SISTEMAS DE GESTIÓN DE DATOS

PROYECTO ENTREGA 2

INTEGRANTES:

MARIANA SANCHEZ ARAQUE

SAMUEL MONCADA MEJÍA

DOCENTE:

EDWIN NELSON MONTOYA

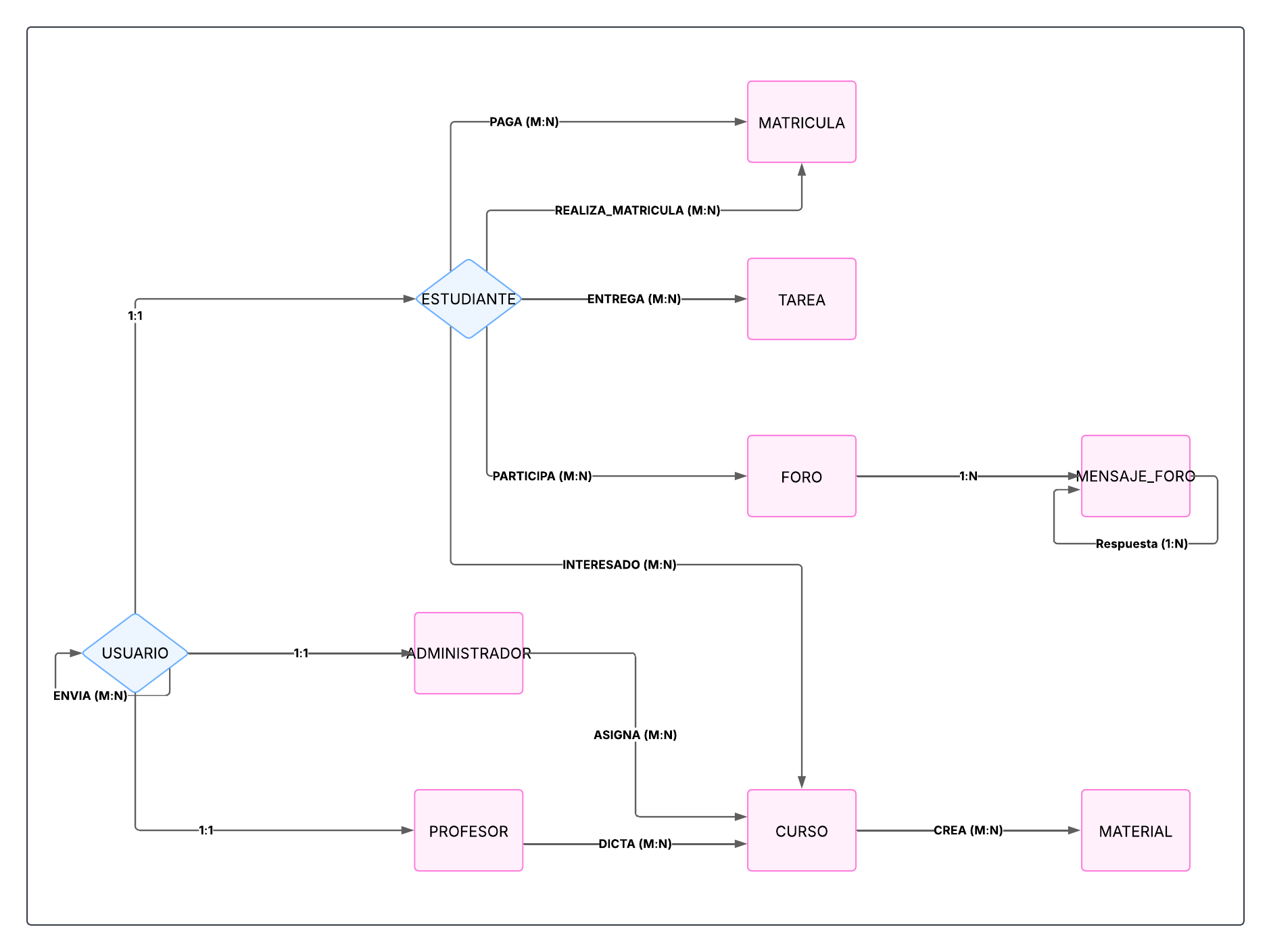
UNIVERSIDAD EAFIT

MEDELLÍN, COLOMBIA

2025

Modelo relacional (lógico)

a. Esquema de las relaciones.



<https://lucid.app/lucidchart/b37c175a-3002-468f-9e1d-8d4fb7af7d83/edit?viewport_loc=-20%2C562%2C2218%2C938%2C0_0&invitationId=inv_2cb1e50c-b3cd-4ec2-9ac4-c40f8d300412>

b. Proceso de normalización.

PASO 1:

1FN:

Como cambios al diseño SQL inicial, eliminamos la duplicación de datos en “MENSAJE\_FORO”, además separamos la información que no es atómica en “TAREA”

CREATE TABLE MENSAJE\_FORO (

id\_mensaje INT PRIMARY KEY,

id\_foro INT,

id\_usuario\_emisor INT,

id\_mensaje\_respuesta INT,

contenido TEXT,

FOREIGN KEY (id\_foro) REFERENCES FORO(id\_foro),

FOREIGN KEY (id\_usuario\_emisor) REFERENCES USUARIO(id\_usuario),

FOREIGN KEY (id\_mensaje\_respuesta) REFERENCES MENSAJE\_FORO(id\_mensaje)

);

CREATE TABLE TAREA (

id\_tarea INT PRIMARY KEY,

nombre\_tarea VARCHAR(100),

descripcion\_tarea TEXT,

fecha\_creacion DATE

);

PASO 2:

2FN:

Ahora debemos eliminar las dependencias parciales, entonces creamos una tabla aparte “ENTREGA”

CREATE TABLE ENTREGA (

id\_entrega INT PRIMARY KEY,

id\_estudiante INT,

id\_tarea INT,

archivo\_estudiante TEXT,

fecha\_entrega DATE,

puntaje DECIMAL(5,2),

FOREIGN KEY (id\_estudiante) REFERENCES ESTUDIANTE(id\_usuario),

FOREIGN KEY (id\_tarea) REFERENCES TAREA(id\_tarea)

);

PASO 3:

3FN:

Debemos eliminar dependencias transitivas, separamos “ADMINISTRADOR”, “ESTUDIANTE” y “PROFESOR” y evitamos que atributos dependan de no-claves.

-- USUARIO general

CREATE TABLE USUARIO (

id\_usuario INT PRIMARY KEY,

documento VARCHAR(20),

nombre\_usuario VARCHAR(100),

email VARCHAR(100),

contrasena VARCHAR(100),

genero VARCHAR(10)

);

CREATE TABLE ESTUDIANTE (

id\_usuario INT PRIMARY KEY,

especializacion VARCHAR(100),

id\_nodo INT,

FOREIGN KEY (id\_usuario) REFERENCES USUARIO(id\_usuario)

);

CREATE TABLE PROFESOR (

id\_usuario INT PRIMARY KEY,

nombre\_profesor VARCHAR(100),

telefono VARCHAR(20),

area\_principal\_conocimiento VARCHAR(100),

area\_alternativa\_conocimiento VARCHAR(100),

contrasena\_pg VARCHAR(100),

FOREIGN KEY (id\_usuario) REFERENCES USUARIO(id\_usuario)

);

CREATE TABLE ADMINISTRADOR (

id\_usuario INT PRIMARY KEY,

nombre\_completo\_Ad VARCHAR(100),

correo\_Admin VARCHAR(100),

contrasena\_Admin VARCHAR(100),

FOREIGN KEY (id\_usuario) REFERENCES USUARIO(id\_usuario)

);

SQL FINAL:

CREATE TABLE USUARIO (

id\_usuario INT AUTO\_INCREMENT PRIMARY KEY,

nombre\_usuario VARCHAR(100),

documento VARCHAR(50),

genero VARCHAR(20),

email VARCHAR(100),

contrasena VARCHAR(100),

tipo\_usuario VARCHAR(20) -- estudiante, profesor, administrador

);

CREATE TABLE ESTUDIANTE (

id\_usuario INT PRIMARY KEY,

especializacion VARCHAR(100),

id\_nodo INT,

FOREIGN KEY (id\_usuario) REFERENCES USUARIO(id\_usuario)

);

CREATE TABLE PROFESOR (

id\_usuario INT PRIMARY KEY,

telefono VARCHAR(20),

area\_principal\_conocimiento VARCHAR(100),

area\_alternativa\_conocimiento VARCHAR(100),

FOREIGN KEY (id\_usuario) REFERENCES USUARIO(id\_usuario)

);

CREATE TABLE ADMINISTRADOR (

id\_usuario INT PRIMARY KEY,

FOREIGN KEY (id\_usuario) REFERENCES USUARIO(id\_usuario)

);

CREATE TABLE CURSO (

id\_curso INT AUTO\_INCREMENT PRIMARY KEY,

nombre\_curso VARCHAR(100),

categoria VARCHAR(50),

ruta VARCHAR(100),

precio DECIMAL(10,2),

fecha\_inicio DATE,

fecha\_fin DATE,

semestre VARCHAR(20),

año INT

);

CREATE TABLE TAREA (

id\_tarea INT AUTO\_INCREMENT PRIMARY KEY,

nombre\_tarea VARCHAR(100),

descripcion\_tarea TEXT,

fecha\_creacion DATE,

fecha\_entrega DATE,

puntaje DECIMAL(5,2)

);

CREATE TABLE ENTREGA (

id\_usuario INT,

id\_tarea INT,

archivo\_estudiante TEXT,

fecha\_entrega DATE,

puntaje DECIMAL(5,2),

PRIMARY KEY (id\_usuario, id\_tarea),

FOREIGN KEY (id\_usuario) REFERENCES ESTUDIANTE(id\_usuario),

FOREIGN KEY (id\_tarea) REFERENCES TAREA(id\_tarea)

);

CREATE TABLE MATERIAL (

id\_material INT AUTO\_INCREMENT PRIMARY KEY,

titulo VARCHAR(100),

descripcion TEXT

);

CREATE TABLE FORO (

id\_foro INT AUTO\_INCREMENT PRIMARY KEY,

nombre\_foro VARCHAR(100),

descripcion\_foro TEXT,

fecha\_creacion DATE,

fecha\_terminacion DATE

);

CREATE TABLE MENSAJE\_FORO (

id\_mensaje INT AUTO\_INCREMENT PRIMARY KEY,

id\_foro INT,

id\_usuario\_emisor INT,

id\_mensaje\_respuesta INT,

contenido TEXT,

FOREIGN KEY (id\_foro) REFERENCES FORO(id\_foro),

FOREIGN KEY (id\_usuario\_emisor) REFERENCES USUARIO(id\_usuario),

FOREIGN KEY (id\_mensaje\_respuesta) REFERENCES MENSAJE\_FORO(id\_mensaje)

);

CREATE TABLE MATRICULA (

id\_matricula INT AUTO\_INCREMENT PRIMARY KEY,

nombre\_matricula VARCHAR(100),

contrasena VARCHAR(100)

);

-- RELACIONES

CREATE TABLE DICTA (

id\_usuario INT,

id\_curso INT,

PRIMARY KEY (id\_usuario, id\_curso),

FOREIGN KEY (id\_usuario) REFERENCES PROFESOR(id\_usuario),

FOREIGN KEY (id\_curso) REFERENCES CURSO(id\_curso)

);

CREATE TABLE INTERESADO (

id\_usuario INT,

id\_curso INT,

PRIMARY KEY (id\_usuario, id\_curso),

FOREIGN KEY (id\_usuario) REFERENCES ESTUDIANTE(id\_usuario),

FOREIGN KEY (id\_curso) REFERENCES CURSO(id\_curso)

);

CREATE TABLE CREA (

id\_curso INT,

id\_material INT,

PRIMARY KEY (id\_curso, id\_material),

FOREIGN KEY (id\_curso) REFERENCES CURSO(id\_curso),

FOREIGN KEY (id\_material) REFERENCES MATERIAL(id\_material)

);

CREATE TABLE PARTICIPA (

id\_usuario INT,

id\_foro INT,

PRIMARY KEY (id\_usuario, id\_foro),

FOREIGN KEY (id\_usuario) REFERENCES ESTUDIANTE(id\_usuario),

FOREIGN KEY (id\_foro) REFERENCES FORO(id\_foro)

);

CREATE TABLE REALIZA\_MATRICULA (

id\_usuario INT,

id\_matricula INT,

PRIMARY KEY (id\_usuario, id\_matricula),

FOREIGN KEY (id\_usuario) REFERENCES ESTUDIANTE(id\_usuario),

FOREIGN KEY (id\_matricula) REFERENCES MATRICULA(id\_matricula)

);

CREATE TABLE PAGA (

id\_usuario INT,

id\_matricula INT,

comprobante\_pago VARCHAR(100),

PRIMARY KEY (id\_usuario, id\_matricula),

FOREIGN KEY (id\_usuario) REFERENCES ESTUDIANTE(id\_usuario),

FOREIGN KEY (id\_matricula) REFERENCES MATRICULA(id\_matricula)

);

CREATE TABLE ENVIA (

id\_usuario\_emisor INT,

id\_usuario\_receptor INT,

PRIMARY KEY (id\_usuario\_emisor, id\_usuario\_receptor),

FOREIGN KEY (id\_usuario\_emisor) REFERENCES USUARIO(id\_usuario),

FOREIGN KEY (id\_usuario\_receptor) REFERENCES USUARIO(id\_usuario)

);

CREATE TABLE ASIGNA (

id\_usuario INT,

id\_curso INT,

PRIMARY KEY (id\_usuario, id\_curso),

FOREIGN KEY (id\_usuario) REFERENCES ADMINISTRADOR(id\_usuario),

FOREIGN KEY (id\_curso) REFERENCES CURSO(id\_curso)

);

--POBLAR LA BASE DE DATOS

INSERT INTO USUARIO (nombre\_usuario, documento, genero, email, contrasena, tipo\_usuario)

VALUES

('Juan Pérez', '12345678', 'Masculino', 'juanp@example.com', '1234pass', 'estudiante'),

('Laura Gómez', '87654321', 'Femenino', 'laurag@example.com', 'abcd1234', 'profesor'),

('Ana Torres', '11223344', 'Femenino', 'ana.admin@example.com', 'adminpass', 'administrador');

INSERT INTO ESTUDIANTE (id\_usuario, especializacion, id\_nodo)

VALUES (1, 'Inteligencia Artificial', 101);

INSERT INTO PROFESOR (id\_usuario, telefono, area\_principal\_conocimiento, area\_alternativa\_conocimiento)

VALUES (2, '555-1234', 'Bases de Datos', 'Redes');

INSERT INTO ADMINISTRADOR (id\_usuario)

VALUES (3);

INSERT INTO CURSO (nombre\_curso, categoria, ruta, precio, fecha\_inicio, fecha\_fin, semestre, año)

VALUES

('Programación I', 'Computación', 'ruta\_prog1', 300.00, '2025-03-01', '2025-07-01', '1', 2025),

('Redes de Computadoras', 'Sistemas', 'ruta\_redes', 350.00, '2025-03-01', '2025-07-01', '1', 2025);

INSERT INTO MATRICULA (nombre\_matricula, contrasena)

VALUES ('Matricula 2025-A', 'mat2025pass');

INSERT INTO TAREA (nombre\_tarea, descripcion\_tarea, fecha\_creacion, fecha\_entrega, puntaje)

VALUES

('Tarea 1', 'Resolver ejercicios de recursividad', '2025-03-05', '2025-03-20', 10.0),

('Tarea 2', 'Diseñar una red LAN', '2025-03-10', '2025-03-25', 15.0);

INSERT INTO ENTREGA (id\_usuario, id\_tarea, archivo\_estudiante, fecha\_entrega, puntaje)

VALUES

(1, 1, 'recursividad.pdf', '2025-03-18', 9.5),

(1, 2, 'redLAN.zip', '2025-03-24', 14.0);

INSERT INTO MATERIAL (titulo, descripcion)

VALUES

('PDF Recursividad', 'Guía completa sobre recursividad en Python'),

('Presentación Redes', 'Slides sobre topologías de red');

INSERT INTO FORO (nombre\_foro, descripcion\_foro, fecha\_creacion, fecha\_terminacion)

VALUES

('Foro Introducción a la Programación', 'Discusión sobre el curso', '2025-03-01', '2025-07-01');

INSERT INTO MENSAJE\_FORO (id\_foro, id\_usuario\_emisor, id\_mensaje\_respuesta, contenido)

VALUES

(1, 1, NULL, '¿Alguien entendió el ejercicio 3?');

INSERT INTO MENSAJE\_FORO (id\_foro, id\_usuario\_emisor, id\_mensaje\_respuesta, contenido)

VALUES

(1, 2, 1, 'Sí, puedo explicártelo mañana.');

-- Profesor dicta cursos

INSERT INTO DICTA (id\_usuario, id\_curso)

VALUES (2, 1), (2, 2);

-- Estudiante interesado en cursos

INSERT INTO INTERESADO (id\_usuario, id\_curso)

VALUES (1, 1), (1, 2);

-- Curso crea material

INSERT INTO CREA (id\_curso, id\_material)

VALUES (1, 1), (2, 2);

-- Estudiante participa en foro

INSERT INTO PARTICIPA (id\_usuario, id\_foro)

VALUES (1, 1);

-- Estudiante realiza matrícula

INSERT INTO REALIZA\_MATRICULA (id\_usuario, id\_matricula)

VALUES (1, 1);

-- Estudiante paga matrícula

INSERT INTO PAGA (id\_usuario, id\_matricula, comprobante\_pago)

VALUES (1, 1, 'comp123.pdf');

-- Comunicación entre usuarios

INSERT INTO ENVIA (id\_usuario\_emisor, id\_usuario\_receptor)

VALUES (1, 2), (2, 1);

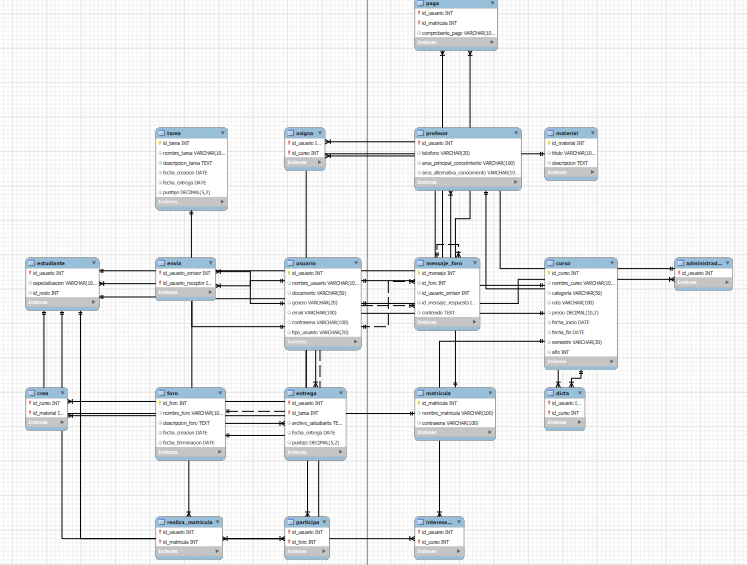
-- Administrador asigna curso

INSERT INTO ASIGNA (id\_usuario, id\_curso)

VALUES (3, 1);

c. Diagrama del esquema de la base de datos. (Diagrama workbench mysql.)

Ver archivo adjunto diagrama workbench.mwb



MODELO FISICO

a. Implementación en la base de datos MySQL. Archivo proyecto\_ddl.sql realizando la creación de la base de datos y las tablas con su integridad referencia.

Ver archivo adjunto proyecto\_ddl.sql

b. Crear un archivo scritpt sql para ‘poblar’ la base de datos en sus tablas maestras y algunas transacciones (INSERT INTO) Para tablas como Usuarios, Administrado, Cursos, etc. (archivo proyecto\_dml.sql).

ver script ajunto script.sql

CONSULTAS

-- 1. Listar estudiantes (nombre completo, matrícula) ordenados alfabéticamente por año y semestre

SELECT u.nombre\_usuario, m.nombre\_matricula, c.año, c.semestre

FROM USUARIO u

JOIN ESTUDIANTE e ON u.id\_usuario = e.id\_usuario

JOIN REALIZA\_MATRICULA rm ON e.id\_usuario = rm.id\_usuario

JOIN MATRICULA m ON rm.id\_matricula = m.id\_matricula

JOIN CURSO c ON c.año IS NOT NULL

ORDER BY c.año, c.semestre, u.nombre\_usuario;

-- 2. Estudiantes por año y semestre concretos (ejemplo: año = 2025, semestre = '1')

SELECT u.nombre\_usuario

FROM USUARIO u

JOIN ESTUDIANTE e ON u.id\_usuario = e.id\_usuario

JOIN INTERESADO i ON e.id\_usuario = i.id\_usuario

JOIN CURSO c ON i.id\_curso = c.id\_curso

WHERE c.año = 2025 AND c.semestre = '1';

-- 3. Cursos entre dos fechas

SELECT nombre\_curso, fecha\_inicio, fecha\_fin

FROM CURSO

WHERE fecha\_inicio BETWEEN '2025-03-01' AND '2025-09-01';

-- 4. Profesores (nombre + usuarios activos)

SELECT u.nombre\_usuario, u.documento

FROM USUARIO u

JOIN PROFESOR p ON u.id\_usuario = p.id\_usuario;

-- 5. Cursos ordenados por categoría

SELECT nombre\_curso, categoria

FROM CURSO

ORDER BY categoria;

-- 6. Cursos entre un rango de precio

SELECT nombre\_curso, precio

FROM CURSO

WHERE precio BETWEEN 100.00 AND 900.00;

-- 7. Usuarios registrados pero no matriculados en cursos

SELECT u.nombre\_usuario

FROM USUARIO u

LEFT JOIN ESTUDIANTE e ON u.id\_usuario = e.id\_usuario

LEFT JOIN REALIZA\_MATRICULA rm ON u.id\_usuario = rm.id\_usuario

WHERE rm.id\_usuario IS NULL AND u.tipo\_usuario = 'estudiante';

-- 8. Cursos por categoría (ejemplo: Computación)

SELECT nombre\_curso

FROM CURSO

WHERE categoria = 'Computación';

-- 9. Estudiantes que se han inscrito en un curso (ejemplo: id\_curso = 1)

SELECT u.nombre\_usuario

FROM USUARIO u

JOIN ESTUDIANTE e ON u.id\_usuario = e.id\_usuario

JOIN INTERESADO i ON e.id\_usuario = i.id\_usuario

WHERE i.id\_curso = 1;

-- 10. Materiales creados por el profesor para un curso (ejemplo: curso id = 1)

SELECT m.titulo, m.descripcion

FROM MATERIAL m

JOIN CREA cr ON m.id\_material = cr.id\_material

JOIN CURSO c ON cr.id\_curso = c.id\_curso

WHERE c.id\_curso = 1;

-- 11. Mensajes en un foro (ejemplo foro id = 1)

SELECT mf.contenido, u.nombre\_usuario

FROM MENSAJE\_FORO mf

JOIN USUARIO u ON mf.id\_usuario\_emisor = u.id\_usuario

WHERE mf.id\_foro = 1;

-- 12. Consulta adicional: Cantidad de tareas entregadas por cada estudiante

SELECT u.nombre\_usuario, COUNT(e.id\_tarea) AS tareas\_entregadas

FROM USUARIO u

JOIN ESTUDIANTE s ON u.id\_usuario = s.id\_usuario

JOIN ENTREGA e ON s.id\_usuario = e.id\_usuario

GROUP BY u.nombre\_usuario;  
  
Ver archivo adjunto **proyecto\_query.sql**