

# INSTITUTO TECNOLOGICO DE MEXICALI

Carrera:  
ING. en Sistemas.

Materia:  
Fundamentos de base de datos.

Alumno:  
Hernandez Garcia Martin 22490354

Profesor:  
Jose Ramón Bogarin Valenzuela.

Mexicali, Baja California a 18 de Marzo del 2025.

## 1. Sistema de Gestión de Hospitales

- Un hospital necesita gestionar información de pacientes, médicos y citas médicas.
- Identificar entidades clave: Paciente, Médico, Cita, Tratamiento.
- Diseñar el modelo E-R con sus relaciones y atributos principales.
- Transformar el modelo en un esquema relacional con claves primarias y foráneas.
- Implementar la base de datos en SQL mediante sentencias LDD.
- Usar LMD para insertar datos y consultar las citas de un paciente específico.

### Entidades:

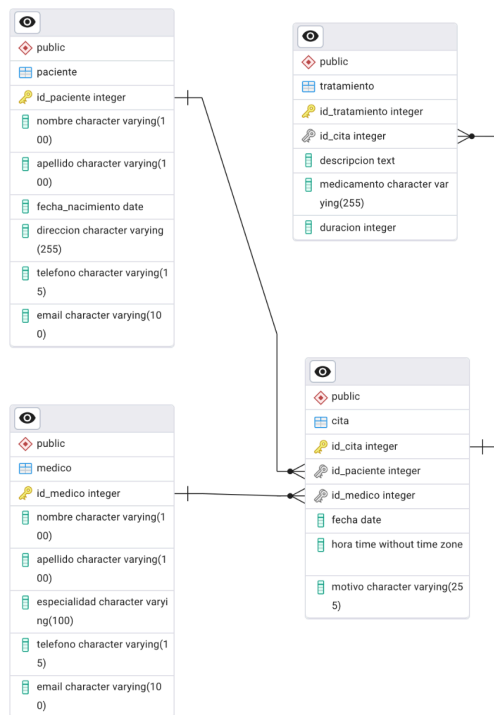
- **Paciente:** Es quien recibe el tratamiento en el hospital.
- **Médico:** Es quien atiende a los pacientes.
- **Cita:** Representa la visita medica
- **Tratamiento:** Un tratamiento asignado a un paciente, por un médico.

### Atributos:

- **Paciente:** id\_paciente (clave primaria), nombre, apellido, fecha\_nacimiento, sexo, direccion, telefono
- **Médico:** id\_medico (clave primaria), nombre, apellido, especialidad, telefono
- **Cita:** id\_cita (clave primaria), id\_paciente (clave foránea de la tabla Paciente), id\_medico (clave foránea de la tabla Medico), fecha, hora
- **Tratamiento:** id\_tratamiento (clave primaria), id\_paciente (clave foránea de la tabla Paciente), id\_medico (clave foránea de la tabla Medico), descripcion, fecha\_inicio, fecha\_fin

### Modelo Entidad-Relación (E-R)

- Un **Paciente** puede tener muchas **Citas** (relación uno a muchos).
- Un **Médico** puede atender a muchas **Citas** (relación uno a muchos).
- Un **Paciente** puede tener varios **Tratamientos**, pero cada tratamiento es asignado por un solo **Médico** (relación uno a muchos).



## QUERY:

### Tablas

```

CREATE TABLE Paciente (
  id_paciente INT PRIMARY KEY,
  nombre VARCHAR(100),
  apellido VARCHAR(100),
  fecha_nacimiento DATE,
  direccion VARCHAR(255),
  telefono VARCHAR(15),
  email VARCHAR(100)
);
  
```

```

CREATE TABLE Medico (
  id_medico INT PRIMARY KEY,
  nombre VARCHAR(100),
  apellido VARCHAR(100),
  especialidad VARCHAR(100),
  telefono VARCHAR(15),
  email VARCHAR(100)
);
  
```

```

CREATE TABLE Cita (
  id_cita INT PRIMARY KEY,
  id_paciente INT,
  id_medico INT,
  descripcion TEXT,
  medicamento VARCHAR(255),
  duracion INT
);
  
```

```

id_cita INT PRIMARY KEY,
id_paciente INT,
id_medico INT,
fecha DATE,
hora TIME,
motivo VARCHAR(255),
FOREIGN KEY (id_paciente) REFERENCES Paciente(id_paciente),
FOREIGN KEY (id_medico) REFERENCES Medico(id_medico)
);

```

```

CREATE TABLE Tratamiento (
id_tratamiento INT PRIMARY KEY,
id_cita INT,
descripcion TEXT,
medicamento VARCHAR(255),
duracion INT, -- Duración en días
FOREIGN KEY (id_cita) REFERENCES Cita(id_cita)
);

```

### Insertar datos

```

INSERT INTO Paciente (id_paciente, nombre, apellido, fecha_nacimiento, direccion,
telefono, email)
VALUES (1, 'Juan', 'Pérez', '1985-06-15', 'Calle Ficticia 123', '555-1234',
'juan.perez@email.com'),
(2, 'Ana', 'González', '1992-03-22', 'Avenida Libertad 456', '555-5678',
'ana.gonzalez@email.com');

```

```

INSERT INTO Medico (id_medico, nombre, apellido, especialidad, telefono, email)
VALUES (1, 'Dr. Carlos', 'Lopez', 'Cardiología', '555-1122',
'carlos.lopez@hospital.com'),
(2, 'Dra. Marta', 'Martínez', 'Pediatría', '555-3344',
'marta.martinez@hospital.com');

```

```

INSERT INTO Cita (id_cita, id_paciente, id_medico, fecha, hora, motivo)
VALUES (1, 1, 1, '2025-03-25', '09:00:00', 'Chequeo general'),
(2, 2, 2, '2025-03-26', '10:00:00', 'Revisión infantil');

```

```

INSERT INTO Tratamiento (id_tratamiento, id_cita, descripcion, medicamento,
duracion)
VALUES (1, 1, 'Chequeo de salud general', 'Ninguno', 0),
(2, 2, 'Vacunación', 'Vacuna infantil', 1);

```

### Consulta

```

SELECT P.nombre AS paciente_nombre, P.apellido AS paciente_apellido,
P.telefono,
      C.id_cita, C.fecha, C.hora,
      M.nombre AS medico_nombre, M.apellido AS medico_apellido
FROM Cita C
JOIN Paciente P ON C.id_paciente = P.id_paciente
JOIN Medico M ON C.id_medico = M.id_medico
WHERE P.id_paciente = 2;

```

Data Output Messages Notifications									
	paciente_nombre character varying (100)	paciente_apellido character varying (100)	telefono character varying (15)	id_cita integer	fecha date	hora time without time zone	medico_nombre character varying (100)	medico_apellido character varying (100)	
1	Ana	González	555-5678	2	2025-03-26	10:00:00	Dra. Marta	Martínez	

## 2. Tienda en Línea

- Una empresa quiere mejorar la administración de sus pedidos en línea.
- Definir entidades: Cliente, Producto, Pedido, DetallePedido.
- Crear el diagrama E-R que refleje las relaciones entre las entidades.
- Convertir el modelo en un esquema de tablas relacionales.
- Implementar la base de datos en SQL con restricciones de integridad.
- Consultar los productos comprados por un cliente específico usando SQL.

### Entidades:

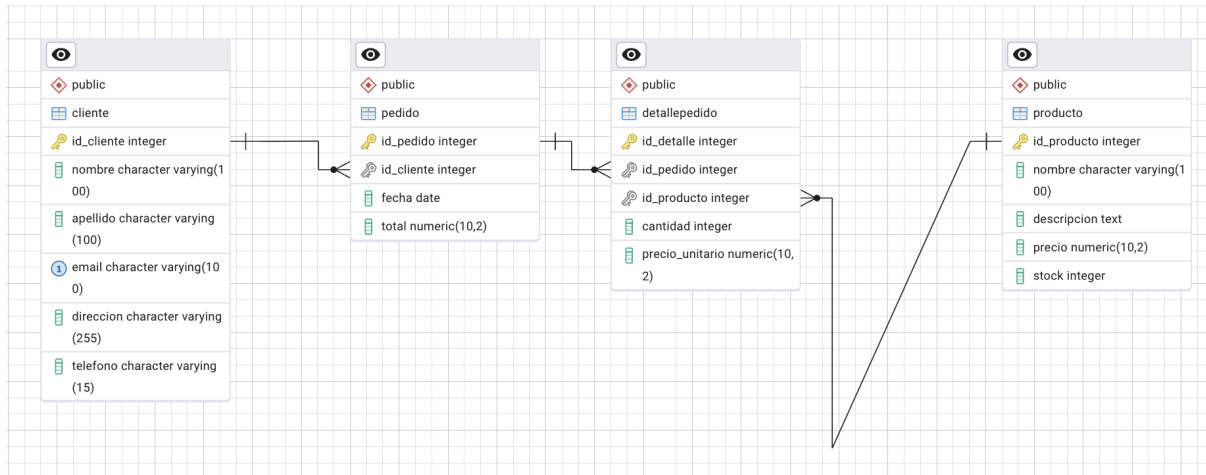
- **Cliente:** Es quien realiza el pedido.
- **Producto:** Es el artículo disponible para la venta en la tienda.
- **Pedido:** Es la compra realizada por un cliente.
- **DetallePedido:** Especifica los productos comprados dentro de un pedido.

### Atributos:

- **Cliente:** id\_cliente (clave primaria), nombre, apellido, direccion, telefono, email
- **Producto:** id\_producto (clave primaria), nombre, descripcion, precio, stock
- **Pedido:** id\_pedido (clave primaria), id\_cliente (clave foránea de la tabla Cliente), fecha\_pedido, estado (e.g., "pendiente", "enviado", "entregado")
- **DetallePedido:** id\_detalle (clave primaria), id\_pedido (clave foránea de la tabla Pedido), id\_producto (clave foránea de la tabla Producto), cantidad, precio\_unitario

## Modelo Entidad-Relación (E-R)

- Un **Cliente** puede realizar muchos **Pedidos** (relación uno a muchos).
- Un **Pedido** puede tener muchos **DetallePedido** (relación uno a muchos).
- Un **Producto** puede estar en muchos **DetallePedido** (relación uno a muchos).



## QUERY:

### Tablas

```

CREATE TABLE Cliente (
  id_cliente INT PRIMARY KEY,
  nombre VARCHAR(100),
  apellido VARCHAR(100),
  email VARCHAR(100) UNIQUE,
  direccion VARCHAR(255),
  telefono VARCHAR(15)
);
  
```

```

CREATE TABLE Producto (
  id_producto INT PRIMARY KEY,
  nombre VARCHAR(100),
  descripcion TEXT,
  precio DECIMAL(10, 2),
  stock INT
);
  
```

```

CREATE TABLE Pedido (
  id_pedido INT PRIMARY KEY,
  id_cliente INT,
  fecha DATE,
  total NUMERIC(10,2)
);
  
```

```
id_cliente INT,  
fecha DATE,  
total DECIMAL(10, 2),  
FOREIGN KEY (id_cliente) REFERENCES Cliente(id_cliente)  
);
```

```
CREATE TABLE DetallePedido (  
id_detalle INT PRIMARY KEY,  
id_pedido INT,  
id_producto INT,  
cantidad INT,  
precio_unitario DECIMAL(10, 2),  
FOREIGN KEY (id_pedido) REFERENCES Pedido(id_pedido),  
FOREIGN KEY (id_producto) REFERENCES Producto(id_producto)  
);
```

### **Insertar datos**

```
INSERT INTO Cliente (id_cliente, nombre, apellido, email, direccion, telefono)  
VALUES (1, 'Carlos', 'Gómez', 'carlos.gomez@email.com', 'Calle 123, Ciudad',  
'555-9876'),  
(2, 'Ana', 'Martínez', 'ana.martinez@email.com', 'Avenida 456, Ciudad', '555-6543');
```

```
INSERT INTO Producto (id_producto, nombre, descripcion, precio, stock)  
VALUES (1, 'Laptop', 'Laptop gaming', 1200.00, 50),  
(2, 'Smartphone', 'Smartphone de alta gama', 800.00, 100),  
(3, 'Teclado', 'Teclado mecánico', 100.00, 200);
```

```
INSERT INTO Pedido (id_pedido, id_cliente, fecha, total)  
VALUES (1, 1, '2025-03-20', 1800.00),  
(2, 2, '2025-03-21', 900.00);
```

```
INSERT INTO DetallePedido (id_detalle, id_pedido, id_producto, cantidad,  
precio_unitario)  
VALUES (1, 1, 1, 1, 1200.00),  
(2, 1, 3, 1, 100.00),  
(3, 2, 2, 1, 800.00);
```

### **Consulta**

```
SELECT P.nombre AS producto_nombre, P.descripcion AS producto_descripcion,  
DP.cantidad, DP.precio_unitario, (DP.cantidad * DP.precio_unitario) AS  
total_producto  
FROM DetallePedido DP  
JOIN Producto P ON DP.id_producto = P.id_producto
```

JOIN Pedido O ON DP.id\_pedido = O.id\_pedido  
WHERE O.id\_cliente = 2;

Data Output Messages Notifications					
<div> <div> <div>+</div> <div>📄</div> <div>▼</div> <div>📋</div> <div>▼</div> <div>🗑️</div> <div>🗄️</div> <div>⬇️</div> <div>📈</div> <div>SQL</div> </div> <div>Show</div> </div>					
	producto_nombre character varying (100)	producto_descripcion text	cantidad integer	precio_unitario numeric (10,2)	total_producto numeric
1	Smartphone	Smartphone de alta gama	1	800.00	800.00

### 3. Biblioteca Digital

- Se requiere un sistema para administrar préstamos de libros en una biblioteca digital.
- Identificar entidades clave: Usuario, Libro, Préstamo.
- Diseñar el modelo E-R que representa los préstamos y relaciones.
- Transformar el modelo en un conjunto de tablas relacionales.
- Implementar la base de datos en un DBMS.
- Realizar consultas SQL para obtener los préstamos activos de un usuario.

#### Entidades:

- **Usuario:** Es quien realiza el préstamo de los libros.
- **Libro:** Es el recurso disponible para préstamo en la biblioteca digital.
- **Préstamo:** Representa el préstamo de un libro a un usuario con detalles como la fecha de préstamo y la fecha de devolución.

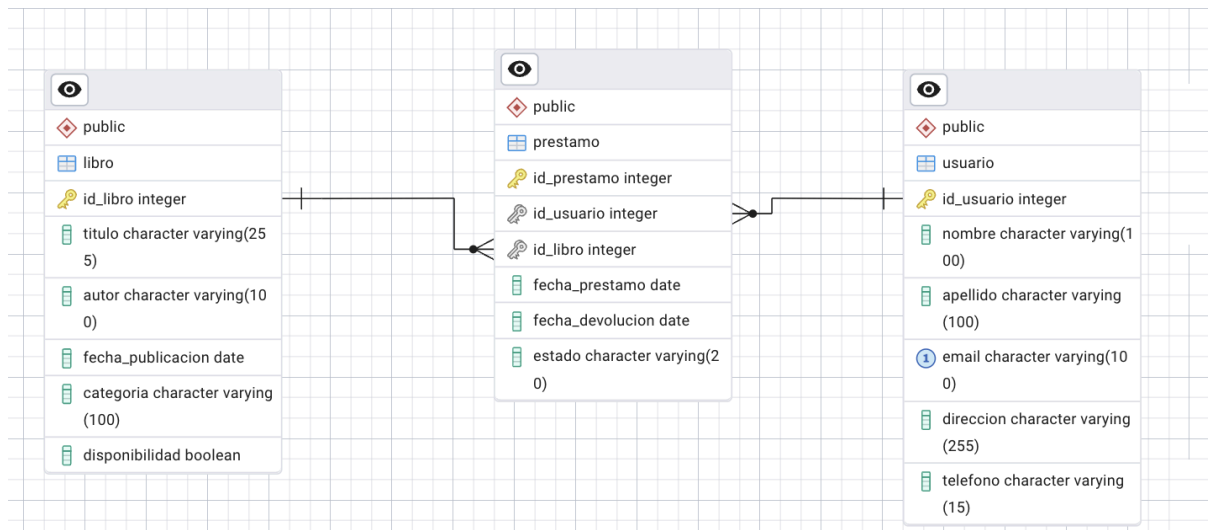
#### Atributos:

- **Usuario:** id\_usuario (clave primaria), nombre, apellido, email, fecha\_registro
- **Libro:** id\_libro (clave primaria), titulo, autor, anio\_publicacion, genero
- **Préstamo:** id\_prestamo (clave primaria), id\_usuario (clave foránea de la tabla Usuario), id\_libro (clave foránea de la tabla Libro), fecha\_prestamo, fecha\_devolucion

#### Modelo Entidad-Relación (E-R)

- Un **Usuario** puede tener muchos **Préstamos** (relación uno a muchos).
- Un **Libro** puede ser prestado varias veces a diferentes **Usuarios** (relación uno a muchos).





## Query:

### Tablas

```

CREATE TABLE Usuario (
  id_usuario INT PRIMARY KEY,
  nombre VARCHAR(100),
  apellido VARCHAR(100),
  email VARCHAR(100) UNIQUE,
  direccion VARCHAR(255),
  telefono VARCHAR(15)
);
  
```

```

CREATE TABLE Libro (
  id_libro INT PRIMARY KEY,
  titulo VARCHAR(255),
  autor VARCHAR(100),
  fecha_publicacion DATE,
  categoria VARCHAR(100),
  disponibilidad BOOLEAN
);
  
```

```

CREATE TABLE Prestamo (
  id_prestamo INT PRIMARY KEY,
  id_usuario INT,
  id_libro INT,
  fecha_prestamo DATE,
  fecha_devolucion DATE,
  estado VARCHAR(20),
  );
  
```

```
FOREIGN KEY (id_usuario) REFERENCES Usuario(id_usuario),
FOREIGN KEY (id_libro) REFERENCES Libro(id_libro)
);
```

### Insertar datos


```
INSERT INTO Usuario (id_usuario, nombre, apellido, email, direccion, telefono
VALUES (1, 'Juan', 'Pérez', 'juan.perez@email.com', 'Calle Ficticia 123', '555-1234'),
(2, 'Ana', 'González', 'ana.gonzalez@email.com', 'Avenida Libertad 456', '555-5678');
```

```
INSERT INTO Libro (id_libro, titulo, autor, fecha_publicacion, categoria,
disponibilidad)
VALUES (1, 'El Quijote', 'Miguel de Cervantes', '1605-01-01', 'Novela', TRUE),
(2, '1984', 'George Orwell', '1949-06-08', 'Distopía', TRUE),
(3, 'Cien Años de Soledad', 'Gabriel García Márquez', '1967-06-05', 'Realismo
Mágico', TRUE);
```

```
INSERT INTO Prestamo (id_prestamo, id_usuario, id_libro, fecha_prestamo,
fecha_devolucion, estado)
VALUES (1, 1, 1, '2025-03-15', NULL, 'Activo'),
(2, 2, 2, '2025-03-10', '2025-03-20', 'Devuelto');
```

### Consulta

```
SELECT L.titulo AS libro_titulo,
L.autor AS libro_autor,
P.fecha_prestamo, P.estado
FROM Prestamo P
JOIN Libro L ON P.id_libro = L.id_libro
WHERE P.id_usuario = 1 AND P.estado = 'Activo';
```

Data Output Messages Notifications				
<div> <div> <div>≡</div> <div>📄</div> <div>▼</div> <div>📋</div> <div>▼</div> <div>🗑️</div> <div>🗄️</div> <div>⬇️</div> <div>📈</div> <div>SQL</div> </div> <div>Showing rows: 1 to 1  Page No: 1 of 1 <div>⏪ ⏴ ⏵ ⏩</div></div> </div>				
	libro_titulo character varying (255) 🔒	libro_autor character varying (100) 🔒	fecha_prestamo date 🔒	estado character varying (20) 🔒
1	El Quijote	Miguel de Cervantes	2025-03-15	Activo

## 4. Sistema de Recursos Humanos

- Una empresa necesita gestionar sus empleados y departamentos.
- Definir entidades: Empleado, Departamento, Empresa.
- Elaborar el diagrama E-R que represente la estructura organizacional.
- Transformar el modelo en un esquema relacional.
- Implementar la base de datos en SQL, asegurando la integridad referencial.

- Consultar empleados por departamento mediante sentencias SQL.

### Entidades:

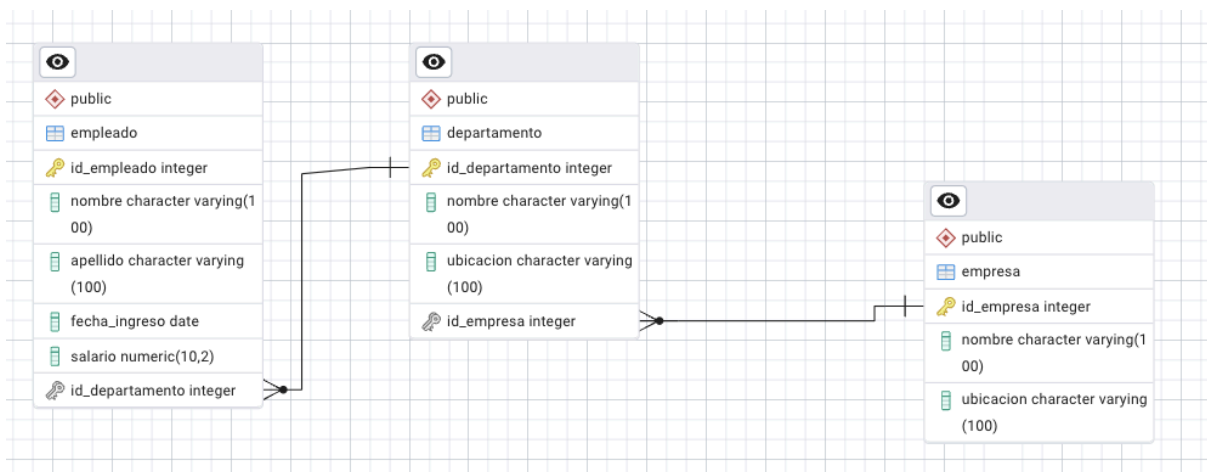
- **Empleado:** Es la persona que trabaja en la empresa.
- **Departamento:** Es una unidad dentro de la empresa que agrupa empleados según sus funciones.
- **Empresa:** Es la entidad que agrupa todos los departamentos y empleados.

### Atributos:

- **Empleado:** id\_empleado (clave primaria), nombre, apellido, fecha\_nacimiento, email, telefono, id\_departamento (clave foránea de la tabla Departamento)
- **Departamento:** id\_departamento (clave primaria), nombre\_departamento, ubicacion, id\_empresa (clave foránea de la tabla Empresa)
- **Empresa:** id\_empresa (clave primaria), nombre\_empresa, ubicacion, fecha\_fundacion

### Modelo Entidad-Relación (E-R)

- Un **Empleado** pertenece a un **Departamento** (relación muchos a uno).
- Un **Departamento** pertenece a una **Empresa** (relación muchos a uno).



### Query:

#### Tablas

```

CREATE TABLE Empresa (
  id_empresa INT PRIMARY KEY,
  nombre VARCHAR(100),
  ubicacion VARCHAR(100)
);
  
```

```
CREATE TABLE Departamento (  
id_departamento INT PRIMARY KEY,  
nombre VARCHAR(100),  
ubicacion VARCHAR(100),  
id_empresa INT,  
FOREIGN KEY (id_empresa) REFERENCES Empresa(id_empresa)  
);
```

```
CREATE TABLE Empleado (  
id_empleado INT PRIMARY KEY,  
nombre VARCHAR(100),  
apellido VARCHAR(100),  
fecha_ingreso DATE,  
salario DECIMAL(10, 2),  
id_departamento INT,  
FOREIGN KEY (id_departamento) REFERENCES Departamento(id_departamento)  
);
```

### **Insertar datos**

```
INSERT INTO Empresa (id_empresa, nombre, ubicacion)  
VALUES (1, 'Tech Solutions', 'Ciudad A'),  
(2, 'Innovatech', 'Ciudad B');
```

```
INSERT INTO Departamento (id_departamento, nombre, ubicacion, id_empresa)  
VALUES (1, 'Recursos Humanos', 'Piso 1', 1),  
(2, 'Desarrollo', 'Piso 2', 1),  
(3, 'Ventas', 'Piso 3', 2);
```

```
INSERT INTO Empleado (id_empleado, nombre, apellido, fecha_ingreso, salario,  
id_departamento)  
VALUES (1, 'Juan', 'Pérez', '2023-01-15', 3500.00, 1),  
(2, 'Ana', 'González', '2022-06-10', 4500.00, 2),  
(3, 'Carlos', 'Martínez', '2021-03-05', 5500.00, 2),  
(4, 'María', 'López', '2024-02-20', 3000.00, 1),  
(5, 'Luis', 'Sánchez', '2023-10-15', 4000.00, 3);
```

### **Consulta**

```
SELECT E.id_empleado, E.nombre, E.apellido, E.fecha_ingreso, E.salario,  
D.nombre AS departamento  
FROM Empleado E  
JOIN Departamento D ON E.id_departamento = D.id_departamento  
WHERE D.nombre = 'Desarrollo';
```

Data Output Messages Notifications						
<div> <div> <div>≡</div> <div>+</div> </div> <div> <div>📄</div> <div>▼</div> </div> <div> <div>📋</div> <div>▼</div> </div> <div> <div>🗑️</div> <div>🔄</div> </div> <div> <div>📥</div> <div>📤</div> </div> <div> <div>📊</div> <div>SQL</div> </div> </div> <div>Showing rows: 1 to 2 <span>✎</span> Page</div>						
	id_empleado integer	nombre character varying (100)	apellido character varying (100)	fecha_ingreso date	salario numeric (10,2)	departamento character varying (100)
1	2	Ana	González	2022-06-10	4500.00	Desarrollo
2	3	Carlos	Martínez	2021-03-05	5500.00	Desarrollo

## 5. Plataforma de Cursos en Línea

- Se requiere un sistema para gestionar la inscripción de usuarios en cursos en línea.
- Identificar entidades clave: Usuario, Curso, Inscripción.
- Diseñar un modelo E-R para representar la relación entre usuarios y cursos.
- Convertir el modelo en un conjunto de tablas con relaciones adecuadas.
- Implementar la base de datos en SQL con restricciones adecuadas.
- Consultar los cursos inscritos por un usuario en la base de datos.

### Entidades:

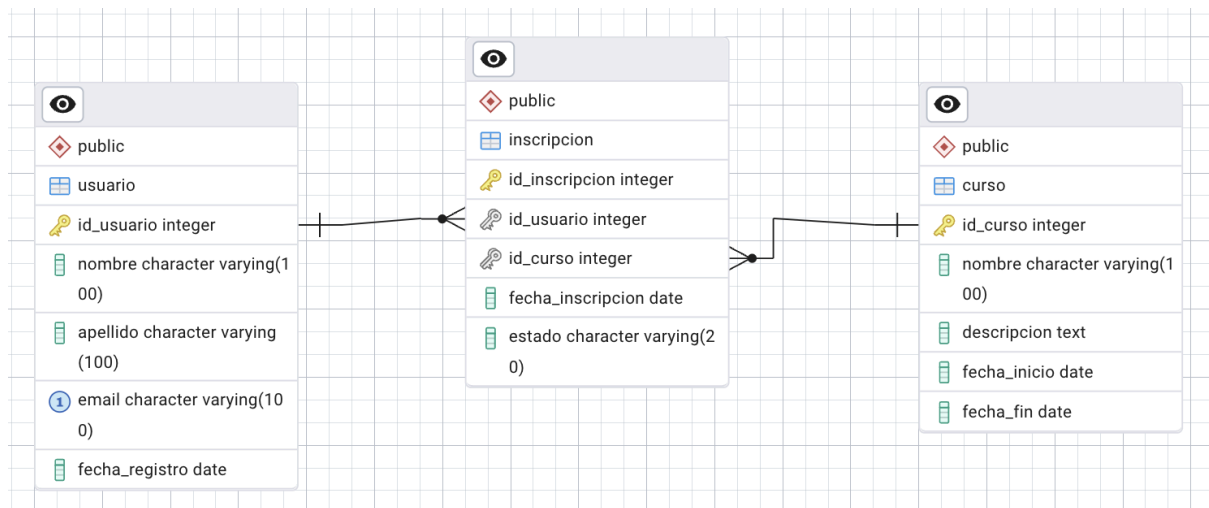
- **Usuario:** Es la persona que se inscribe en los cursos.
- **Curso:** Es el curso que se ofrece en la plataforma.
- **Inscripción:** Representa la inscripción de un usuario en un curso específico.

### Atributos:

- **Usuario:** id\_usuario (clave primaria), nombre, apellido, email, fecha\_registro
- **Curso:** id\_curso (clave primaria), titulo, descripcion, fecha\_inicio, fecha\_fin
- **Inscripción:** id\_inscripcion (clave primaria), id\_usuario (clave foránea de la tabla Usuario), id\_curso (clave foránea de la tabla Curso), fecha\_inscripcion

### Modelo Entidad-Relación (E-R)

- Un **Usuario** puede inscribirse en muchos **Cursos** (relación muchos a muchos).
- Un **Curso** puede tener muchos **Usuarios** inscritos (relación muchos a muchos).



## Query:

### Tablas

```

CREATE TABLE Usuario (
  id_usuario INT PRIMARY KEY,
  nombre VARCHAR(100),
  apellido VARCHAR(100),
  email VARCHAR(100) UNIQUE,
  fecha_registro DATE
);
  
```

```

CREATE TABLE Curso (
  id_curso INT PRIMARY KEY,
  nombre VARCHAR(100),
  descripcion TEXT,
  fecha_inicio DATE,
  fecha_fin DATE
);
  
```

```

CREATE TABLE Inscripcion (
  id_inscripcion INT PRIMARY KEY,
  id_usuario INT,
  id_curso INT,
  fecha_inscripcion DATE,
  estado VARCHAR(20), -- Ejemplo: 'activo', 'completado', 'cancelado'
  FOREIGN KEY (id_usuario) REFERENCES Usuario(id_usuario),
  FOREIGN KEY (id_curso) REFERENCES Curso(id_curso)
);
  
```

### Insertar dato

```
INSERT INTO Usuario (id_usuario, nombre, apellido, email, fecha_registro)
VALUES (1, 'Juan', 'Pérez', 'juan.perez@email.com', '2025-01-01'),
(2, 'Ana', 'González', 'ana.gonzalez@email.com', '2025-02-15'),
(3, 'Carlos', 'Martínez', 'carlos.martinez@email.com', '2025-03-10');
```

```
INSERT INTO Curso (id_curso, nombre, descripcion, fecha_inicio, fecha_fin)
VALUES (1, 'Curso de SQL', 'Aprende SQL desde cero', '2025-04-01', '2025-06-01'),
(2, 'Curso de Python', 'Fundamentos de Python', '2025-04-15', '2025-06-15'),
(3, 'Curso de Desarrollo Web', 'Desarrollo Web con HTML, CSS y JavaScript',
'2025-05-01', '2025-07-01');
```

```
INSERT INTO Inscripcion (id_inscripcion, id_usuario, id_curso, fecha_inscripcion,
estado)
VALUES (1, 1, 1, '2025-03-20', 'activo'),
(2, 1, 2, '2025-03-21', 'activo'),
(3, 2, 1, '2025-03-22', 'completado'),
(4, 3, 3, '2025-03-23', 'activo');
```

## Consulta

```
SELECT C.id_curso, C.nombre AS curso_nombre, C.descripcion, C.fecha_inicio,
C.fecha_fin, I.fecha_inscripcion, I.estado
FROM Inscripcion I
JOIN Curso C ON I.id_curso = C.id_curso
WHERE I.id_usuario = 1;
```

Data Output Messages Notifications							
	id_curso integer	curso_nombre character varying (100)	descripcion text	fecha_inicio date	fecha_fin date	fecha_inscripcion date	estado character varying (20)
1	1	Curso de SQL	Aprende SQL desde cero	2025-04-01	2025-06-01	2025-03-20	activo
2	2	Curso de Python	Fundamentos de Python	2025-04-15	2025-06-15	2025-03-21	activo

