INSTITUTO TECNOLOGICO DE MEXICO INGENIERO EN SISTEMAS COMPUTACIONALES

Armas Diaz Erick Hidekio N.ctrol: 23490411

FUNDAMENTOS DE BASE DE DATOS

Tarea 2 Unidad 2 Fecha: 21/03/25



Codigos SQLS y sus Entidades

1. Sistema de Gestion de Inventarios

Entidades:

- Producto (id, nombre, precio, id_categoria, id_proveedor)
- · Categoría (id, nombre)
- Proveedor (id, nombre, contacto)
- Inventario (id_producto, stock, ubicacion)

```
CREATE TABLE Categoria (
id SERIAL PRIMARY KEY,
nombre VARCHAR(100) NOT NULL
);
```

```
CREATE TABLE Proveedor (
id SERIAL PRIMARY KEY,
nombre VARCHAR(100) NOT NULL,
contacto VARCHAR(100)
);
```

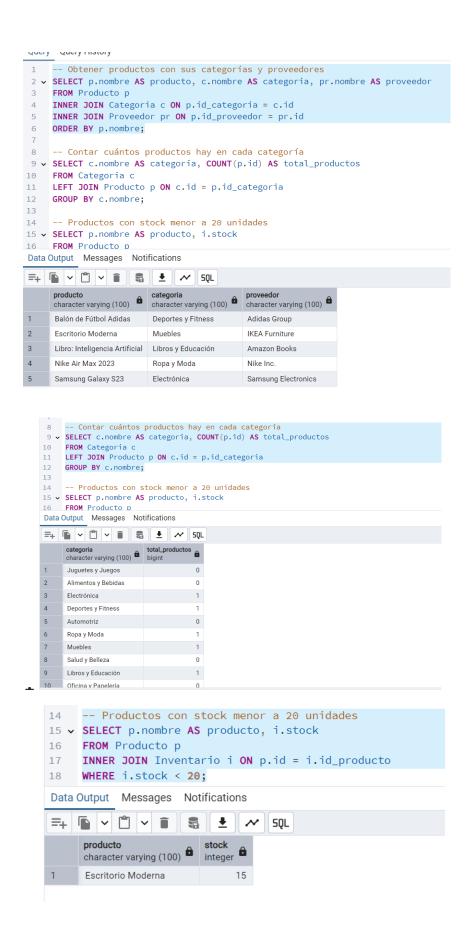
```
CREATE TABLE Producto (
 id SERIAL PRIMARY KEY,
 nombre VARCHAR(100) NOT NULL,
 precio DECIMAL(10,2) NOT NULL,
 id_categoria INT REFERENCES Categoria(id),
 id_proveedor INT REFERENCES Proveedor(id)
);
CREATE TABLE Inventario (
 id producto INT PRIMARY KEY REFERENCES
Producto(id),
 stock INT NOT NULL,
 ubicacion VARCHAR(100) NOT NULL
);
ERD y Consultas
2. Sistema de Gestion de Eventos.
Codigo SQL y Entidades.
Entidades:
```

- Evento (id, nombre, fecha, id_ubicacion, id_organizador)
- · Ubicación (id, direccion)
- Organizador (id, nombre, contacto)
- · Participante (id, nombre, email)
- Registro (id_evento, id_participante)
 CREATE TABLE Ubicacion (
 id SERIAL PRIMARY KEY,
 direccion VARCHAR(255) NOT NULL
);

```
CREATE TABLE Organizador (
id SERIAL PRIMARY KEY,
nombre VARCHAR(100) NOT NULL,
contacto VARCHAR(100)
);
```

CREATE TABLE Evento (
id SERIAL PRIMARY KEY,

```
nombre VARCHAR(100) NOT NULL,
 fecha DATE NOT NULL,
 id_ubicacion INT REFERENCES Ubicacion(id),
 id_organizador INT REFERENCES Organizador(id)
);
CREATE TABLE Participante (
 id SERIAL PRIMARY KEY,
 nombre VARCHAR(100) NOT NULL,
 email VARCHAR(100) UNIQUE NOT NULL
);
CREATE TABLE Registro (
 id_evento INT REFERENCES Evento(id),
 id_participante INT REFERENCES Participante(id),
 PRIMARY KEY (id_evento, id_participante)
);
ERD y Consultas.
```



2. Sistema de Gestión de Eventos Entidades:

- Evento (id, nombre, fecha, id_ubicacion, id_organizador)
- Ubicación (id, direccion)
- Organizador (id, nombre, contacto)
- Participante (id, nombre, email)
- Registro (id_evento, id_participante)Sql.

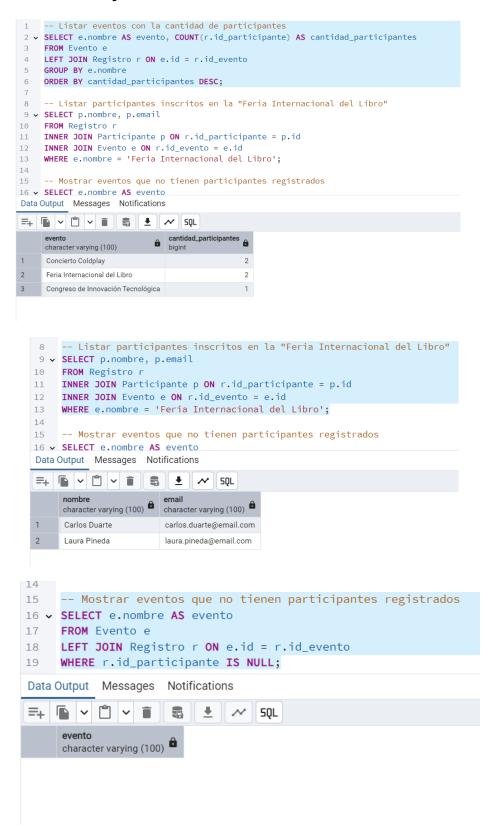
```
CREATE TABLE Ubicacion (
id SERIAL PRIMARY KEY,
direccion VARCHAR(255) NOT NULL
);
```

```
CREATE TABLE Organizador (
id SERIAL PRIMARY KEY,
nombre VARCHAR(100) NOT NULL,
contacto VARCHAR(100)
);
```

```
CREATE TABLE Evento (
```

```
id SERIAL PRIMARY KEY,
 nombre VARCHAR(100) NOT NULL,
 fecha DATE NOT NULL,
 id_ubicacion INT REFERENCES Ubicacion(id),
 id_organizador INT REFERENCES Organizador(id)
);
CREATE TABLE Participante (
 id SERIAL PRIMARY KEY,
 nombre VARCHAR(100) NOT NULL,
 email VARCHAR(100) UNIQUE NOT NULL
);
CREATE TABLE Registro (
 id_evento INT REFERENCES Evento(id),
 id_participante INT REFERENCES Participante(id),
 PRIMARY KEY (id_evento, id_participante)
);
```

ERD y Consultas.



3. Plataforma de Sreaming de Musica.

Codigo SQL y Entidades.

Entidades:

- · Usuario (id, nombre, email)
- Artista (id, nombre)

id SERIAL PRIMARY KEY,

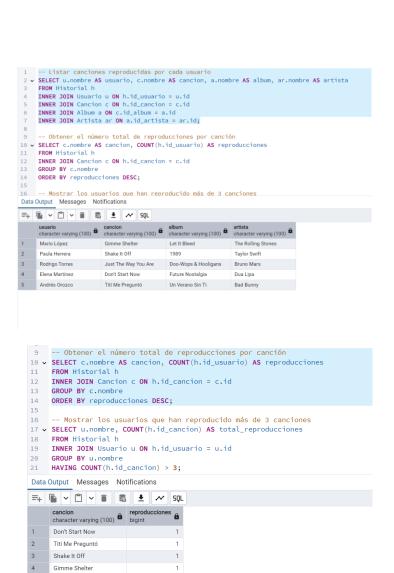
);

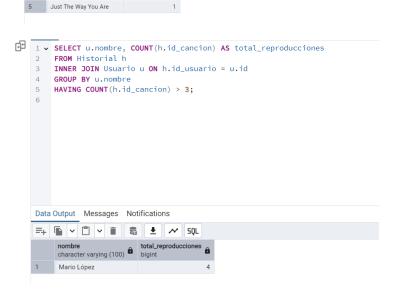
nombre VARCHAR(100) NOT NULL

- Álbum (id, nombre, id_artista)
- Canción (id, nombre, id_album)
- Historial (id_usuario, id_cancion, fecha)
 SQL.

```
CREATE TABLE Usuario (
id SERIAL PRIMARY KEY,
nombre VARCHAR(100) NOT NULL,
email VARCHAR(100) UNIQUE NOT NULL
);
CREATE TABLE Artista (
```

```
CREATE TABLE Album (
 id SERIAL PRIMARY KEY,
 nombre VARCHAR(100) NOT NULL,
 id_artista INT REFERENCES Artista(id)
);
CREATE TABLE Cancion (
 id SERIAL PRIMARY KEY,
 nombre VARCHAR(100) NOT NULL,
 id_album INT REFERENCES Album(id)
);
CREATE TABLE Historial (
 id_usuario INT REFERENCES Usuario(id),
 id_cancion INT REFERENCES Cancion(id),
 fecha TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
ERD y Consultas.
```





4. Sistema de Control de Proyectos. Codigo SQL y Entidades . Proyecto (id, nombre, id_empleado) . Empleado (id, nombre, puesto) . Tarea (id, descripcion, id_proyecto, fecha_vencimiento, estado) SQL. CREATE TABLE Empleado (id SERIAL PRIMARY KEY, nombre VARCHAR(100) NOT NULL, puesto VARCHAR(100) NOT NULL); **CREATE TABLE Proyecto (** id SERIAL PRIMARY KEY, nombre VARCHAR(100) NOT NULL, id_empleado INT REFERENCES Empleado(id));

CREATE TABLE Tarea (

```
id SERIAL PRIMARY KEY,

descripcion TEXT NOT NULL,

id_proyecto INT REFERENCES Proyecto(id),

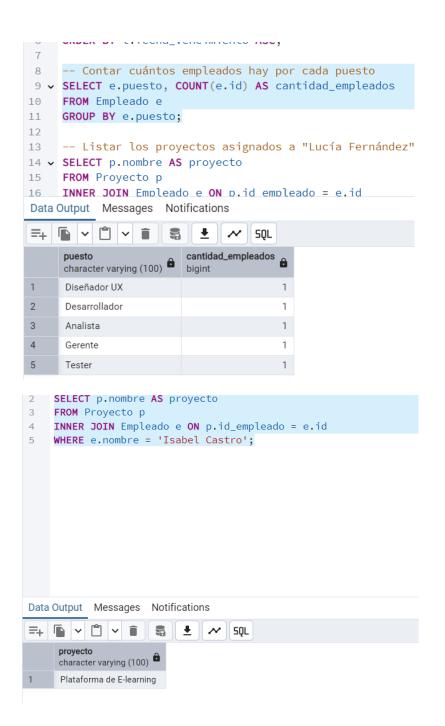
fecha_vencimiento DATE NOT NULL,

estado VARCHAR(20) CHECK (estado IN ('pendiente', 'en progreso', 'completada'))

);
```

ERD y Consultas.

```
1 -- Mostrar tareas pendientes ordenadas por fecha
 2 SELECT t.descripcion, p.nombre AS proyecto, t.fecha_vencimiento
   FROM Tarea t
 4 INNER JOIN Proyecto p ON t.id_proyecto = p.id
   WHERE t.estado = 'pendiente'
    ORDER BY t.fecha_vencimiento ASC;
 8 -- Contar cuántos empleados hay por cada puesto
9 - SELECT e.puesto, COUNT(e.id) AS cantidad_empleados
10 FROM Empleado e
11 GROUP BY e.puesto;
12
    -- Listar los proyectos asignados a "Lucía Fernández"
13
14 v SELECT p.nombre AS proyecto
15 FROM Proyecto p
16 INNER JOIN Empleado e ON p.id empleado = e.id
Data Output Messages Notifications
=+ 🖺 ∨ 🖺 ∨ 🛊 👼 👲 🕢 SQL
                                     fecha_vencimiento
                  character varying (100) date
    Diseño de interfaz Desarrollo de App
                                     2024-07-01
```



5. Sistema de Evaluacion Academica.

Codigos SQL y Entidades

Entidades:

- Estudiante (id, nombre, email)
- Curso (id, nombre)
- Profesor (id, nombre)
- Calificación (id_estudiante, id_curso, id_profesor, nota)

CREATE TABLE Estudiante (
id SERIAL PRIMARY KEY,
nombre VARCHAR(100) NOT NULL,
email VARCHAR(100) UNIQUE NOT NULL

```
);
CREATE TABLE Curso (
 id SERIAL PRIMARY KEY,
 nombre VARCHAR(100) NOT NULL
);
CREATE TABLE Profesor (
 id SERIAL PRIMARY KEY,
 nombre VARCHAR(100) NOT NULL
);
CREATE TABLE Calificacion (
 id_estudiante INT REFERENCES Estudiante(id),
 id_curso INT REFERENCES Curso(id),
 id_profesor INT REFERENCES Profesor(id),
 nota DECIMAL(5,2) NOT NULL,
 PRIMARY KEY (id_estudiante, id_curso, id_profesor)
);
```

ERD y Consultas.

