TAREA 4 UNIDAD 2

Fundamentos de base de datos

22 DE MARZO DE 2025

DAMIAN ALEXANDER LUGO AGUILAR- 23490386
Prof. Jose Ramon Bogarin Valenzuela

INDICE

Sistema de gestion de inventarios 2

Sistema de gestion de eventos 5

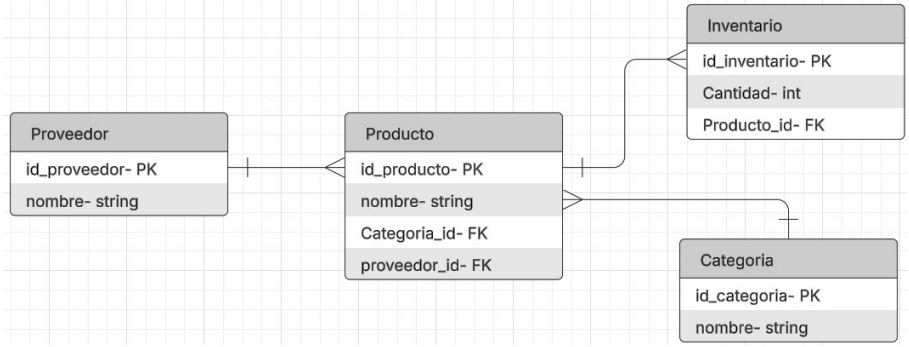
Plataforma de streaming de musica 7

Sistema de control de proyectos 10

Sistema de evaluacion academica 12

Sistema de gestion de inventarios

Una empresa desea controlar su inventario de productos y proveedores, tenemos las siguientes entidades clave: producto, proveedor, categoria e inventario.



Asi nos quedaria nuestro diagrama E-R, con esto podemos crear nuestras tablas

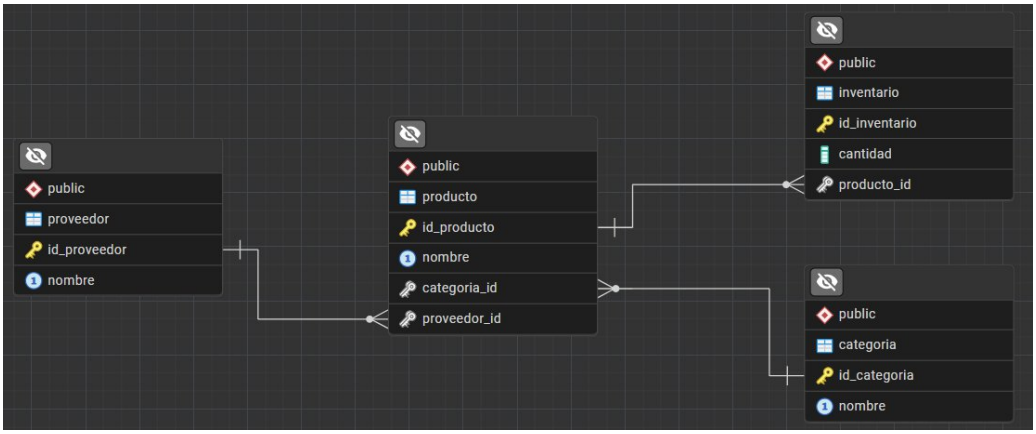
```sql
CREATE TABLE proveedor (
    id_proveedor SERIAL PRIMARY KEY,
    nombre VARCHAR(100) UNIQUE NOT NULL
);

CREATE TABLE categoria (
    id_categoria SERIAL PRIMARY KEY,
    nombre VARCHAR(100) UNIQUE NOT NULL
);

CREATE TABLE producto (
    id_producto SERIAL PRIMARY KEY,
    nombre VARCHAR(100) UNIQUE NOT NULL,
    categoria_id INT NOT NULL,
    proveedor_id INT NOT NULL,
    FOREIGN KEY (categoria_id) REFERENCES categoria(id_categoria),
    FOREIGN KEY (proveedor_id) REFERENCES proveedor(id_proveedor)
);

CREATE TABLE inventario (
    id_inventario SERIAL PRIMARY KEY,
    cantidad INT NOT NULL,
    producto_id INT NOT NULL,
    FOREIGN KEY (producto_id) REFERENCES producto(id_producto)
)
```

Y podemos comprobar que nuestro diagrama inicial coincide con nuestro query



Ahora insertamos los datos en las tablas

```sql
INSERT INTO Categoria (nombre) VALUES
('Electrónica'),
('Hogar'),
('Ropa'),
('Deportes'),
('Juguetes'),
('Alimentos'),
('Salud'),
('Belleza'),
('Automotriz'),
('Libros');

INSERT INTO Proveedor (nombre) VALUES
('Tech Supplier'),
('Hogar Plus'),
('Moda Express'),
('Deportes Global'),
('Juguetería Central'),
('Alimentos Frescos'),
('Salud y Vida'),
('Estética Total'),
('Auto Parts'),
('Editorial Alfa');
```

```sql
INSERT INTO Producto (nombre, categoria_id, proveedor_id) VALUES
('Televisor 55"', 21, 21),
('Licuadora ProMix', 22, 22),
('Camiseta Deportiva', 23, 23),
('Bicicleta Montaña', 24, 23),
('Muñeca Interactiva', 25, 24),
('Chocolate Orgánico', 26, 25),
('Vitaminas C', 27, 26),
('Shampoo Natural', 28, 27),
('Aceite de Motor', 29, 29),
('Libro de Ciencia Ficción', 30, 30);

INSERT INTO Inventario (producto_id, cantidad) VALUES
(22, 50),
(23, 30),
(24, 100),
(25, 25),
(26, 75),
(27, 120),
(28, 60),
(29, 40),
(30, 15),
(31, 90);
```

Y ahora hacemos nuestras consultas de datos

```sql
--Con ORDER BY (principal)
SELECT p.nombre AS producto, c.nombre AS categoria, pr.nombre AS proveedor, p.proveedor_id
FROM Producto p LEFT JOIN Categoria c
ON p.categoria_id = c.id_categoria
LEFT JOIN proveedor pr
ON pr.id_proveedor= p.proveedor_id
ORDER BY p.nombre;
```

	producto character varying (100)	categoria character varying (100)	proveedor character varying (100)	proveedor_id integer
1	Aceite de Motor	Automotriz	Auto Parts	29
2	Bicicleta Montaña	Deportes	Moda Express	23
3	Camiseta Deportiva	Ropa	Moda Express	23
4	Chocolate Orgánico	Alimentos	Juguetería Central	25
5	Libro de Ciencia Ficción	Libros	Editorial Alfa	30
6	Licuadora ProMix	Hogar	Hogar Plus	22
7	Muñeca Interactiva	Juguetes	Deportes Global	24
8	Shampoo Natural	Belleza	Salud y Vida	27
9	Televisor 55"	Electrónica	Tech Supplier	21
10	Vitaminas C	Salud	Alimentos Frescos	26

```sql
--con GROUP BY
SELECT p.nombre AS producto, c.nombre AS categoria, pr.nombre AS proveedor, p.proveedor_id
FROM Producto p LEFT JOIN Categoria c
ON p.categoria_id = c.id_categoria
LEFT JOIN proveedor pr
ON pr.id_proveedor= p.proveedor_id
GROUP BY p.id_producto, c.nombre,pr.nombre
```

	producto character varying (100)	categoria character varying (100)	proveedor character varying (100)	proveedor_id integer
1	Aceite de Motor	Automotriz	Auto Parts	29
2	Licuadora ProMix	Hogar	Hogar Plus	22
3	Bicicleta Montaña	Deportes	Moda Express	23
4	Shampoo Natural	Belleza	Salud y Vida	27
5	Muñeca Interactiva	Juguetes	Deportes Global	24
6	Televisor 55"	Electrónica	Tech Supplier	21
7	Camiseta Deportiva	Ropa	Moda Express	23
8	Chocolate Orgánico	Alimentos	Juguetería Central	25
9	Libro de Ciencia Ficción	Libros	Editorial Alfa	30
10	Vitaminas C	Salud	Alimentos Frescos	26

```sql
--con WHERE
SELECT p.nombre AS producto, c.nombre AS categoria, pr.nombre AS proveedor, p.proveedor_id
FROM Producto p LEFT JOIN Categoria c
ON p.categoria_id = c.id_categoria
LEFT JOIN proveedor pr
ON pr.id_proveedor= p.proveedor_id
WHERE p.nombre='Aceite de Motor';
```

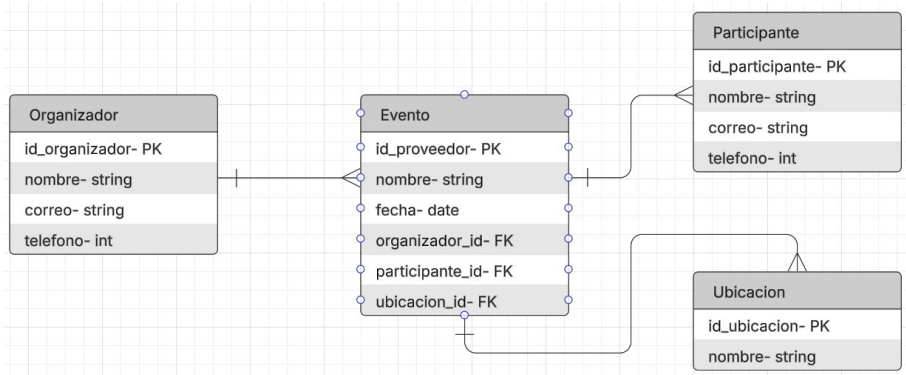
	producto character varying (100)	categoria character varying (100)	proveedor character varying (100)	proveedor_id integer
1	Aceite de Motor	Automotriz	Auto Parts	29

```sql
--con LIKE (WHERE)
SELECT p.nombre AS producto, c.nombre AS categoria, pr.nombre AS proveedor, p.proveedor_id
FROM Producto p LEFT JOIN Categoria c
ON p.categoria_id = c.id_categoria
LEFT JOIN proveedor pr
ON pr.id_proveedor= p.proveedor_id
WHERE p.nombre LIKE '%C%'
```

	producto character varying (100)	categoria character varying (100)	proveedor character varying (100)	proveedor_id integer
1	Camiseta Deportiva	Ropa	Moda Express	23
2	Chocolate Orgánico	Alimentos	Juguetería Central	25
3	Vitaminas C	Salud	Alimentos Frescos	26
4	Libro de Ciencia Ficción	Libros	Editorial Alfa	30

Sistema de gestion de eventos

Una empresa de organizacion de eventos necesita administrar sus eventos y participantes, tenemos las siguentes entidades clave: Evento, participante, ubicacion y organizador.



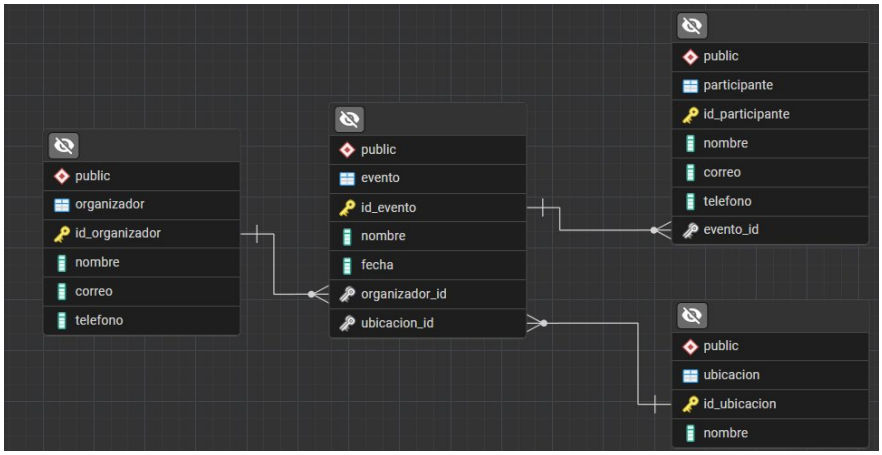
Con esto podemos crear nuestras tablas

```sql
CREATE TABLE ubicacion(
    id_ubicacion SERIAL PRIMARY KEY,
    nombre VARCHAR(100)
);

CREATE TABLE organizador(
    id_organizador SERIAL PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    correo VARCHAR(100) NOT NULL,
    telefono INT NOT NULL
);

CREATE TABLE evento(
    id_evento SERIAL PRIMARY KEY,
    nombre VARCHAR(100),
    fecha TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    organizador_id INT NOT NULL,
    ubicacion_id INT UNIQUE NOT NULL,
    FOREIGN KEY (organizador_id) REFERENCES organizador(id_organizador),
    FOREIGN KEY (ubicacion_id) REFERENCES ubicacion(id_ubicacion)
);

CREATE TABLE participante(
    id_participante SERIAL PRIMARY KEY,
    nombre VARCHAR(100),
    correo VARCHAR(100) NOT NULL,
    telefono INT NOT NULL,
    evento_id INT NOT NULL,
    FOREIGN KEY (evento_id) REFERENCES evento(id_evento)
);
```



Ahora insertamos los datos en las tablas

```sql
INSERT INTO Ubicacion (nombre) VALUES
('Centro de Convenciones'),
('Sala de Conferencias A'),
('Auditorio Central'),
('Salón VIP'),
('Centro Tecnológico'),
('Parque Empresarial'),
('Biblioteca Nacional'),
('Hotel Internacional'),
('Universidad Local'),
('Teatro Municipal');

INSERT INTO organizador (nombre, correo, telefono) VALUES
('Empresa Eventos SA', 'eventos@empresa.com', 123456789),
('Congresos Globales', 'info@congresos.com', 987654321),
('TechMeetings', 'contacto@techmeetings.com', 555666777),
('Organización Científica', 'ciencia@org.com', 666777888),
('Eventos Corporativos', 'corporativos@eventos.com', 777888999),
('Startup Innovadora', 'startup@innovadora.com', 888999000),
('Fundación Educativa', 'educacion@fundacion.com', 999000111),
('Asociación Cultural', 'cultura@asociacion.com', 111222333),
('Conferencias Médicas', 'medicas@conferencias.com', 222333444),
('Comité Empresarial', 'empresarial@comite.com', 333444555);
```

```sql
INSERT INTO evento (nombre, fecha, organizador_id, ubicacion_id) VALUES
('Conferencia de Tecnología', '2025-04-10', 1, 1),
('Foro Empresarial', '2025-05-15', 2, 2),
('Congreso Científico', '2025-06-20', 3, 3),
('Seminario de Innovación', '2025-07-05', 4, 4),
('Feria Educativa', '2025-08-12', 5, 5),
('Encuentro Cultural', '2025-09-25', 6, 6),
('Simposio Médico', '2025-10-30', 7, 7),
('Cumbre Empresarial', '2025-11-18', 8, 8),
('Hackathon Global', '2025-12-05', 9, 9),
('Jornada de Networking', '2026-01-15', 10, 10);

INSERT INTO participante (nombre, correo, telefono, evento_id) VALUES
('Gerardo Diaz', 'gerardo.diaz@email.com', 123356789,6),
('Juan Pérez', 'juan.perez@email.com', 123456789, 1),
('María López', 'maria.lopez@email.com', 987654321, 2),
('Carlos Gómez', 'carlos.gomez@email.com', 555666777, 3),
('Ana Torres', 'ana.torres@email.com', 666777888, 4),
('Pedro Ramírez', 'pedro.ramirez@email.com', 777888999, 5),
('Laura Fernández', 'laura.fernandez@email.com', 888999000, 6),
('Javier Herrera', 'javier.herrera@email.com', 999000111, 7),
('Sofía Martínez', 'sofia.martinez@email.com', 111222333, 8),
('Luis García', 'luis.garcia@email.com', 222333444, 9),
('Elena Rodríguez', 'elena.rodriguez@email.com', 333444555, 10);
```

Ahora hacemos las consultas de datos

```sql
--Con GROUP BY y ORDER BY (principal)
SELECT e.id_evento, e.nombre AS nombre_evento, e.fecha, COUNT(p.id_participante) AS cantidad_participantes
FROM evento e
LEFT JOIN participante p ON e.id_evento = p.evento_id
GROUP BY e.id_evento, e.nombre, e.fecha
ORDER BY e.fecha;
```

	id_evento [PK] integer	nombre_evento character varying (100)	fecha timestamp without time zone	cantidad_participantes bigint
1	1	Conferencia de Tecnología	2025-04-10 00:00:00	1
2	2	Foro Empresarial	2025-05-15 00:00:00	1
3	3	Congreso Científico	2025-06-20 00:00:00	1
4	4	Seminario de Innovación	2025-07-05 00:00:00	1
5	5	Feria Educativa	2025-08-12 00:00:00	1
6	6	Encuentro Cultural	2025-09-25 00:00:00	2
7	7	Simposio Médico	2025-10-30 00:00:00	1
8	8	Cumbre Empresarial	2025-11-18 00:00:00	1
9	9	Hackathon Global	2025-12-05 00:00:00	1
10	10	Jornada de Networking	2026-01-15 00:00:00	1

```sql
--Con LIKE
SELECT e.id_evento, e.nombre AS nombre_evento, e.fecha, COUNT(p.id_participante) AS cantidad_participantes
FROM evento e
LEFT JOIN participante p ON e.id_evento = p.evento_id
WHERE e.nombre LIKE '%Encuentro%'
GROUP BY e.id_evento, e.nombre, e.fecha
ORDER BY e.fecha;
```

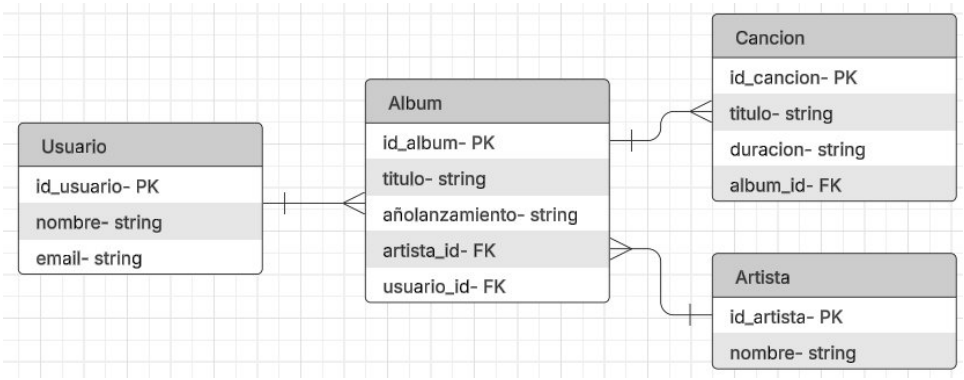
	id_evento [PK] integer	nombre_evento character varying (100)	fecha timestamp without time zone	cantidad_participantes bigint
1	6	Encuentro Cultural	2025-09-25 00:00:00	2

```sql
--Con WHERE
SELECT e.id_evento, e.nombre AS nombre_evento, e.fecha, COUNT(p.id_participante) AS cantidad_participantes
FROM evento e LEFT JOIN participante p
ON e.id_evento = p.evento_id
WHERE e.nombre = 'Encuentro Cultural'
GROUP BY e.id_evento, e.nombre, e.fecha
ORDER BY e.fecha;
```

	id_evento [PK] integer	nombre_evento character varying (100)	fecha timestamp without time zone	cantidad_participantes bigint
1	6	Encuentro Cultural	2025-09-25 00:00:00	2

Plataforma de streaming de musica

Se desea un sistema para administrar ususarios, artistas y sus canciones, aparter de las ya dichas otra de nuestras entidades clave es album



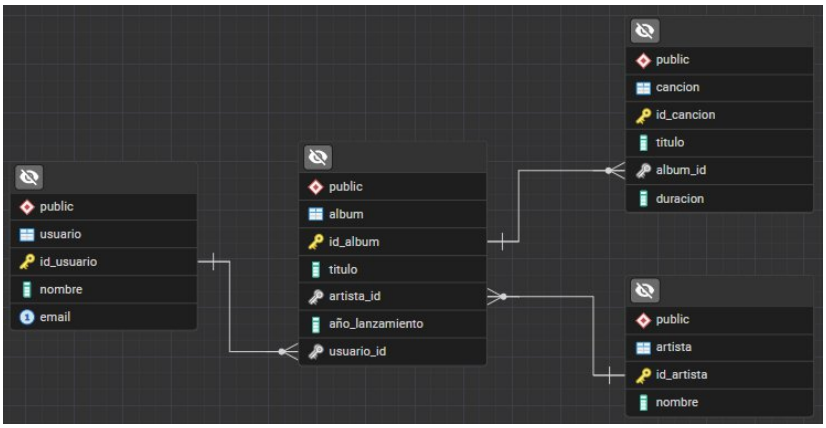
Una vez teniendo esto podemos crear nuestras tablas

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

CREATE TABLE Artista (
    id_artista SERIAL PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL
);

CREATE TABLE Album (
    id_album SERIAL PRIMARY KEY,
    titulo VARCHAR(200) NOT NULL,
    artista_id INT NOT NULL,
    año_lanzamiento INT CHECK (año_lanzamiento > 1900),
    usuario_id INT NOT NULL,
    FOREIGN KEY (artista_id) REFERENCES Artista(id_artista) ON DELETE CASCADE,
    FOREIGN KEY (usuario_id) REFERENCES Usuario(id_usuario)
);

CREATE TABLE Cancion (
    id_cancion SERIAL PRIMARY KEY,
    titulo VARCHAR(200) NOT NULL,
    album_id INT NOT NULL,
    duracion INTERVAL NOT NULL,
    FOREIGN KEY (album_id) REFERENCES Album(id_album) ON DELETE CASCADE
);
```



Ahora insertamos los datos

```sql
INSERT INTO Usuario (nombre, email) VALUES
('Carlos Pérez', 'carlos@email.com'),
('Ana Gómez', 'ana@email.com'),
('Luis Ramírez', 'luis@email.com'),
('María López', 'maria@email.com'),
('Pedro Fernández', 'pedro@email.com'),
('Laura Torres', 'laura@email.com'),
('Diego Castro', 'diego@email.com'),
('Elena Sánchez', 'elena@email.com'),
('Javier Mendoza', 'javier@email.com'),
('Sofía Herrera', 'sofia@email.com');

INSERT INTO Artista (nombre) VALUES
('Coldplay'),
('Ed Sheeran'),
('Adele'),
('Shakira'),
('Queen'),
('The Beatles'),
('Bruno Mars'),
('Taylor Swift'),
('Bad Bunny'),
('Imagine Dragons');
```

```sql
INSERT INTO Album (titulo, artista_id, año_lanzamiento, usuario_id) VALUES
('Parachutes', 21, 2000, 21),
('Divide', 22, 2017, 22),
('25', 23, 2015, 23),
('El Dorado', 24, 2017, 24),
('A Night at the Opera', 25, 1975, 25),
('Abbey Road', 26, 1969, 26),
('Doo-Wops & Hooligans', 27, 2010, 27),
('1989', 28, 2014, 28),
('YHLQMDLG', 29, 2020, 29),
('Evolve', 30, 2017, 30);

INSERT INTO Cancion (titulo, album_id, duracion) VALUES
('Yellow', 21, '00:04:26'),
('Shape of You', 22, '00:03:53'),
('Hello', 23, '00:04:55'),
('Chantaje', 24, '00:03:16'),
('Bohemian Rhapsody', 25, '00:05:55'),
('Here Comes The Sun', 26, '00:03:05'),
('Just the Way You Are', 27, '00:03:40'),
('Shake It Off', 28, '00:03:39'),
('Safaera', 29, '00:04:55'),
('Believer', 30, '00:03:24');
```

Ahora hacemos nuestras consultas

```
--Con where
SELECT  u.nombre AS usuario, c.titulo AS cancion, a.titulo AS album, ar.nombre AS artista
FROM album a
INNER JOIN Cancion c ON a.id_album = c.album_id
INNER JOIN usuario u ON u.id_usuario = a.usuario_id
INNER JOIN Artista ar ON a.artista_id = ar.id_artista
WHERE u.id_usuario = 21
```

	usuario character varying (100)	cancion character varying (200)	album character varying (200)	artista character varying (100)
1	Carlos Pérez	Yellow	Parachutes	Coldplay

```
--Con LIKE
SELECT  u.nombre AS usuario, c.titulo AS cancion, a.titulo AS album, ar.nombre AS artista
FROM album a
INNER JOIN Cancion c ON a.id_album = c.album_id
INNER JOIN usuario u ON u.id_usuario = a.usuario_id
INNER JOIN Artista ar ON a.artista_id = ar.id_artista
WHERE u.nombre LIKE '%Carlos Pérez%'
```

	usuario character varying (100)	cancion character varying (200)	album character varying (200)	artista character varying (100)
1	Carlos Pérez	Yellow	Parachutes	Coldplay

```
--Con ORDER BY
SELECT  u.nombre AS usuario, c.titulo AS cancion, a.titulo AS album, ar.nombre AS artista
FROM album a
INNER JOIN Cancion c ON a.id_album = c.album_id
INNER JOIN usuario u ON u.id_usuario = a.usuario_id
INNER JOIN Artista ar ON a.artista_id = ar.id_artista
ORDER BY u.nombre
```

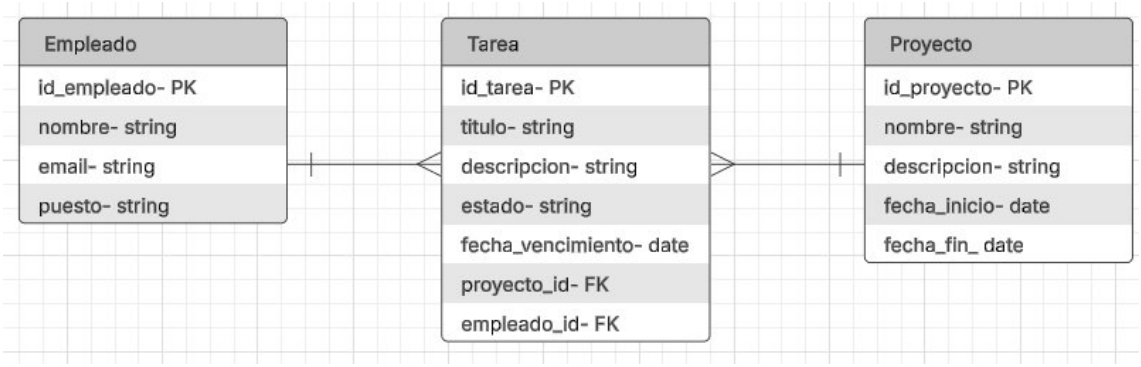
	usuario character varying (100)	cancion character varying (200)	album character varying (200)	artista character varying (100)
1	Ana Gómez	Shape of You	Divide	Ed Sheeran
2	Carlos Pérez	Yellow	Parachutes	Coldplay
3	Diego Castro	Just the Way You Are	Doo-Wops & Hooligans	Bruno Mars
4	Elena Sánchez	Shake It Off	1989	Taylor Swift
5	Javier Mendoza	Safaera	YHLQMDLG	Bad Bunny
6	Laura Torres	Here Comes The Sun	Abbey Road	The Beatles
7	Luis Ramírez	Hello	25	Adele
8	María López	Chantaje	El Dorado	Shakira
9	Pedro Fernández	Bohemian Rhapsody	A Night at the Opera	Queen
10	Sofía Herrera	Believer	Evolve	Imagine Dragons

```
--Con GROUP BY
SELECT  u.nombre AS usuario, c.titulo AS cancion, a.titulo AS album, ar.nombre AS artista
FROM album a
INNER JOIN Cancion c ON a.id_album = c.album_id
INNER JOIN usuario u ON u.id_usuario = a.usuario_id
INNER JOIN Artista ar ON a.artista_id = ar.id_artista
GROUP BY u.nombre,ar.nombre,c.titulo, a.titulo
```

	usuario character varying (100)	cancion character varying (200)	album character varying (200)	artista character varying (100)
1	Ana Gómez	Shape of You	Divide	Ed Sheeran
2	Javier Mendoza	Safaera	YHLQMDLG	Bad Bunny
3	Diego Castro	Just the Way You Are	Doo-Wops & Hooligans	Bruno Mars
4	Elena Sánchez	Shake It Off	1989	Taylor Swift
5	Carlos Pérez	Yellow	Parachutes	Coldplay
6	Pedro Fernández	Bohemian Rhapsody	A Night at the Opera	Queen
7	Sofía Herrera	Believer	Evolve	Imagine Dragons
8	María López	Chantaje	El Dorado	Shakira
9	Luis Ramírez	Hello	25	Adele
10	Laura Torres	Here Comes The Sun	Abbey Road	The Beatles

Sistema de control de proyectos

Una empresa desea hacer seguimiento de sus proyectos y tareas, sus entidades clave son las siguientes: Proyecto, empleado y tarea

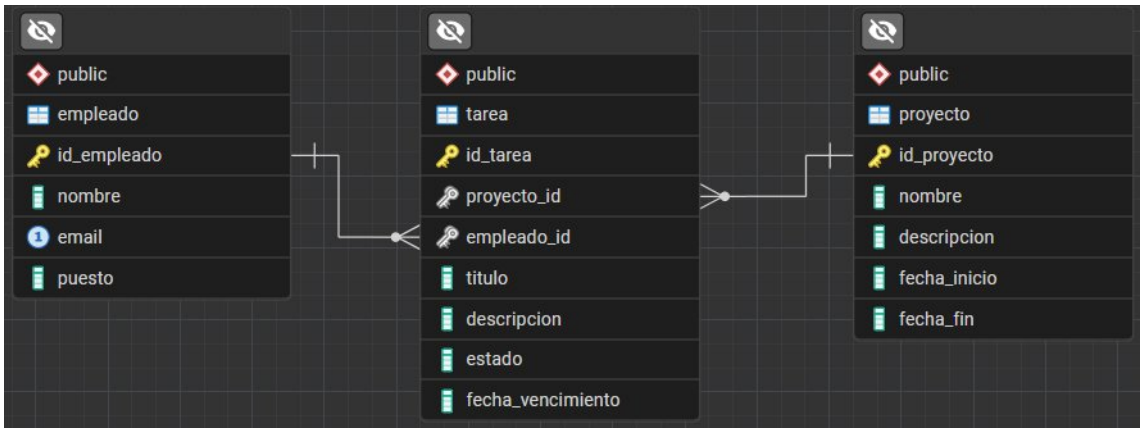


Con esto podemos crear nuestras tablas

```sql
CREATE TABLE Proyecto (
    id_proyecto SERIAL PRIMARY KEY,
    nombre VARCHAR(200) NOT NULL,
    descripcion TEXT,
    fecha_inicio DATE NOT NULL,
    fecha_fin DATE
);

CREATE TABLE Empleado (
    id_empleado SERIAL PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    puesto VARCHAR(100) NOT NULL
);

CREATE TABLE Tarea (
    id_tarea SERIAL PRIMARY KEY,
    proyecto_id INT NOT NULL,
    empleado_id INT,
    titulo VARCHAR(200) NOT NULL,
    descripcion TEXT,
    estado VARCHAR(20) CHECK (estado IN ('Pendiente', 'En Proceso', 'Completada')) DEFAULT 'Pendiente',
    fecha_vencimiento DATE NOT NULL,
    FOREIGN KEY (proyecto_id) REFERENCES Proyecto(id_proyecto) ON DELETE CASCADE,
    FOREIGN KEY (empleado_id) REFERENCES Empleado(id_empleado) ON DELETE SET NULL
);
```



Ahora insertamos datos en las tablas

```sql
INSERT INTO Proyecto (nombre, descripcion, fecha_inicio, fecha_fin) VALUES
('Desarrollo de Software', 'Crear una nueva plataforma web', '2024-01-10', '2024-06-30'),
('Marketing Digital', 'Estrategia para redes sociales', '2024-02-15', '2024-07-15'),
('Implementación ERP', 'Migración a nuevo sistema ERP', '2024-03-01', '2024-08-31'),
('Investigación de Mercado', 'Análisis de tendencias', '2024-01-20', '2024-05-15'),
('Construcción de Infraestructura', 'Nuevas oficinas', '2024-04-01', '2024-12-01'),
('Desarrollo de App Móvil', 'Aplicación para clientes', '2024-02-01', '2024-09-30'),
('Automatización de Procesos', 'Mejorar eficiencia', '2024-03-10', '2024-11-20'),
('Capacitación Interna', 'Formación de empleados', '2024-01-05', '2024-06-10'),
('Expansión Internacional', 'Nuevos mercados', '2024-05-01', '2024-12-31'),
('Optimización SEO', 'Mejorar posicionamiento web', '2024-02-20', '2024-07-10');

INSERT INTO Empleado (nombre, email, puesto) VALUES
('Carlos Pérez', 'carlos@email.com', 'Desarrollador'),
('Ana Gómez', 'ana@email.com', 'Gerente de Marketing'),
('Luis Ramírez', 'luis@email.com', 'Analista de Negocios'),
('María López', 'maria@email.com', 'Ingeniera de Software'),
('Pedro Fernández', 'pedro@email.com', 'Diseñador UX/UI'),
('Laura Torres', 'laura@email.com', 'Consultora ERP'),
('Diego Castro', 'diego@email.com', 'Especialista en SEO'),
('Elena Sánchez', 'elena@email.com', 'Project Manager'),
('Javier Mendoza', 'javier@email.com', 'Especialista en Datos'),
('Sofía Herrera', 'sofia@email.com', 'Coordinadora de Ventas');
```

```sql
INSERT INTO Tarea (proyecto_id, empleado_id, titulo, descripcion, estado, fecha_vencimiento) VALUES
(21, 21, 'Desarrollar backend', 'Implementar la API REST', 'Pendiente', '2024-05-10'),
(21, 24, 'Crear interfaz de usuario', 'Diseñar y desarrollar frontend', 'En Proceso', '2024-05-20'),
(22, 22, 'Plan de contenido', 'Definir estrategia de publicaciones', 'Pendiente', '2024-04-15'),
(23, 26, 'Migrar base de datos', 'Transferir datos al nuevo ERP', 'Pendiente', '2024-06-01'),
(24, 23, 'Encuesta de clientes', 'Recopilar datos de clientes', 'Completada', '2024-03-30'),
(25, 25, 'Supervisión de obras', 'Revisar avances de construcción', 'En Proceso', '2024-10-10'),
(26, 21, 'Implementar autenticación', 'Login y registro de usuarios', 'Pendiente', '2024-07-01'),
(27, 23, 'Automatizar informes', 'Generar reportes automáticos', 'Pendiente', '2024-09-15'),
(28, 28, 'Organizar capacitaciones', 'Definir agenda y temas', 'Pendiente', '2024-04-05'),
(29, 30, 'Análisis de mercado', 'Evaluar viabilidad en nuevos países', 'Pendiente', '2024-08-20');
```

Ahora podemos hacer las consultas

```sql
--con ORDER BY (principal)
SELECT  t.id_tarea, t.titulo, t.descripcion, e.nombre AS asignado_a, t.fecha_vencimiento
FROM Tarea t
LEFT JOIN Empleado e ON t.empleado_id = e.id_empleado
ORDER BY t.fecha_vencimiento;
```

	id_tarea integer	titulo character varying (200)	descripcion text	asignado_a character varying (100)	fecha_vencimiento date
1	15	Encuesta de clientes	Recopilar datos de clientes	Luis Ramírez	2024-03-30
2	19	Organizar capacitaciones	Definir agenda y temas	Elena Sánchez	2024-04-05
3	13	Plan de contenido	Definir estrategia de publicaciones	Ana Gómez	2024-04-15
4	11	Desarrollar backend	Implementar la API REST	Carlos Pérez	2024-05-10
5	12	Crear interfaz de usuario	Diseñar y desarrollar frontend	María López	2024-05-20
6	14	Migrar base de datos	Transferir datos al nuevo ERP	Laura Torres	2024-06-01
7	17	Implementar autenticación	Login y registro de usuarios	Carlos Pérez	2024-07-01
8	20	Análisis de mercado	Evaluar viabilidad en nuevos países	Sofía Herrera	2024-08-20
9	18	Automatizar informes	Generar reportes automáticos	Luis Ramírez	2024-09-15
10	16	Supervisión de obras	Revisar avances de construcción	Pedro Fernández	2024-10-10

```sql
--con WHERE
SELECT  t.id_tarea, t.titulo, t.descripcion, e.nombre AS asignado_a, t.fecha_vencimiento
FROM Tarea t
LEFT JOIN Empleado e ON t.empleado_id = e.id_empleado
WHERE t.proyecto_id = 21
```

	id_tarea integer	titulo character varying (200)	descripcion text	asignado_a character varying (100)	fecha_vencimiento date
1	11	Desarrollar backend	Implementar la API REST	Carlos Pérez	2024-05-10
2	12	Crear interfaz de usuario	Diseñar y desarrollar frontend	María López	2024-05-20

```sql
--con LIKE
SELECT  t.id_tarea, t.titulo, t.descripcion, e.nombre AS asignado_a, t.fecha_vencimiento
FROM Tarea t
LEFT JOIN Empleado e ON t.empleado_id = e.id_empleado
WHERE e.nombre LIKE '%María López%'
```

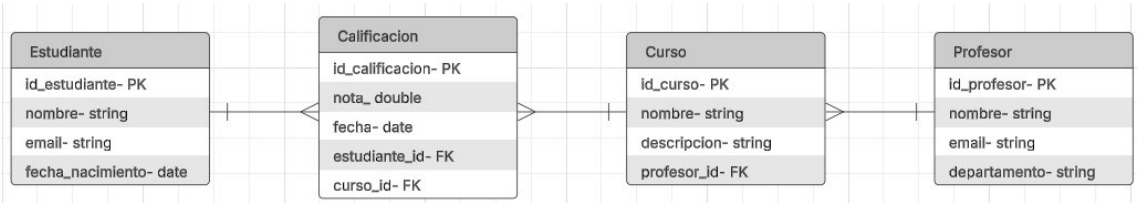
	id_tarea integer	titulo character varying (200)	descripcion text	asignado_a character varying (100)	fecha_vencimiento date
1	12	Crear interfaz de usuario	Diseñar y desarrollar frontend	María López	2024-05-20

```sql
--con GROUP BY
SELECT  t.id_tarea, t.titulo, t.descripcion, e.nombre AS asignado_a, t.fecha_vencimiento
FROM Tarea t
LEFT JOIN Empleado e ON t.empleado_id = e.id_empleado
GROUP BY t.id_tarea, t.titulo, t.descripcion, e.nombre
```

	id_tarea integer	titulo character varying (200)	descripcion text	asignado_a character varying (100)	fecha_vencimiento date
1	20	Análisis de mercado	Evaluar viabilidad en nuevos país...	Sofía Herrera	2024-08-20
2	19	Organizar capacitaciones	Definir agenda y temas	Elena Sánchez	2024-04-05
3	14	Migrar base de datos	Transferir datos al nuevo ERP	Laura Torres	2024-06-01
4	16	Supervisión de obras	Revisar avances de construcción	Pedro Fernández	2024-10-10
5	12	Crear interfaz de usuario	Diseñar y desarrollar frontend	María López	2024-05-20
6	17	Implementar autenticación	Login y registro de usuarios	Carlos Pérez	2024-07-01
7	18	Automatizar informes	Generar reportes automáticos	Luis Ramírez	2024-09-15
8	13	Plan de contenido	Definir estrategia de publicaciones	Ana Gómez	2024-04-15
9	15	Encuesta de clientes	Recopilar datos de clientes	Luis Ramírez	2024-03-30
10	11	Desarrollar backend	Implementar la API REST	Carlos Pérez	2024-05-10

Sistema de evaluacion academica

Una institucion educativa necesita administrar las calificaciones de los estudiantes, las entidades claves son las siguientes: Estudiante, curso, profesor y calificacion



Con esto podemos crear nuestras tablas

```sql
CREATE TABLE Profesor (
    id_profesor SERIAL PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    departamento VARCHAR(100) NOT NULL
);

CREATE TABLE Curso (
    id_curso SERIAL PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    descripcion TEXT,
    profesor_id INT NOT NULL,
    FOREIGN KEY (profesor_id) REFERENCES Profesor(id_profesor) ON DELETE CASCADE
);

CREATE TABLE Estudiante (
    id_estudiante SERIAL PRIMARY KEY,
    nombre VARCHAR(100) NOT NULL,
    email VARCHAR(100) UNIQUE NOT NULL,
    fecha_nacimiento DATE NOT NULL
);

CREATE TABLE Calificacion (
    id_calificacion SERIAL PRIMARY KEY,
    estudiante_id INT NOT NULL,
    curso_id INT NOT NULL,
    nota DECIMAL(4,2) NOT NULL,
    fecha DATE NOT NULL,
    FOREIGN KEY (estudiante_id) REFERENCES Estudiante(id_estudiante) ON DELETE CASCADE,
    FOREIGN KEY (curso_id) REFERENCES Curso(id_curso) ON DELETE CASCADE
);
```

Ahora insertamos neustros datos

```sql
INSERT INTO Profesor (nombre, email, departamento) VALUES
('Juan Pérez', 'juan.perez@universidad.edu', 'Matemáticas'),
('María López', 'maria.lopez@universidad.edu', 'Física'),
('Carlos Gómez', 'carlos.gomez@universidad.edu', 'Informática'),
('Ana Torres', 'ana.torres@universidad.edu', 'Historia'),
('Luis Rodríguez', 'luis.rodriguez@universidad.edu', 'Biología'),
('Elena Fernández', 'elena.fernandez@universidad.edu', 'Química'),
('Pedro Sánchez', 'pedro.sanchez@universidad.edu', 'Filosofía'),
('Sofía Martínez', 'sofia.martinez@universidad.edu', 'Economía'),
('Ricardo Díaz', 'ricardo.diaz@universidad.edu', 'Psicología'),
('Laura Moreno', 'laura.moreno@universidad.edu', 'Ingeniería');

INSERT INTO Curso (nombre, descripcion, profesor_id) VALUES
('Álgebra', 'Curso de álgebra básica', 1),
('Mecánica Clásica', 'Fundamentos de la mecánica', 2),
('Programación en Python', 'Curso introductorio a Python', 3),
('Historia Contemporánea', 'Análisis del siglo XX', 4),
('Genética', 'Introducción a la genética', 5),
('Química Orgánica', 'Conceptos básicos de química orgánica', 6),
('Ética Filosófica', 'Principios de ética y moral', 7),
('Macroeconomía', 'Principios de macroeconomía', 8),
('Psicología Cognitiva', 'Estudio de los procesos mentales', 9),
('Electrónica Digital', 'Fundamentos de la electrónica', 10);
```

```sql
INSERT INTO Estudiante (nombre, email, fecha_nacimiento) VALUES
('Andrés Herrera', 'andres.herrera@email.com', '2000-05-15'),
('Beatriz Sánchez', 'beatriz.sanchez@email.com', '2001-07-20'),
('Carlos Jiménez', 'carlos.jimenez@email.com', '2002-09-10'),
('Diana Torres', 'diana.torres@email.com', '1999-12-05'),
('Eduardo Martínez', 'eduardo.martinez@email.com', '2000-03-25'),
('Fernanda Rojas', 'fernanda.rojas@email.com', '2001-06-30'),
('Gabriel Díaz', 'gabriel.diaz@email.com', '2002-11-14'),
('Hugo Castro', 'hugo.castro@email.com', '1999-08-19'),
('Isabel Pérez', 'isabel.perez@email.com', '2000-04-10'),
('Jorge Gómez', 'jorge.gomez@email.com', '2001-10-23');

INSERT INTO Calificacion (estudiante_id, curso_id, nota, fecha) VALUES
(1, 1, 8.5, '2024-02-15'),
(2, 2, 9.0, '2024-02-16'),
(3, 3, 7.5, '2024-02-17'),
(4, 4, 6.8, '2024-02-18'),
(5, 5, 8.2, '2024-02-19'),
(6, 6, 9.5, '2024-02-20'),
(7, 7, 7.8, '2024-02-21'),
(8, 8, 6.5, '2024-02-22'),
(9, 9, 9.0, '2024-02-23'),
(10, 10, 8.7, '2024-02-24');
```

Ahora podemos consultar los datos

```sql
--con GROUP BY
SELECT e.nombre AS estudiante,
AVG(c.nota) AS promedio --AVG calcula el promedio
FROM Calificacion c
INNER JOIN Estudiante e ON c.estudiante_id = e.id_estudiante
GROUP BY e.nombre;
```

	estudiante character varying (100)	promedio numeric
1	Beatriz Sánchez	9.0000000000000000
2	Eduardo Martínez	8.2000000000000000
3	Isabel Pérez	9.0000000000000000
4	Jorge Gómez	8.7000000000000000
5	Gabriel Díaz	7.8000000000000000
6	Fernanda Rojas	9.5000000000000000
7	Andrés Herrera	8.5000000000000000
8	Carlos Jiménez	7.5000000000000000
9	Diana Torres	6.8000000000000000
10	Hugo Castro	6.5000000000000000

```sql
--con where
SELECT e.nombre AS nombre,
AVG(c.nota) AS promedio
FROM Calificacion c
JOIN Estudiante e ON c.estudiante_id = e.id_estudiante
WHERE e.nombre = 'Beatriz Sánchez'
GROUP BY e.nombre;
```

	nombre character varying (100)	promedio numeric
1	Beatriz Sánchez	9.0000000000000000

```sql
--con LIKE
SELECT e.nombre AS nombre,
AVG(c.nota) AS promedio
FROM Calificacion c
JOIN Estudiante e ON c.estudiante_id = e.id_estudiante
WHERE e.nombre LIKE '%Andrés Herrera%'
GROUP BY e.nombre;
```

	nombre character varying (100)	promedio numeric
1	Andrés Herrera	8.5000000000000000

```sql
--con ORDER BY
SELECT e.nombre AS estudiante,
AVG(c.nota) AS promedio --AVG calcula el promedio
FROM Calificacion c
INNER JOIN Estudiante e ON c.estudiante_id = e.id_estudiante
GROUP BY e.nombre
ORDER BY e.nombre;
```

	estudiante character varying (100)	promedio numeric
1	Andrés Herrera	8.5000000000000000
2	Beatriz Sánchez	9.0000000000000000
3	Carlos Jiménez	7.5000000000000000
4	Diana Torres	6.8000000000000000
5	Eduardo Martínez	8.2000000000000000
6	Fernanda Rojas	9.5000000000000000
7	Gabriel Díaz	7.8000000000000000
8	Hugo Castro	6.5000000000000000
9	Isabel Pérez	9.0000000000000000
10	Jorge Gómez	8.7000000000000000