PRÁCTICA DE LABORATORIO - BASES DE DATOS ORIENTADAS A OBJETOS PREREQUISITOS

- PostgreSQL instalado (versión 12 o superior)
- pgAdmin 4 o acceso a psql

PARTE 1: MODELADO RELACIONAL TRADICIONAL

Paso 1: Crear la base de datos

Abrir pgAdmin o psql y ejecutar:

```
1   CREATE DATABASE lab_bdoo;
```

Data Output | Messages | Notifications

```
CREATE DATABASE

Query returned successfully in 237 msec.
```

Conectarse a la base de datos:

```
sql

\c lab_bdoo
```

Paso 2: Crear tablas relacionales tradicionales

Vamos a modelar un sistema de biblioteca con el enfoque relacional tradicional:

```sql
-- Tabla de direcciones
CREATE TABLE direcciones (
    id SERIAL PRIMARY KEY,
    calle VARCHAR(100),
    numero VARCHAR(10),
    ciudad VARCHAR(50),
    codigo_postal VARCHAR(10)
);

-- Tabla de autores
CREATE TABLE autores (
    id SERIAL PRIMARY KEY,
    nombre VARCHAR(100),
    apellido VARCHAR(100),
    fecha_nacimiento DATE,
    direccion_id INTEGER REFERENCES direcciones(id)
);

-- Tabla de libros
CREATE TABLE libros (
    id SERIAL PRIMARY KEY,
    titulo VARCHAR(200),
    isbn VARCHAR(20),
    año_publicacion INTEGER,
    precio NUMERIC(10, 2),
    autor_id INTEGER REFERENCES autores(id)
);
```

Data Output | Messages | Notifications

```
CREATE TABLE

Query returned successfully in 42 msec.
```

Paso 3: Insertar datos en el modelo relacional

```sql
INSERT INTO direcciones (calle, numero, ciudad, codigo_postal)
VALUES ('Av. Arequipa', '1234', 'Lima', '15001');

-- Insertar autor
INSERT INTO autores (nombre, apellido, fecha_nacimiento, direccion_id)
VALUES ('Mario', 'Vargas Llosa', '1936-03-28', 1);

-- Insertar libros
INSERT INTO libros (titulo, isbn, año_publicacion, precio, autor_id)
VALUES
('La ciudad y los perros', '978-8420471839', 1963, 45.50, 1),
('Conversación en La Catedral', '978-8466326070', 1969, 52.00, 1);
```

Data Output | Messages | Notifications

```
INSERT 0 2

Query returned successfully in 42 msec.
```

Paso 4: Consultar datos (requiere JOINs)

```sql
SELECT
    l.titulo,
    l.isbn,
    a.nombre || ' ' || a.apellido AS autor,
    d.ciudad
FROM libros l
JOIN autores a ON l.autor_id = a.id
JOIN direcciones d ON a.direccion_id = d.id;
```

Data Output | Messages | Notifications

Showing rows: 1 to 2 | Page No: 1

	titulo character varying (200)	isbn character varying (20)	autor text	ciudad character varying (50)
1	La ciudad y los perros	978-8420471839	Mario Vargas Llosa	Lima
2	Conversación en La Catedral	978-8466326070	Mario Vargas Llosa	Lima

OBSERVACIÓN: Note que necesitamos 2 JOINs para obtener información completa.

PARTE 2: MODELADO OBJETO-RELACIONAL

Paso 5: Crear tipos de datos personalizados

PostgreSQL permite crear tipos compuestos (característica objeto-relacional):

```sql
-- Tipo personalizado para dirección
CREATE TYPE tipo_direccion AS (
    calle VARCHAR(100),
    numero VARCHAR(10),
    ciudad VARCHAR(50),
    codigo_postal VARCHAR(10)
);

-- Tipo personalizado para autor
CREATE TYPE tipo_autor AS (
    nombre VARCHAR(100),
    apellido VARCHAR(100),
    fecha_nacimiento DATE,
    direccion tipo_direccion
);
```

Data Output | Messages | Notifications

```
CREATE TYPE

Query returned successfully in 52 msec.
```

Paso 6: Crear tabla con tipos compuestos

```sql
CREATE TABLE libros_oo (
    id SERIAL PRIMARY KEY,
    titulo VARCHAR(200),
    isbn VARCHAR(20),
    año_publicacion INTEGER,
    precio NUMERIC(10, 2),
    autor tipo_autor
);
```

Data Output | Messages | Notifications

```
CREATE TABLE

Query returned successfully in 43 msec.
```

OBSERVACIÓN: Note que el campo 'autor' es un tipo complejo que contiene toda la información del autor, incluyendo su dirección anidada.

Paso 7: Insertar datos en el modelo objeto-relacional

```sql
-- Insertar libro con autor y dirección anidados
INSERT INTO libros_oo (titulo, isbn, año_publicacion, precio, autor)
VALUES
(
'La ciudad y los perros',
'978-8420471839',
1963,
45.50,
ROW('Mario', 'Vargas Llosa', '1936-03-28',
    ROW('Av. Arequipa', '1234', 'Lima', '15001'))::tipo_autor
);

INSERT INTO libros_oo (titulo, isbn, año_publicacion, precio, autor)
VALUES
(
'Conversación en La Catedral',
'978-8466326070',
1969,
52.00,
ROW('Mario', 'Vargas Llosa', '1936-03-28',
    ROW('Av. Arequipa', '1234', 'Lima', '15001'))::tipo_autor
);
```

Data Output | Messages | Notifications

```
INSERT 0 1

Query returned successfully in 42 msec.
```

Paso 8: Consultar datos sin JOINs (navegación por puntos)

```sql
-- Acceso directo a campos anidados usando notación de punto
SELECT
    titulo,
    isbn,
    (autor).nombre || ' ' || (autor).apellido AS nombre_autor,
    ((autor).direccion).ciudad AS ciudad_autor
FROM libros_oo;
```

Data Output | Messages | Notifications

Showing rows: 1 to 2 | Page No: 1

	titulo character varying (200)	isbn character varying (20)	nombre_autor text	ciudad_autor character varying (50)
1	La ciudad y los perros	978-8420471839	Mario Vargas Llosa	Lima
2	Conversación en La Catedral	978-8466326070	Mario Vargas Llosa	Lima

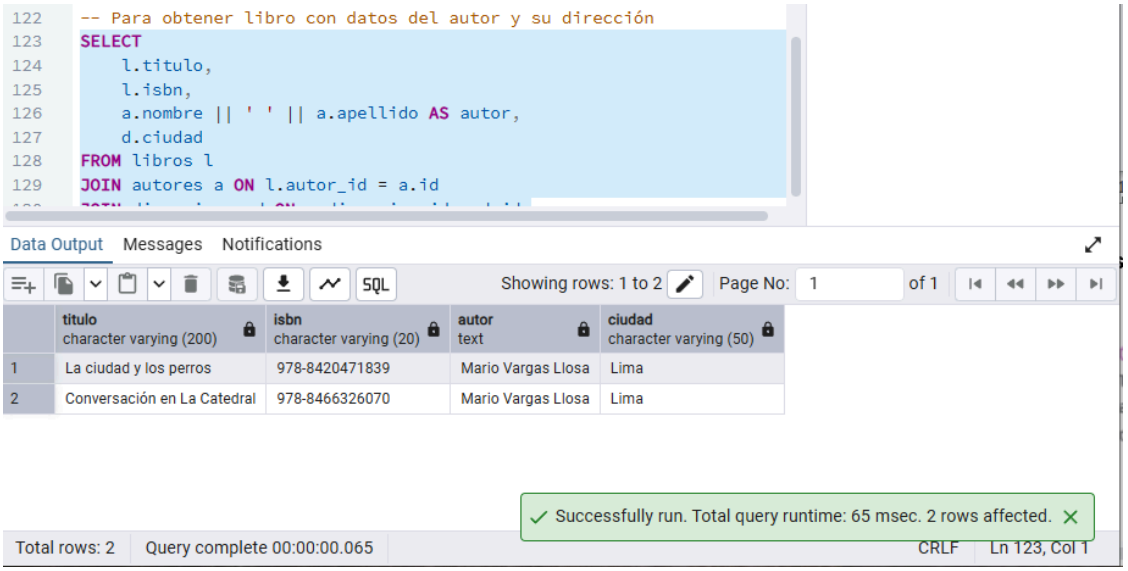
OBSERVACIÓN: No necesitamos JOINs. Accedemos directamente a los datos anidados usando notación de punto, similar a la programación orientada a objetos.

PARTE 3: COMPARACIÓN Y ANÁLISIS

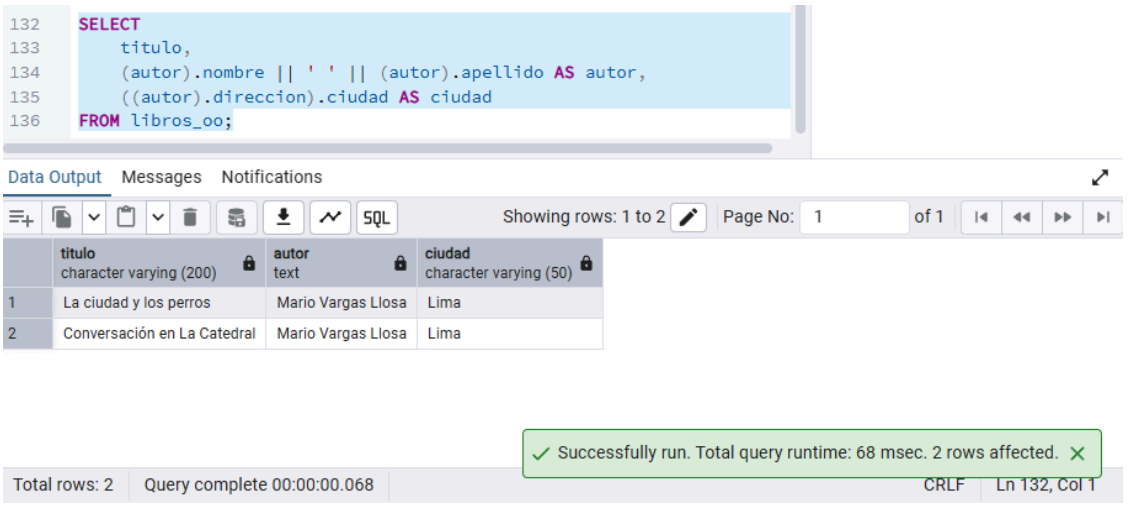
Paso 9: Comparar consultas

Ejecute ambas consultas y compare los resultados:

Consulta Relacional:



Consulta Objeto-Relacional:



Paso 10: Ventajas y desventajas observadas

Complete la siguiente tabla basándose en su experiencia:

Característica	Modelo Relacional	Modelo Objeto-Relacional
Número de tablas necesarias	**3**	**1**
Complejidad de consultas	**Alta (Requiere join)**	**Baja (Acceso a la información anidada)**
Normalización	**Alta (Sigue reglas estrictas de normalización, eliminando redundancia de datos).**	**Baja (La información del autor y dirección está duplicada dentro de cada registro de libro en `libros_oo`).**

Facilidad de actualización	**Alta (Si cambia la dirección del autor, solo se actualiza un registro en la tabla `direcciones`).**	**Baja (Si cambia la dirección del autor, se deben actualizar todos los registros en `libros_oo` que contenga a ese autor).**
Consistencia de datos	**Alta (Asegurada por Claves Foráneas y la Normalización.**	**Dependiente del programador (No hay claves foráneas en los tipos compuestos).**