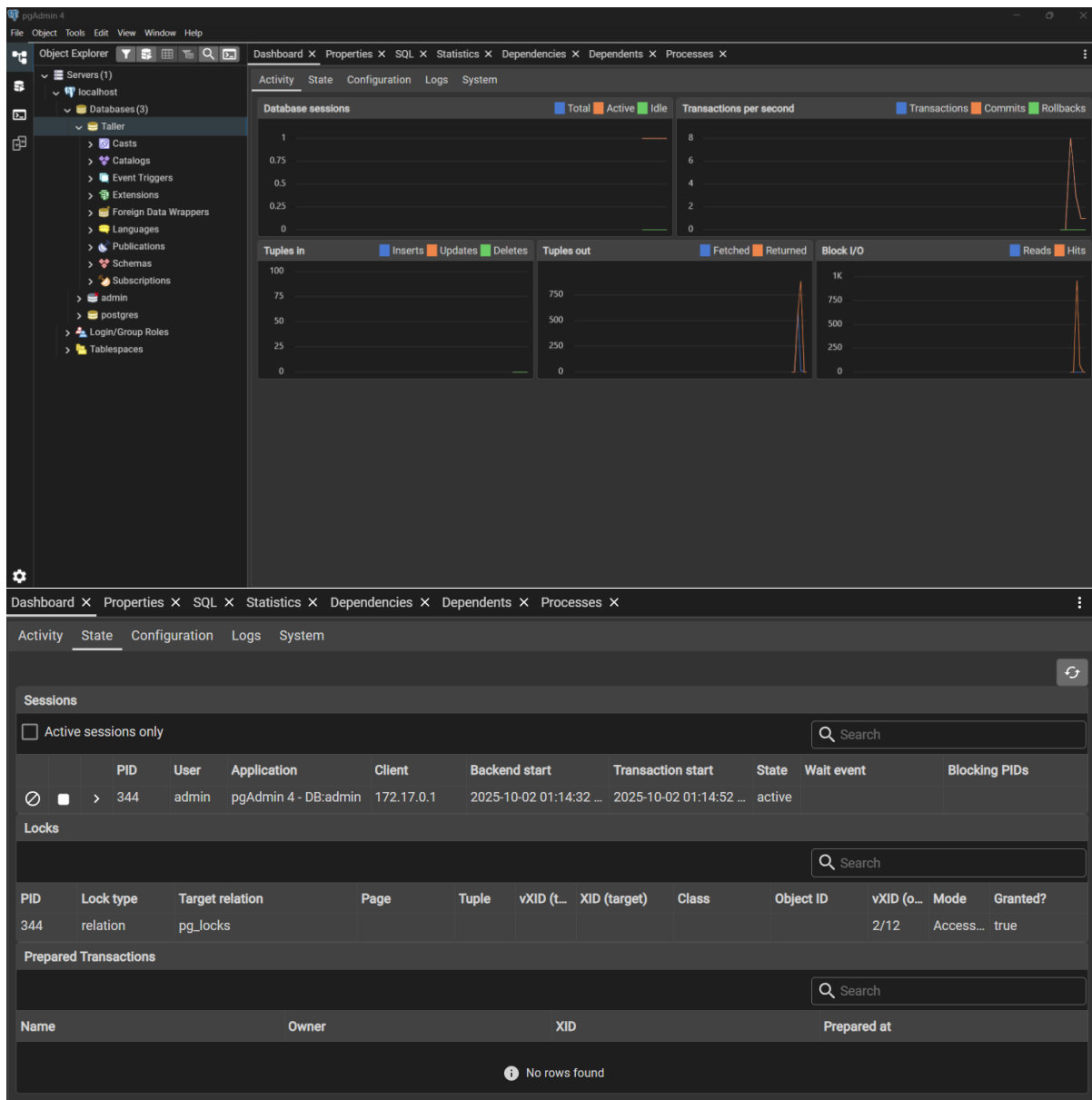
Database: Taller

OID:

Owner: admin

Comment:

Close Reset Save



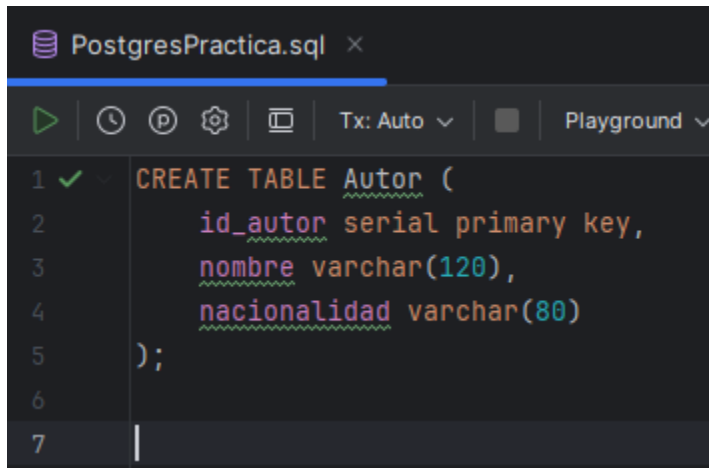
Parte 5. Actividades de aplicación

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)



```
PostgresPractica.sql x
Tx: Auto Playground
1 ✓ CREATE TABLE Autor (
2     id_autor serial primary key,
3     nombre varchar(120),
4     nacionalidad varchar(80)
5 );
6
7
```

2. Libro

Campo	Tipo de Dato	Clave	Descripción
id_libro	INT	PK	Identificador del libro
título	VARCHAR(200)		Título del libro
isbn	VARCHAR(20)		Código ISBN (único)
año_publicación	INT		Año de publicación
id_autor	INT	FK	Autor del libro (1 autor por libro)
stock	INT		Unidades disponibles para préstamo (≥ 0)

```

6
7 ✓
8 CREATE TABLE Libro (
9     id_libro serial primary key,
10    titulo varchar(200) not null ,
11    isbn varchar(20) not null ,
12    anio_publicacion int not null,
13    id_autor int,
14    stock int,
15    CONSTRAINT fk_autor FOREIGN KEY (id_autor)
16        REFERENCES Autor(id_autor)
17        ON UPDATE CASCADE
18        ON DELETE SET NULL
19 );

```

3. Usuario

Campo	Tipo de Dato	Cla ve	Descripción
id_usuar io	INT	PK	Identificador del usuario
docum ento	VARCHAR(30)		Documento de identidad (único)
nombre	VARCHAR(120)		Nombre completo
email	VARCHAR(120)		Correo electrónico (opcional)

```

19
20 ✓ CREATE TABLE Usuario(
21     id_usuario serial primary key ,
22     documento varchar(30) not null unique ,
23     nombre varchar(120) not null ,
24     email varchar(120)
25 );

```

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

```
CREATE TABLE Prestamo (
    id_prestamo serial primary key ,
    id_usuario int not null ,
    id_libro int not null ,
    fecha_prestamo DATE not null ,
    fecha_devolucion_max DATE not null ,
    fecha_devuelto DATE,
    estado varchar(15)
);

ALTER TABLE Prestamo ADD CONSTRAINT fk_usuario FOREIGN KEY (id_usuario)
    REFERENCES Usuario(id_usuario) ON UPDATE CASCADE ON DELETE SET NULL;

ALTER TABLE Prestamo ADD CONSTRAINT fk_libro FOREIGN KEY (id_libro)
    REFERENCES Libro(id_libro) ON UPDATE CASCADE ON DELETE SET NULL;
```

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.

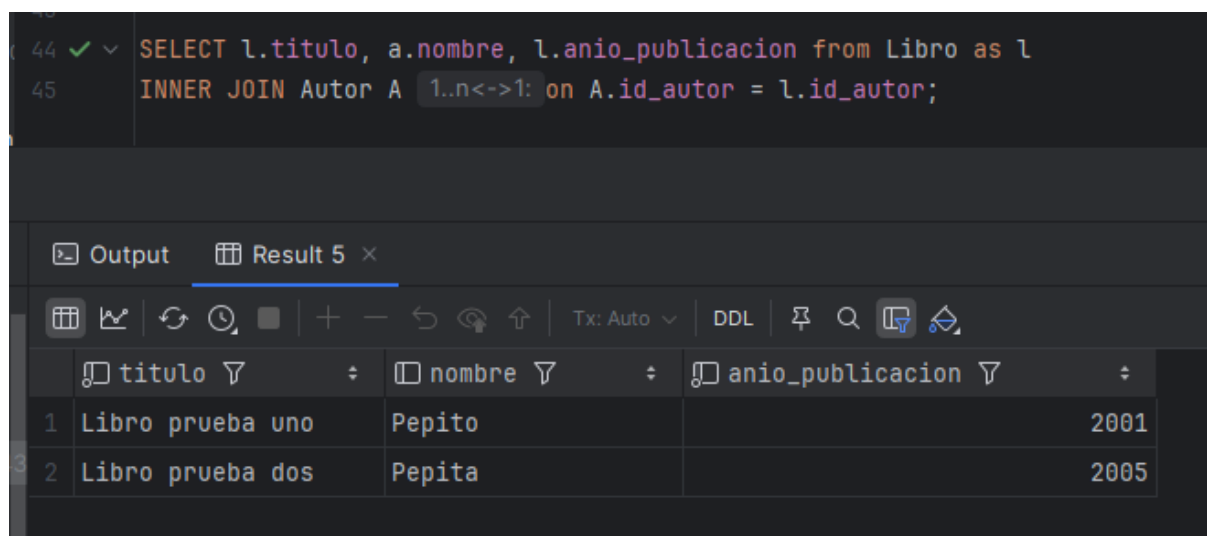
Usuario — Prestamo

- 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.



```

44 ✓ SELECT l.titulo, a.nombre, l.anio_publicacion from Libro as l
45     INNER JOIN Autor A 1..n<->1: on A.id_autor = l.id_autor;
  
```

Output Result 5 x

	titulo	nombre	anio_publicacion
1	Libro prueba uno	Pepito	2001
2	Libro prueba dos	Pepita	2005

2. **Préstamos abiertos con datos del usuario y del libro** Columnas: id_prestamo, usuario, titulo_libro, fecha_prestamo, fecha_devolucion_max, estado.

```
SELECT p.id_prestamo,
       u.nombre,
       u.documento,
       l.titulo,
       p.fecha_prestamo,
       p.fecha_devolucion_max,
       p.estado
FROM prestamo as p
inner join Libro l on p.id_libro = l.id_libro
inner join Usuario u on p.id_usuario = u.id_usuario
WHERE p.estado = 'ABIERTO';
```

Output Result 6

id_prestamo	nombre	documento	titulo	fecha_prestamo	fecha_devolucion_max	estado
3	Nombre 2	4324324	Libro prueba uno	2025-09-22	2025-09-30	ABIERTO
1	Nombre 1	12010201	Libro prueba uno	2025-09-08	2025-10-15	ABIERTO

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

Dado un documento, listar titulo_libro, fecha_prestamo, fecha_devuelto, estado.

```
SELECT p.id_prestamo, l.titulo, p.fecha_prestamo,
       p.fecha_devuelto, p.estado FROM prestamo AS p
inner join Libro L on p.id_libro = L.id_libro
inner join Usuario u on p.id_usuario = u.id_usuario
WHERE u.documento = '12010201';
```

Output Result 7

id_prestamo	titulo	fecha_prestamo	fecha_devuelto	estado
1	Libro prueba uno	2025-09-08	2025-09-29	ABIERTO
2	Libro prueba dos	2025-09-02	2025-09-13	CERRADO

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

Columnas: autor, cantidad_libros. Ordenar descendente.

```

5 ✓ SELECT
6     a.nombre AS autor,
7     COUNT(l.id_libro) AS cantidad_libros
8 FROM Autor a
9 LEFT JOIN Libro l 1<->0..n: ON a.id_autor = l.id_autor
10 GROUP BY a.id_autor, a.nombre
11 ORDER BY cantidad_libros DESC;
12

```

Output Result 9 x

	autor	cantidad_libros
1	Pepita	1
2	Pepito	1

5. **Disponibilidad actual de cada libro**

Columnas: titulo, stock. Filtrar libros con stock = 0 (sin unidades disponibles).

```

72
73 ✓ SELECT titulo, stock
74 FROM Libro
75 WHERE stock = 0;
76
77

```

Output prueba.public.libro x

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

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.)

```

78 ✓ SELECT
79     l.titulo,
80     u.nombre AS usuario,
81     p.fecha_prestamo,
82     p.fecha_devolucion_max
83 FROM Prestamo p
84 INNER JOIN Libro l ON p.id_libro = l.id_libro
85 INNER JOIN Usuario u ON p.id_usuario = u.id_usuario
86 WHERE p.estado = 'ABIERTO'
87     OR p.fecha_devuelto IS NULL;
88
89

```

Output Result 11 ×

	titulo	usuario	fecha_prestamo	fecha_devolucion_max
1	Libro prueba uno	Nombre 2	2025-09-22	2025-09-30
2	Libro prueba uno	Nombre 1	2025-09-08	2025-10-15

7. Usuarios con cantidad de préstamos abiertos

Columnas: usuario, prestamos_abiertos. (Agrupar por usuario.)

```

90 ✓ SELECT
91     u.nombre AS usuario,
92     COUNT(p.id_prestamo) AS prestamos_abiertos
93 FROM Usuario u
94 INNER JOIN Prestamo p ON u.id_usuario = p.id_usuario
95 WHERE p.estado = 'ABIERTO'
96     OR p.fecha_devuelto IS NULL
97 GROUP BY u.id_usuario, u.nombre
98 ORDER BY prestamos_abiertos DESC;
99
100

```

Output Result 12 ×

	usuario	prestamos_abiertos
1	Nombre 1	1
2	Nombre 2	1

8. Búsqueda por texto (título o autor)

Dado un término, mostrar título, autor, anio_publicacion donde el término aparezca en título o nombre de autor (insensible a mayúsculas).

```
SELECT
    l.titulo,
    a.nombre AS autor,
    l.anio_publicacion
FROM Libro l
INNER JOIN Autor a ON l.id_autor = a.id_autor
WHERE l.titulo ILIKE '%prueba%'
    OR a.nombre ILIKE '%pep%';
```

Output Result 13 x

titulo	autor	anio_publicacion
Libro prueba uno	Pepito	2001
Libro prueba dos	Pepita	2005

9. Préstamos vencidos

Listar usuario, titulo, dias_atraso cuando CURRENT_DATE > fecha_devolucion_max y el préstamo siga abierto.

```

109 ✓ SELECT
110     u.nombre AS usuario,
111     l.titulo,
112     CURRENT_DATE - p.fecha_devolucion_max AS dias_atraso
113 FROM Prestamo p
114 INNER JOIN Usuario u ON p.id_usuario = u.id_usuario
115 INNER JOIN Libro l ON p.id_libro = l.id_libro
116 WHERE (p.estado = 'ABIERTO' OR p.fecha_devuelto IS NULL)
117 AND CURRENT_DATE > p.fecha_devolucion_max;
118
119

```

10. Libros por década de publicación

Mostrar década (ej. 1990s, 2000s) y cantidad de libros. (Tip: agrupa por `anio_publicacion/10`.)

```
118
119 ✓ SELECT
120     ⚡ (FLOOR(l.ano_publicacion / 10) * 10)::TEXT || 's' AS decada,
121         COUNT(*) AS cantidad_libros
122 FROM Libro l
123 GROUP BY FLOOR(l.ano_publicacion / 10)
124 ORDER BY decada;
125
126
```

Output

Result 16 ×

Grid

Table

Refresh

Undo

Copy

Paste

decada ▼

÷

cantidad_libros ▼

÷

1

2000s

2