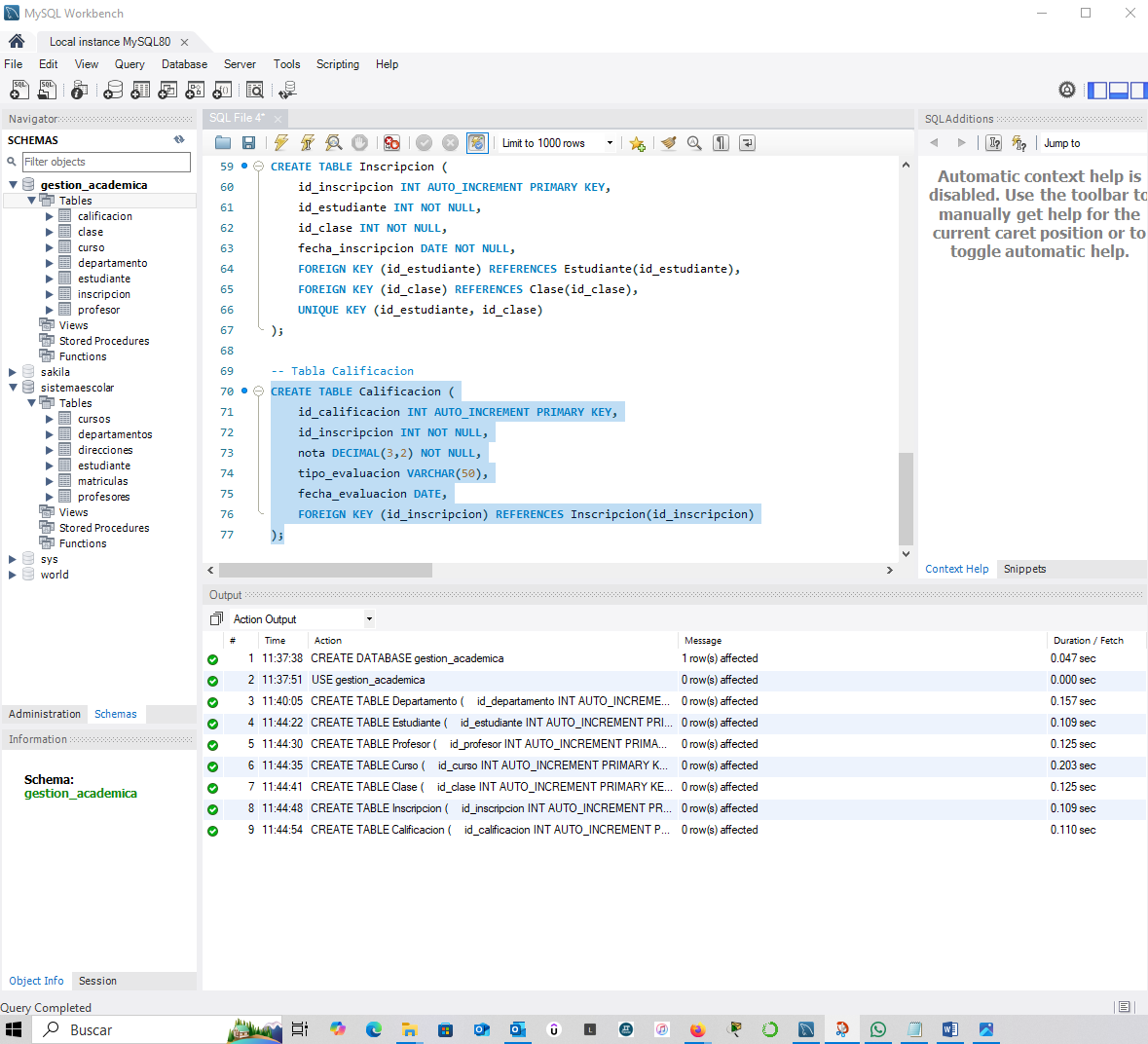
**Proyecto Final Introducción a las Bases de datos**

Diseñar y desarrollar una base de datos en MySQL que refleje la relación 1:N entre tablas, aplicando claves primarias, foráneas, consultas básicas, joins, estadísticas y manipulación de datos (insertar, actualizar y eliminar).

**Estructura del proyecto**

Tendremos un sistema de gestión académica simple con estas entidades:

* **Departamento**
* **Estudiante**
* **Profesor**
* **Curso**
* **Clase**
* **Inscripcion**
* **Calificacion**

****

**Insertar**

Insertar 100 Registro en cada tabla

-- Insertar 10 departamentos

INSERT INTO Departamento (nombre, edificio, presupuesto) VALUES

-- Insertar 100 estudiantes (ejemplo de 10)

INSERT INTO Estudiante (nombre, apellido, email, id\_departamento, fecha\_ingreso) VALUES

-- Insertar 100 profesores (ejemplo de 10, repetir patrones similares)

INSERT INTO Profesor (nombre, apellido, email, id\_departamento, fecha\_contratacion) VALUES

-- Insertar 40 cursos (relacionados con departamentos)

INSERT INTO Curso (codigo, nombre, creditos, id\_departamento) VALUES

-- Insertar 100 clases (relacionando cursos y profesores)

INSERT INTO Clase (id\_curso, id\_profesor, semestre, año, aula, horario) VALUES

-- Insertar 100 inscripciones (relacionando estudiantes y clases)

INSERT INTO Inscripcion (id\_estudiante, id\_clase, fecha\_inscripcion) VALUES

-- Insertar 150 calificaciones (relacionadas con inscripciones)

INSERT INTO Calificacion (3 por cada inscripción hasta la 50) ( id\_inscripcion, nota, tipo\_evaluacion, fecha\_evaluacion) VALUES

**Consultas Basicas Y avanzada**

1. Mostrar el nombre y apellido de todos los estudiantes ordenados alfabéticamente por apellido.

-- 1. Mostrar el nombre y apellido de todos los estudiantes ordenados alfabéticamente por apellido

SELECT nombre, apellido

FROM Estudiante

ORDER BY apellido;

1. Mostrar todos los cursos que tienen más de 3 créditos.

-- 2. Mostrar todos los cursos que tienen más de 3 créditos

SELECT nombre, creditos

FROM Curso

WHERE creditos > 3;

1. Mostrar el nombre del estudiante y el nombre del curso en el que está inscrito (INNER JOIN).

-- 3. Mostrar el nombre del estudiante y el nombre del curso en el que está inscrito (INNER JOIN)

SELECT e.nombre, e.apellido, c.nombre AS curso

FROM Estudiante e

JOIN Inscripcion i ON e.id\_estudiante = i.id\_estudiante

JOIN Clase cl ON i.id\_clase = cl.id\_clase

JOIN Curso c ON cl.id\_curso = c.id\_curso;

1. Mostrar todos los estudiantes y, si están inscritos, el curso correspondiente (LEFT JOIN).

-- 4. Mostrar todos los estudiantes y, si están inscritos, el curso correspondiente (LEFT JOIN)

SELECT e.nombre, e.apellido, c.nombre AS curso

FROM Estudiante e

LEFT JOIN Inscripcion i ON e.id\_estudiante = i.id\_estudiante

LEFT JOIN Clase cl ON i.id\_clase = cl.id\_clase

LEFT JOIN Curso c ON cl.id\_curso = c.id\_curso;

1. Mostrar todos los cursos, incluyendo los que aún no tienen estudiantes inscritos (RIGHT JOIN).

-- 5. Mostrar todos los cursos, incluyendo los que aún no tienen estudiantes inscritos (RIGHT JOIN)

SELECT c.nombre AS curso, COUNT(i.id\_inscripcion) AS estudiantes\_inscritos

FROM Inscripcion i

RIGHT JOIN Clase cl ON i.id\_clase = cl.id\_clase

RIGHT JOIN Curso c ON cl.id\_curso = c.id\_curso

GROUP BY c.id\_curso;

1. Contar cuántos estudiantes hay por departamento.

-- 6. Contar cuántos estudiantes hay por departamento

SELECT d.nombre AS departamento, COUNT(e.id\_estudiante) AS total\_estudiantes

FROM Departamento d

LEFT JOIN Estudiante e ON d.id\_departamento = e.id\_departamento

GROUP BY d.id\_departamento;

1. Calcular el promedio de notas por estudiante.

-- 7. Calcular el promedio de notas por estudiante

SELECT e.nombre, e.apellido, AVG(ca.nota) AS promedio\_notas

FROM Estudiante e

JOIN Inscripcion i ON e.id\_estudiante = i.id\_estudiante

JOIN Calificacion ca ON i.id\_inscripcion = ca.id\_inscripcion

GROUP BY e.id\_estudiante;

1. Mostrar la nota máxima y mínima por clase.

-- 8. Mostrar la nota máxima y mínima por clase

SELECT cl.id\_clase, cu.nombre AS curso,

MAX(ca.nota) AS nota\_maxima,

MIN(ca.nota) AS nota\_minima

FROM Clase cl

JOIN Curso cu ON cl.id\_curso = cu.id\_curso

JOIN Inscripcion i ON cl.id\_clase = i.id\_clase

JOIN Calificacion ca ON i.id\_inscripcion = ca.id\_inscripcion

GROUP BY cl.id\_clase;

1. Mostrar los **5 estudiantes** con el **mayor promedio de notas**.

-- 9. Mostrar los 5 estudiantes con el mayor promedio de notas

SELECT e.nombre, e.apellido, AVG(ca.nota) AS promedio\_notas

FROM Estudiante e

JOIN Inscripcion i ON e.id\_estudiante = i.id\_estudiante

JOIN Calificacion ca ON i.id\_inscripcion = ca.id\_inscripcion

GROUP BY e.id\_estudiante

ORDER BY promedio\_notas DESC

LIMIT 5;

.

**Actualizar y borrar**

1. Cambiar el correo electrónico de un estudiante

-- Actualizar y Borrar

-- 10. Cambiar el correo electrónico de un estudiante

UPDATE Estudiante

SET email = 'nuevo.email@estudiante.edu'

WHERE id\_estudiante = 1;

-- Actualizar los créditos de un curso

UPDATE Curso

SET creditos = 5

WHERE codigo = 'CS101';

1. Eliminar una inscripción de un estudiante

-- Eliminar una inscripción

DELETE FROM Inscripcion

WHERE id\_estudiante = 5 AND id\_clase = 3;

-- Eliminar todas las calificaciones menores a 3.0

DELETE FROM Calificacion

WHERE nota < 3.0;

-- 1: Eliminar por ID de estudiante y ID de clase

-- Eliminar la inscripción del estudiante con id=5 en la clase con id=10

DELETE FROM Inscripcion

WHERE id\_estudiante = 5 AND id\_clase = 10;

-- 2: Eliminar usando nombres en lugar de IDs (más amigable

-- Eliminar la inscripción de "Juan Pérez" en la clase de "Cálculo I"

DELETE FROM Inscripcion

WHERE id\_estudiante = (

SELECT id\_estudiante FROM Estudiante

WHERE nombre = 'Juan' AND apellido = 'Pérez'

)

AND id\_clase = (

SELECT id\_clase FROM Clase

JOIN Curso ON Clase.id\_curso = Curso.id\_curso

WHERE Curso.nombre = 'Cálculo I'

);

-- 3: Eliminar todas las inscripciones de un estudiante

-- Eliminar todas las inscripciones del estudiante con id=8

DELETE FROM Inscripcion

WHERE id\_estudiante = 8;

-- eliminar las calificaciones asociadas:

-- Primero eliminar las calificaciones relacionadas

DELETE FROM Calificacion

WHERE id\_inscripcion IN (

SELECT id\_inscripcion FROM Inscripcion

WHERE id\_estudiante = 5 AND id\_clase = 10

);

-- Luego eliminar la inscripción

DELETE FROM Inscripcion

WHERE id\_estudiante = 5 AND id\_clase = 10;

-- Otra manera de eliminar

START TRANSACTION;

DELETE FROM Calificacion

WHERE id\_inscripcion IN (

SELECT id\_inscripcion FROM Inscripcion

WHERE id\_estudiante = 5 AND id\_clase = 10

);

DELETE FROM Inscripcion

WHERE id\_estudiante = 5 AND id\_clase = 10;

Como entregar:

En el mismo repositorio de Bases de datos subir una carpeta que diga Trabajo final y agregar el link del repositorio al siguiente URL:

<https://docs.google.com/spreadsheets/d/1Aok9n7cke_G16fXLlgS_V_WXCdAkiCCPOu8bgR63zUo/edit?gid=0#gid=0>

Fecha de entrega final: Lunes 5 de Mayo 2025